Struct Multidisc Optim (2012) 46:853–870
DOI 10.1007/s00158-012-0800-x

RESEARCH PAPER

A local multiobjective optimization algorithm using neighborhood field

Zhou Wu · Tommy W. S. Chow

Received: 22 September 2011 / Revised: 20 February 2012 / Accepted: 29 March 2012 / Published online: 24 April 2012

Abstract A new local search algorithm for multiobjective optimization problems is proposed to find the global optima accurately and diversely. This paper models the cooperatively local search as a potential field, which is called neighborhood field model (NFM). Using NFM, a new Multiobjective Neighborhood Field Optimization (MONFO) algorithm is proposed. In MONFO, the neighborhood field can drive each individual moving towards the superior neighbor and away from the inferior neighbor. MONFO is compared with other popular multiobjective algorithms under twelve test functions. Intensive simulations show that MONFO is able to deliver promising results in the respects of accuracy and diversity, especially for multimodal problems.

Keywords Multiobjective optimization · Evolutionary algorithms · Local search · Neighborhood field model · Contour gradient optimization · Multiobjective neighborhood field optimization

1 Introduction

Most real world problems need to optimize several incommensurable objectives or competing objectives at the same time, such as the design of combinational logic circuits, autonomous vehicles navigation, and DNA sequence description (Coello and Lamont 2004; Jin 2006). These are called multiobjective optimization problems (MOOPs), in which no single solution can be optimal satisfying all objectives. For MOOPs, certain tradeoffs called Pareto optima can be found to compromise different objectives. A number of novel works on multiobjective evolutionary algorithms (MOEAs) have then been proposed (Schaffer 1984; Horn et al. 1994; Srinivas and Deb 1994; Zitzler and Thiele 1999; Knowles and Corne 2000). Most MOEAs were largely motivated by Goldberg's paper on non-dominance sorting (Goldberg 1989; Horn et al. 1994). For instance, the non-dominated sorting GA-II (NSGA-II) (Deb et al. 2002a) and strength Pareto EA-II (SPEA2) (Zitzler et al. 2001) were proposed to improve the computation complexity, density estimation and fitness assignment when solving MOOPs. NSGA-II and SPEA2 have been empirically proven to be two of the most efficient MOEAs and so become the benchmark algorithms of evaluating other multiobjective algorithms. Other existed global search meta-heuristic approaches, such as Particle Swarm Optimization (PSO) (Eberhart and Shi 2001), Differential Evolution (DE) (Storn and Price 1997), are also feasible for the multiobjective optimization. Their extended algorithms are called multiobjective Particle Swarm Optimization algorithms (MOPSOs) (Hu and Eberhart 2002; Parsopoulos and Vrahatis 2002; Coello et al. 2004) and multiobjective Differential Evolution algorithms (MODEs) (Abbass and Sarker 2002; Xue et al. 2003; Kukkonen and Lampinen 2005). The development in the multiobjective optimization area has been well reviewed in Veldhuizen and Lamont (2000), Zitzler (1999), Veldhuizen and Lamont (1998), Durillo et al. (2010). Despite these fruitful outputs, there are many open issues in MOOPs, such as the fitness assignment method and the design of effective search meta-heuristics.

Unlike the single objective optimization problem in which the optimum is clearly defined as the global best solution in the search space, MOOPs only have certain

Z. Wu · T. W. S. Chow (✉)
Department of Electronic Engineering,
City University of Hong Kong, Kowloon, Hong Kong, China
e-mail: eetchow@cityu.edu.hk

appropriate trade-offs with comprehensively good fitness in each objective. These trade-off solutions are called Pareto optima, which are non-dominated by all other solutions in the search space. Generally, for a problem $F(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})]$ of minimizing the M objectives, we call a decision vector $\boldsymbol{x}_1$ dominates another decision vector $\boldsymbol{x}_2$, if and only if each component of the objective vector $F(\boldsymbol{x}_1)$ is not larger than that of the objective vector $F(\boldsymbol{x}_2)$ with at least jth component f_j satisfying $f_j(\boldsymbol{x}_1) < f_j(\boldsymbol{x}_2)$. A solution is the global Pareto optimum only if no other solutions in the whole search space can dominate it. The Pareto set (PS) is the set of all global Pareto optima in the decision space, and the Pareto front (PF) is defined as the image of the Pareto set in the objective space (Coello and Lamont 2004). The target of MULTIOBJECTIVE algorithm is to obtain a well-dispersed set of solutions close to the Pareto optima. In other words, a certain multiobjective algorithm is expected to converge towards the Pareto set with two goals, namely accuracy and diversity. To achieve the two goals at the same time, multiobjective algorithms employ different search meta-heuristics, which can be categorized in the following three types.

The first type of algorithms refers to local search strategies, such as the gradient-based algorithms (Shukla 2007; Fliege and Svaiter 2000; Brown and Smith 2003) and the Pareto simulated annealing algorithm (PSA) (Czyzak and Jaszkiewicz 1998). This type of algorithms can find certain acceptable solutions efficiently with little computation time. But since these algorithms only exploit the good solution in the local region, their results are easily getting trapped in the local optima. Another drawback is that these local search algorithms are difficult to maintain the diversity, as they are separately applied to solve MOOPs.

The second type is global search algorithms, such as MOPSO, MODE and other MOEAs. They deal with a population of individuals and can obtain several Pareto optima diversely in a single run. These algorithms employing global search strategies can explore the whole search space. But these algorithms tend to require a relatively large number of iterations for obtaining acceptable results, because they cannot exploit the local area sufficiently compared with the local search methods. Furthermore, these global search algorithms often suffer from the drawback of stagnation because of the excessive exploration. When they are handling certain complicated multimodal functions, they stop converging to any optimum at the beginning of search process.

The existing conflicts between exploitation and exploration make the above mentioned algorithms difficult to achieve accuracy and diversity at the same time. To overcome this drawback, the third type is derived to balance between global exploration and local exploitation. This type of algorithms, called memetic algorithms (MAs) (Molina et al. 2010), is proposed to hybridize MOEAs with local search strategies. During the optimization process, MAs can do exploration at one time and exploitation at another time. The rationale behind MAs is to provide an improved method by compensating the deficiency of EA amid the local exploitation and the inadequacy of local search amid the global exploration. Many authors have reported successful hybridization of local search techniques with MOEAs (Lara et al. 2010; Wanner et al. 2008; Adra et al. 2005; Hu et al. 2003). In Adra et al. (2005), three different local search techniques, i.e. simulated annealing, hill climbing, and tabu search, were hybridized with multiobjective genetic algorithm. The obtained results can outperform the original multiobjective genetic algorithm. In Hu et al. (2003), a gradient based local search algorithm (sequential quadratic programming) was used in combination with NSGA-II and SPEA2 to solve some benchmark functions. The author concluded that if there are no local Pareto fronts, the hybrid MOEA has faster convergence speed than the original NSGA-II and SPEA2. Apparently, the third type can deliver better performances than the second type in terms of the accuracy and diversity. But the performance improvement is not remarkable because of poor local search strategies.

People may prefer to use the evolutionary algorithms and the MAs for real-world problems. The local search algorithms may be applied as a part of MAs, but most local search strategies are more easily getting trapped into local optima than the global algorithms like GA, PSO and DE (Molina et al. 2010). The poor performance of local search strategies has also limited MAs' overall performances. So it is necessary to emphasize on improving the local search strategies in the evolutionary computation area. Recently, a new search algorithm based on self-organizing map for single objective optimization was proposed to exploit the search space based on the neighbors' cooperative field. The cooperative field is called the Neighborhood Field Model (NFM), which can provide an efficient and effective exploitation method for finding the global optimum (Xu and Chow 2010). The scope of this paper is to discuss whether the neighborhood field can be employed as local search to deliver a global multiobjective algorithm with better performance than certain MOEAs.

The contributions of this paper include three aspects. First, MONFO is newly proposed with the concept of neighborhood field model (NFM) in multiobjective optimization. MONFO utilizes the non-dominance sorting scheme to obtain several sets of solutions with decent fitness values. The field between the neighboring sets has a descent direction towards the superior region. Second, the proposed MONFO is a new local search algorithm. While most multiobjective algorithms comply with the principle of learning from the best individuals (the non-dominated solutions), MONFO complies with a different principle of "learning

from its neighbors". Third, we propose an extensive dominance (E-dominance) relationship for the selection of offspring. This method provides a steady tournament evaluation, which has potential applications in other MOEAs. Despite being a local search, it can be noticed in our studies that MONFO is able to find the Pareto optima with accuracy and diversity, especially for the multimodal functions with high dimensional objective space. The results of MONFO is better than or at least comparable with NSGA-II, SPEA2, the multiobjective PSO (MOPSO) (Coello et al. 2004) and the third version generalized DE (GDE3) (Kukkonen and Lampinen 2005) in terms of accuracy and diversity.

The paper is organized as follows. Section 2 introduces previous work about the algorithm based on neighborhood field model (NFM), and describes four popular multiobjective algorithms NSGA-II, SPEA2, MOPSO and GDE3. Section 3 illustrates the proposed MONFO algorithm. Section 4 analyzes the property of MONFO. Section 5 gives the computational comparisons of NSGA-II, SPEA2, MOPSO, GDE3 and MONFO on the test functions. Section 6 concludes this paper.

2 Previous work

This section introduces a single optimization algorithm called Contour Gradient Optimization (CGO), which is based the neighborhood field model. In NFM, each individual is attracted by superior neighbors and is repulsed by inferior neighbors (Xu and Chow 2010). In the following, we present the neighborhood field model; then give the procedure of CGO; at last reviews some popular multiobjective algorithms.

2.1 Neighborhood field model

Neighborhood field model (NFM) is proposed to emulate the cooperation behavior in a local environment (Xu and Chow 2010). It is worth noting that agents in the real-world networks are likely to cooperate with their neighbors in the local environment rather than the others in the global environment (Shi and Eberhart 1998). This kind of cooperation can be modeled to deliver the global optimization. It is stated that in NFM an agent is influenced by superior neighbors positively and by other inferior neighbors negatively. Note that NFM is similar to the potential field model (PFM) (Khatib 1986). In PFM, the overall force on the robot is composed of the attractive force of the target and the repulsive force of the obstacle; the overall force drives the robot to approach the targets without collision. In NFM, each individual x_i like a robot regards superior neighbors as targets to follow, and regards inferior neighbors as obstacles to evade. The neighbor field of the individual $\boldsymbol{x}_i$ driven by one target and one obstacle can be expressed as

$$\boldsymbol{NF}_i = \Phi\left(\boldsymbol{xc}_i - \boldsymbol{x}_i\right) - \Phi\left(\boldsymbol{xw}_i - \boldsymbol{x}_i\right), \tag{1}$$

where $\boldsymbol{NF}_i$ is the overall force driving on $\boldsymbol{x}_i$, $\boldsymbol{xc}_i$ is the superior neighbor, $\boldsymbol{xw}_i$ is the inferior neighbor, and $\Phi(\cdot)$ is the dynamical force function related with the position difference. In the right hand side of (1), the first component represents the attractive force of the superior neighbor and the second component represents the repulsive force of the inferior neighbor.

