Self-Adaptive Multi-Objective Differential Evolution with Direction Information Provided by Archived Inferior Solutions

Jingqiao Zhang and Arthur C. Sanderson

Abstract: We propose a new self-adaptive differential evolution algorithm for multi-objective optimization problems. To address the challenges in multi-objective optimization, we introduce an archive to store recently explored inferior solutions whose difference with the current population is utilized as direction information about the optimum, and also consider a fairness measure in calculating crowding distances to prefer the solutions whose distances to nearest neighbors are large and close to be uniform. As a result, the obtained solutions can spread well over the computed non-dominated front and the front can be moved fast toward the Pareto-optimal front. In addition, the control parameters of the algorithm are adjusted in a self-adaptive manner, avoiding parameter tuning for problems of different characteristics. The proposed algorithm, named JADE2, achieves better or at least competitive results compared to NSGA-II and GDE3 for a set of twenty-two benchmark problems.

I. Introduction

MULTI-OBJECTIVE optimization (MOO) exists everywhere in real-world applications such as engineering, financial, and scientific applications [1], because the outcome is directly linked to cost, profit and/or many other criteria that have heavy impacts on performance, safety, environment etc. It is difficult to provide an ultimate comparison among different outcomes via only one dimension, as the involved multiple criteria/objectives are generally competing and non-commensurable. For example, a financial manager needs to take both return and risk into consideration when making an investment decision; an air traffic controller needs to consider both the reduction of system-level airspace congestion and the satisfaction of different stakeholders' preferences.

The presence of multiple objectives in the decision making of real world applications introduces extra challenges to researchers. More efforts are required for these problems beyond traditional techniques like linear and nonlinear programming as well as single-objective evolutionary algorithms. According to the interaction with decision making process, the multi-objective optimization can be classified as follows [2]:

1) Priori preference articulation: The multiple objectives are linearly or non-linearly aggregated into a scalar function based on priori preference structure from decision makers. This transforms the problem into a single-objective problem prior to optimization.
2) Progressive preference articulation: The optimization is intertwined with decision making process when partial preference information is available.
3) Posteriori preference articulation: A family of tradeoff solutions is found before a decision is made to determine the best solution.

The aggregation method in priori preference articulation is mainly applied in the early development of multi-objective evolutionary algorithms [3]. While the progressive preference articulation is also a current research interest [4], the posteriori preference have been gaining significant attention from researchers in proposing multi-objective evolutionary algorithms (MOEAs) to search for tradeoff solutions [5−10].

There are two key issues in the design of multi-objective evolutionary algorithms: to maximize the diversity and spread of solutions, and to increase the speed of convergence to the Pareto-optimal front. In the literature, the first issue is addressed by various diversity maintenance mechanisms that are mainly based on the estimation of crowding density of solutions. For example, two commonly used crowding estimation techniques are proposed in SPEA2 [8] and NSGA-II [10]. In SPEA2, a decreasing function of the distance to the *k*-th nearest neighbor is used to estimate the crowding density of a solution. In NSGA-II, a crowding distance is calculated as the summation of distances of two points on either side of a solution along each dimension in the objective space. In both methods, solutions with smaller crowding distances or higher crowding densities are considered inferior, and thus are more likely pruned from the population to improve the diversity.

One method to address the second issue is parameter tuning or parameter control. The former requires tedious trial and error, while the latter is "a very interesting topic that very few researchers have addressed in the specialized literature" of MOEAs [11]. A second method is to integrate the promising features in existing MOEAs into recently developed single objective evolutionary algorithms (SOEAs) that have shown great success. However, the reason for the success of SOEAs may become invalid in the case of multi-objective optimization. For example, in SOEAs, it has been shown that the best solutions carry direction information towards the optimum and thus can be utilized to speed up convergence. This advantage becomes invalid or less significant in multi-objective optimization, because all or most solutions in the population may become non-dominated from one another after only a few

Jingqiao Zhang and Arthur C. Sanderson are with the Center for Automation Technologies and Systems, Rensselaer Polytechnic Institute, Troy, NY 12180, USA. (Emails: zhangj14@rpi.edu; sandea@rpi.edu).

