Memetic Comp. (2010) 2:3–24
DOI 10.1007/s12293-009-0012-0

SPECIAL ISSUE

Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization

Yu Wang · Bin Li

Received: 12 November 2008 / Accepted: 13 August 2009 / Published online: 1 September 2009

Abstract Dynamic optimization and multi-objective optimization have separately gained increasing attention from the research community during the last decade. However, few studies have been reported on dynamic multi-objective optimization (dMO) and scarce effective dMO methods have been proposed. In this paper, we fulfill these gabs by developing new dMO test problems and new effective dMO algorithm. In the newly designed dMO problems, Pareto-optimal decision values (i.e., Pareto-optimal solutions: POS) or both POS and Pareto-optimal objective values (i.e., Pareto-optimal front: POF) change with time. A new multi-strategy ensemble multi-objective evolutionary algorithm (MS-MOEA) is proposed to tackle the challenges of dMO. In MS-MOEA, the convergence speed is accelerated by the new offspring creating mechanism powered by adaptive genetic and differential operators (GDM); a Gaussian mutation operator is employed to cope with premature convergence; a memory like strategy is proposed to achieve better starting population when a change takes place. In order to show the advantages of the proposed algorithm, we experimentally compare MS-MOEA with several algorithms equipped with traditional restart strategy. It is suggested that such a multi-strategy ensemble approach is promising for dealing with dMO problems.

Y. Wang · B. Li (✉)
Nature Inspired Computation and Applications Laboratory,
Department of Electronic Science and Technology,
University of Science and Technology of China,
230027 Hefei, Anhui, China
e-mail: binli@ustc.edu.cn

Y. Wang
e-mail: wyustc@mail.ustc.edu.cn

Keywords Dynamic environment · Multi-objective optimization · Evolutionary algorithm · Multi-strategy ensemble · Gaussian mutation

1 Introduction

In real world, change and multiple objective are common features of many important engineering and scientific problems [2,3,23]. For example in controller designing domain, the accuracy, robustness and response time are all important objectives that the controller under-design should pursuit [3]. While when the aging effect of the systems, as well as other elements such as using different fuels, that will cause the systems' performances change at different time, are taken into consideration, the problem turns to be a dynamic multi-objective optimization problem, which means that the controller should be able to adapt to the changing system with the pursuit of multiple objectives. Another example of dynamic multi-objective problem is the routing of vehicles. people always want to get to the target with the shortest transport time and the least transport cost. As is known to all, the road condition is changing from time to time, which makes the routing a dynamic multi-objective optimization problem.

In recent years, there has been a growing interest in studying evolutionary algorithms (EAs) for dynamic optimization problems [30,54]. Previously, various EAs have been proposed to tackle such a challenging task [4,5,7–12,14,21,48,51] with many successful applications. However, most of these studies focused on dynamic single objective (dSO) optimization problems, but few studies have been reported on dynamic multi-objective (dMO) optimization. As far as the authors know, the only works reported for dMO are [2,6,23,24]. In a certain sense, dMO is a hard task, because it synthesizes the difficulties of both static multi-objective optimization (sMO) and dSO [23]. Such difficulties are

summarized as follows, and further discussion is given in Sect. 2:

- Compared with sMO problems, dMO problems set a higher demand on convergence speed of algorithms, which can be described as that POS should be found within the stable duration. Meanwhile, the conflicts between convergence speed and diversity maintenance are much more serious for dMO [19,33,59]: Accelerating unilaterally the convergence speed will not only result in loss of diversity in decision space, but also result in loss of some parts of POF. Static multi-objective optimization evolutionary algorithms (sMOEAs) are believed to be able to cope with dMO problems at the very beginning [23], but will inevitably lose their effectiveness when the environment changes.
- Compared with that a separate environment usually has only one or a limited number of global optimal solutions in most cases of dSO, the goal of dMO is to find an approximation to the true POS [23]. Most multi-objective problems may have many or even infinite optimal solutions (i.e., POS). Therefore, besides the similar goal of approaching as quick as possible the global optimal solutions in dSO, dMO has one another important goal of maintaining a set of optimal solutions widely distributed along the true POS.

Benchmark problems are important for designing and evaluating algorithms for dMO [23]. Motivated by the fact that few dMO benchmark problems are available by now, this paper designs a suit of four test instances offering changes in POS and one test instance offering changes in both POS and POF. Compared with the test suite FDAs proposed in [23], the POFs of these new instances can be mathematically described. Furthermore, their POFs have different characteristics, such as convex, non-convex and disconnected. Besides, the stable duration of these test instants can be tuned to simulate various practical cases.

In the perspective of algorithm designing, two important issues are analyzed in this paper: convergence speed and dynamic environment handling strategy. Then, a multi-strategy ensemble framework is proposed. In this framework, different components (strategies), such as memory like strategy, GDM and Gaussian mutation, are designed to address different issues of dMO.

The remainder of this paper is structured as follows: In Sect. 2, the related works on sMOEAs and dynamic optimization are reviewed. Then, the necessity of developing dMOEAs is discussed. In Sect. 3, the definition of dMO is presented. Then, five new test instances are introduced. In addition, two crucial factors of dMOEAs are discussed. In Sect. 4, the framework of the new algorithm, multi-strategy ensemble multi-objective evolutionary algorithm (MS-MOEA), is proposed. Then, the multiple strategies are presented in detail. In Sect. 5, to evaluate the effect of GDM and Gaussian mutation operator, multi-strategy ensemble static multi-objective evolutionary algorithm (MS-MOEA(s)) is compared with several effective sMOEAs on sMO problems. Moreover, multi-strategy ensemble dynamic multi-objective evolutionary algorithm (MS-MOEA(d)) is experimentally compared with several dMOEAs with traditional dynamic environment handling strategies. In Sect. 6, a brief conclusion is given and the future work is outlined.

2 Related works and necessity of developing dMOEAs

Historically, the attempt of making use of the population based evolutionary approaches for sMO problems goes back as far as 1985 [40]. The early works have been summarized in the books [15,16]. Currently, the widely used sMOEAs in engineering applications [1,25,38] include non-dominated sorting genetic algorithm II (NSGA-II) [19], strength Pareto evolutionary algorithm II [59] and Pareto develop based selection algorithm [33]. For many sMO tasks, these approaches have shown excellent performances. Due to the arising of new challenges, several effective sMOEAs with improved strategies have been proposed in recent years, such as ϵ-MOEA [20], hypervolume based MOEA [22], regularity model-based multi-objective estimation of distribution algorithm [56], MOEA based on decomposition [55] and fast hypervolume based MOEA (FH-MOEA) [43]. In these sMOEAs, either more effective new offspring creating mechanism or elite maintenance with stronger selection pressure was adopted. Besides, another branch is to incorporate the classical mathematical optimization tools [29,34,46], such as sequential quadratic programming, into the sMOEAs to form the memetic MOEAs, whose major idea is inherited by MS-MOEA. It is apparent that simple detect-and-restart strategy can be used to make sMOEAs suitable for dMO problems, but the effect of such improvement was far from satisfactory, which is similar to the matter met in dSO [52]. This viewpoint will be confirmed in the experimental study of this paper. Therefore, further studies are required to develop dynamic environment handling strategies for dMO.

To handle dynamic multi-objective environment, a natural choice is to directly adopt the effective single dynamic environment handling strategies, because some single objective meta-heuristic algorithms with effective dynamic environment handling strategies have exhibited promising search abilities [4,5,7–12,14,21,48,51]. For example, a memory-based technique was proposed in [11] to record useful information from past generations. If the environment did not change severely, the stored information can provide a much better starting point for the new environment [30]. Multi-population (i.e., multi-swarm in particle swarm

optimization (PSO)) strategy is another promising branch to handle dynamic environment [12]. In detail, the whole population is partitioned into a number of sub-populations (i.e., swarms in PSO). Different swarms are forced to search different promising regions simultaneously. Estimation of distribution algorithms (EDAs), such as dual population-based incremental learning (DPBIL) algorithms proposed in [48,51,52], were also applied to dynamic problems, in which a probabilistic memory scheme was used to improve the adaptability in dynamic environment. Besides, the immigrants strategies, which re-initialize the worst individuals throughout the optimization process, have been proved to be beneficial to maintain the diversity of the population [49,50]. Generally speaking, the essential goal of the above dSO strategies are to find a single global optimal 'point' in the search space. However, the goal of dMO is to provide a 'set of moving optimal points', which contains certain individuals along the changing POS. The relationship of their objectives and global optimal solutions can be uniformly described as one vs. many. Therefore, the above dSO optimization strategies may fail on dMO problems due to various reasons: (1) the consideration of isolated optimal points of memory strategy and multi-swarm strategies in dSO are not so effective to maintain the general shape of POS in dMO, which was discussed in [44]; (2) based on the experience obtained by [56], an extremely expensive computational cost is necessary for building the accurate probabilistic model of POS. The univariate probability of DPBIL is not so effective as being used in dSO problems; (3) unlike in single objective optimization, many solutions in multi-objective optimization are non-dominated, thus, it is hard to determine the worst-quality solutions in multi-objective optimization.

For multi-objective optimization in uncertain environment, several algorithms have been proposed. In [24], a multi-objective noisy environment was discussed in depth, moreover, several effective techniques were proposed to cope with the noisy environment. However, the dMO problems, whose true POSs change, are out of the coverage of these techniques. Although an ALife-inspired evolutionary algorithm for variating fitness landscape search was presented in [2], the validity of this algorithm has not been assessed via sufficient experimental analysis, that is, no comparison study is carried out. Furthermore, the main drawbacks of such approach can be summarized as: (1) the convergence speed is too slow; (2) the convergence speed strongly depends on the distribution of the population in search space [2].

In summary of these reviewed literatures, we want to make and emphasize the following remarks:

- sMOEAs are lack of strategies for handling dynamic environments.
- The dSO optimization strategies are not suitable for dMO problems.
- Previously, few studies have been reported on developing dMOEAs.

3 Dynamic multi-objective optimization problem

Without loss of generality, the dMO problems considered in this paper can be stated as follows:

$$\begin{aligned} &\text{minimize } f = (f_1(x,t), f_2(x,t), \ldots, f_M(x,t)) \\ &\text{subject to } x \in X, \end{aligned} \tag{1}$$

where $X \subset R^D$ denotes the decision space with D dimensions; $x = (x_1, x_2, \ldots, x_D) \in R^D$ is the decision variable vector; $f : X \to R^M$ stands for M objective functions that map from D dimensional variable space to M dimensional objective space f. Different from sMO problems, a special case of dMO can be defined as that the parameter time t is introduced where the environmental change takes place when time t reaches some pre-defined breakpoints.

3.1 Related concepts

In dMO, the concept 'Pareto dominate' is the same as that in sMO.

Definition 1 (Pareto dominate, POS and POF) x_1 'Pareto dominate' (is better than) x_2 is true when the following two conditions hold: (1) $f_j(x_1) \leq f_j(x_2)$, $j = 1, 2, \ldots, M$, which means that all objective values of x_1 are not worse than x_2. (2) $\exists j \in 1, 2, \ldots, M$, st. $f_j(x_1) < f_j(x_2)$, which means that at least one objective of x_1 is better than x_2. Based on the 'Pareto dominate' concept, POS is defined as a set of optimal solutions in the decision space that none of the other feasible solutions dominates any one of them. POF is the mapping of POS in the objective space.

The ultimate goal of sMOEAs is to achieve a set of solutions spread uniformly along the POF.

Definition 2 We call POS(t) and POF(t) the set of Pareto-optimal solutions and their objective values at time t in decision space and objective domain, respectively [2,23].

Based on Definition 2, there are four possible ways that a dMO problem can demonstrate time varying change [23]:

- Type I: The POS changes, whereas the POF does not change.
- Type II: Both POS and POF change.
- Type III: POS does not change, whereas POF changes.
- Type IV: Both POS and POF do not change, although the problem can dynamically change.

Table 1 Static multi-objective test instances

Instance	Variables	Objective functions
MZDT1	$x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_i = [-1, 1]$	$f_2(x) = g(x)[1 - \sqrt{x_1/g(x)}]$
	$i = 2, \ldots D$	$g(x) = 1 + 9(\sum_{i=2}^{D} \lvert x_i \rvert)/(D-1)$
	$D = 30$	
MZDT2	$x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_i = [-1, 1]$	$f_2(x) = g(x)[1 - (x_1/g(x))^2]$
	$i = 2, \ldots D$	$g(x) = 1 + 9(\sum_{i=2}^{D} \lvert x_i \rvert)/(D-1)$
	$D = 30$	
MZDT3	$x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_i = [-1, 1]$	$f_2(x) = g(x)[1 - \sqrt{x_1/g(x)} - \frac{x_1}{g(x)} \sin(10\pi x_1)]$
	$i = 2, \ldots D$	$g(x) = 1 + 9(\sum_{i=2}^{D} \lvert x_i \rvert)/(D-1)$
	$D = 30$	
MZDT4	$x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_i = [-1, 1]$	$f_2(x) = g(x)[1 - \sqrt{x_1/g(x)}]$
	$i = 2, \ldots D$	$g(x) = 10D - 9 + \sum_{i=2}^{D}[\lvert x_i \rvert^2 - 10\cos(4\pi \lvert x_i \rvert)]$
	$D = 10$	

It is observed that sMOEAs may be directly adopted on type III and type IV problems, since no change takes place on POS. Especially, it is expected that sMOEAs are able to perform well for type III and type IV in the case that a good Pareto non-dominated set is found before the first change occurs. Therefore, we concentrate on the first two types of changes when conducting the new test instances.

3.2 Dynamic test instances evolved from static test problems

Previous extensions of sMO problems for dMO have been developed in [23,31]. The POFs of most of these problems can not be mathematically prescribed. In these cases, some important performance metrics, such as inverted generational distance (IGD) [60], can not be easily calculated. In this paper, we design several new test instances based on the widely used test suit ZDTs [58]. The suite of selected ZDTs are structured in the same manner and consist of various landscapes, such as convex POF of ZDT1, non-convex POF of ZDT2, disconnected POF of ZDT3, and many local Pareto optima in ZDT4. It is expected that the new test instances share the following important features:

(1) they have different characteristics in POF;
(2) their dimensionality can be scaled to any size needed;
(3) their POS can not be simply located on the bounds of the decision space;
(4) the major advantage over others is that the change of POS can be controlled via simply shifting or distorting the previous POS.