2.2 Contour gradient optimization algorithm

We introduce a Contour Gradient Optimization algorithm (CGO) based on the neighborhood field (Xu and Chow 2010). CGO considers a single objective minimization problem $y = f\left(\left[x_{D,i}, x_{2,i}, \ldots, x_{D,i}\right]^T\right)$ (D is dimension of the search space). A population of N individuals cooperatively evolves in an attempt of searching the global optimum. These individuals are ranked by their fitness values from the best to the worst and are sorted into m levels evenly.

The individuals in the same level are assumed having the same fitness. In CGO algorithm, the neighbors $\boldsymbol{xc}_i$ and $\boldsymbol{xw}_i$ are specified as the nearest individuals in the neighboring levels as (2), which are called contour neighbors. Base on NFM, CGO is proceeding as the follows:

1) Initialization: randomize the initial N individuals in the search space.
2) Contouring: at the generation G, rank all individuals by their function value in ascendant order, and sort them into m levels. We denote the ith individual $\boldsymbol{x}_{i,G}$'s level number as $L(\boldsymbol{x}_{i,G})$. For each individual $\boldsymbol{x}_{i,G}$, recognize the superior contour neighbor $\boldsymbol{xc}_{i,G}$ in the level $L(\boldsymbol{x}_{i,G}) - 1$ and the inferior contour neighbor $\boldsymbol{xw}_{i,G}$ in the level $L(\boldsymbol{x}_{i,G}) + 1$ as (2). Especially if $\boldsymbol{x}_{i,G}$ is in the first level, $\boldsymbol{xc}_{i,G}$ is defined as $\boldsymbol{x}_{i,G}$. If $\boldsymbol{x}_{i,G}$ is in the last level, $\boldsymbol{xw}_{i,G}$ is defined as $\boldsymbol{x}_{i,G}$.

$$\begin{cases} \boldsymbol{xc}_{i,G} = arg \underset{L(x_{k,G})=L(x_{i,G})-1}{min} \left\|\boldsymbol{x}_{k,G} - \boldsymbol{x}_{i,G}\right\| \\ \boldsymbol{xw}_{i,G} = arg \underset{L(x_{k,G})=L(x_{i,G})+1}{min} \left\|\boldsymbol{x}_{k,G} - \boldsymbol{x}_{i,G}\right\| \end{cases}, \tag{2}$$

3) Mutation: perturb each individual $\boldsymbol{x}_i$ as (3).

$$\begin{aligned} \boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} &+ \alpha \cdot \boldsymbol{r}_1 \cdot \left(\boldsymbol{xc}_{i,G} - \boldsymbol{x}_{i,G}\right) \\ &- \alpha \cdot \boldsymbol{r}_2 \cdot \left(\boldsymbol{xw}_{i,G} - \boldsymbol{x}_{i,G}\right), \end{aligned} \tag{3}$$

where $\boldsymbol{xc}_{i,G}$ is the superior neighbor, $\boldsymbol{xw}_{i,G}$ is the inferior neighbor at Gth generation; $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are random

vectors uniformly distributed in [0, 1], and α is the learning rate. $\boldsymbol{v}_{i,G}$ is the obtained mutant vector.

4) Crossover: recombine the mutation vector with the target vector $\boldsymbol{x}_i$.

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if rand(0,1)} \le Cr \text{ or } j = j_{rand} \\ x_{j,i,G}, & \text{otherwise} \end{cases}, \tag{4}$$

where $j = 1, 2, \ldots, D$ is the dimension index; Cr is the crossover probability; rand(0,1) is a uniformly distributed random number in the scale of [0, 1]; j_{rand} is a random component to accept the new mutant vector so that the trial vector is different from the target vector.

5) Selection: in the next generation, the ith individual will be updated as the better one between $\boldsymbol{x}_{i,G}$ and $\boldsymbol{u}_{i,G}$ as

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G}, & \text{if } f\left(\boldsymbol{u}_{i,G}\right) \le f\left(\boldsymbol{x}_{i,G}\right) \\ \boldsymbol{x}_{i,G}, & \text{otherwise} \end{cases}. \tag{5}$$

6) If the stopping criteria are not satisfied, go to step 2.

Based on the neighborhood field, CGO has placed equal emphasis on each individual, which plays the same important role in refining other neighbors. As a result, all individuals will not move towards the single direction but towards diverse directions instead, which is helpful to maintain the population's diversity. This paper shows how NFM can be extended from a single objective problem to a multiobjective problem for finding several different Pareto optima at the same time.

2.3 Four multiobjective algorithms

Four popular multiobjective algorithms are briefly described as they have been considered in comparison studies. They are non-dominated sorting genetic algorithm-II (NSGA-II) (Deb et al. 2002a), strength Pareto evolutionary algorithm 2 (SPEA2) (Zitzler et al. 2001), multiobjective particle swarm optimization (MOPSO) (Coello et al. 2004), and generalized differential evolution 3 (GDE3) (Kukkonen and Lampinen 2005).

The NSGA-II algorithm was proposed by Deb et al. (2002a). It is a genetic algorithm based on obtaining a new population from the original one by applying the typical genetic operators (selection, crossover, and mutation). The individuals in the two populations are then sorted according to their Pareto dominance ranks. The best solutions are chosen to create a new population. In case of selecting some individuals with the same rank, a density estimation based on measuring the crowding distance to the surrounding individuals in the same rank is used to choose the most sparse solutions.

SPEA2 was proposed by Zitzler et al. (2001). In this algorithm, each individual has a fitness value that is the sum of its strength raw fitness plus an estimation of density. The strength of a certain individual is proportional to the number of solutions that dominated by the individual. SPEA2 applies the selection, crossover, and mutation operators to fill an archive of individuals; then the non-dominated individuals of both the original population and the archive are copied into a new population. If the number of archive is greater than the population size, a truncation operator based on calculating the distances to the kth nearest neighbor is used. In this way, the individuals with the minimum distance to any other individual are truncated out of the archive.

In the evaluated multiobjective particle swarm optimization (MOPSO) (Coello et al. 2004), an external repository is used to store the non-dominated solutions found by the population. Every particle has a position vector $\boldsymbol{x}_i$ and a velocity vector $\boldsymbol{v}_i$. Each particle is updated towards its own best position $\boldsymbol{x}_p$ and the best position $\boldsymbol{x}_g$ chosen from the repository. When a certain particle finds the best solution, other particles are informed to move towards the best solution with an adaptive velocity. The new velocity of the ith particle are updated by

$$\boldsymbol{v}_i = \omega \cdot \boldsymbol{v}_i + c_1 \cdot \boldsymbol{r}_1 \cdot \left(\boldsymbol{x}_p - \boldsymbol{x}_i\right) + c_2 \cdot \boldsymbol{r}_2 \cdot \left(\boldsymbol{x}_g - \boldsymbol{x}_i\right), \tag{6}$$

where ω is called the inertia weight; c_1, c_2 are positive learning rates; $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are random vectors uniformly distributed in the interval of [0, 1]. The non-dominated solutions among the new positions are then inserted into the repository. If the repository is full, truncation is preceded according to the crowding distance of NSGA-II.

Among many multiobjective DE algorithms (MODEs), the third version of generalized DE algorithm (GDE3) proposed in Kukkonen and Lampinen (2005) is one of the most successful MODEs. In GDE3, three random vectors chosen from the population are mutated to generate a mutant vector as

$$\boldsymbol{v}_i = \boldsymbol{x}_{r0} + F\left(\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}\right), \tag{7}$$

where $r0$, $r1$ and $r2$ are three distinct random indices; $\boldsymbol{x}_{r0}$, $\boldsymbol{x}_{r1}$, $\boldsymbol{x}_{r2}$ are three random individuals in the population. The mutant vector is then combined with the target vector to generate a trial vector. The trial vector is accepted as offspring if it weakly constraint-dominates the target vector as follows. In case of both infeasible vectors, the trial vector is selected if it weakly dominated the target in the constraint violation space; in case of the feasible and infeasible vectors, the feasible vector is selected; in case of both feasible vectors, the trail vector is selected if it weakly dominates the target vector. If both vectors cannot constraint-dominate each other, they are saved in the population. In the next generation, the

population is pruned based on the non-dominated sorting and crowding distance as NSGA-II.

3 Multiobjective neighborhood field algorithm

This paper proposes a new local search multiobjective algorithm—Multiobjective Neighborhood Field Optimization algorithm (MONFO), which belongs to the Pareto based algorithms. It is not in any other current frameworks such as PSO, DE, and other EAs. Unlike other local search algorithms based on individual refinement (Shukla 2007; Fliege and Svaiter 2000; Brown and Smith 2003), MONFO deals with a population of individuals cooperatively, in which each individual is updated close to the superior neighbor and away from the inferior neighbor. In MONFO, the population can often be sorted into several levels, which seems to classify the population into several sets from the best to the worst. The individual in certain level will learn from the nearest individuals in the neighboring levels. Finally, the updating directions as the neighborhood field can push each individual towards the non-dominated sets. Unlike other local algorithms, MONFO is a novel local algorithm that can balance the local exploitation and the global exploration with fast convergence speed. In MONFO, an extensive dominance relationship is newly defined to select the offspring.

3.1 Multiobjective neighborhood field optimization algorithm

Generally, a D dimensional multiobjective problem for minimization with M objectives can be expressed as $\boldsymbol{F}(\boldsymbol{x}) = \left[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})\right]$, where $\boldsymbol{x}$ is a D dimensional solution, $f_j(\boldsymbol{x})$ is the objective function for the jth objective. An external archive Ex is used to store the non-dominated solutions in the past generations. MONFO handles the problem as the following:

1) Initialization: randomize the initial N individuals in the search space.
2) Contouring: in the Gth generation, sort the population into several fronts based on the dominance, and rank the within-front individuals based on a randomly chosen objective. According to the overall rankings, divide all individuals into m levels evenly. We denote the ith individual $\boldsymbol{x}_{i,G}$'s level as $L(\boldsymbol{x}_{i,G})$. For each individual $\boldsymbol{x}_{i,G}$, recognize the superior contour neighbor $\boldsymbol{xc}_{i,G}$ in the level $L(\boldsymbol{x}_{i,G}) - 1$, and the inferior contour neighbor $\boldsymbol{xw}_{i,G}$ in the level $L(\boldsymbol{x}_{i,G}) + 1$ as (2). Especially, if $\boldsymbol{x}_{i,G}$ is in the first level, $\boldsymbol{xc}_{i,G}$ is defined as $\boldsymbol{x}_{i,G}$; if $\boldsymbol{x}_{i,G}$ is in the last level, $\boldsymbol{xw}_{i,G}$ is defined as $\boldsymbol{x}_{i,G}$.
3) Mutation: perturb each individual $\boldsymbol{x}_{i,G}$ as (3).
4) Crossover: recombine the mutation vector with the target solution $\boldsymbol{x}_{i,G}$ as (4).
5) Selection: in the next generation the ith individual will be updated as the better one between $\boldsymbol{x}_{i,G}$ and $\boldsymbol{u}_{i,G}$. If $\boldsymbol{u}_{i,G}$ can extensively dominate $\boldsymbol{x}_{i,G}$, $\boldsymbol{u}_{i,G}$ will be selected in the next generation. Otherwise the target solution $\boldsymbol{x}_{i,G}$ will be selected. The extensive dominance relationship denoted as $\lesssim$ will be defined later. The selection can be express as

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G}, & \text{if } F\left(\boldsymbol{u}_{i,G}\right) \lesssim F\left(\boldsymbol{x}_{i,G}\right) \\ \boldsymbol{x}_{i,G}, & \text{otherwise} \end{cases}. \tag{8}$$

6) Add the non-dominated solutions of the current population into the external archive Ex. If the archive size is large than N, prune the archive according to the density estimation. The first N sparse solutions are maintained in Ex. Go to step 2 until the stopping criteria are satisfied.