Funding for this work was provided in part by grant number IIS-0329837 from the National Science Foundation. This work is also supported in part by the Center for Automation Technologies and Systems (CATS) under a block grant from the New York State Office of Science, Technology, and Academic Research (NYSTAR).

generations, especially when the dimension of the objective space is very high. In this case, as illustrated later in Fig. 1, the best solutions are identified according to their secondary metric of crowding density, instead of the Pareto dominance rank, and thus do not necessarily indicate the progress direction to the Pareto-optimal front.

Addressing the above issues, we propose in this paper a self-adaptive multi-objective differential evolution algorithm. Its control parameters are adjusted in a self-adaptive manner, and thus avoid users' interaction before and during the optimization process and avoid degraded performance due to inappropriate parameter settings. An external archive is introduced to store recently explored inferior solutions whose difference with the current population indicates a promising direction towards the Pareto-optimal front. It is worth noting that this is different from the archives used in many existing MOEAs to store the best-so-far non-dominated solutions. In addition, we calculate the crowding distance according to a fairness measure to prefer the solutions whose distances to nearest neighbors are large and close to be uniform. This is helpful to better spread solutions over the computed non-dominated front.

The rest of this paper is organized as follows. Section II briefly reviews several classic multi-objective evolutionary algorithms and introduces existing studies of differential evolution in multi-objective optimization. Section III describes the basic procedure of differential evolution and then introduces a self-adaptive DE algorithm JADE [12]. Section IV analyzes the challenges in multi-objective optimization and proposes new methods of estimating crowding density and extracting direction information from archived inferior solutions. These features, together with the self-adaptive parameter control in JADE, compose the basic operations in JADE2 for multi-objective optimization. Simulation results are presented in Section V to demonstrate the performance of JADE2. Finally, concluding remarks are presented in Section VI.

II. Multi-Objective Evolutionary Algorithms

This section briefly reviews several classic Pareto-dominance based multi-objective evolutionary algorithms and the applications of differential evolution in this area. It provides a convenient way to compare the diversity maintenance mechanisms and archive techniques in these algorithms with those proposed in JADE2. Interested readers are referred to [3, 5] for comprehensive reviews of MOEAs.

A. PAES

Knowles and Corne [6] proposed the Pareto archived evolution strategy (PAES) with a reference archive of non-dominated solutions found in the evolution. The simplest form of PAES employs (1+1) evolution strategy (ES) and uses a mutation operator for local search. The archive approximates the Pareto front and identifies the fitness of current and potential solutions. In addition, a map of grid is applied in the algorithm to maintain diversity of the archive.

B. SPEA2

The SPEA2 [8] is an improved version of SPEA, the strength Pareto evolutionary algorithm (SPEA) [7]. Both the archive and population in the algorithm are assigned a fitness based on the strength and density estimation. The strength of an individual is denoted as the number of individuals that dominates it, while the density estimation is based on the *k*-th nearest neighbor method [13] where the density of any solution is a decreasing function of the distance to the *k*-th nearest neighbor. A truncation method based on the density is applied to keep the archive at a fixed size.

C. PESA

The Pareto envelope-based selection algorithm (PESA) [9] is motivated from SPEA and PAES. It uses an external population to store the computed Pareto front and an internal population to generate new candidate solutions. The PESA maintains a hyper grid based scheme to keep track of the degree of crowding in different regions of the archive, which is applied to maintain the diversity of external population and to select the internal population from the external population.

D. NSGA and NSGA-II

Srinivas and Deb [14] developed an approach called non-dominated sorting genetic algorithm (NSGA). The fitness assignment is carried out in several steps. In each step, the solutions constituting a non-dominated front are assigned the same dummy fitness value. Then, these solutions are ignored in the following step and the next non-dominated front is extracted. This procedure repeats until all individuals in the population are classified. In [14], this fitness assignment method was combined with a stochastic remainder selection.

In [10], Deb et al proposed NSGA-II as an improved version of NSGA. It uses elitism and a comparison operator that ranks the population based on Pareto dominance ranking and crowding density at a lexicographic order: In the case of a tie in dominance rank, the individual with a lower density succeeds in the selection. The dominance ranking has the same meaning as in NSGA but is calculated in a more efficient manner. The crowding distance of a solution is calculated as the sum of distances to two adjacent points on its either side along each dimension in the objective space. The comparison operator makes the NSGA-II considerably faster than NSGA while producing very good results.