To satisfy feature 1 and feature 2, we utilize the basic component functions with various landscapes as presented above. To satisfy feature 3, the decision space of ZDTs is extended as follows:

$$\Omega = \Pi_{i=1}^{D}[a_i, b_i] \in R^D \tag{2}$$

where $a_1 = 0$, $b_1 = 1$ and $a_i = -1$, $b_i = 1$ for $i = 2, \ldots, D$. The formal definitions of modified ZDT (MZDT) test instances are shown in Table 1.

In the newly designed dMO problems, we set the static duration as a fixed number of fitness evaluations FES_c. In this case, the landscape remains stable in the steady period within FES_c fitness evaluations, which is similar to the dSO optimization generator in [37]. To highlight the importance of convergence speed, we set $FES_c = 1{,}250, 2{,}500, 5{,}000$ and $10{,}000$ for all test problems, while the size of fitness evaluations is always set to be 20,000 or more for ZDTs [19,33,59]. Since the size of fitness evaluations is very small for an individual environment, the requirement of convergence speed is very high. Another important aspect is the size of changing step. Similar to [23], the size of changing step can be controlled through tuning the metric n_T, which is used to control the changing step of the environment. A smaller n_T means larger changing step. A typical approach of generating time dependent problems is defined as follows:

$$\begin{aligned} &t = \lfloor f_c/FES_c \rfloor \\ &H_i(t) = \max\{|b_i - t/n_T|, |a_i - t/n_T|\}, \quad i = 2, \ldots, D \\ &y_1 = x_1 \\ &y_i = |x_i - t/n_T|/H_i(t), \quad i = 2, \ldots, D \end{aligned} \tag{3}$$

Table 2 Dynamic ZDT multi-objective test instances

Instance	Variables	Objective functions
DMZDT1	$x_1 = [0, 1]$	$f_1(y) = y_1$
	$x_i = [-1, 1]$	$f_2(y) = g(y)[1 - \sqrt{x_1/g(y)}]$
	$i = 2, \ldots D$	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
	$D = 30$	$t = \lfloor f_c/FES_c \rfloor$
		$y_1 = x_1,$
		$y_i = \lvert x_i - t/n_T \rvert / H(t), i = 2, \ldots, D$
		$H(t) = \max\{\lvert 1 - t/n_T/\rvert, \lvert -1 - t/n_T/\rvert\}$
DMZDT2	$x_1 = [0, 1]$	$f_1(y) = y_1$
	$x_i = [-1, 1]$	$f_2(y) = g(y)[1 - (x_1/g(y))^2]$
	$i = 2, \ldots D$	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
	$D = 30$	$t = \lfloor f_c/FES_c \rfloor$
		$y_1 = x_1,$
		$y_i = \lvert x_i - t/n_T \rvert / H(t), i = 2, \ldots, D$
		$H(t) = \max\{\lvert 1 - t/n_T/\rvert, \lvert -1 - t/n_T/\rvert\}$
DMZDT3	$x_1 = [0, 1]$	$f_1(y) = y_1$
	$x_i = [-1, 1]$	$f_2(y) = g(y)[1 - \sqrt{x_1/g(y)} - \frac{x_1}{g(y)} \sin(10\pi y_1)]$
	$i = 2, \ldots D$	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
	$D = 30$	$t = \lfloor f_c/FES_c \rfloor$
		$y_1 = x_1,$
		$y_i = \lvert x_i - t/n_T \rvert / H(t), i = 2, \ldots, D$
		$H(t) = \max\{\lvert 1 - t/n_T/\rvert, \lvert -1 - t/n_T/\rvert\}$
DMZDT4	$x_1 = [0, 1]$	$f_1(y) = y_1$
	$x_i = [-1, 1]$	$f_2(y) = g(y)[1 - \sqrt{y_1/g(y)}]$
	$i = 2, \ldots D$	$g(y) = 10D - 9 + \sum_{i=2}^{D} [\lvert y_i \rvert^2 - 10\cos(4\pi \lvert y_i \rvert)]$
	$D = 10$	$t = \lfloor f_c/FES_c \rfloor$
		$y_1 = x_1,$
		$y_i = \lvert x_i - t/n_T \rvert / H(t), i = 2, \ldots, D$
		$H(t) = \max\{\lvert 1 - t/n_T/\rvert, \lvert -1 - t/n_T/\rvert\}$

where f_c stands for the current computational cost (i.e., the number of function evaluations used so far); a_i and b_i are the lower and upper bounds of the problem; $H(t)$ is the normalized metric; when time t increases one, one environmental change takes place. Under the translation of Eq. (3), the original variable vector x is mapped to the new one y. During the optimization period, the true POS keeps fixed in y space while that in x space changes with step t/n_T. Then, the type I test instance dynamic MZDT1 (DMZDT1) can be defined as follows:

$$
\begin{aligned}
f_1(y) &= y_1 \\
f_2(y) &= g(y)\left[1 - \sqrt{x_1/g(y)}\right] \\
g(y) &= 1 + 9\left(\sum_{i=2}^{D} |y_i|\right)/(D-1)
\end{aligned}
\tag{4}
$$

Other test instances DMZDT2–4 adopt the similar generated approaches. The details of DMZDTs are shown in Table 2.

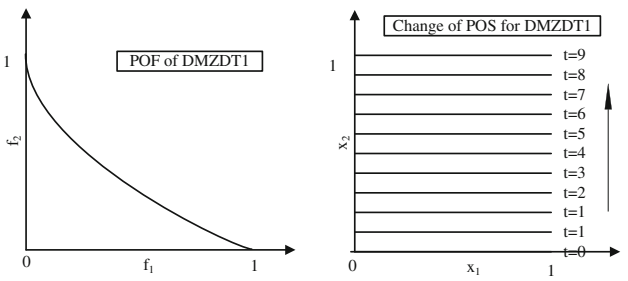

Fig. 1 POF(t) for DMZDT1 on the *left* and POS(t) for DMZDT1 on the *right*. Variation on only the first two decision variables are shown for 10 time steps

Horizontal lines in the right part of Fig. 1 represent how the first two variables of all POS(t) solutions vary in a particular time instantiation of the test problem.

It is shown in Fig. 1 that the POF of type I problems remains fixed during the optimization process. To generate

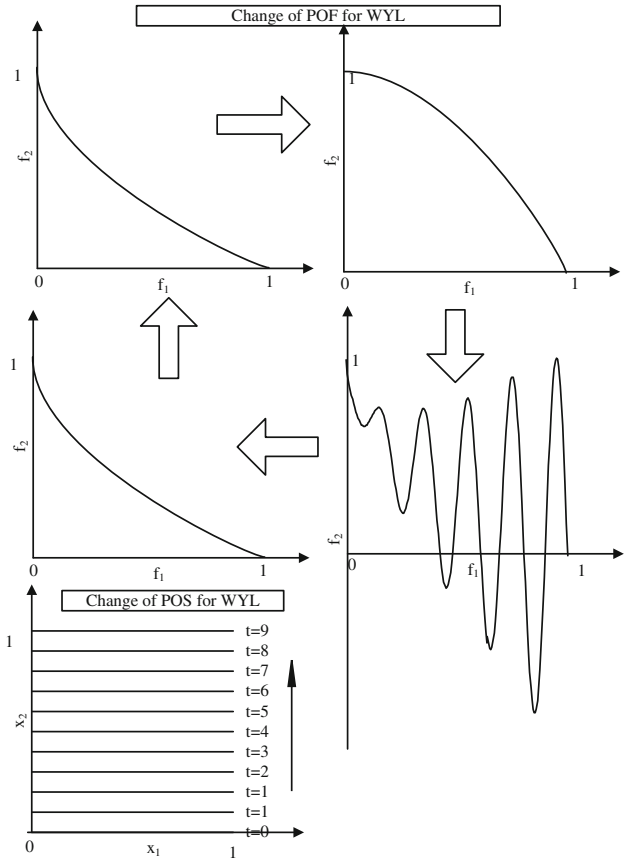

Fig. 2 POF(t) and POS(t) for WYL. Variation on only the first two decision variables are shown for 10 time steps

changes in POF, several component functions are serially adopted in one framework. The changes in terms of both POF and POS of the new problem 'WYL' are shown in Fig. 2. The detailed definition of WYL is shown in Table 3.

3.3 Discussion of memory like strategy in DMO

For sMO problems, sMOEAs aim at finding out the set of Pareto optimal solutions that form the Pareto front in objective space. The question that arises in dynamic cases is: how to track and predict the changes of POS [26]? This is especially important when the changing frequency is too high for MOEAs to search from scratch. In the work of [52], it is indicated that the memory strategy used in dSO is a good choice for reusing useful information from the past environment. The major problem in designing the memory like strategy for DMO is: how to appropriately reuse a set of non-dominated solutions.

Compared with memory strategies adopted in dSO, which only store some good solutions found in the past generations, it is much more difficult in dMO to store the information of non-dominated set that may contains infinite points. In this paper, memory like strategy is implemented in the

Table 3 Dynamic WYL multi-objective test instance

Variables	Objective functions
$x_1 = [0, 1]$	$t = \lfloor f_c / FES_c \rfloor$
$x_i = [-1, 1]$	$y_1 = x_1,$
$i = 2, \ldots D$	$y_i = \lvert x_i - t/n_T \rvert / H(t), i = 2, \ldots, D$
	$H(t) = \max\{\lvert 1 - t/n_T \rvert, \lvert -1 - t/n_T \rvert\}$
$t \ mod \ 4 == 0$	$f_1(y) = y_1$
	$f_2(y) = g(y)[1 - \sqrt{x_1/g(y)}]$
	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
$t \ mod \ 4 == 1$	$f_1(y) = y_1$
	$f_2(y) = g(y)[1 - (x_1/g(y))^2]$
	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
$t \ mod \ 4 == 2$	$f_1(y) = y_1$
	$f_2(y) = g(y)[1 - \sqrt{x_1/g(y)} - \frac{x_1}{g(y)} \sin(10\pi y_1)]$
	$g(y) = 1 + 9(\sum_{i=2}^{D} \lvert y_i \rvert)/(D-1)$
$t \ mod \ 4 == 3$	$f_1(y) = y_1$
	$f_2(y) = g(y)[1 - \sqrt{y_1/g(y)}]$
	$g(y) = 10D - 9 + \sum_{i=2}^{D}[\lvert y_i \rvert^2 - 10\cos(4\pi \lvert y_i \rvert)]$

re-initialization step. To discuss the merits and demerits of different re-initialization strategies, we take the population with current distribution in the first two variables shown in Fig. 3 for example. For the re-evaluation strategy, the locations of the population stay the same while the objective fitness values are re-evaluated. Note that such case may cause serious loss of optimization propulsion of differential information or estimation of distribution based algorithms. Moreover, the proportion of shrinking towards the local optima is high when the previous POS is near to the local optimal POS in the new environment. To cope with the demerits of re-evaluation strategy, a natural choice is to restart the whole population immediately when an environment change is detected. It performs as a serial of independent runs of EAs for separate problems. Such an approach is beneficial in the case that the landscape is full of local POS. However, in the cases that the environments only incur a small-step change, the restart strategy inevitably results in expensive cost to achieve the new POS. The restart strategy is also inadequate when the stable period of the environment is shorter than the convergence time of EAs. The memory like strategy proposed in this paper can be treated as a tradeoff between re-evaluation and restart strategies. It can be observed in Fig. 3 that a part of new offspring solutions are generated by sampling Gaussian distributions with their centers locating at the memory solutions. These offspring solutions are put on exploiting around the previous optimal solutions (memory), while the other part of new offspring solutions widely distribute with pursuit of exploration. This memory like strategy is similar to the case-based initialization proposed in [39]. The difference lies in the aspect of reusing the optimal solutions.

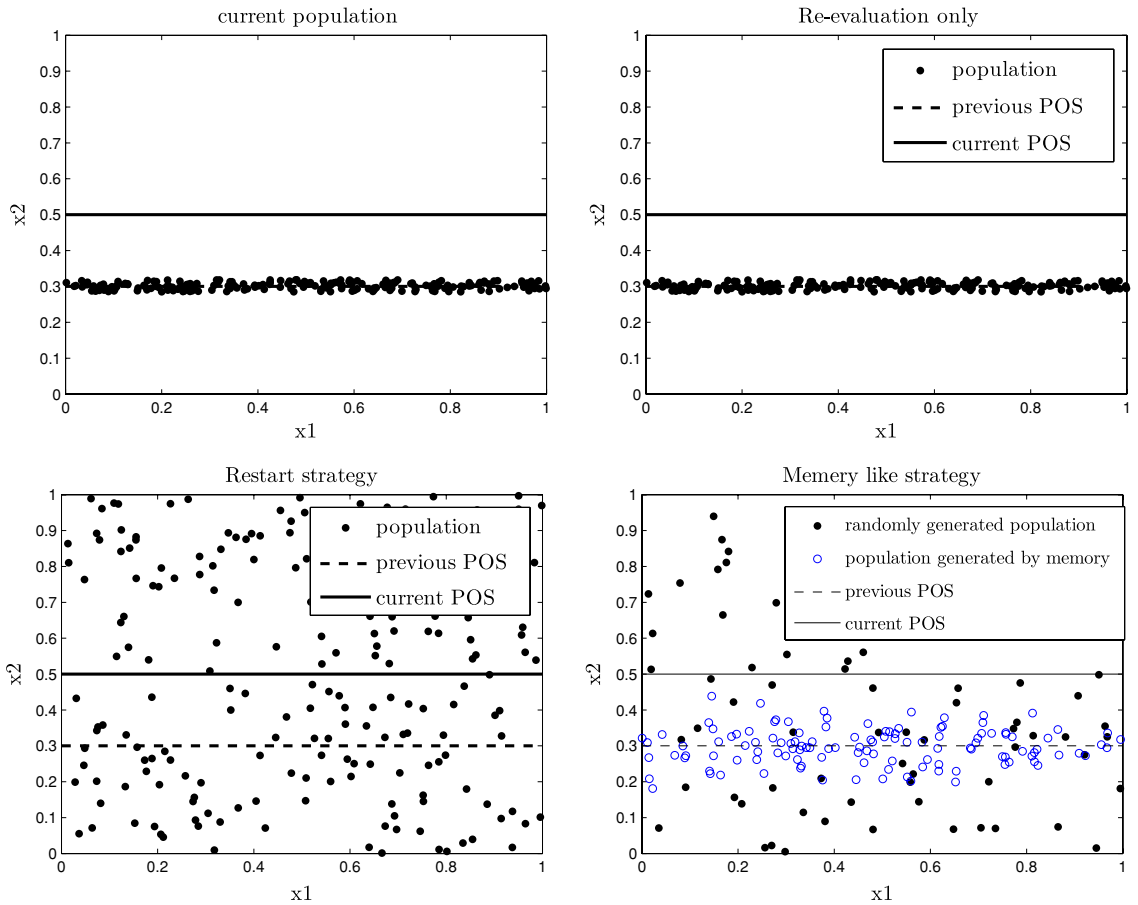

Fig. 3 Illustration of re-initialization strategies. *Up-left*: the current population; *up-right*: re-evaluation only; *down-left*: restart strategy; *down-right*: memory like strategy

In principle, the major idea of memory like strategy is to gain attraction around the memory solutions without losing the global search ability. However, overmuch exploitation around the memory optimal POS may cause shrinking towards the local optimal POS. Therefore, a new re-initialization strategy that adaptively sampling around memory is presented in the following section.