In the contouring step, MONFO ranks the population by the non-dominated sorting of NSGA-II. For the solutions in the same front, individuals are ranked based on a randomly chosen objective. They are ranked from the minimum to the maximum on that objective. For example, for the solutions in the first front, we randomly choose an objective f_j, and then all the solutions in the first front are ranked according to their jth objective value. For $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in the first front, $\boldsymbol{x}_1$ is ranked before $\boldsymbol{x}_2$ if and only if $f_j(\boldsymbol{x}_1) < f_j(\boldsymbol{x}_2)$.

After sorting the population, the population is evenly divided into m levels. Each individual $\boldsymbol{x}_{i,G}$ is perturbed based on NFM. Each individual is repulsed by the contour neighbor in its next level, and attracted by the contour neighbor in the previous level. Especially, for the individual located at the first level, it can only be repulsed by the contour neighbor in the second level. For the individual located in the mth level, it can only be attracted by the contour neighbor in the $(m-1)$ th level. After mutation and crossover, the new solution $\boldsymbol{u}_{i,G}$ may have some components out of the constrained search space, and then the boundary action is needed. As the boundary action is not the main focus of this paper, we simply utilize the reinitializing method for the violent components.

In the last step, the external archive is pruned if its size is larger than the predefined number. The density estimation technique has been widely used for pruning the archive in most MOEAs. A good density estimation method can express the population's distribution accurately. How to calculate each individual's density is still an open issue. Currently there are mainly three methods for the density estimation. First, an individual's density can be calculated as the inverse of the summation of distances to its first k-nearest neighbors. Second, the inverse of the distance to the

kth nearest individual is used for the density estimation in SPEA2. Third, evaluate the density by the crowding distance, which is defined as the side-length of the cuboids enclosing the evaluated individual, proposed in NSGA-II. It must be noted that these different estimation methods can be used in MONFO. Here we use the inverse of the crowding distance to estimate the density mentioned in Xue et al. (2003).

3.2 Extensive dominance (E-dominance)

An extensive dominance (E-dominance) mentioned in the selection procedure is newly introduced in this paper. In general, tournament selection in multiobjective optimization is much difficult compared to that in the single objection optimization. How to select an offspring between two parent vectors is an interesting issue. Because two parent vectors may be non-dominated with each other in tournament selection, and then it is hard to say which is better based on the Pareto dominance. Unlike certain MOEAs using "one-to-one" Pareto dominance comparison, MONFO employs "one-to-many" comparisons between the trail vector and the parents, called extensive dominance (E-dominance). E-dominance can be expressed as

$$F(\boldsymbol{u}_i) \lesssim F(\boldsymbol{x}_i), \quad if \begin{cases} \exists \boldsymbol{x}_e, LF(\boldsymbol{x}_e) \le LF(\boldsymbol{x}_i),\ F(\boldsymbol{u}_i) \prec F(\boldsymbol{x}_e) \\ or \\ \exists j \in [1,\ldots,M],\ f_j(\boldsymbol{u}_i) < \min f_j \end{cases}, \quad (9)$$

where $\lesssim$ is the notation of the E-dominance relation, $\prec$ is the notation of the normal Pareto dominance relation, and $LF(\boldsymbol{x}_i)$ means the front level of $\boldsymbol{x}_i$ in non-dominated sorting. For example, $LF(\boldsymbol{x}_i) = 1$ means that $\boldsymbol{x}_i$ is in the first front. Definitely, $\boldsymbol{u}_i$ can E-dominate $\boldsymbol{x}_i$ if $\boldsymbol{u}_i$ has at least one objective function value less than the found minimum objective value, or it can dominate any individual in its current and before fronts. In MONFO, the trial vector is selected if it can E-dominate the target vector, otherwise the target vector is selected into the next generation.

Figure 1 gives a simple illustration of E-dominance. Although the individuals A_1 and A_8 are non-dominated with each other by "one-to-one" Pareto dominance comparison, A_1 can E-dominate A_8, because A_1, A_8 are in the different fronts. In the figure, the shaded region is the set of solutions that E-dominate A_8. It can be noticed that the E-dominance in fact includes "one-to-many" comparisons, which can enlarge the search region (the shaded region). Therefore, the selection in MONFO turns out to be less greedy than the selection based on "one-to-one" comparison.

E-dominance is a general scheme for dominance evaluation, which suits to be applied in the tournament selection of other MOEAs. E-dominance is an extension of Pareto dominance, which utilizes the whole population's information to compare the two peers' fitness. We find that in MONFO the E-dominance can deliver the final results more accurately and diversely than the normal Pareto dominance.

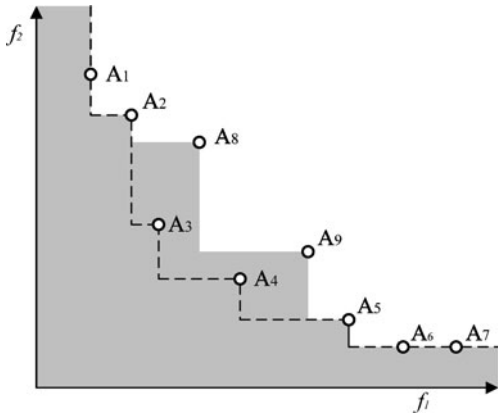

Fig. 1 E-dominance in a minimization example. The *dashed line* is the first front, and the *shaded region* is the set of solutions that can E-dominate A_8

4 Analysis of neighborhood field in MONFO

In MONFO, the neighbor field is desired to push the superior neighbors or pull the inferior neighbors towards better regions in the local region efficiently. Although multiobjective optimization is more complex than the single objective optimization, we will analyze the neighborhood field with some local characteristics.

The neighborhood field in MONFO can approximate a descent direction of MOOP. For a given minimization function $F(\boldsymbol{x})$ if $\boldsymbol{x}_i$ is not a local Pareto optima, the multiobjective gradient of $F(\boldsymbol{x})$ at $\boldsymbol{x}_i$ can be given (Shukla 2007),

$$\boldsymbol{g}(\boldsymbol{x}_i) = J(\boldsymbol{x}_i)\,\lambda^*, \quad (10)$$

where $\boldsymbol{g}(\boldsymbol{x}_i)$ is the multiobjective gradient of $F(\boldsymbol{x})$ at $\boldsymbol{x}_i$; $J = \left[\nabla f_1, \nabla f_2, \ldots, \nabla f_M\right]^T$ is the Jacobian matrix of $F(\boldsymbol{x})$ at $\boldsymbol{x}_i$. In (10) λ^* is the optimal solution of a quadratic programming problem as follows

$$\begin{aligned} &\lambda^* = \arg\min \tfrac{1}{2}\left\|J(x_i)\,\lambda\right\|^2 \\ &st\ \lambda \ge 0, \textstyle\sum_j \lambda_j = 1 \end{aligned}. \quad (11)$$

Lemma 1 *Assume that $\boldsymbol{x}_i$ is not a local Pareto optimum. In its neighborhood we can find a superior individual $\boldsymbol{xc}_i$ that dominate $\boldsymbol{x}_i$, and also find an inferior neighbor $\boldsymbol{xw}_i$ that*

dominated by $\boldsymbol{x}_i$. The neighborhood field in MONFO can approximate a descend direction satisfying

$$\begin{cases} -g\left(x_i\right)^T\left(xc_i - x_i\right) \geq 0 \\ g\left(x_i\right)^T\left(xw_i - x_i\right) \geq 0 \end{cases}. \tag{12}$$

Proof Since $\boldsymbol{xc}_i$ is the neighbor close to $\boldsymbol{x}_i$, using the linear approximation we can obtain

$$f_j\left(xc_i\right) \approx f_j\left(x_i\right) + \nabla f_j\left(x_i\right)\left(xc_i - x_i\right), j = 1, \ldots, M.$$

Then according to (10), we can obtain

$$\begin{aligned} g\left(x_i\right)^T\left(xc_i - x_i\right) &= \lambda^{*T} J\left(x_i\right)^T\left(xc_i - x_i\right) \\ &= \lambda^{*T}\left[\nabla f_1, \nabla f_2, \ldots, \nabla f_M\right]\left(xc_i - x_i\right) \\ &\approx \sum\nolimits_j \lambda_j^*\left(f_j\left(xc_i\right) - f_j\left(x_i\right)\right) \end{aligned}$$

Since $\boldsymbol{x}_c$ dominates $\boldsymbol{x}_i$, then

$$f_j(xc_i) - f_j(x_i) \leq 0 \text{ and } \sum\nolimits_j f_j(xc_i) - f_j(x_i) < 0.$$

Therefore, we can obtain $\sum_j \lambda_j^*\left(f_j\left(x_c\right) - f_j\left(x_c\right)\right) \leq 0\,(\lambda^* \geq 0)$. The first conclusion is proven. The second conclusion can be proven in the same way. In all, the neighborhood field $\boldsymbol{NF}_i$ will satisfy $-g\left(x_i\right)^T \cdot NF_i = -g\left(x_i\right)^T \cdot \left[\Phi\left(xc_i - x_i\right) - \Phi\left(xw_i - x_i\right)\right] \geq 0$. □

According to Lemma 1, the neighborhood field is close to the inverse multiobjective gradient based on the cosine similarity. It is reasonable that the neighborhood field can be used as a tool to approximate the MOP, especially for some engineering applications with complicated functions. When $\boldsymbol{xc}_i$, $\boldsymbol{xw}_i$ are the local best and worst individuals, the field direction is precisely the inverse multiobjective gradient. Since the neighborhood field can approximate a descend direction of the MOOP, the search direction in MONFO is efficient.

Figure 2 shows an instance of the neighborhood field in the objective space and decision space. The neighborhood field includes two differential vectors $\boldsymbol{xc}_i - \boldsymbol{x}_i$ and $\boldsymbol{x}_i - \boldsymbol{xw}_i$. To recognize the neighbors $\boldsymbol{x}_c$ and $\boldsymbol{xw}_i$, MONFO needs to order a population of individuals from the best to the worst and sort them into the different level sets. The non-dominated sorting scheme appears to be a good choice of this paper, in which a population of individuals can be sorted into several levels based on the Pareto dominance. MONFO utilizes the fast non-dominated sorting approach in NSGA-II.

In Fig. 2, a series of level sets with descend fitness can be obtained by non-dominated sorting. The set of solutions in the same front have the same fitness value. It is noticed that the individuals distribute in three fronts displayed in the 2-D decision space in Fig. 2. For a solution $\boldsymbol{x}_i$ in the figure, choose $\boldsymbol{xc}_i$ and $\boldsymbol{xw}_i$ as contour neighbors defined in (2). These contour neighbors have the minimum Euclidian distance in the neighboring two fronts. The neighbor $\boldsymbol{xc}_i$ is superior to $\boldsymbol{x}_i$, and the neighbor $\boldsymbol{xw}_i$ is inferior to $\boldsymbol{x}_i$ in the lower and higher fronts respectively. When we use these contour neighbors for the mutation, Fig. 2 shows that the two differential vectors of $\boldsymbol{xc}_i - \boldsymbol{x}_i$ and $\boldsymbol{x}_i - \boldsymbol{xw}_i$ can approximate a descend direction at $\boldsymbol{x}_i$, which is the same with Lemma 1.