E. Differential Evolution Based MOEAs

As a recent development of evolutionary algorithms, differential evolution has shown its success in various single objective optimization problems. The extensions of DE to multi-objective optimization, such as PDE [15, 16], MODE [17], DEMO [18], and GDE3 [19], and DEMORS [20], incorporate successful features in existing MOEAs and introduce various new operations suitable to multi-objective optimization.

In [21], Zielinski and Laur compared a set of DE variants for multi-objective optimization, where (1) we *may* or *may not* conduct a one-to-one comparison between each pair of parent and offspring vectors (i.e., the original DE selection scheme) before the dominance ranking and crowding density sorting among all solutions, (2) the most crowded solutions are pruned from the population *all in once* or *one by one*. It is shown [21, 19] that the original DE selection scheme is beneficial for performance improvement and enables a multi-objective algorithm fall back to the original DE algorithm if there is a single objective. Also, it is helpful to improve the population diversity by removing the most crowed solutions one by one and update the crowding density estimation after each removal [22].

In addition, special care needs to be taken to calculate the crowding density in the case of three or more objectives: compared to previous methods, the product [23] or harmonic mean [24] of the distances to nearest neighbors may serve as an effective, though relatively complex, crowding density estimation metric to improve the diversity of solutions.

F. A Brief Comparison of MOEAs in the Literature

In [10], NSGA-II is shown to outperform PESA and SPEA in terms of finding a diverse set of solutions and in converging near the true Pareto front. In [25], Khare et al compared the scalability of NSGA-II, PESA, and SPEA2, by considering different number of objectives (2 to 8). The results show that PESA is the best in terms of converging to the Pareto-optimal front but it has poor diversity maintenance. SPEA2 and NSGA-II perform equally well in terms of convergence and diversity maintenance whereas NSGA-II runs much faster.

In [23], GDE3, when using the product distance as a crowding estimation metric and pruning the most crowded solutions one-by-one, obtained solution that spread clearly better than NSGA-II and SPEA2 while other performance metrics are competitive.

III. DE and JADE for Single-objective Optimization

In this section, we describe the basic operations of differential evolution and introduce a self-adaptive DE algorithm, JADE, that has shown promising results in single-objective optimization.

Differential evolution follows the general procedure of an evolutionary algorithm. The initial population is randomly generated according to a uniform distribution between the lower and upper limits ($x_{j,\text{low}}$ and $x_{j,\text{up}}$) defined for each component x_j of an individual. After initialization, DE enters a loop of evolutionary operations: mutation, crossover and selection.

Mutation: At each generation g, this operation creates mutation vectors $\mathbf{v}_{i,g}$ based on the current parent population $\{\mathbf{x}_{i,g} \mid i = 1, 2, \ldots, NP\}$, where NP is the population size. The following are different mutation strategies frequently used in the literature

- 'DE/rand/1'

$$\mathbf{v}_{i,g} = \mathbf{x}_{r0,g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}), \qquad (1)$$

- 'DE/current-to-best/1'

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F(\mathbf{x}_{\text{best},g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}), \qquad (2)$$

- 'DE/best/1'

$$\mathbf{v}_{i,g} = \mathbf{x}_{\text{best},g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}), \qquad (3)$$

where the indices r_0, r_1 and r_2 are distinct integers uniformly chosen from the set $\{1, 2, \ldots, NP\}\backslash\{i\}$, $\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}$ is a difference vector to mutate the parent, and $F \in (0, 1]$ is the mutation factor that is fixed throughout the optimization process. Different from (1) - (3), JADE adopts a mutation strategy 'DE/current-to-p-best/1':

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i[(\mathbf{x}^p_{\text{best},g} - \mathbf{x}_{i,g}) + (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g})], \qquad (4)$$

where $\mathbf{x}^p_{\text{best},g}$ is randomly chosen as one of the top 100p%, $p \in (0, 1]$, individuals in the current population, and $F_i \in (0, 1]$ is the mutation factor associated with each individual $\mathbf{x}_{i,g}$ and is randomly generated by the parameter self-adaptation described below.