3.4 Discussion of convergence speed in dMO

Similar to dSO [30], the convergence speed is equally important in dMO. In the previous works on sMO [20,55,56], it has been shown that the new offspring creating mechanism and elite maintenance strategy can remarkably influence the convergence speed.

The most widely used new offspring creating mechanism is GA's genetic operators [1,19,38,41,59]. In recent years, some new strategies were reported to improve the convergence speed of EAs, which is subject to high success rate of finding the true POS. In [13], a jumping gene paradigm evolutionary algorithm was proposed to accelerate the convergence speed. Another branch of multi-objective optimization that also requires extremely fast convergence speed is expensive multi-objective optimization, where each solution evaluation is financially and/or temporally expensive. In [32], Knowles fully studied an effective single objective new offspring creating mechanism, then, an expensive multi-objective optimization algorithm is proposed based on this mechanism. The experimental results on real-valued, low-dimensional functions demonstrated that a significantly better performance was achieved by the new algorithm. However, this algorithm is not so effective for general use, due to the specific design for expensive problems. In some recent works of sMOEAs assessment [42], it is very interesting to note that the GA based algorithm [41] always provides top performance in the early optimization process, although the major operators, polynomial mutation and simulated binary crossover, were proposed in 1999 [17]. Taking the advantage of GA, the new algorithm framework proposed in this paper adopts a new offspring creating mechanism to enhance the convergence ability.

Elite maintenance strategy is another factor that plays a crucial role in accelerating convergence speed [43]. A large number of elite maintenance strategies have been proposed,

Table 4 Procedure of MS-MOEA

Algorithm: Multi-strategy ensemble dynamic multi-objective evolutionary algorithm (MS-MOEA)
Step 0: Randomly initialize a population $P(0)$. The non-dominated solution of $P(0)$ are copied to an archive population $A(0)$ one by one. Set the iteration counter $t = 0$. Randomly chose 5 sentry individuals and evaluate the fitness of them
Step 1: If $(t \ mod \ \frac{N_p}{2}) == 0$, evaluate the fitness values of sentry individuals. If the new values are different from the old ones, re-initialize $P(t)$ and $A(t)$ by the memory like strategy (shown in table 5). (mod is the rem operator)
Step 2: Select parent solutions from $P(t)$ and $A(t)$
Step 3: Two offspring solutions c_1 and c_2 are created by adaptive genetic and differential operators (shown in Table 6)
Step 4: If $t \ mod \ N_p == 0$, a new offspring solution is created by Gaussian mutation shown in Eq. (5), and is set to be c_1
Step 5: Update the population by c_1 and c_2 using the same strategy of [20]
Step 6: Update the archive by c_1 and c_2 using fast hypervolume strategy (shown in Table 7)
Step 7: If termination criterion is not satisfied, set $t = t + 1$ and go to step 1, else report $A(t)$

which have made significant progresses compared with the most widely adopted non-dominated sorting strategy [19]. For example, K. Deb presented a steady state ϵ-MOEA based on the ϵ-domination elite strategy [20]. In such an approach, the fitness value space is divided into a number of subspaces beforehand. Throughout the optimization process, only one solution is permitted to enter one subspace. It is experimentally shown that this strategy is beneficial to obtain fast convergence speed. Furthermore, the ϵ-domination strategy remarkably improves the diversity metric in comparison with several classical sMOEAs. However, the requirement of prior knowledge makes it difficult to apply to dMO problems. It is due to the fact that the best ϵ value has to be tuned when the environment changes. Some other researches tried to design elite maintenance with strong pressure by incorporating hyper-volume metric S, which was originally proposed in [57]. In [22], a new algorithm directly adopted S metric as the selection pressure. Although the experiments strongly assessed the good performance of this algorithm, the computational cost is still too expensive, which is $O(8k^2)$ for two objectives problems with k population size at each generation. This demerit inevitably limits its application to dMO problems. To fix up this, Y. Wang [43] proposed a fast hypervolume contribution calculation approach called 'FH', of which the computational cost of conducting hypervolume contribution decreases remarkably. In MS-MOEA, the 'FH' approach is adopted because of its excellent performance reported.

4 Multi-strategy ensemble dynamic multi-objective evolutionary algorithm

In this section, we elaborate the general framework of MS-MOEA. In addition, multiple strategies, such as adaptive genetic and differential operators (GDM), Gaussian mutation, memory like re-initialization strategy and FH, are presented in detail.

4.1 Framework of MS-MOEA

In general, MS-MOEA adopts several strategies for different purposes. To emphasize this, the major focuses of all strategies are briefly introduced as follows:

- GDM: provide fast convergence speed in a short time after an environment change is detected and provide higher-quality non-dominated set when the non-dominated solutions get close to the true POS.
- Gaussian mutation: strengthen the ability of escaping from the local optima.
- Memory like strategy: gain attraction around the memory solutions without losing the global search ability.
- FH: provide fast convergence speed and maintain well diversity.

At each generation, the proposed algorithm maintains: a population of N_P solutions (i.e points in decision space) with their objective values and an external archive to record the non-dominated solutions found previously. For the memory like strategy, the external archive can be utilized as a memory storage. MS-MOEA works as shown in Table 4. Moreover, Fig. 4 depicts a flowchart of MS-MOEA.

Step 1 is designed to handle the dynamic environment. Several randomly chosen "sentry" individuals are re-evaluated to detect the environment change. In this paper, the rules of setting the number and locations of the sentries are inherited from [21], that is, the locations are randomly initialized in the initialization step and the number is set to be 5. At each generation, only two offspring solutions are generated under steps 2–4, which are denoted as selection, creation and Gaussian mutation respectively. For selection step, different

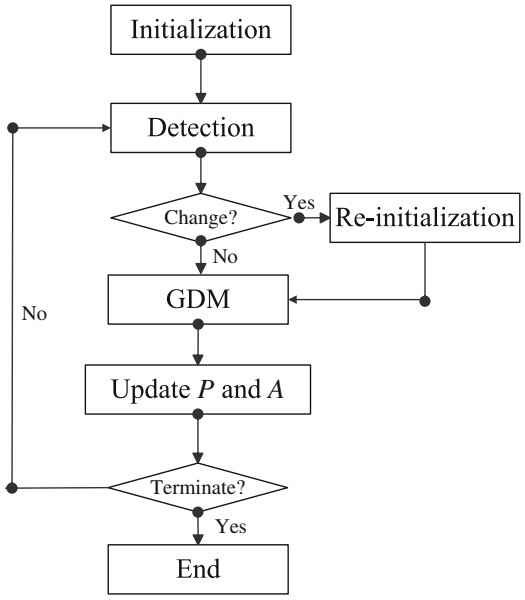

Fig. 4 Flowchart of MS-MOEA

numbers of parent solutions are needed for different new offspring creating strategies. After a fixed number of solutions are generated, the Gaussian mutation is launched. In step 4, the Gaussian mutation operator is defined as follows:

$$\begin{aligned} &a = (a_1, a_2, \ldots, a_D) \\ &c = (c_1, c_2, \ldots, c_D) \\ &\text{for} \quad i = 1, 2, \ldots, D \\ &c_i = \begin{cases} a_i + N(0, \sigma), & \text{if } rand < \frac{2}{D} \\ a_i, & \text{otherwise,} \end{cases} \end{aligned} \tag{5}$$

where c is the new offspring solution generated by the randomly selected solution a from the archive via the Gaussian mutation; D is the dimensional size of search space; $rand$ is a uniformly generated number within [0, 1]; σ is a positive constant. The Gaussian mutation strategy defined as Eq. (5) has been testified by many researches to be a good strategy to enhance the ability of escaping the local optima [53]. To inherit information from the archive solution a, the Gaussian mutation is adopted with probability $\frac{2}{D}$. By performing such Gaussian mutation operator, some randomly selected archive solutions can search for superior solution around its current position. Therefore, it might be beneficial to find the superior optimum in multimodal environments. In step 6, the archive is updated by the offspring c_1 and c_2 one by one, which is based on the strategy FH. The detail of re-initialization strategy, GDM and FH will be introduced following.

4.2 Re-initialization strategy

As is described in Sect. 3.3, a memory like re-initialization strategy is adopted for the new environment. Generally, the new generated population are formed by two parts, which are called part I and part II here. The part I solutions are randomly generated within the bounds of the search space. The part II solutions are generated by the Gaussian local search operator defined as follows:

$$\begin{aligned} &a = (a_1, a_2, \ldots, a_D) \\ &c = (c_1, c_2, \ldots, c_D) \\ &\text{for} \quad i = 1, 2, \ldots, D \\ &c_i = a_i + N(0, \sigma) \end{aligned} \tag{6}$$

Different from the Gaussian mutation operator presented above in Eq. 5, all of the variables are generated by sampling Gaussian distribution with the mean vector locating at the randomly selected achieve solution a.

The procedure of re-initialization of MS-MOEA is shown in Table 5. In line 2, the proportion of local search is controlled by a probability p_l. Therefore, each archive solution has a chance to provide a neighborhood solution in the new population. The ideal situation is that the distribution of individuals generated by local search inherits general manifold of the old POS.

Table 5 Procedure of re-initialization of MS-MOEA

Re-initialization strategy of MS-MOEA
0 Empty P.
1 for $i = 1$ to N_P
2 if $rand > p_l$
3 Randomly initialize the ith solution of the population.
4 else
5 Randomly select a parent solution a from A.
6 Create the ith solution using Eq. (6).
7 end if
8 end for
9 Empty A. Update A by P.

p_l Is set to be 0.2 in MS-MOEA

4.3 New offspring creating mechanism powered by adaptive genetic and differential operators

It has been discussed in Sect. 3.4 that new offspring creating mechanism plays an important role in accelerating the convergence speed in dMO. Suganthan [42] introduced that the GA based sMOEAs [36,41] always emerged top convergence speed within a low computational cost, while differential evolution (DE) based sMOEAs [28,35] can achieve well-distributed Pareto front after a long optimization process. In the new strategy, GA and DE are adopted serially with the purpose of benefitting from both GA and DE. The pseudocode of GDM is expressed in Table 6.

Table 6 Procedure of new offspring creating mechanism of MS-MOEA

New offspring creating mechanism of MS-MOEA
0 if $fullcount < givencount$
1 Create the offspring c_1 and c_2 by SBX and polynomial
2 mutation operators [17,18].
3 else
4 Randomly select 8 solutions $p_1, p_2, \ldots p_8$ from P;
5 Randomly select 1 solution a from A.
6 $c_1 = p_1 + N(0.3, 0.1) \cdot (p_1 - a) + N(0.3, 0.1) \cdot (p_2 - p_3)$,
7 $c_2 = p_4 + N(0.3, 0.1) \cdot (p_5 - p_6)$.
8 Crossover c_1, c_2 with p_7 p_8 by multi-points crossover using $p_c = N(0.3, 0.5)$.
9 end
10 Output c_1 and c_2.

$fullcount$ Stands for the generations that the size of archive is full. $givencount$ is set to be 30 in MS-MOEA. $N(0.3, 0.1)$ Stands for Gaussian random number with mean 0.3 and deviation 0.1. p_c Is set to be 0.3 in MS-MOEA

In the first stage of GDM, GA operators, including polynomial mutation and simulated binary crossover [17], are adopted. In the second stage, DE operators *DE/rand/1* and *DE/current to best/1* are adopted, see lines 6–7 of Table 6. The DE operators are shown in Table 6 lines 4–8, where $N(0.3, 0.1)$ is a normal random number with mean 0.3 and deviation 0.1. The parameters setting of DE is inherited from [28].

The crucial problem remained is how to determine when DE operator is launched. In GDM, the trigger is defined as the number of generations when the size of archive is full ($fullcount$ exceeds a given fixed size $givencount$). It can be described as: GA provides fast convergence speed in the early stage; when a set of solutions with good convergence property have been achieved, DE is implemented to improve diversity of the non-dominated solutions. Note that only GA is used when the trigger condition is not met till the end of operation. That is, only GA operators are implemented for some extremely complex tasks. A similar trigger condition has been discussed in detailed in [45].

4.4 Fast hypervolume contribution calculation approach

Benefitting from the excellent property discussed in [43], hyper-volume mechanism is expected to possess superior performance in elite maintenance. However, the computational cost is quite expensive to conduct global hyper-volume contribution model for the whole population [22,60]. In the previous work of [47], a faster algorithm for calculating global hyper-volume is proposed to reduce the computational cost to approximate polynomial time. In [43], 'FH' makes use of an external archive and sliced technology, which is similar to [47].