Due to the descent property, each individual is driven towards global Pareto optima accurately and diversely when searching in the neighborhood field direction. The final results are accurate because each front iteratively moves towards the neighboring superior level and away from the neighboring inferior level. Furthermore, the MONFO can also maintain the diversity of the population. For each individual only one target in the neighboring superior layer has an attractive force, and one target in the neighboring inferior layer has a repulsive force. Each individual driven by these two forces will approximately move on their specific directions diversely.

In all, MONFO can efficiently solve the MOOPs problem based on the local cooperation. The neighborhood field can provide an efficient search heuristic, which can find the global optima accurately and diversely at the same time. The running time of MONFO is obviously more than GDE3 due

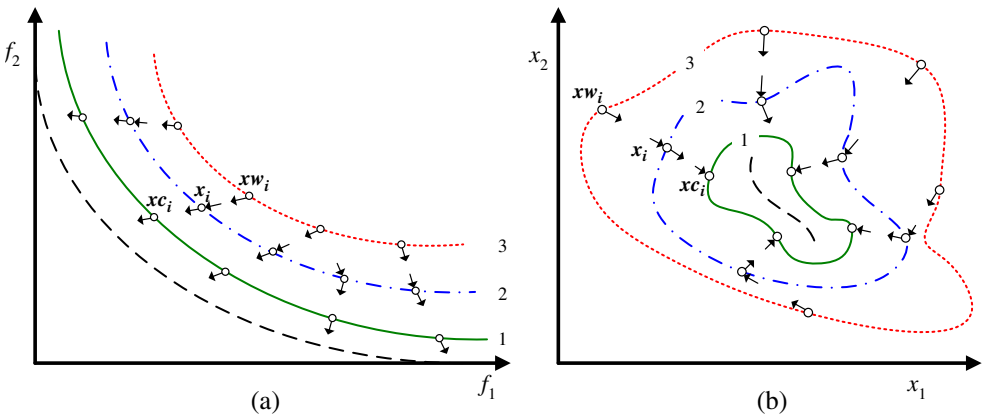

Fig. 2 An illustration of the neighborhood field when the population is sorted into three fronts. **a** The distribution of fronts in the objective space, and the global Pareto front is denoted as the *bottom left dashed line*. **b** The corresponding distribution of fronts in the decision space, in which the global Pareto set is denoted as the *dashed line*

to the additional contouring. The contouring has the computation complexity O(GN). The other parts of MONFO have the same computation complexity with GDE3, i.e., $O\left(GN\log^{M-1}N\right)$ (Kukkonen and Lampinen 2005). So MONFO's computation complexity is the same with GDE3 as $O\left(GN\log^{M-1}N\right)$, where G, N and M is the maximum generations, the population size and the number of objectives.

5 Simulations

The proposed MONFO is a fundamental framework for multiobjective optimization. MONFO is compared other two widely used MOEA frameworks NSGA-II and SPEA2. In addition, as MONFO exhibits similar with other vector field optimization techniques, such as MOPSO and GDE3, they are included in this comparison study.

Compared with MOPSO, MONFO is different in many aspects. First, MOPSO has a blackboard recording the memories of each agent, but MONFO does not need the blackboard because it only utilizes the information in the current iteration. Second, each particle in MOPSO is updated towards the historical best positions and the local best positions positively. In contrast each individual in MONFO is influenced by the neighborhood field, including positive attracting force and negative repulsing force. Third, each particle in MOPSO is updated in fully dimensions, but in MONFO the individual is updated in the random chosen dimensions.

The differences between GDE3 and MONFO are their differential mutation and the selection scheme. First, GDE3 chooses the differential vectors randomly among the whole population, but MONFO chooses two deterministic differential vectors in a local environment. The differential vector in MONFO can be clearly classified as positive and negative dynamics. This ensures each individual's refinement with a larger chance than GDE3. Second, the selection in GDE3 is base on the weakly dominance, but the selection in MONFO utilizes the E-dominance.

We compare MONFO with NSGA-II, SPEA2, MOPSO, and GDE3 on the following twelve benchmark functions in Table 1. They are ranging from two objectives to five objectives. We implement these algorithms with the software Matlab on the computer with Core-2 CPU Q9650 and 4G RAM.

5.1 Test functions and simulation settings

Many classical benchmark functions have been proposed which can cover some essential features of MOOPS with two objectives (Tan et al. 2005). However, they cannot systematically express overall features of the MOOPs, and some of them cannot even formulate the definition of Pareto optima mathematically. Deb (1999) pointed out several features that pose challenges of accuracy and diversity for multiobjective optimization, including multimodality, deception, isolated optima, non-continuity and convexity property. They also proposed a series of two-objective benchmark functions ZDTs to cover these features systematically (Zitzler et al. 2000). Since two-objective functions can be easily understood through visualization, the ZDTs functions have been widely used as benchmarks in the evaluation of MOEAs. For more complex problems with three and more objectives, Deb et al. (2002b) proposed a bottom-up approach to generate a series of test functions DTLZs. Their proposed method is generic and scalable to an arbitrary number of objectives. However, ZDTs and DTLZs are often criticized for their simple and regular Pareto set shapes that consist of either line segments or plane areas. Additionally, some of ZDTs and DTLZs have their optima laying on the boundary of the search space, which have decreased their difficulty. The MOEAs can easily converge towards the optima at the boundary due to the boundary violation action. Li and Zhang (2009) thus introduced a general framework to construct test instances with complicated Pareto set shapes, and gave a series of test function LZs. We construct a comprehensive test set chosen from ZDTs (Zitzler et al. 2000), DTLZs (Deb et al. 2002b) and LZs (Li and Zhang 2009). Table 1 lists these functions, their input dimensions, their search spaces, and the objective spaces.

In order to allow a quantitative evaluation of the obtained results, two issues have to be taken into consideration. The

Table 1 A set of twelve test functions

F_i	Name	Dimension (D)	Search space	Objectives
1	ZDT3	30	$[0,1]^D$	2
2	ZDT4	10	$[0,1]\times[-5,5]^{D-1}$	2
3	ZDT6	10	$[0,1]^D$	2
4	LZ1	30	$[0,1]^D$	2
5	LZ2	30	$[0,1]\times[-1,1]^{D-1}$	2
6	LZ3	30	$[0,1]\times[-1,1]^{D-1}$	2
7	DTLZ1	10	$[0,1]^D$	3
8	DTLZ2	30	$[0,1]^D$	3
9	DTLZ5	30	$[0,1]^D$	3
10	DTLZ1-5D	10	$[0,1]^D$	5
11	DTLZ2-5D	30	$[0,1]^D$	5
12	DTLZ5-5D	30	$[0,1]^D$	5

D is the dimension of the decision space; DTLZs-5D means the function DTLZs with 5-dimensional objective space. The unimodal functions ZDT1, ZDT2 and the discrete function ZDT5 are omitted in the set, because the evaluated algorithms are continuous and they have similarly good performance on the unimodal functions

first issue is about accuracy, i.e., the distance of the nondominated results to global optima. The other issue is about diversity, i.e., the spread of the final solutions in the Pareto set. Based on these two considerations, three metrics are adopted to evaluate the performance in this paper, generational distance metric (GD) (Bosman and Thierens 2003), and inversed generational distance metric (IGD) (Veldhuizen and Lamont 1998), and hypervolume (HV) (Durillo et al. 2010) defined in the Appendix. Note that a small value of GD and IGD, and a large value of HV are preferred, which indicate the results are accurate and diverse.

We run all the five algorithms 30 times for each benchmark in our simulation. For each run, the number of maximum function evaluations (FEs) is set to 50,000, which is large enough for the convergence of valid algorithms (Chowdhury et al. 2009). The mean values and the standard derivations of GD, IGD, and HV are computed for statistical analysis. The parameters within the compared algorithms are set the same with their original papers, which have been verified with the best performance. For NSGA-II and SPEA2, we use the real-valued representation of the decision variables, and the parameters are set as suggested in Deb et al. (2002a) and Zitzler et al. (2001). The population size N is set to 100, and the probability of crossover p_c is set to 0.8. The probability of mutation is $p_m = 1/D$ (D is the number of decision variables). For crossover, the simulated binary crossover (SBX) operator with $\mu_c = 20$ is used. For mutation, the polynomial mutation operator with $\mu_m = 20$ is used. For MOPSO, the population size and the repository size is set to 100. The inertia weigh is 0.4, and the learn rate is 1.0 (Coello et al. 2004). For GDE3, the population size is 100. The mutation factor is 0.9, and the crossover probability is 0.1 (Kukkonen and Lampinen 2005). For MONFO, the population size N is set to be 100, the number of levels m is set to 10, the learning rate α is 1.5, and the crossover probability Cr is 0.1.

Table 2 The mean results of IGD with different parameters when $m = 10$(30 runs)

α	Cr = 0.1	Cr = 0.3	Cr = 0.5	Cr = 0.7	Cr = 0.9	Cr = 0.1	Cr = 0.3	Cr = 0.5	Cr = 0.7	Cr = 0.9
	F_1					F_2				
0.5	1.90e−03	2.45e−02	1.48e−01	3.88e−01	6.25e−01	2.07e+00	1.65e+00	2.24e+00	2.79e+00	4.03e+00
0.7	1.21e−03	9.90e−03	2.24e−02	4.20e−02	1.46e−01	2.49e−05	1.04e−04	1.16e+00	2.58e+00	3.89e+00
0.9	1.21e−03	1.21e−03	1.21e−03	1.21e−03	7.89e−03	3.54e−06	3.54e−06	3.54e−06	3.03e−05	1.71e+00
1.1	1.21e−03	1.21e−03	1.21e−03	1.21e−03	1.21e−03	3.54e−06	3.54e−06	3.54e−06	7.11e−05	1.65e+00
1.3	1.21e−03	1.21e−03	1.21e−03	1.21e−03	1.21e−03	3.54e−06	3.54e−06	3.54e−06	2.15e−05	5.18e−05
1.5	**1.21e−03**	**1.21e−03**	**1.21e−03**	**1.21e−03**	**1.21e−03**	**3.54e−06**	**3.54e−06**	**3.54e−06**	**3.54e−06**	**3.54e−06**
1.7	1.21e−03	1.21e−03	1.21e−03	1.21e−03	1.21e−03	3.54e−06	3.54e−06	3.54e−06	3.54e−06	3.54e−06
1.9	1.21e−03	1.21e−03	1.21e−03	1.21e−03	1.21e−03	3.54e−06	3.54e−06	3.54e−06	3.54e−06	3.54e−06
	F_3					F_4				
0.5	1.08e−01	5.05e−01	1.07e+00	5.74e+00	1.77e+01	1.90e+00	1.78e+00	2.48e+00	6.39e+00	1.34e+00
0.7	3.69e−02	9.50e−02	2.20e−01	4.48e−01	2.49e−14	3.47e+00	1.08e+00	1.08e+00	3.03e+00	5.09e+00
0.9	2.57e−06	2.57e−06	9.95e−01	2.98e+00	9.95e+00	1.23e−01	1.23e−01	1.23e−01	3.68e−01	1.65e−01
1.1	4.06e−05	2.57e−06	9.86e−03	1.23e−02	3.20e−02	1.23e−01	1.23e−01	1.23e−01	3.03e−01	2.35e−01
1.3	2.57e−06	2.57e−06	2.57e−06	2.57e−06	2.98e+00	1.23e−01	1.23e−01	1.23e−01	6.46e−01	1.65e−01
1.5	**2.57e−06**	**2.57e−06**	**2.57e−06**	7.50e−03	4.18e−02	**1.23e−01**	**1.23e−01**	**1.23e−01**	**1.23e−01**	1.07e+00
1.7	4.18e−05	1.31e−05	1.06e−05	1.35e−02	3.94e−02	1.07e−02	1.23e−01	1.23e−01	1.23e−01	1.23e−01
1.9	2.57e−06	2.57e−06	2.57e−06	2.04e+00	1.11e+01	1.23e−01	1.23e−01	1.23e−01	2.22e−01	2.09e−01
	F_5					F_6				
0.5	1.32e+00	1.92e+00	2.41e+00	3.27e+00	4.06e+00	1.65e+00	1.65e+00	1.98e+00	1.91e+00	2.18e+00
0.7	9.97e−01	1.99e+00	6.99e+00	1.29e+00	2.13e+00	1.74e+00	8.15e+00	3.67e+00	4.13e+00	7.07e+00
0.9	3.55e−01	3.55e−01	1.49e+00	1.65e+00	2.58e+00	8.67e−01	9.86e−01	2.22e+00	4.67e+00	1.62e+00
1.1	2.64e−01	2.64e−01	9.95e−01	1.99e+00	8.96e+00	1.87e−01	1.87e−01	1.87e−01	1.87e−01	2.22e−01
1.3	3.55e−01	3.55e−01	2.64e−01	3.55e−01	3.55e−01	1.11e+00	1.99e−01	8.11e−01	9.86e+00	2.71e+00
1.5	**2.64e−01**	**2.64e−01**	**2.64e−01**	2.84e−01	7.83e+00	**1.87e−01**	**1.87e−01**	**1.87e−01**	**1.87e−01**	1.22e+00
1.7	7.83e+00	2.64e−01	2.64e−01	1.31e+00	1.15e+01	1.87e−01	1.87e−01	1.87e−01	1.87e−01	2.22e−01
1.9	3.55e−01	3.55e−01	3.55e−01	2.64e−01	3.55e−01	4.43e−01	7.40e−01	7.40e−01	3.45e+00	5.95e+00

5.2 Parameter evaluation

There are two main parameters in MONFO, learning rate α and the crossover probability Cr. The effect of α is to control the scale of search region, and Cr is used to control the convergence speed. For users, the two parameters need to be insensitive and problem independent. In this experiment, we have evaluated how α and Cr affect the performance of MONFO. Different settings of α and Cr are tested on the functions F_1, F_2, F_3, F_4, F_5, F_6. And α is set to 0.5, .0.7, 0.9, 1.1, 1.3, 1.5, 1.7 and 1.9. For each learning rate, Cr is set to 0.1, 0.3, 0.5, 0.7, and 0.9 for comparisons. With each pair of parameters, the mean IGD values of final results are listed in Table 2, in which the best mean value are highlighted as **bold**. It can be noticed that appropriate α lies in the scale from 0.7 to 1.7. For α values smaller than 0.7, MONFO is easily getting trapped in the local minimum due to the insufficient search. Furthermore, Cr is robust enough in the scale [0.1, 0.7]. If Cr is large than 0.7, NFO cannot converge to the global optimum regardless to the learning rate. In the above parameter evaluation, we set the contour level $m = 10$.

In the following comparison studies, α and Cr are set to 1.5 and 0.1 in MONFO. We also find that the contour level is less sensitive than Cr and α. In Table 3, we study the effect of using different contour level number m. We test MONFO in all functions with different M values when $\alpha = 1.5$ and $Cr = 0.1$. The contour level is set $M = 2, 3, 4, 5, 10, 20, 25$ respectively. The mean over 30 runs are listed in Table 3. The best results are marked in **bold** for each function. It can be notice that when the contour level m is larger than 5, MONFO can converge to the global optimum on most functions. The optimal value of m is 10, which is used in the following experiments.

5.3 Comparison studies

The mean values and the standard derivations of GD, IGD and HV on each test function are listed in Tables 4, 5 and 6 respectively. The best results of GD, IGD and HV for each function are highlighted in **bold**. The overall performance of accuracy and diversity can be analyzed using GD, IGD and HV metrics. To illustrate the results of MONFO, NSGA-II, SPEA2, MOPSO and GDE3 more clearly, we plot the final results of the best run with the largest HV value in Figs. 3, 4 and 5. They have shown the results on ZDTs, DTLZs and LZs respectively.

For F_1, the Pareto front of ZDT3 consists of several non-contiguous convex parts. The first column of Fig. 3 shows the results of MONFO, NSGA-II, SPEA2, MOPSO and GDE3 on function F_1. It is confirmed that all the evaluated algorithms can solve this function with well-dispersed distribution on the Pareto front.

For F_2, ZDT4 is a multimodal function that contains 21^{10} local optimal fronts. The second column of Fig. 3 shows the final results of each algorithm on F_2. Only MONFO and GDE3 can obtain some of the final results on the global Pareto front, while NSGA-II, SPEA2 and MOPSO cannot obtain any result on the front shown as Fig. 3e, h and k. It is evident that MONFO can solve this multimodal function. In Table 6, the HV metrics of NSGA-II, SPEA2 and MOPSO are all zeros, while MONFO exhibits a comparably large HV with GDE3.

For F_3, ZDT6 has two difficulties caused by the non-uniformity of the search space. First, its Pareto solutions are non-uniformly distributed in the search space, sparsely near the Pareto front and densely away from the Pareto front. Second, the Pareto optimal solutions are also non-uniformly distributed in the search space. The distribution is dense

Table 3 The mean results of IGD with different contour levels when Cr = 0.1 and $\alpha = 1.5$(30 runs)

m	2	3	4	5	10	20	25
F_i							
1	9.12e−03	7.89e−03	6.54e−03	2.36e−03	**1.21e−03**	**1.21e−03**	3.15e−03
2	7.86e−05	7.21e−05	5.18e−05	6.53e−06	**3.54e−06**	4.41e−06	5.17e−06
3	4.53e−04	9.13e−05	7.34e−05	3.12e−06	2.57e−06	**2.51e−06**	4.87e−06
4	2.33e+00	1.37e+00	6.35e−01	1.29e−01	**1.23e−01**	1.26e−01	1.64e−01
5	4.61e+00	1.65e+00	5.32e−01	4.19e−01	**2.64e−01**	2.87e−01	3.91e−01
6	3.87e+01	2.13e+00	3.45e−01	1.91e−01	**1.87e−01**	1.94e−01	2.53e−01
7	9.37e−01	5.82e−01	2.39e−01	**1.54e−01**	**1.54e−01**	1.65e−01	3.12e−01
8	2.33e−02	1.13e−02	1.07e−02	9.84e−03	**1.46e−03**	1.49e−03	1.75e−03
9	7.91e−02	3.52e−02	2.03e−02	1.47e−02	**7.58e−03**	7.67e−03	8.23e−03
10	1.79e−01	9.14e−01	3.83e−01	**1.25e−01**	**1.25e−01**	2.07e−01	6.64e−01
11	4.06e−01	3.76e−01	8.72e−02	6.55e−02	**2.15e−02**	**2.15e−02**	5.03e−02
12	9.87e−01	6.72e−01	5.63e−01	5.63e−01	3.09e−01	**3.02e−01**	4.67e−01

Table 4 The mean and standard derivation of GD on the benchmarks (30 runs)

F_i	NSGA-II Mean±Std. Dev.	SPEA2 Mean±Std. Dev.	MOPSO Mean±Std. Dev.	GDE3 Mean±Std. Dev.	MONFO Mean±Std. Dev.
1	3.60e−03 ± 5.42e−04	2.41e−03 ± 4.45e−19	**9.03e−04 ± 1.17e−04**	2.92e−03 ± 3.68e−03	1.42e−03 ± 2.92e−03
2	2.65e+01 ± 9.00e+00	7.54e−01 ± 1.14e−16	6.27e+00 ± 3.21e+00	**1.21e−05 ± 4.90e−02**	6.02e−05 ± 6.34e−05
3	4.28e−01 ± 6.15e−01	1.03e+00 ± 2.28e−16	7.92e−02 ± 1.40e−01	2.46e−03 ± 2.94e−03	**1.71e−03 ± 2.54e−03**
4	5.92e−02 ± 2.36e−03	**5.79e−03 ± 1.78e−18**	1.40e−02 ± 2.25e−03	1.17e−02 ± 9.95e−04	8.11e−03 ± 2.16e−03
5	1.50e−01 ± 5.34e−02	**9.89e−03 ± 1.78e−18**	8.37e−02 ± 5.05e−02	6.88e−02 ± 3.78e−03	1.71e−02 ± 3.25e−03
6	7.81e−02 ± 3.11e−02	**7.75e−03 ± 2.67e−18**	1.75e−02 ± 2.56e−03	1.073−02 ± 1.23e−03	9.91e−03 ± 3.29e−03
7	2.08e+01 ± 7.14e+00	2.11e+00 ± 7.31e−01	5.76e+01 ± 3.77e+00	**2.40e−01 ± 2.90e−01**	1.98+00 ± 4.20e−01
8	1.91e−01 ± 5.78e−02	3.42e−02 ± 1.42e−17	3.94e−01 ± 2.08e−01	**1.79e−03 ± 8.97e−05**	9.32e−03 ± 1.53e−02
9	**7.80e−03 ± 6.98e−04**	2.33e−02 ± 7.12e−18	6.78e−02 ± 7.88e−02	1.26e−02 ± 1.11e−04	3.43e−02 ± 2.91e−04
10	3.64e+01 ± 8.13e+00	3.64e+02 ± 1.17e−13	1.27e+02 ± 1.77e+01	7.63e+00 ± 7.45e−01	**2.07e+00 ± 6.11e−01**
11	1.06e+02 ± 4.03e+01	5.82e+00 ± 1.82e−15	5.61e+00 ± 2.61e+00	8.24e−01 ± 1.90e−01	**1.42e−01 ± 2.18e−01**
12	3.77e+02 ± 1.40e+01	2.87e+01 ± 3.65e−15	1.22e+02 ± 2.13e+01	4.55e+02 ± 3.74e+01	**2.77e+01 ± 2.63e+00**

when the first objective value near 1. Figure 3 plots the final results of F_3 in the third column. From these subfigures, we can see that MONFO, NSGA-II and GDE3 are capable of solving this function, whilst SPEA2 and MOPSO cannot solve the function. The statistical results in Tables 4, 5 and 6 agree with the same conclusion. SPEA2 has the zero HV, and MONFO has the smallest GD, IGD on F_3. Overall, it is clear that MONFO can deliver remarkable performance on multimodal and non-uniformly functions.