In FH, the contributions of each solution does not have to be calculated at each generation. Only the contributions of a small sub-set of population are tuned when a new solution is added. It is also apparent that FH is designed to update the archive with one added solution at a time. Compared with the approach in [22], only the contributions of a small sub-set are tuned, which reduces the computational cost remarkably.

5 Experimental study

In MS-MOEA, there are several incorporated strategies, of which FH, GDM, Gaussian mutation are not designed specially for dMO problems. In order to provide experimental evidence to study how these incorporated strategies improve the performance of traditional sMOEA separately, MS-MOEA is compared with several sMOEAs without one or more of these incorporated strategies on a suite of sMO problems. After that, a suite of classical dMO problems FDAs and a suite of newly designed dMO problems with four different $FESc$s are used to assess the advantages of this multi-strategy approach. The algorithms under comparison and their purposes are emphasized as follows:

- In order to show the evidence how GDM improves the classical GA and DE based algorithms, we compared MS-MOEA with an improved NSGA-II (INSGA-II), MOEA with FH strategy only (FH-MOEA) [43].
- To show the advantage of memory like strategy, the algorithm without such strategy (FH-MOEADE) is used.

In this experimental study, the algorithm versions for dMO are named: INSGA-II(d), FH-MOEA(d), FH-MOEADE(d) and MS-MOEA(d); the algorithm versions for sMO are named: INSGA-II(s), FH-MOEA(s), FH-MOEADE(s) and MS-MOEA(s). In this study, we will briefly introduce the other algorithms in comparison and present the performance indicators. Then, the empirical study is shown in terms of statistical metrics, evolution process figures and POF figures.

5.1 Algorithms in comparison

The original NSGA-II algorithm is proposed in [19]. It implements a fast non-dominated sorting approach to effectively conduct the selection strategy. An improved non-dominated selection has been proposed in [56], which removes dense solutions one by one and recalculates the crowding distances before deciding which solution should be deleted. It can improve the diversity property compared with the selection operator of NSGA-II. The only difference between INSGA-II and NSGA-II is the implementation of the improved

Table 7 Procedure of update strategy

Framework of fast hypervolume contribution approach
Step 0: Get A from the main procedure
Step 1: When a new solution x is added. If A is empty, add x into A directly, and set the contribution of x to ∞, otherwise go to step 2
Step 2: Compare x with the value from the slice 1 a_1 to the top slice and find out the first solution a_i, which is larger in objective 1 than x. If x is dominated by x, remove x and go to step 6, else if a_i is dominated by x, go to step 3, otherwise go to step 5
Step 3: Remove solution a_i and add x into the ith position of A. Then, tune the contribution of a_{i-1}, a_i and a_{i+1}, set $i = i + 1$, go to step 4
Step 4: Pareto dominated comparison between a_i and x. If a_i is dominated by x, go to step 3, otherwise go to step 6
Step 5: Insert x into the ith position and tune the contributions a_{i-1} and a_i. Remove the archive solutions ($> i$) that are dominated by x until the first non-dominated solution or the last solution
Step 6: Terminate and return the new archive to the main procedure

Table 8 Procedure of FH-MOEA

Algorithm: Multi-objective evolutionary algorithm based-on fast hyper-volume contribution approach (FH-MOEA)
Step 0: Randomly initialize a population $P(0)$. The non-dominated solution of $P(0)$ are copied to an archive population $A(0)$ one by one. Set the iteration counter $t = 0$
Step 1: One parent solution x is chosen from the population $P(t)$ using the league selection
Step 2: Another parent solution a is chosen from the archive population $A(t)$ randomly
Step 3: Two offspring solutions c_1 and c_2 are created by genetic crossover and mutation
Step 4: Update the population by c_1 and c_2 using the same strategy as [20]
Step 5: Update the archive by c_1 and c_2 using the strategy in Table 7
Step 6: If termination criterion is not satisfied, set $t = t + 1$ and go to step 1, else report $A(t)$

non-dominated selection and an external archive. The parameters of GA is set to be: mutation rate $p_m = 0.1$, crossover rate $p_c = 0.9$.

FH-MOEA is proposed in [43]. The major difference compared with ϵ-MOEA [20] lies in the archive updating strategy. The details of FH are shown in Table 7 and the procedure of FH-MOEA is presented in Table 8. In these experiments, the parameters for GA are the same as INSGA-II. Another FH induced algorithm MS-MOEADE is similar to MS-MOEA but without the memory like re-initialization strategy. The population size N_P is set to be 50 for all algorithms. The size of sentry individuals is 5. At each iteration, the sentry individuals are re-evaluated.

5.2 Performance indicators

As is presented in Sect. 3.2, the inverted generational distance (IGD) metric is hard to calculated for some classical test instances [56]. In those cases (FDA2 and FDA3), hypervolume (H) [60,57] are used for assessing the performance of the algorithms. These two metrics IGD_i and H_i are calculated right before $i + 1$th change happens. The mean IGD and hypervolume metrics are defined as:

$$\overline{IGD} = \frac{\sum_{i=1}^{num\_of\_change} IGD_i}{num\_of\_change} \tag{7}$$

$$\overline{H} = \frac{\sum_{i=1}^{num\_of\_change} H_i}{num\_of\_change} \tag{8}$$

where num_of_change stands for the numbers of changes in a run.

Let P^* be a set of uniformly distributed points along the POF (in the objective space). Let A be an approximate set to the POF, the average distance from P^* to A is defined as:

$$IGD(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|} \tag{9}$$

where $d(v, A)$ is the minimum Euclidean distance between v and the points in A. "If $|P^*|$ is large enough to represent the POF very well, $IGD(A, P^*)$ could measure both the diversity and convergence of A in a sense. To have a low value of $IGD(A, P^*)$, the set A must be very close to the POF and cannot miss any part of the whole POF" [56].

The hypervolume H is another metric that can simultaneously compare the algorithms in both convergence and diversity. It is defined as the space dominated by the set of non-dominated solutions in relation to the reference point.

5.3 Experiment on static multi-objective optimization

The main purpose of this study is to evaluate the effect of proposed GDM and Gaussian mutation. The test functions MZDTs are shown in Table 1. In MZDTs, the POSs of ZDTs have been shifted to non-bound position. We set the total size of fitness evaluations $FES = 15{,}000$ for MZDT1–3, $FES = 20{,}000$ for MZDT4. The statistical experimental metrics IGD and H of 50 independent runs are summarized in Table 9.

Table 9 Comparison of *IGD* metric for static multi-objective optimization

	MZDT1	MZDT2	MZDT3	MZDT4
MS-MOEA(s)	**1.00E-02 (9.34E-04)**	**1.25E-02 (2.22E-03)**	**7.92E-03 (1.44E-03)**	**2.38E-02 (4.02E-02)**
INSGA-II(s)	2.21E-02 (2.63E-03)	3.52E-02 (6.12E-03)	1.65E-02 (4.83E-03)	2.38E-02 (4.08E-02)
FH-MOEA(s)	1.39E-02 (1.14E-02)	1.40E-02 (2.83E-03)	8.27E-03 (2.05E-03)	7.20E-02 (1.04E-01)

Table 10 Comparison on FDAs

	FDA1 ($\overline{IGD}$)	FDA2 (25-$\overline{H}$)	FDA3 (25-$\overline{H}$)
MS-MOEA(d)	**7.63E-03 (3.09E-05)**	**7.80E-03(9.41E-05)**	3.55E-01(2.03E-01)
INSGA-II(d)	1.53E-02 (3.70E-03)	1.10E-02 (1.11E-03)	**2.55E-01 (1.82E-01)**
FH-MOEA(d)	2.40E-02 (1.15E-02)	1.58E-02 (2.46E-03)	5.54E+00 (1.39E+00)
FH-MOEADE(d)	7.72E-03 (7.04E-05)	8.10E-03 (9.94E-05)	6.96E-01 (4.70E-01)

As is easy to be observed in Table 9, the proposed algorithm MS-MOEA(s) provides remarkably better performances on MZDT1, MZDT2 and MZDT3 than INSGA-II(s), but only slightly outperforms FH-MOEA(s). After observing the difference among these three algorithms, MS-MOEA(s) and FH-MOEA(s) are more effective for these problems due to the implementation of FH. For the instance with many local Pareto optima MZDT4, MS-MOEA(s) and INSGA-II(s) provide comparable results, while FH-MOEA(s), which is beneficial for the former three problems with relatively smooth landscape, deteriorates sharply. It is interesting to see that MS-MOEA(s) always provides the top performances for all problems. FH-MOEA(s), without Gaussian mutation, fail to obtain satisfactory performance on MZDT4. INSGA-II(s), without GDM, perform badly on MZDT1–3. All these results tell us that the cooperation of GDM and FH works quite well in MS-MOEA(s) and is suitable for the problems with different landscapes, while the missing of either strategy may cause the deterioration or failure on some problems. This is in virtue of the fact that GDM strategy strengthens convergence ability compared with DE based sMOEA, and diversity ability compared with GA based sMOEA and furthermore, the Gaussian mutation improves the performance on complex problem.

Table 11 Dynamic FDA multi-objective test instances

Instance	Variables	Objective functions
FDA1	$x_I = x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_{II} = x_i$	$f_2(x) = g(x)h(x)$
	$x_i = [-1, 1]$	$g(x) = 1 + \sum_{x_i \in x_{II}} (x_i - G(t))^2)$
	$i = 2, \ldots D$	$h(x) = 1 - \sqrt{f_1/g}$
		$G(t) = \sin(0.5\pi t), t = \lfloor f_c/FES_c \rfloor / n_T$
FDA2	$x_I = x_1 = [0, 1]$	$f_1(x) = x_1$
	$x_{II} = x_i$	$f_2(x) = g(x)h(x)$
	$i = 2, \ldots 16$	$g(x_{II}) = 1 + \sum_{x_i \in x_{II}} x_i^2$
	$x_{III} = x_i$	$h(x) = 1 - (f_1/g)^P$
	$i = 17, \ldots 31$	$P = [H(t) + \sum_{x_i \in x_{III}} (x_i - H(t))^2]^{-1}$
	$x_{II,III} = [-1, 1]$	$H(t) = 0.75 + 0.7 \sin(0.5\pi t)$
		$t = \lfloor f_c/FES_c \rfloor / n_T$
FDA3	$x_I = x_1 = [0, 1]$	$f_1(x) = x_1^{F(t)}$
	$x_{II} = x_i$	$f_2(x) = g(x)h(x)$
	$x_i = [-1, 1]$	$g(x) = 1 + \sum_{x_i \in x_{II}} (x_i - G(t))^2)$
	$i = 2, \ldots D$	$h(x) = 1 - \sqrt{f_1/g}$
		$G(t) = \lvert \sin(0.5\pi t) \rvert, t = \lfloor f_c/FES_c \rfloor / n_T$
		$F(t) = 10^{2sin(0.5\pi t)}$

In these test instances, the $FES_c = 5{,}000$ stands for the static duration; $n_T = 10$

5.4 Experiment on dynamic FDAs

The test instances FDAs are proposed in [23]. So far, FDAs are widely used in almost all of the dMO papers [2,44]. Three test instances with different characteristics and different dimensional sizes are selected: FDA1 belongs to type I; FDA2 and FDA3 belong to type II. The definitions of FDAs are summarized in Table 11. More details are available in [23]. The statistical experimental metrics $\overline{IGD}$ for FDA1 and $(25 - \overline{H})$ for FDA2 and FDA3 of 50 independent runs are summarized in Table 10. For hypervolume metric, the reference point [5, 5] is used, so the space that is not dominated by the population can be calculated as $(25 - \overline{H})$. We set fitness evaluation size $FES = 50{,}000$, set static duration $FESc = 5{,}000$ for all test instances.

In Table 10, MS-MOEA(d) shows a distinct advantage over the other approaches for FDA1. As to Fig. 5, the memory like re-initialization strategy tends to carry out the search in a more reasonable region when the change is detected. Also, MS-MOEA(d) slightly outperforms FH-MOEADE(d). For FDA2, the performance of the proposed algorithm indicates that it is steady and robust for such problems with both changing POS and POF. For FDA3, INSGA-II(d) demonstrates top

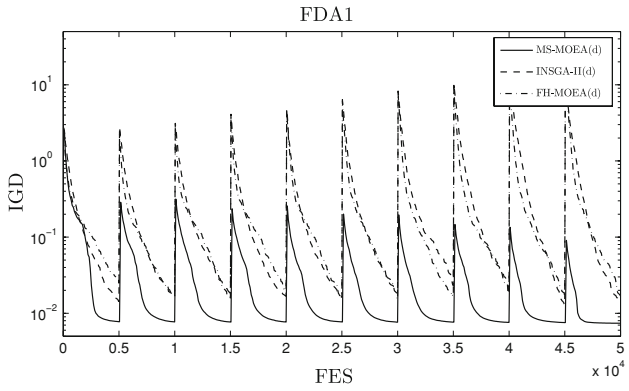

Fig. 5 Comparison of IGD metric curves for FDA1

performance. It could be attributed to the fact that the DE of GDM is launched too early for this problem. Obviously, such situation may delay the search and emerge as poor distributed and converged non-dominated solutions. It is still interesting to see that MS-MOEA(d) shows significantly better performance than FH-MOEADE(d), see Table 13. Compared with MS-MOEA(d), FH-MOEADE(d) lacks Gaussian mutation and memory like strategy. Therefore, the effectiveness of these two strategies is validated. We can conclude from the above results that the results of MS-MOEA(d) for the classical dynamic test instances are encouraging and promising.

5.5 Experiment on dynamic DMZDTs and WYL

As defined in Table 2, DMZDT suite has 4 test instances, and belongs to type I. The basic static functions MZDTs of DMZDTs have been fully studied in many works [19,20,55]. The POFs of DMZDTs can be mathematically analyzed and easily generated. The MZDT test cases are expanded to build a test instance of type II, named WYL. The formal definition of WYL is shown in Table 3. We set static duration $FESc = 1{,}250$, $2{,}500$, $5{,}000$ and $10{,}000$ for all test instances, set the size of environments 100. The total size of function evaluations is $FES = 100 \times FESc$.