For the three LZs functions F_4, F_5, F_6 , the final results of the evaluated algorithms are plotted in Fig. 4. For F_4, the simulation results show that all the five algorithms can find certain results on the Pareto front with almost the same performance shown in the first column of Fig. 4. SPEA2 exhibits the smallest GD metric; MONFO shows the smallest IGD and the largest HV. It is worth noting that F_5 is a difficult function that can only be solved by using MONFO. For F_5, MONFO can find better-dispersed results than other algorithms shown in Fig. 4b, e, h, k and n. For F_6, all algorithms have almost the same performance, but MONFO can obtain the results with the smallest IGD on F_6. Generally, MONFO can consistently deliver well-dispersed results on the series of LZs functions.

For the three-objective DTLZs functions F_7, F_8, F_9, Fig. 5 shows the results obtained by MONFO, NSGA-II, SPEA2, MOPSO, and GDE3. It is clear that MONFO can deliver a comparably performance among all the evaluated algorithms on DTLZs. For F_7, its front is a linear hyperplane. It is worth noting that the difficulty of this function is that the search space contains $(11^{D-2}-1)$ local Pareto-optimal fronts. The first column of Fig. 5 shows that the results on F_7. MONFO and GDE3 can find certain Pareto

Table 5 The mean and standard derivation of IGD on the benchmarks (30 runs)

F_i	NSGA-II Mean±Std. Dev.	SPEA2 Mean±Std. Dev.	MOPSO Mean±Std. Dev.	GDE3 Mean±Std. Dev.	MONFO Mean±Std. Dev.
1	2.92e−02 ± 3.29e−03	1.70e−02 ± 7.12e−18	6.13e−03 ± 7.54e−04	1.76e−03 ± 2.00e−03	**1.21e−03 ± 2.34e−03**
2	5.36e+00 ± 8.95e−01	8.65e−01 ± 1.14e−16	2.70e−06 ± 6.28e−01	**2.47e−06 ± 1.52e−07**	3.54e−06 ± 2.46e−06
3	2.97e−04 ± 6.44e−04	9.57e−04 ± 1.11e−19	9.00e−06 ± 2.37e−05	2.60e−06 ± 1.38e−07	**2.57e−06 ± 2.19e−07**
4	4.90e−01 ± 6.54e−02	2.27e−01 ± 8.54e−17	3.51e−01 ± 2.54e−02	3.27e−01 ± 1.07e−02	**1.23e−01 ± 1.35e−02**
5	9.70e−01 ± 1.46e−01	2.30e−01 ± 2.85e−17	8.67e−01 ± 3.12e−01	**2.18e−01 ± 4.50e−02**	2.64e−01 ± 1.83e−02
6	6.75e−01 ± 9.41e−02	2.35e−01 ± 2.93e−02	3.53e−01 ± 1.90e−02	3.23e−01 ± 1.59e−02	**1.87e−01 ± 1.56e−02**
7	7.28e−01 ± 1.28e−01	2.45e−01 ± 2.85e−17	5.03e−01 ± 4.17e−02	**5.48e−02 ± 5.62e−02**	1.54e−01 ± 1.63e−02
8	4.25e−02 ± 6.89e−03	1.85e−02 ± 7.12e−18	6.06e−02 ± 1.69e−02	**1.28e−03 ± 7.09e−05**	1.46e−03 ± 9.85e−03
9	**7.10e−03 ± 3.38e−03**	1.44e−02 ± 1.78e−18	2.17e−02 ± 1.33e−02	1.26e−02 ± 8.48e−05	7.58e−03 ± 3.21e−03
10	9.50e−01 ± 1.15e−01	1.01e+00 ± 2.28e−16	5.13e−01 ± 5.55e−02	8.07e−02 ± 5.95e−02	**1.25e−01 ± 1.19e−03**
11	1.09e+00 ± 2.08e−01	2.40e−01 ± 5.70e−17	2.26e−01 ± 5.66e−02	9.22e−02 ± 8.97e−03	**2.15e−02 ± 3.02e−03**
12	2.16e+00 ± 6.83e−02	5.07e−01 ± 2.28e−16	1.03e+00 ± 8.61e−02	2.03e+00 ± 8.17e−02	**3.09e−01 ± 3.08e−02**

Table 6 The mean and standard derivation of HV on the benchmarks (30 runs)

F_i	NSGA-II Mean±Std. Dev.	SPEA2 Mean±Std. Dev.	MOPSO Mean±Std. Dev.	GDE3 Mean±Std. Dev.	MONFO Mean±Std. Dev.
1	3.77−01 ± 6.47e−03	3.70e−01 ± 1.71e−16	2.51e−01 ± 3.95e−03	2.52e−01 ± 5.73e−03	**3.90e−01 ± 1.01e−02**
2	0 ± 0	0 ± 0	0 ± 0	**2.57e−01 ± 4.28e−03**	2.01e−01 ± 3.44e−02
3	3.49e−02 ± 2.79e−02	0 ± 0	0 ± 0	**8.05e−02 ± 1.09e−03**	6.91e−02 ± 3.50e−03
4	1.91e−01 ± 1.49e−02	2.40e−01 ± 0	2.57e−01 ± 6.56e−03	2.49e−01 ± 5.08e−03	**2.85e−01 ± 9.87e−03**
5	1.52e−01 ± 3.07e−02	**2.59e−01 ± 1.05e−02**	1.98e−01 ± 5.15e−02	2.54e−01 ± 2.51e−02	2.54e−01 ± 1.16e−02
6	1.86e−01 ± 2.02e−02	**2.80e−01 ± 4.50e−03**	2.66−01 ± 5.81e−03	2.66e−01 ± 1.21e−02	2.76e−01 ± 7.90e−03
7	0 ± 0	0 ± 0	0 ± 0	**1.13e−02 ± 1.21e−03**	1.73e−03 ± 1.30e−02
8	1.09e−02 ± 6.69e−03	4.08e−02 ± 2.14e−017	4.29e−03 ± 5.80e−03	**5.45e−02 ± 1.01e−03**	4.25e−02 ± 5.97e−03
9	7.22e−03 ± 6.43e−04	5.11e−03 ± 0	4.13e−03 ± 2.41e−03	7.91e−03 ± 2.89e−04	**8.27e−03 ± 6.99e−04**
10	0 ± 0	0 ± 0	0 ± 0	6.99e−04 ± 7.66e−04	**1.21e−03 ± 4.59e−04**
11	0 ± 0	0 ± 0	1.36e−06 ± 6.09e−06	1.54e−03 ± 7.10e−04	**3.37e−02 ± 5.97e−03**
12	0 ± 0	0 ± 0	0 ± 0	0 ± 0	**9.37e−06 ± 9.59e−06**

optima, while the other algorithms cannot solve the problem completely as shown in the first column of Fig. 5. For F_8, its Pareto front is a unit sphere in the 3-D objective space. For F_9, its difficulty is different from the above two functions that its Pareto front is not a plane but a continuous curve. The results on F_8, F_9 are illustrated in the second and third columns of Fig. 5. All the algorithms show they can find well-dispersed results, but the results of GDE3 and MONFO appear to be closer to the Pareto optima.

Although we cannot visualize the results on high dimensional objective functions, it is clear that MONFO is the best in Tables 4, 5 and 6. For the F_{10}, F_{11} and F_{12}, MONFO outperforms others dominantly, with the best GD, IGD and HV metrics. The HV values of NSGA-II and SPEA2 on these functions are all zeros, which demonstrate that they cannot find any result within the referred area (that dominates the nadir point). MONFO can find some good results in the referred area with the largest HV.

The two-side Wilcoxon's rank-sum tests can be conducted to clarify the statistical differences of GD, IGD and HV between MONFO and other algorithms NSGA-II, SPEA2, MOPSO, and GDE3. Table 7 shows the results of Wilcoxon's test at the significance level of 5% between the performance of MONFO and other algorithms. The notation "+" indicates that the performance of MONFO is significantly better than the compared algorithms; the notation "−" indicates that the MONFO's result is significantly worse than the compared algorithm; "≈" implies that two compared algorithms are not statistically different with (i.e., statistically insignificant) each other. For the GD metric, the numbers of functions, on which MONFO is significantly better than, worse than, and similar with NSGA-II, are 11/1/0; the numbers of functions, on which MONFO is significantly better than, worse than, and similar with SPEA2, MOPSO and GDE3, are 8/3/1, 11/1/0 and 8/3/1, respectively. It can be noticed that MONFO is significant better than the compared algorithms on most functions. For the IGD and HV, the same conclusions can be obtained according to Table 7. Generally, the performance of MONFO statistically better than the other four algorithms on most evaluated functions in the respects of GD, IGD and HV. Besides the Wilcoxon's tests, one-side t-tests at significance level of 5% are also conducted between MONFO and the compared algorithms. The results of t-tests are the same as Table 7.

In all, MONFO delivers the minimum GD on four different functions F_3, F_{10}, F_{11} and F_{12} in Table 4, while SPEA2, GDE3, NSGA-II and MOPSO deliver the minimal GD on three, three, one and one functions respectively. From IGD results in Table 5, MONFO delivers the minimal IGD on seven functions F_1, F_3, F_4, F_6, F_{10}, F_{11} and F_{12}, while GDE3, NSGA-II, SPEA2 and MOPSO deliver the minimal IGD on four, one ,zero and zero functions respectively. From HV results in Table 6, MONFO exhibits its superiority in both accuracy and diversity. SPEA2 delivers the best HV on the functions F_5 and F_6; GDE3 delivers the best HV on four different functions F_2, F_3, F_8 and F_9; MONFO delivers the maximal HV on the other six functions.

Under the overall evaluations of GD, IGD and HV, it can be concluded that MONFO can obtain excellent performances in terms of accuracy and diversity, compared with NSGA-II SPEA2, MOPSO, and GDE3. MONFO can obtain the significantly better results of GD, IGD and HV, especially for multimodal functions with the high dimensional objective space. For some real-world problems with

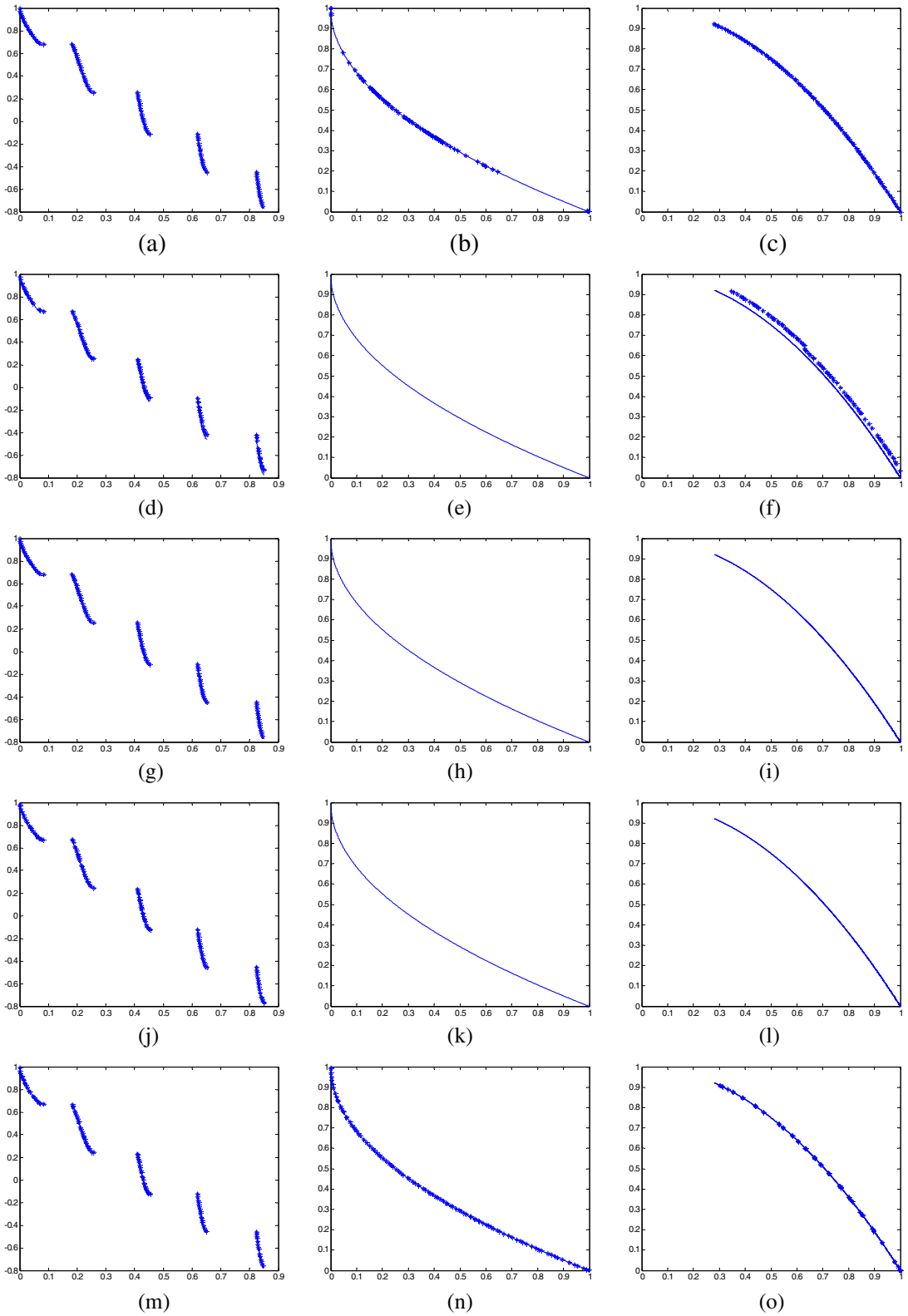

Fig. 3 Final results of the run with the largest HV value on ZDTs. (**a**), (**d**), (**g**), (**j**) and (**m**) in the first column are the results on ZDT3 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**b**), (**e**), (**h**), (**k**) and (**n**) in the second column are the results on ZDT4 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**c**), (**f**), (**i**), (**l**) and (**o**) in the third column are the results on ZDT6 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively. In each subfigure, the *solid line* denotes Pareto front of each function

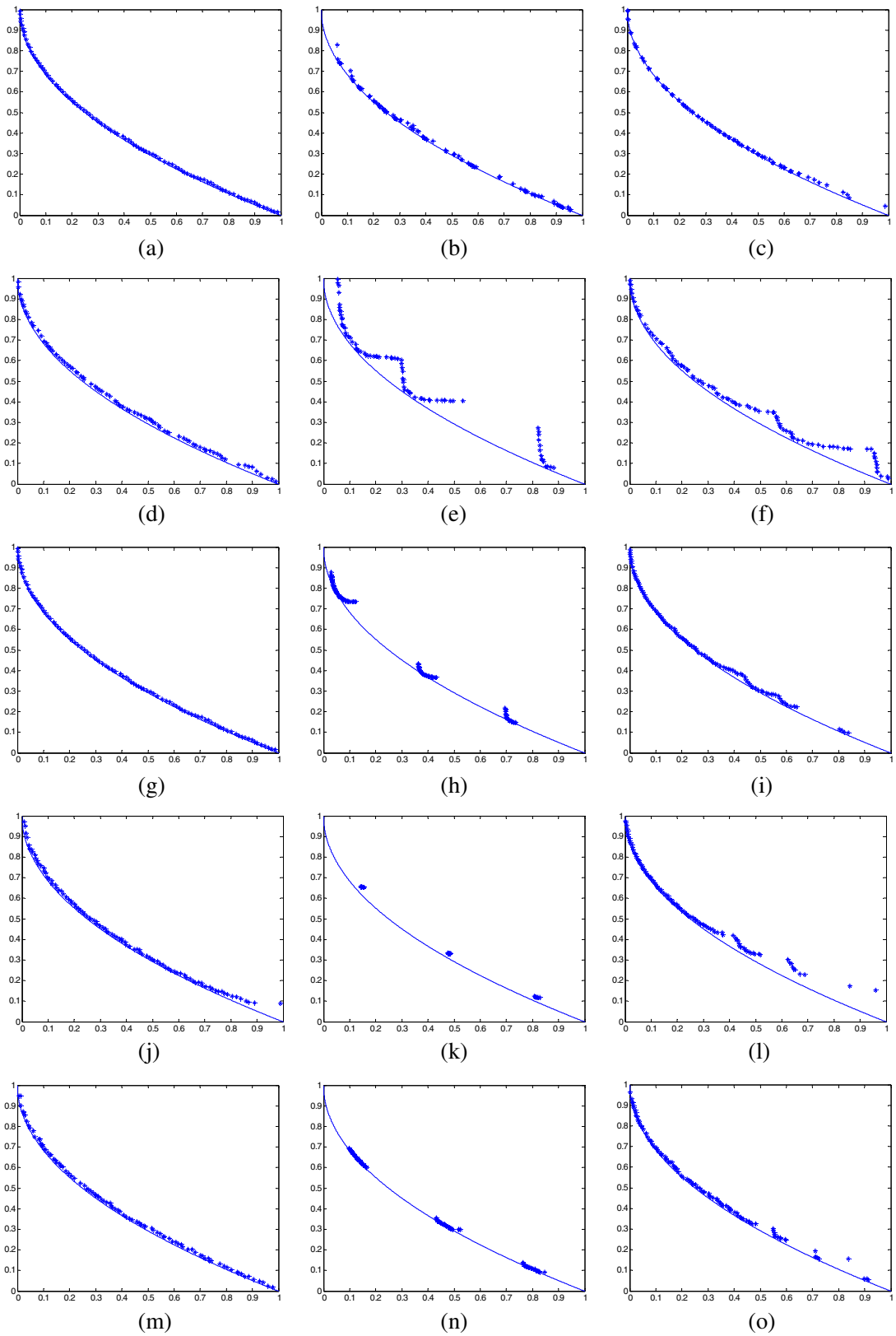

Fig. 4 Final results of the run with the largest HV value on LZs. (**a**), (**d**), (**g**), (**j**) and (**m**) in the first column are the results on LZ1 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**b**), (**e**), (**h**), (**k**) and (**n**) in the second column are the results on LZ2 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**c**), (**f**), (**i**), (**l**) and (**o**) in the third column are the results on LZ3 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively. In each subfigure, the *solid line* denotes Pareto front of each function

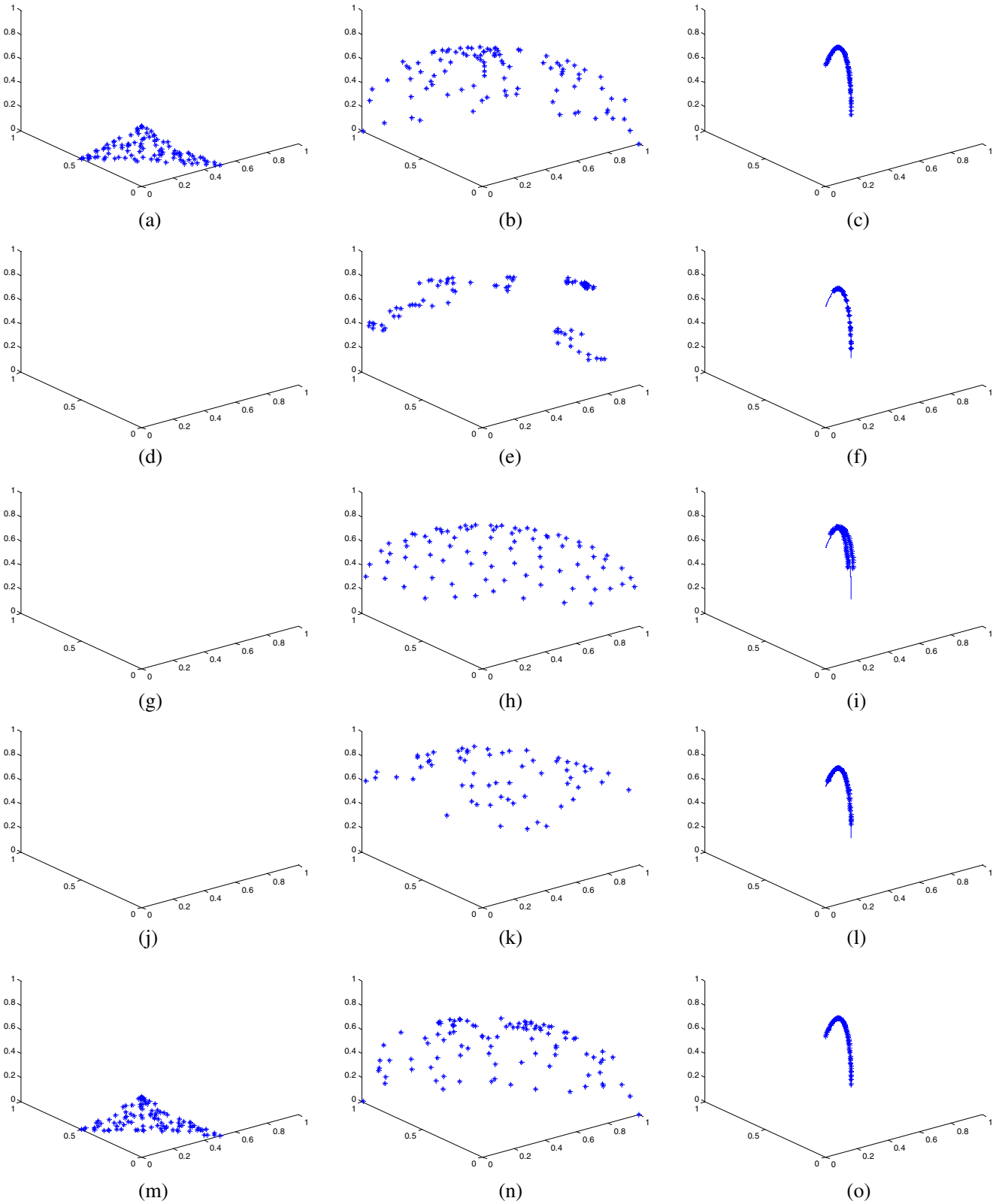

Fig. 5 Final results of the run with the largest HV value on DTLZs. (**a**), (**d**), (**g**), (**j**) and (**m**) in the first column are the results on DTLZ1 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**b**), (**e**), (**h**), (**k**) and (**n**) in the second column are the results on DTLZ2 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively; (**c**), (**f**), (**i**), (**l**) and (**o**) in the third column are the results on DTLZ5 function for MONFO, NSGA-II, SPEA2, MOPSO and GDE3, respectively

Table 7 Results of statistical tests on the performance metrics

F_i	GD				IGD				HV			
	NSGA-II	SPEA2	MOPSO	GDE3	NSGA-II	SPEA2	MOPSO	GDE3	NSGA-II	SPEA2	MOPSO	GDE3
1	+	+	−	+	+	+	+	+	+	+	+	+
2	+	+	+	≈	+	+	−	≈	+	+	+	−
3	+	+	+	+	+	+	+	≈	+	+	+	−
4	+	−	+	+	+	+	+	+	+	+	+	+
5	+	−	+	+	+	−	+	−	+	−	+	+
6	+	≈	+	+	+	+	+	+	+	−	+	+
7	+	+	+	−	+	+	+	−	+	+	+	−
8	+	+	+	−	+	+	+	−	+	+	+	−
9	−	−	+	−	−	+	+	+	+	+	+	+
10	+	+	+	+	+	+	+	+	+	+	+	+
11	+	+	+	+	+	+	+	+	+	+	+	+
12	+	+	+	+	+	+	+	+	+	+	+	+
Total(B/W/I)	11/1/0	8/3/1	11/1/0	8/3/1	11/1/0	11/1/0	11/1/0	7/3/2	12/0/0	10/2/0	12/0/0	8/4/0

Wilcoxon's tests and t-tests at the significance level of 5% are conducted between the performance of MONFO and those of other algorithms. "+" means that MONFO is significant better than the compared algorithms; "−" means that MONFO is significant worse than the compared algorithms; "≈" mean that MONFO is statistically insignificant with the compared algorithms. In the last column, "B/W/I" means the number of functions that MONFO statistical better than, worse than, and insignificant with each compared algorithm

constraints, MONFO can still handle them after defining constraints-dominance relationships similarly with GDE3, or using certain problem-based operations to repair the violating solutions feasibly, which is out of the discussion of this paper.