The statistical experimental metrics $\overline{IGD}$ of 50 independent runs are summarized in Table 12. In order to validate the

Table 12 The t-test results of comparing MS-DMOEA with the other algorithms

	$FESc = 1{,}250$	$FESc = 2{,}500$	$FESc = 5{,}000$	$FESc = 10{,}000$
DMZDT1				
MS-MOEA(d)	**2.16E-01 (1.04E-02)**	**7.46E-02 (5.70E-03)**	**1.96E-02 (2.53E-04)**	**9.78E-03 (4.48E-05)**
IDNSGA-II(d)	5.69E-01 (1.02E-02)	2.43E-01 (3.59E-03)	8.25E-02 (1.70E-03)	2.80E-02 (4.92E-04)
FH-MOEA(d)	4.23E-01 (1.02E-02)	1.79E-01 (4.50E-03)	6.75E-02 (3.82E-03)	1.86E-02 (7.17E-04)
FH-MOEADE(d)	4.37E-01 (4.30E-03)	1.71E-01 (4.30E-03)	5.12E-02 (3.70E-03)	1.56E-02 (7.00E-04)
DMZDT2				
MS-MOEA(d)	**3.19E-01 (3.76E-02)**	**1.31E-01 (1.22E-02)**	**3.26E-02 (4.98E-04)**	**1.21E-02 (5.91E-05)**
IDNSGA-II(d)	1.11E+00 (2.26E-02)	4.93E-01 (1.53E-02)	1.54E-01 (8.68E-03)	4.22E-02 (4.84E-04)
FH-MOEA(d)	6.97E-01 (2.26E-02)	3.71E-01 (1.28E-02)	1.17E-01 (6.67E-03)	2.32E-02 (1.29E-03)
FH-MOEADE(d)	7.09E-01 (1.07E-02)	3.09E-01 (7.50E-03)	6.47E-02 (7.20E-03)	1.84E-02 (1.40E-03)
DMZDT3				
MS-MOEA(d)	**2.22E-01 (1.87E-02)**	**8.50E-02 (9.58E-03)**	**1.70E-02 (6.86E-04)**	**1.01E-02 (3.36E-05)**
IDNSGA-II(d)	4.33E-01 (8.42E-03)	1.86E-01 (1.93E-03)	6.96E-02 (1.69E-03)	2.39E-02 (7.69E-04)
FH-MOEA(d)	3.28E-01 (8.42E-03)	1.24E-01 (4.95E-03)	5.01E-02 (2.53E-03)	2.29E-02 (1.57E-03)
FH-MOEADE(d)	3.22E-01 (5.00E-03)	1.26E-01 (2.50E-03)	3.65E-02 (2.40E-03)	1.82E-02 (1.60E-03)
DMZDT4				
MS-MOEA(d)	**1.21E+00 (1.12E-01)**	**5.79E-01 (6.21E-02)**	**2.95E-01 (2.49E-02)**	2.93E-01 (3.10E-02)
IDNSGA-II(d)	2.69E+00 (9.87E-02)	1.30E+00 (5.72E-02)	6.22E-01 (4.38E-02)	**1.92E-01 (1.94E-02)**
FH-MOEA(d)	2.34E+00 (9.87E-02)	1.39E+00 (7.45E-02)	7.16E-01 (2.94E-02)	3.09E-01 (1.32E-02)
FH-MOEADE(d)	2.51E+00 (7.66E-02)	1.43E+00 (3.04E-02)	7.84E-01 (3.58E-02)	3.92E-01 (1.31E-02)
WYL				
MS-MOEA(d)	**3.34E-01 (2.91E-02)**	**1.52E-01 (1.39E-02)**	**7.22E-02 (2.01E-02)**	**5.32E-02 (3.36E-03)**
IDNSGA-II(d)	8.49E-01 (5.76E-02)	3.88E-01 (3.36E-02)	1.63E-01 (1.79E-02)	6.03E-02 (5.13E-03)
FH-MOEA(d)	7.66E-01 (4.58E-002)	2.75E-01 (2.75E-02)	2.73E-01 (1.91E-02)	1.46E-01 (3.70E-03)
FH-MOEADE(d)	7.61E-01 (2.56E-02)	2.75E-01 (1.94E-02)	2.13E-01 (1.88E-02)	7.03E-02 (3.00E-03)

Table 13 The t-test results of comparing MS-DMOEA with the other algorithms

t-test	FDAs			DMZDT1				DMZDT2			
FESc(DMZDTs and WYL)	FDA1	FDA2	FDA3	1,250	2,500	5,000	10,000	1,250	2,500	5,000	10,000
MS-MOEA(d) vs INSGA-II(d)	$s+$	$s+$	$-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
MS-MOEA(d) vs FH-MOEA(d)	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
MS-MOEA(d) vs FH-MOEADE(d)	$+$	$+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$

t-test	DMZDT3				DMZDT4				WYL			
FESc(DMZDTs and WYL)	1,250	2,500	5,000	10,000	1,250	2,500	5,000	10,000	1,250	2,500	5,000	10,000
MS-MOEA(d) vs INSGA-II(d)	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$-$	$s+$	$s+$	$s+$	$s+$
MS-MOEA(d) vs FH-MOEA(d)	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
MS-MOEA(d) vs FH-MOEADE(d)	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$

effectiveness of the new algorithm, the t-test, which is based on one-tailed t-test with 98 degrees of freedom at a 0.05 level of significance, is provided to statistically show the comparison between MS-MOEA(d) and the other algorithms. In Table 13, the t-test results of dMO problems regarding $algorithm1$ vs. $algorithm2$ are shown as '+', '−', 's+' and 's−' when $algorithm1$ is insignificantly better than, insignificantly worse than, significantly better than, and significantly worse than $algorithm2$ respectively. Figs. 12–16 depict the distributions of non-dominated sets of median run before environment changes. The mean evolution processes of last ten environments are shown in Figs. 6–10. Figure 11 records the experimental results on DMZDTs and WYL of different $FESc$s.

5.5.1 *Test problems with relatively smooth fitness landscape (DMZDT1–3)*

The fitness landscapes of sMO problems MZDT1–3 are relatively smooth. Therefore, the dMO problems extended from them, including DMZDT1–3, have the same fitness landscape features. However, these problems have different characteristics of POF, such as convex, non-convex and disconnected. It is clear from Table 12 that MS-MOEA(d) provides the top performances on all of these problems with different $FESc$ settings. Furthermore, the t-test in Table 13 validates that the advantages of MS-MOEA(d) are significant.

In the sMO experiments, MS-MOEA(s) only slightly outperforms FH-MOEADE(s). However, it is evident from Figs. 6–8 that the memory like strategy provides a better starting point when an environmental change takes place, and thus, makes a remarkable positive effect to MS-MOEA(d). When examining the effect of memory like strategy on the problems with short static duration ($FESc = 1{,}250$), it is observed that the quality of re-initialized population generated by MS-MOEA(d), which is the only algorithm with such strategy, remarkable outperforms the other algorithms. That is, the purpose of designing such a memory like strategy is met. Furthermore, as is shown in Figs. 6–8, it is interesting to see that as the search continues, the quality of the non-dominated set found for each environment tends to raise, while no remarkable difference can be found for the restart strategy.

As the static duration increases, the performances of all algorithms get better, see Table 12 and Fig. 11. The comparisons on the problems with $FECc = 2{,}500$, 5,000 and 10,000 are similar to that on $FESc = 1{,}250$. In Figs. 12–14, we compare the non-dominated solutions obtained of median run for all environments. It is convinced that the non-dominated sets obtained by MS-MOEA(d) are much nearer to the true POF and with much better diversity characteristic. Especially, when $FECc = 5{,}000$ and 10,000, MS-MOEA(d) could locate the population along POF with well distribution for all environments while solutions obtained by INSGA-II(d) can not fully converge to POF and distribute widely in the objective space.

5.5.2 *Test problem with rugged fitness landscape (DMZDT4)*

Since the basic function MZDT4 has many local POFs in each separate environment, DMZDT4 has rugged fitness landscape. Compared with DMZDT1–3, DMZDT4 is much harder. This viewpoint is also verified in Table 12, that is, the IGD metrics of all algorithms on DMZDT4 are much worse than those on DMZDT1–3.

For this hard dMO task, the performances of all algorithms inevitably deteriorate badly. MS-MOEA(d) provides the best performances and significantly outperforms the other algorithms when $FECc = 1{,}250$, 2,500 and 5,000, see Table 11.