6 Conclusion

A new local algorithm called MONFO is proposed to solve MOOPs based on the neighborhood cooperation. This study shows that the proposed local search algorithm can achieve comparable performance with other global search algorithms, such as NSGA-II, SPEA2, MOPSO and GDE3. Especially for some multimodal functions with high dimensional objectives, MONFO can outperform these evaluated algorithms in terms of accuracy and diversity.

The focus of this paper is not discussing which kind of search, locally or globally, should be employed. This paper only emphasizes on the new basic algorithm framework and the comparison study with some other major frameworks. It has been demonstrated that the cooperative neighborhood field can be used to generate an optimization algorithm directly and independently. The neighborhood field model (NFM) utilizes the dynamics from neighbors directly, which can provide an efficient searching mechanism in MONFO. The presented results have shown that the local cooperation can directly enable the algorithm to converge with high accuracy and diversity.

Appendix

(1) Generational distance (GD) (Chowdhury et al. 2009): The concept of generational distance was introduced as a way of estimating the distance between the obtained non-dominated solutions and the global optima in the objective space. GD is defined as

$$GD = \frac{\sum_{i=1}^{N} d_{\min}\left(F\left(x_i\right), F\left(Xr\right)\right)}{N}, \tag{13}$$

where N is the size of non-dominated set, Xr is the referred Pareto set. $d_{\min}(,)$ is the minimum of Euclidean distances from the ith individual $\boldsymbol{x}_i$ to all points in the Pareto set Xr (in the objective space). It is clear that when GD equals to zero all the solutions are in the global Pareto set, and a large GD means the solutions are far from the global Pareto optima. The GD metric can evaluate the accuracy performance.

(2) Inverse Generational distance (IGD) (Veldhuizen and Lamont 1998): The Inverse Generational distance (IGD) is similar with GD to compare the solution's quality in the decision space. IGD inversely compares each referred Pareto optima's distance away from the obtained non-dominated solutions. IGD is defined as

$$GD = \frac{\sum_{i=1}^{Nr} d_{\min}\left(F\left(Xr_i\right), F\left(X\right),\right)}{Nr}, \tag{14}$$

where Xr is referred Pareto set, Nr is the size of Xr, X is the obtained set of non-dominated solutions. $d_{\min}(,)$ is the minimum of Euclidean distances from the ith referred solution Xr_i to all solutions in the obtained set X (in the objective space). A low IGD is preferred for the algorithms, which means they have the good accuracy performance.

(3) Hypervolume (HV) (Durillo et al. 2010): The hypervolume (HV) computes the volume (in objective space) covered by members in a non-dominated set X. For the minimization multiobjective optimization problems, HV measures the total dominated hypervolume between each solution $\boldsymbol{x}_i \in X$ and an inferior reference point W, which should be dominated by all Pareto-optimal solutions. The volume of hypercube between $\boldsymbol{x}_i$ and W is denoted as V_i. The hypervolume is calculated as,

$$HV = \frac{volume\left(\bigcup_{i=1}^{N} V_i\right)}{N}. \tag{15}$$

To evaluate algorithms with different population sizes, the hypervolume metric is usually defined as an average of the total volume to the size. Hence, larger HV values are more desirable, which indicates the better performance of the convergence towards the Pareto front, as well as the better performance of the diversity. For calculating HV, the reference point is chosen as the nadir point. Only the obtained results can dominate the nadir point have positive volumes, otherwise they have volumes with zero.

References

Abbass HA, Sarker R (2002) The Pareto differential evolution algorithm. Proc Int J Artif Intell Tools 11(4):531–552

Adra SF, Griffin I, Fleming PJ (2005) Hybrid multiobjective genetic algorithm with a new adaptive local search process. In: Proc. Genetic Evol. Comput. Conf. (GECCO '05), vol 1. New York, USA, pp 1009–1010

Bosman P, Thierens D (2003) The balance between proximity and diversity in multi-objective evolutionary algorithms. IEEE Trans Evol Comput 7(2):174–188

Brown M, Smith RE (2003) Effective use of directional information in multi-objective evolutionary computation. In: Proc. of the 2003 international conference on Genetic and Evolutionary Computation (GECCO'03). Chicago, IL, USA, pp 778–789

Chowdhury S, Dulikravich GS, Moral RJ (2009) Modified predator–prey algorithm for constrained and unconstrained multi-objective optimization. Int J Math Model Num Optim 1(1–2):1–38

Coello CAC, Lamont GB (2004) Application of multi-objective evolutionary algorithms (advances in natural computation—vol 1). World Scientific Publishing Co. Pte. Inc., Singapore

Coello CAC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 8(3):256–279

Czyzak P, Jaszkiewicz A (1998) Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. J Multi-Crit Decis Anal 7(1):34–47

Deb K (1999) Multi-objective genetic algorithms: problem difficulties and construction of test problems. Evol Comput 7(3):205–230

Deb K, Agarwal S, Pratap A, Meyarivan T (2002a) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Deb K, Thiele L, Laumanns M, Zitzler E (2002b) Scalable multi-objective optimization test problems. In: Proc. of IEEE Congress on Evolutionary Computation (CEC), vol 1. Hawaii, USA, pp 825–830

Durillo JJ, Nebro AJ, Coello CAC, Garciìa-Nieto J, Luna F, Alba E (2010) A study of multiobjective metaheuristics when solving parameter scalable problems. IEEE Trans Evol Comput 14(4):618–635

Eberhart R, Shi Y (2001) Particle swarm optimization: developments, applications and resources. In: Proc. of IEEE Congress on Evolutionary Computation (CEC), vol 1. Seoul, Korea, pp 81–86

Fliege J, Svaiter BF (2000) Steepest descent methods for multicriteria optimization. Math Method Oper Res 51(3):479–494

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning, 1st edn. Addison-Wesley Longman Publishing Co. Inc., Boston

Horn J, Nafpliotis N, Goldberg DE (1994) A Niched Pareto genetic algorithm for multiobjective optimization. In: Proc. of the 1st conference on evolutionary computation, IEEE world congress on computational intelligence, vol 1, pp 82–87

Hu X, Eberhart R (2002) Multiobjective optimization using dynamic neighborhood particles warm optimization. In: Proc. of IEEE Congress on Evolutionary Computation (CEC), vol 2. Honolulu, HI, pp 1677–1681

Hu X, Huang Z, Wang Z (2003) Hybridization of the multiobjective evolutionary algorithms and the gradient based algorithms. In: Proc. IEEE congr. evol. comput., vol 2. Canberra, Australia, pp 870–877

Jin Y (2006) Multi-objective machine learning. Springer, Berlin Heidelberg

Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. Int J Robot Res 5(1):90–98

Knowles JD, Corne D (2000) Approximating the nondominated front using the pareto archived evolution strategy. Evol Comput 8(2):149–172

Kukkonen S, Lampinen J (2005) GDE3: the third evolution step of generalized differential evolution. In: Proc. of IEEE Congress on Evolutionary Computation (CEC), vol 1. Edinburgh, Scotland, pp 443–450

Lara A, Sanchez G, Coello CAC, Schutze O (2010) HCS: a new local search strategy for memetic multiobjective evolutionary algorithms. IEEE Trans Evol Comput 14(1):112–132

Li H, Zhang Q (2009) Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Trans Evol Comput 13(2):284–302

Molina D, Lozan M, Carcía-Martínez C, Herrera F (2010) Memetic algorithms for continuous optimisation based on local search chains. Evol Comput 18(1):27–63

Parsopoulos KE, Vrahatis MN (2002) Particle swarm optimization method in multiobjective problems. In: Proc. of ACM Symptom Applied Computing (SAC). Madrid, Spain, pp 603–607

Schaffer JD (1984) Some experiments in machine learning using vector evaluated genetic algorithms (Artificial Intelligence, Optimization, Adaptation, Pattern Recognition). PhD Dissertation, Vanderbilt University, Nashville, TN, USA

Shi Y, Eberhart RC (1998) A modified particle swarm optimizer. In: Proc. of 1998 IEEE Congress on Evolutionary Computation 1998 (CEC1998). AK, pp 69–73

Shukla PK (2007) Gradient based stochastic mutation operators in evolutionary multi-objective optimization. In: Proc. of the 8th Int. Conference on Adaptive and Natural Computing Algorithms (ICANNGA '07). Warsaw, Poland, pp 58–66

Srinivas N, Deb K (1994) Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol Comput 2(3):221–248

Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous space. J Global Optim 11(4):341–359

Tan KC, Khor EF, Lee TH (2005) Performance assessment and comparison of MOEAs. In: Jain L, Wu X (eds) Multiobjective evolutionary algorithms and applications, advanced information and knowledge processing. Springer, London, pp 125–149

Veldhuizen DV, Lamont G (1998) Multiobjective evolutionary algorithm research: a history and analysis. Air Force Inst. Technol., Dayton, OH, Tech. Rep. TR-98-03

Veldhuizen DAV, Lamont GB (2000) Multiobjective evolutionary algorithms: analyzing the state-of-the-art. Evol Comput 8(2):125–147

Wanner EF, Guimarães FG, Takahashi RHC, Fleming PJ (2008) Local search with quadratic approximations into memetic algorithms for optimization with multiple criteria. Evol Comput 16(2):185–224

Xu L, Chow TWS (2010) Self-organizing potential field network: a new optimization algorithm. IEEE Trans Neural Netw 21(9):1482–1495

Xue F, Sanderson AC, Graves RJ (2003) Pareto-based multi-objective differential evolution. In: Proc. of IEEE Congress on Evolutionary Computation (CEC), vol 2. Canberra, Australia, pp 862–869

Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and applications. PhD thesis, Swiss Federal Institute of Technology Zurich, Switzerland

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195

Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm. In: Proc. of EUROGEN 2001. Evolutionary methods for design, optimization and control with applications to industrial problems. Athens, Greece, pp 95–100