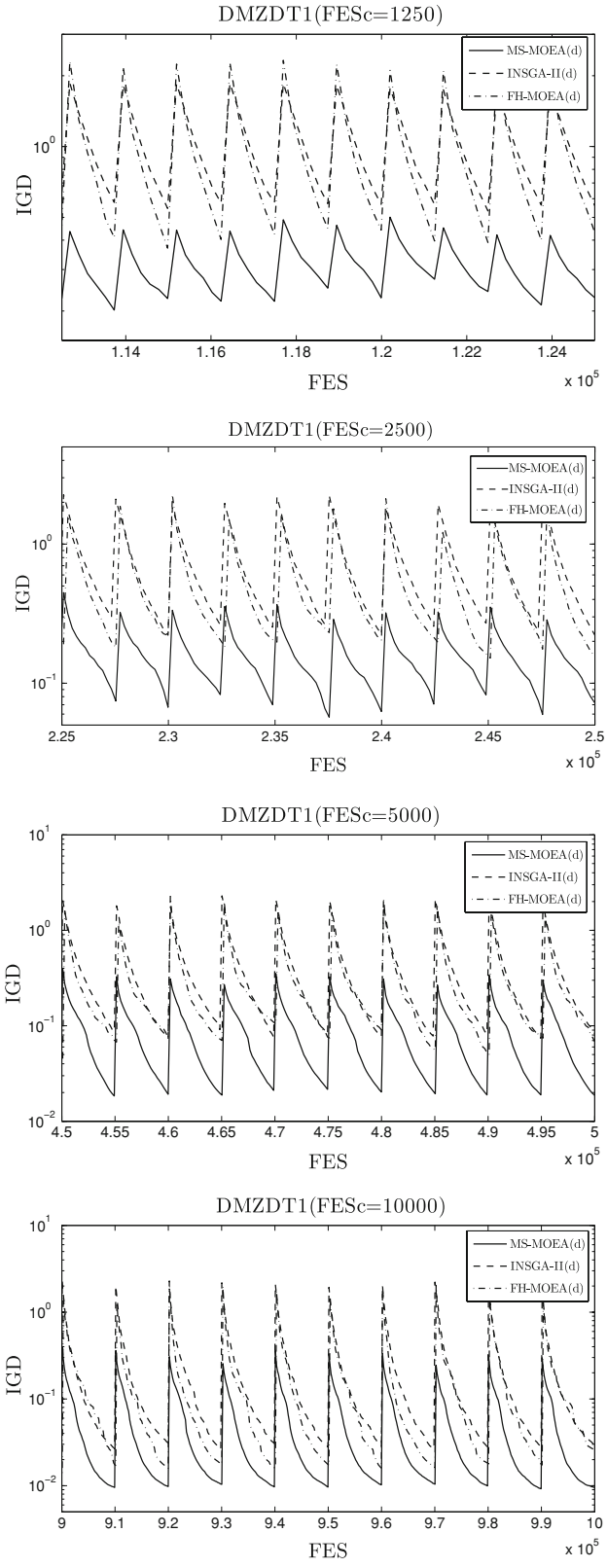

Fig. 6 Comparison of IGD metric curves for DMZDT1

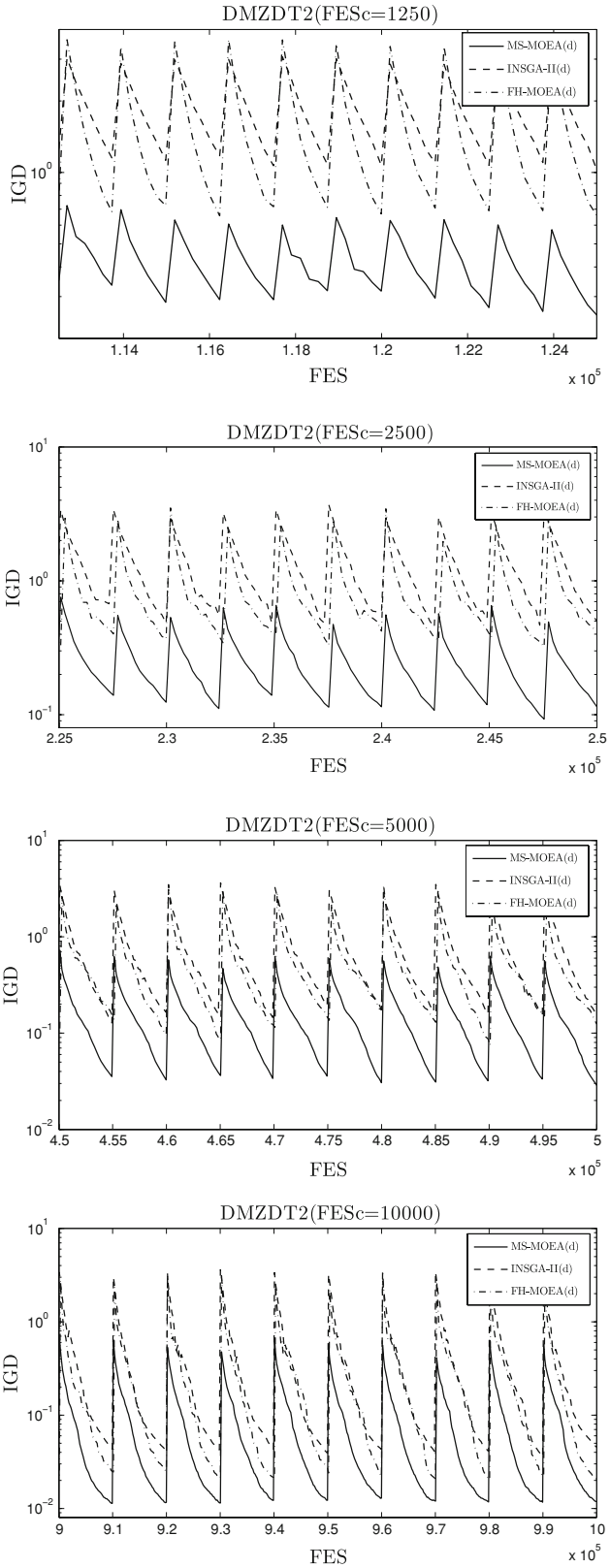

Fig. 7 Comparison of IGD metric curves for DMZDT2

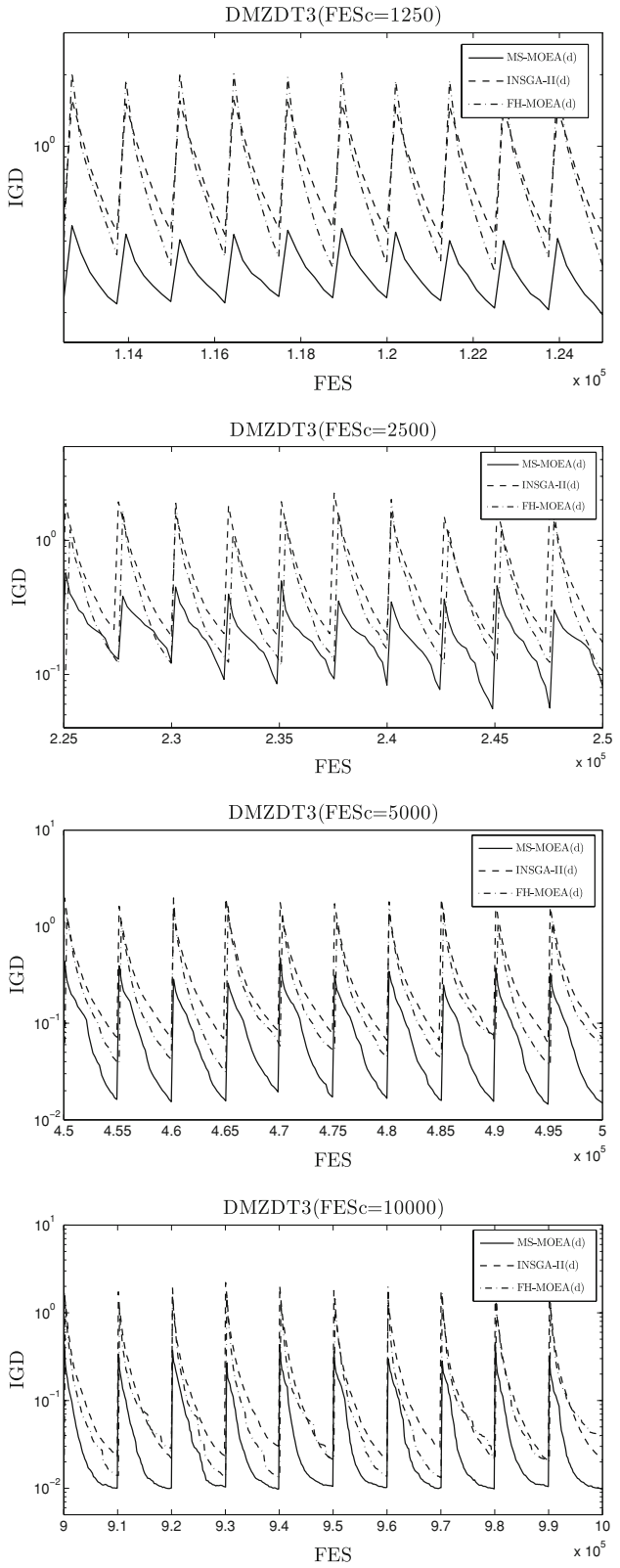

Fig. 8 Comparison of IGD metric curves for DMZDT3

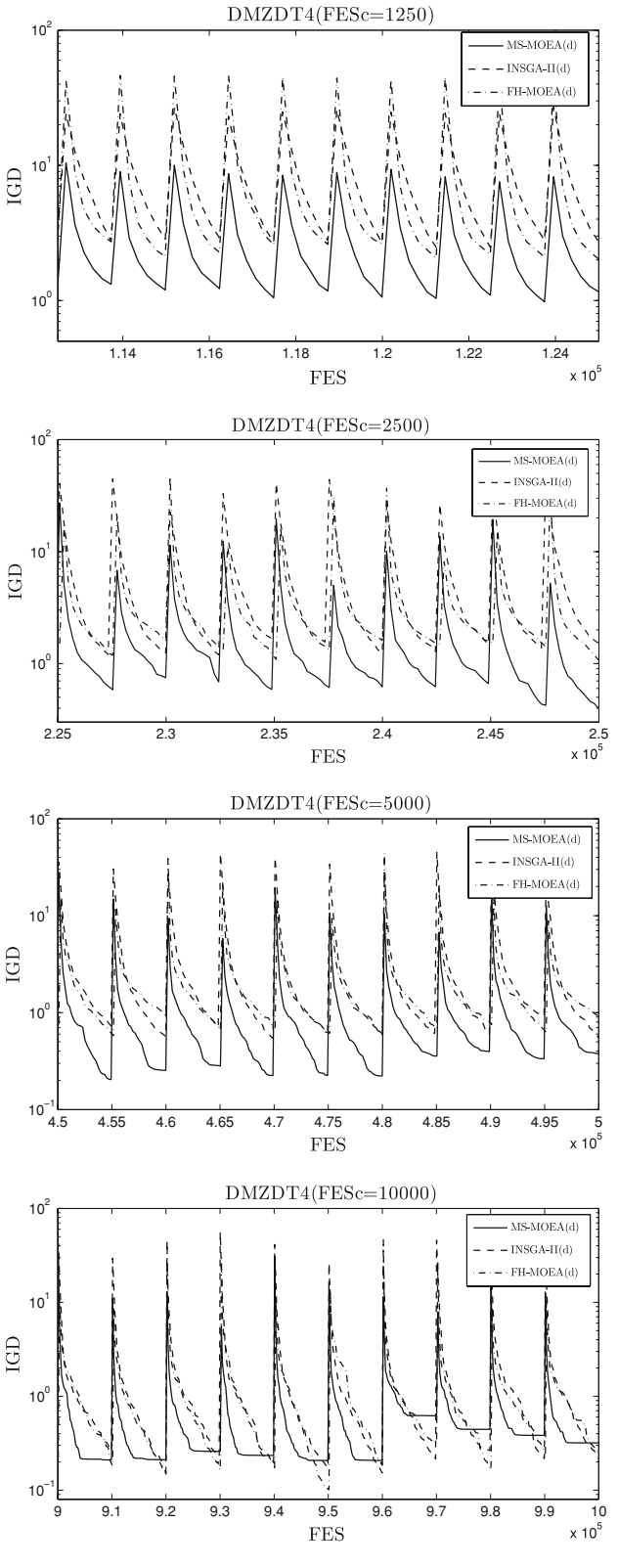

Fig. 9 Comparison of IGD metric curves for DMZDT4

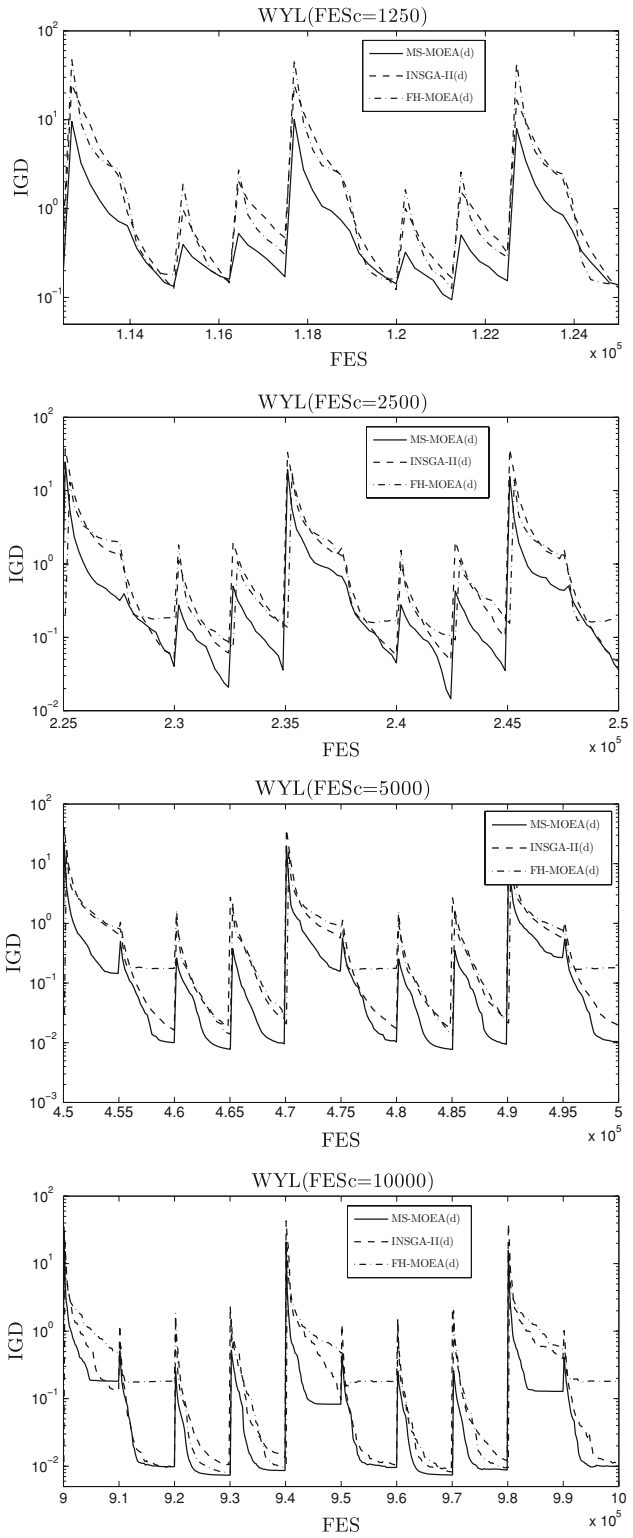

Fig. 10 Comparison of IGD metric curves for WYL

This is not surprising, because the excellent performance of MS-MOEA(s) on MZDT4 has been verified in the above sMO experiment, and furthermore, the memory like strategy leads to better starting point once the environment changes. However, INSGA-II(d) slightly outperforms MS-MOEA(d) when $FECc = 10{,}000$, due to the fact that GA has better global search ability, while in GDM, the DE is launched too earlier [44]. The Fig. 15 suggests that the non-dominated sets provided by MS-MOEA(d) have better diversity and convergence characteristics, except that when $FECc = 10{,}000$, INSGA-II(d) provides slightly better non-dominated sets in terms of convergence.

5.5.3 Experiment on dynamic WYL

In the previous researches, few works have been carried out on such problem with so sharply changing POF and POS.

For all $FESc$ settings, MS-MOEA(d) significantly outperforms the other algorithms, see Table 11. Figure 10 shows that MS-MOEA(d) is able to provide the best non-dominated set at any time, while the performances of other algorithms are identical. It is especially true when the environment is generated by MZDT4. Correspondingly, the evolution curves of other algorithms are almost overlapped. The reason may be about the same as is discussed for DMZDT4. As is shown in Fig. 16, the quality of non-dominated sets provided by MS-MOEA(d) are much better. Similar to the above dMO experiments, the addition of memory like strategy also provides a positive effect on WYL. We can conclude that MS-MOEA(d) is much more suitable for this hard problem of type II.

6 Conclusion

With strong requirement from engineering and scientific application, it is important to study dMO problems and corresponding effective algorithms. Few algorithms have been designed previously. In this paper, a number of dMO test instances are proposed extended from the classical ZDTs. Furthermore, two important issues of dMO algorithms, dynamic environment handling strategy and convergence speed in the stable period, are discussed. Based on the analysis, a new algorithm, multi-strategy ensemble multi-objective evolutionary algorithm (MS-MOEA), is proposed. In MS-MOEA, several strategies are adopted for different purposes: (1) GDM is used to achieve fast convergence speed in a short time after the environment change is detected and provide higher-quality non-dominated set when the non-dominated solutions get close to the true POS; (2) Gaussian mutation is designed to strengthen the ability of escaping from the local optima; (3) to handle dynamic environment, memory like strategy is designed to gain attraction around the memory solutions without losing the global search ability.

In the experiments on sMO problems, the strategies used to accelerate the convergence speed, including GDM and FH, cooperate well and provide top performances on different

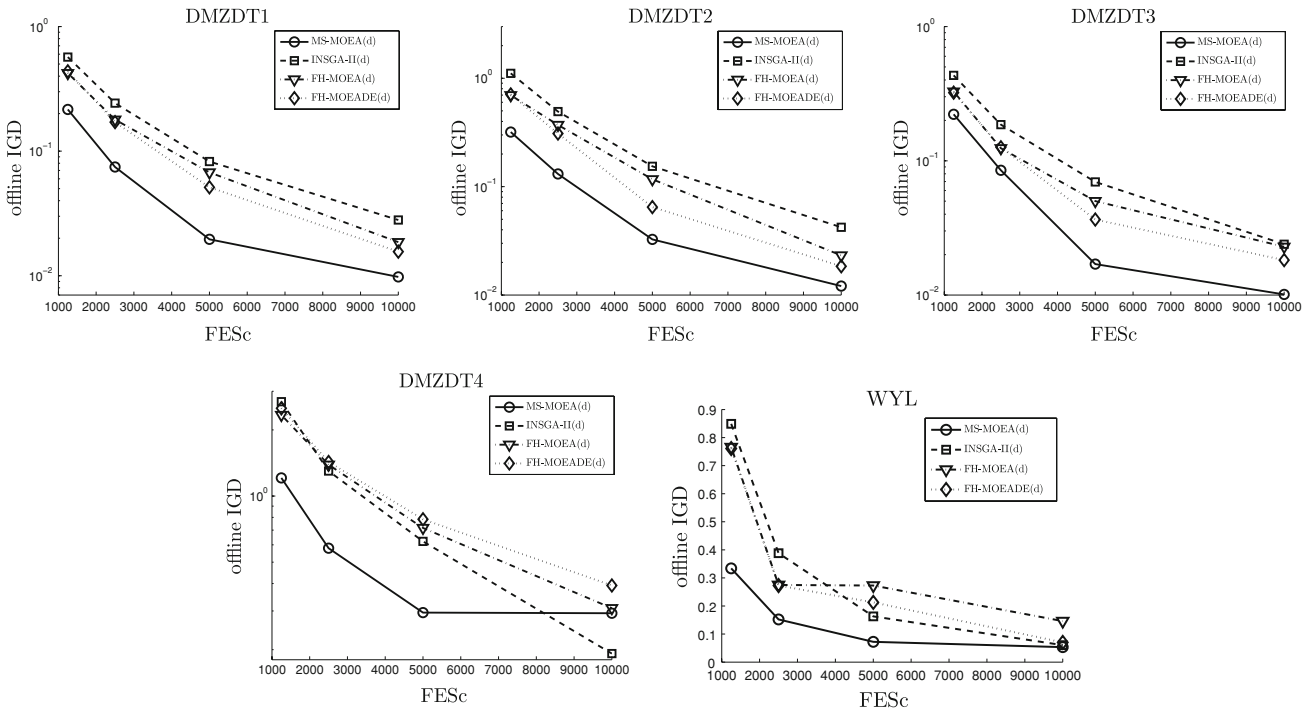

Fig. 11 Experimental results on DMZDTs and WYL

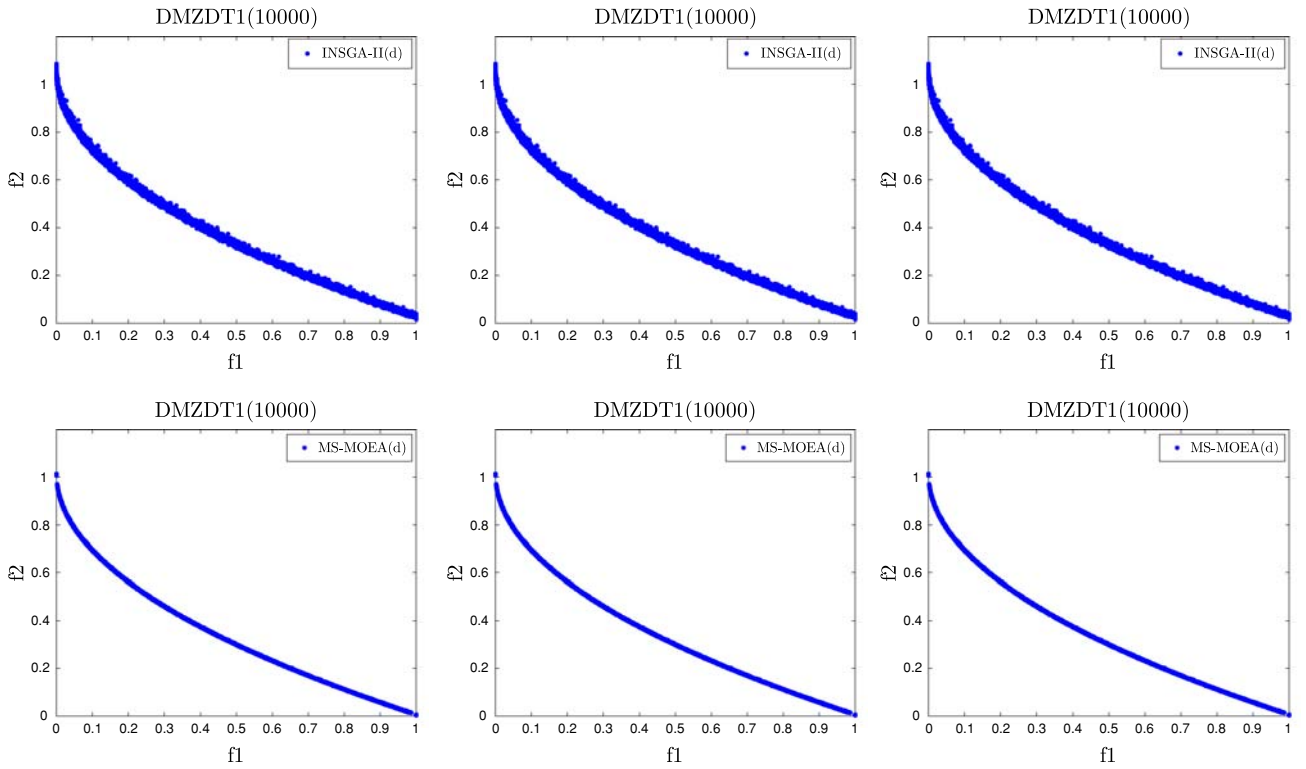

Fig. 12 Comparison of distribution of non-dominated set for dynamic DMZDT1

kinds of problems. Via the experiments on dMO problems, including FDAs, DMZDTs and WYL, the advantages of the multi-strategy ensemble, such as fast convergence speed and maintenance of historical information, are clearly revealed.

This work is also expected to call for more attention on dMO. There are still several unsolved problems, such as how to more appropriately make use of the past information, how to more appropriately generate dMO test instances, and etc.

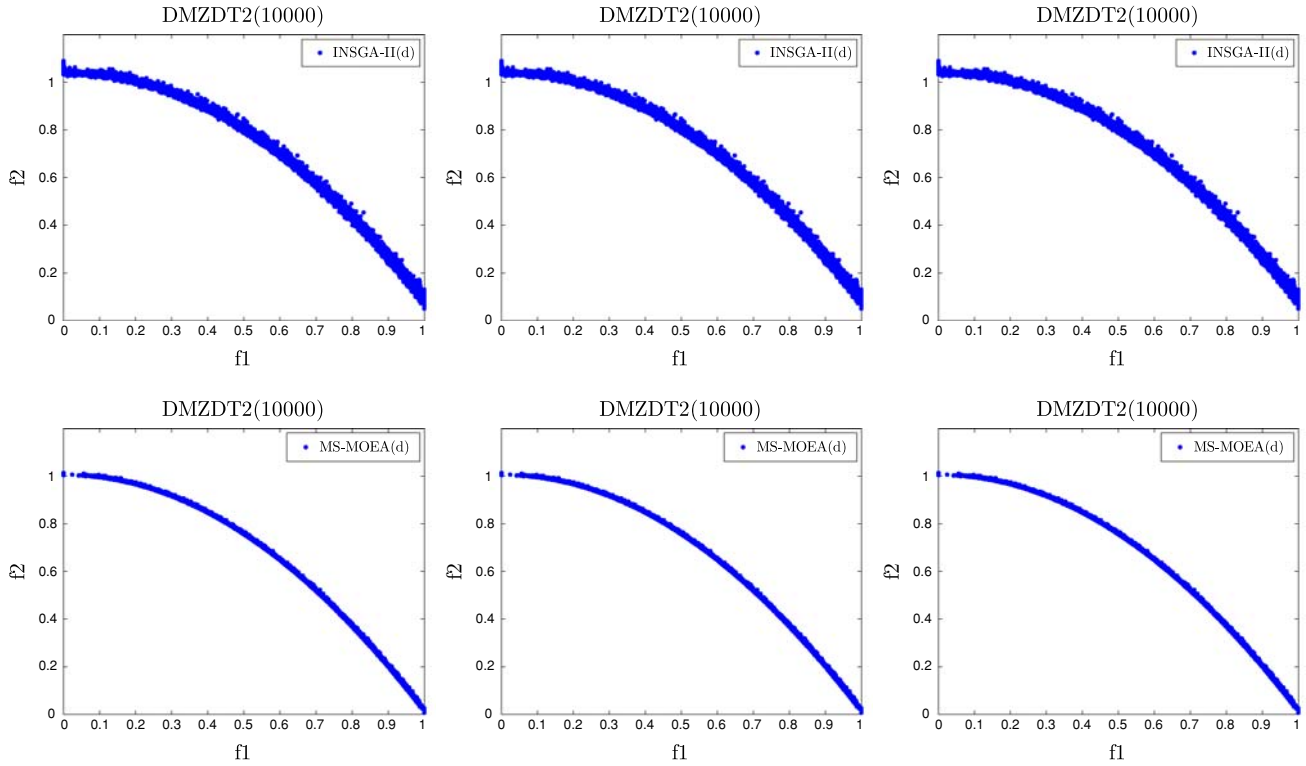

Fig. 13 Comparison of distribution of non-dominated set for DMZDT2

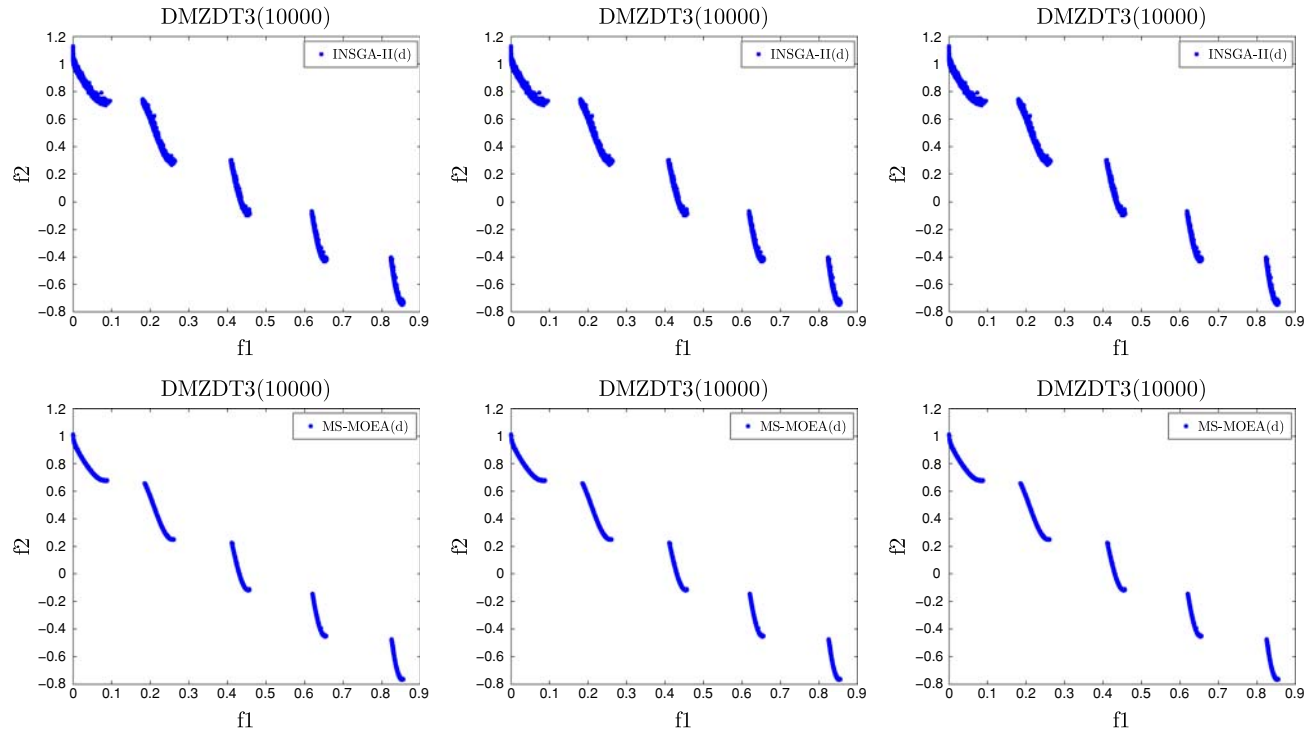

Fig. 14 Comparison of distribution of non-dominated set for DMZDT3

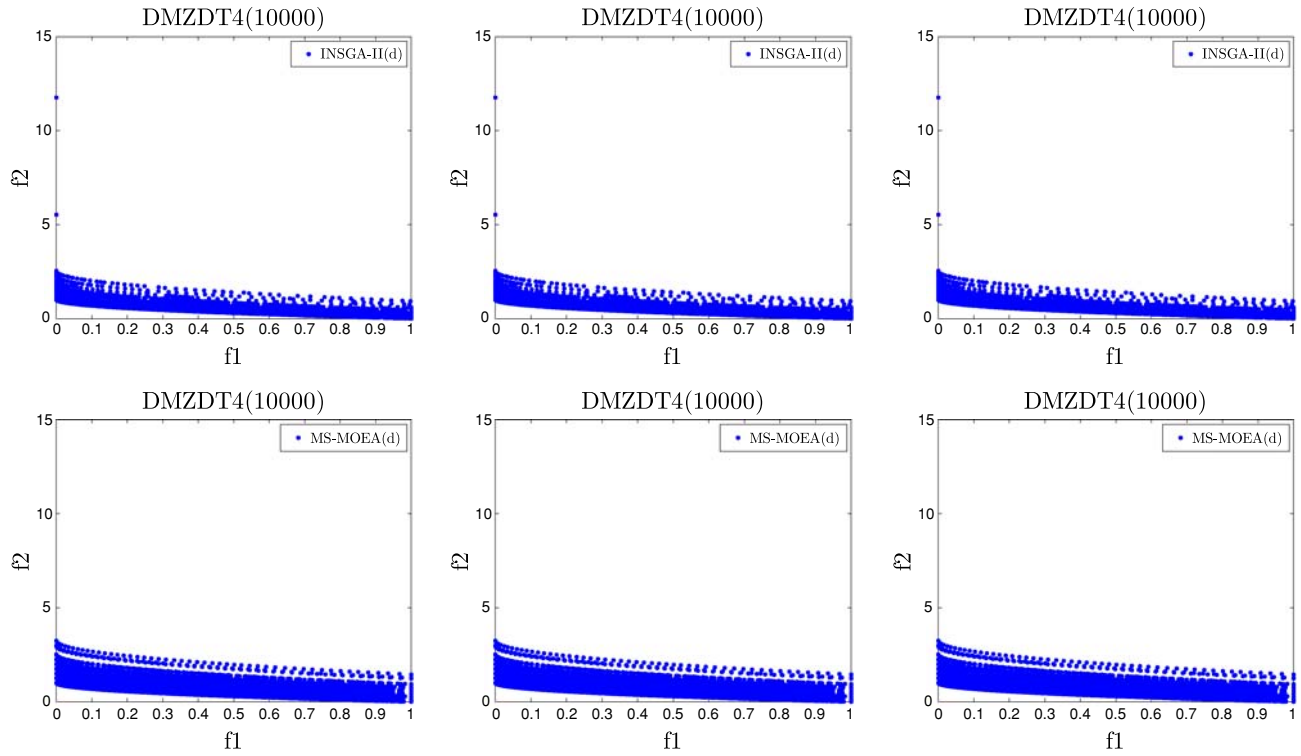

Fig. 15 Comparison of distribution of non-dominated set for DMZDT4

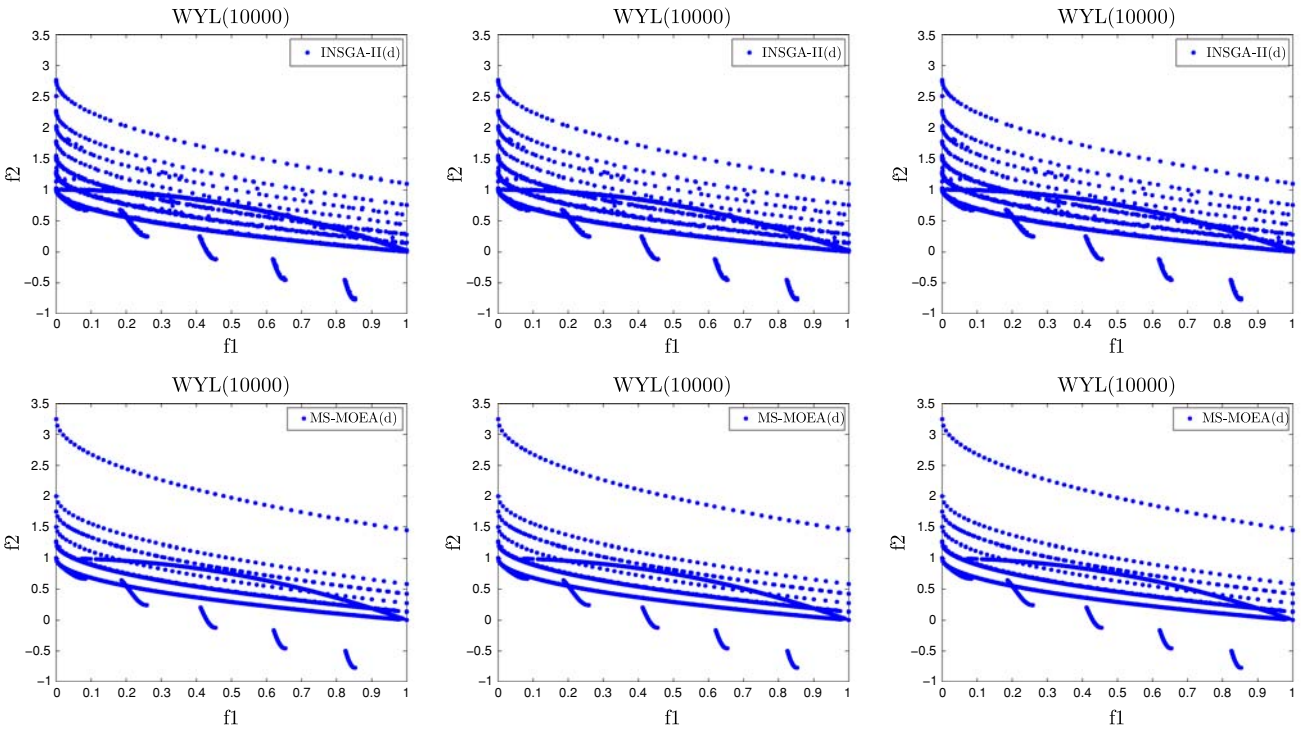

Fig. 16 Comparison of distribution of non-dominated set for WYL

Acknowledgments The authors would like to thank Mr. Deb to make the source code of NSGA-II available. This work was partially supported by the National Natural Science Foundation of China (Grants No. 60401015, U0835002), IEEE Walter Kaplus Student Summer Research Grant, the Chinese Academy of Science (CAS) Special Grant for Postgraduate Research, Innovation and Practice, and Graduate Innovation Fund of University of Science and Technology of China. The first author would like to thank Xiaolei Chen, Zhiwei Xiong, Ruoxi Xiang and Xuexiao Lai for their invaluable contributions.

References

1. Abido MA (2006) Multi-objective evolutionary algorithms for electric power dispatch problem. IEEE Trans Evol Comput 10(3):315–329
2. Amato P, Farina M (2005) An alife-inspired evolutionary algorithm for dynamic multiobjective optimization problems. Adv Soft Comput 1:113–125
3. Annunziato M, Bertini I, Pannicelli A, Pizzuti S (2001) Evolutionary control and optimization: an industrial application for combustion processes. In: Proceedings of EUROGEN, Athens, Greece, September, pp 367–372
4. Bhattacharya M, Lu G (2003) A dynamic approximate fitness-based hybrid EA for optimization problems. In: Proceedings of congress evolutionary computation, pp 1879–1886
5. Bierwirth C, Kopfer H (1994) Dynamic task scheduling with genetic algorithms in manufacturing systems. Technical report, Department of Economics, University of Bremen, Bremen, Germany
6. Bingul Z (2007) Adaptive genetic algorithms applied to dynamic multiobjective problems. Appl Soft Comput 7(3):791–799
7. Blackwell T (2007) Particle swarm optimization in dynamic environments. Evol Comput Dyn Uncertain Environ 51:29–49
8. Blackwell T, Bentley PJ (2002) Dynamic search with charged swarms. In: Proceedings of genetic evolutionary computations conference. Morgan Kaufmann, London, pp 19–26
9. Blackwell T, Branke J (2004) Multi-swarm optimization in dynamic environments. Appl Evol Comput 3005:489–500
10. Blackwell T, Branke J (2006) Multi-swarms, exclusion, and anti-convergence in dynamic environments. IEEE Trans Evol Comput 10(4):459–472
11. Branke J (1999) Memory enhanced evolutionary algorithms for changing optimization problems. Proc Congr Evol Comput 3:1875–1882
12. Branke J, Schmeck H (2002) Designing evolutionary algorithms for dynamic optimization problems. In: Proceedings of theory and applications of evolutionary computation, recent trends. Springer, Heidelberg, pp 239–262
13. Chan TM, Man KF, Kwong S, Tang KS (2008) A jumping gene paradigm for evolutionary multiobjective optimization. IEEE Trans Evol Comput 12(3):143–159
14. Cobb HG (1990) An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Tech Rep AIC-90-001, Naval Res Lab, Washington, DC
15. Coello C, Van Veldhuizen D, Lamont G (2002) Evolutionary algorithms for solving multi-objective problems. In: Ser. Genetic algorithms and evolutionary computation. Kluwer, Norwell
16. Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, New York
17. Deb K, Agrawal S (1999) A niched-penalty approach for constraint handling in genetic algorithms. In: Proceedings of international conference on artifical neural networks and genetic algorithms. Springer, Heidelberg, pp 235–243
18. Deb K. http://www.iitk.ac.in/kangal/code/newnsga/nsga2code.tar
19. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
20. Deb K, Mohan M, Mishra S (2005) Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solution. MIT Evol Comput 13(4):501–502
21. Du WL, Li B (2008) Multi-strategy ensemble particle swarm optimization for dynamic optimization. Inf Sci 178(15):3096–3109
22. Emmerich M, Beume N, Naujoks B (2005) An EMO algorithm using the hypervolume measure as selection criterion. In: Proceedings of third international conference on evolutionary multi-criterion optimization (EMO 2005), vol 3410. Springer, Berlin, pp 62–76
23. Farina M, Amato P, Deb K (2004) Dynamic multi-objective optimization problems: test cases approximations and applications. IEEE Trans Evol Comput 8(5):425–442
24. Goh CK, Tan KC (2007) An investigation on noisy environments in evolutionary multiobjective optimization. IEEE Trans Evol Comput 11(3):354–381
25. Handl J, Kell DB, Knowles J (2007) Multi-objective optimization in bioinformatics and computational biology. IEEE/ACM Trans Comput Biol Bioinform 4(2):279–292
26. Hatzakis I, Wallace D (2006) Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. In: Proceedings of genetic and evolutionary computation conference (GECCO), pp 1201–1208
27. Huang VL, Suganthan PN, Baskar S (2005) Multiobjective differential evolution with external archive. Technical report, Nanyang Technological University, Singapore
28. Huang VL, Qin AK, Suganthan PN, Tasgetiren MF (2007) Multi-objective optimization based on self-adaptive differential evolution. In: Proceedings of IEEE conference on evolutionary computation, 25–28 September 2007, Singapore
29. Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans Evol Comput 7(2):204–223
30. Jin Y, Branke J (2005) Evolutionary optimization in uncertain environments—a survey. IEEE Trans Evol Comput 9(3):303–317
31. Jin YC, Sendhoff B (2004) Constructing dynamic optimization test problems using the multiobjective optimization concept. In: Raidl GR et al (eds) Proceedings of evolutionary workshops 2004, LNCS3005. Springer, Berlin, pp 525–536
32. Knowles J (2006) ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Trans Evol Comput 10(1):50–66
33. Knowles J, Corne D (1999) The Pareto archived evolution strategy: a new baseline algorithm for multiobjective optimization. In: Proceedings of 1999 congress on evolutionary computation, Piscataway, NJ, pp 9–105
34. Knowles JD, Corne DW (2000) M-PAES: a memetic algorithm for multiobjective optimization. In: Proceedings of IEEE Congress on Evolutionary Computation, pp 325–332
35. Kukkonen S, Lampinen J (2007) Performance assessment of generalized differential evolution 3 (GDE3) with a given set of problems. In: Proceedings of IEEE conference on evolutionary computation, 25–28 September 2007, Singapore
36. Kumar A, Sharma D, Deb K (2007) A hybrid multi-objective optimization procedure using PCX based NSGA-II and sequential quadratic programming. In: Proceedings of IEEE conference on evolutionary computation, 25–28 September 2007, Singapore
37. Li C, Yang S, Nguyen TT, Yu EL, Yao X, Jin Y, Beyer H-G, Suganthan PN (2009) Benchmark generator for CEC 2009 competition on dynamic optimization. In: IEEE conference on evolutionary computation. Special session competition on evolutionary

computation in dynamic and uncertain environments. http://www3.ntu.edu.sg/home/EPNSugan/index_files/
38. Mendoza F, Bernal-Agustin JL, Domnguez-Navarro JA (2006) NSGA and SPEA applied to multi-objective design of power distribution systems. IEEE Trans Power Syst 21(4)
39. Ramsey C, Grefensttete J (1993) Case-based initialization of genetic algorithms. In: Proceedings of the fifth international conference on genetic algorithms, pp 84–91
40. Schaffer JD (1985) Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of 1st international conference on genetic algorithms, Pittsburgh, PA, pp 93–100
41. Sharma D, Kumar A, Deb K, Sindhya K (2007) Hybridization of SBX based NSGA-II and sequential quadratic programming for solving multi-objective optimization problems. In: Proceedings of IEEE conference on evolutionary computation, 25–28 September 2007, Singapore
42. Suganthan PN (2007) Performance assessment on multi-objective optimization algorithms. IEEE conference on evolutionary computation special session, competition on performance assessment of multi-objective optimization algorithms. http://www3.ntu.edu.sg/home/EPNSugan/index_files/
43. Wang Y, Li B (2008) FH-MOEA: multi-objective evolutionary algorithm based-on fast hyper-volume contribution approach. J Univ Sci Technol China 38(7):802–809 (special issue on information science for 50th Anniversary)
44. Wang Y, Li B (2009) Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment. In: Proceedings of IEEE congress on evolutionary computation (CEC 2009), pp 630–637 (accepted)
45. Wang Y, Li B (2009) ED-DE: cooperative estimation of distribution based differential evolution for economic dispatch optimization of power system. Inf Sci (accepted)
46. Wanner EF, Guimaraes FG, Takahashi RHC, Fleming PJ (2008) Local search with quadratic approximations into memetic algorithms for optimization with multiple criteria. MIT Evol Comput 16(2):185–224
47. While L, Hingston P, Barone L, Huband S (2006) A faster algorithm for calculating hyper-volume. IEEE Trans Evol Comput 10(1):29–38
48. Yang S (2005) Population-based incremental learning with memory scheme for changing environments. Proc Genet Evol Comput Conf 1:711–718
49. Yang S (2007) Genetic algorithms with elitism-based immigrants for changing optimization problems. Applications of evolutionary computing. In: Lecture Notes in Computer Science, vol 4448. Springer, Berlin, pp 627–636
50. Yang S, Tinos R (2007) A hybrid immigrants scheme for genetic algorithms in dynamic environments. Int J Autom Comput 4(3):243–254
51. Yang S, Yao X (2005) Experimental study on population-based incremental learning algorithms for dynamic optimization problems. Soft Comput 9(11):815–834
52. Yang S, Yao X (2008) Population-based incremental learning with associative memory for dynamic environments. IEEE Trans Evol Comput 12(5):542–561
53. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102
54. Yu X, Tang K, Chen TS, Yao X (2009) Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization. Memet Comput 1(1):3–24
55. Zhang QF, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731
56. Zhang QF, Zhou AM, Jin YC (2008) RM-MEDA: a regularity model-based multiobjective estimation of distribution algorithm. IEEE Trans Evol Comput 12(1):41-63
57. Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative study. In: Proceedings of parallel problem solving from nature V. Springer, Amsterdam, pp 292–301
58. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195
59. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength pareto evolutionary algorithm. Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, Tech Rep 103
60. Zitzler E, Thiele L, Laumanns M, Fonseca CM, Grunertda Fonseca V (2003) Performance assesment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput 7(2):117–132