Crossover: After mutation, a 'binary' crossover operation forms the final trial vector $\mathbf{u}_{i,g} = (u_{1,i,g}, u_{2,i,g}, \ldots, u_{D,i,g})$:

$$u_{j,i,g} = \begin{cases} v_{j,i,g} & \text{if } \text{rand}_j(0,1) \le CR_i \text{ or } j = j_{\text{rand}} \\ x_{j,i,g} & \text{otherwise,} \end{cases} \qquad (5)$$

where $\text{rand}_j(a,b)$ is a uniform random number on the interval (a, b] and newly generated for each j, $j_{\text{rand}} = \text{randint}_i(1, D)$ is an integer randomly chosen from 1 to D and newly generated for each i, and the crossover probability, $CR_i \in (0, 1]$, roughly corresponds to the average fraction of vector components that are inherited from the mutant vector. In classic DE algorithms, a single crossover probability CR (i.e., $CR_i = CR$ for all i) is used to generate all trial vectors during the whole optimization process. In JADE, the crossover probabilities CR_i's are newly generated by the parameter self-adaptation at each generation.

Selection: In the selection operation, we choose the better one from the parent vector $\mathbf{x}_{i,g}$ and the trial vector $\mathbf{u}_{i,g}$ according to their fitness values $f(\cdot)$. For example, if we have a minimization problem, the selected vector is given by

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} & \text{if } f(\mathbf{u}_{i,g}) < f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g} & \text{otherwise,} \end{cases} \qquad (6)$$

and used as a parent vector in the next generation. If the trial vector $\mathbf{u}_{i,g}$ succeeds, the selection is considered as a *successful update* and the corresponding control parameters F_i and CR_i are called a *successful mutation factor* and *successful crossover probability*, respectively.

The above composes a complete evolutionary loop of the classic differential evolution algorithm. The two involved control parameters, F and CR, are usually problem dependent and need to be tuned by trial and error. In JADE, F and CR are updated by a self-adaptation mechanism that is based on a simple principle: Better values of control parameters tend to generate individuals that are more likely to survive and thus these values should be propagated. To be specific, F_i and CR_i are generated by two random processes:

$$CR_i = \text{randn}_i(\mu_{CR}, 0.1), \qquad (7)$$

$$F_i = \text{randc}_i(\mu_F, 0.1), \qquad (8)$$

where $\text{randn}(\mu, \sigma^2)$ denotes a random value from a normal distribution of mean μ and variance σ^2, $\text{randc}(\mu, \delta)$ denotes a random value from a Cauchy distribution with location and scale parameters μ and δ, respectively. The mean μ_{CR} and location parameter μ_F are updated in a self-adaptive manner:

$$\mu_{CR} = (1-c)\mu_{CR} + c \cdot \text{mean}_A(S_{CR}), \qquad (9)$$

$$\mu_F = (1-c)\mu_F + c \cdot \text{mean}_L(S_F). \qquad (10)$$

where S_{CR} and S_F are the respective sets of all successful crossover probabilities and successful mutation factors obtained in the selection (6) at generation g, c is a positive constant between 0 and 1 and $\text{mean}_A(\cdot)$ is the usual arithmetic mean operation and $\text{mean}_L(\cdot)$ is the Lehmer mean:

$$L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F}. \qquad (11)$$

which plays more weight on larger mutation factor F to improve evolutionary progress.

The parameters c and p of JADE determine the adaptation rates of μ_{CR} and μ_F and the greediness of the mutation strategy, respectively. They are shown to be problem-insensitive in a large set of experimental studies [12]. While it works well with c and p in a large range, JADE is shown to perform best with values $1/c \in [5, 20]$ and $p \in [5\%, 20\%]$.

IV. JADE2 FOR MULTI-OBJECTIVE OPTIMIZATION

In differential evolution, the crossover can be conducted in the same way for single- and multi-objective optimization. However, the selection and mutation need to be reconsidered because of the challenges in multi-objective optimization.

First, the selection requires the comparability between any distinct solutions in multi-dimensional objective space. This, however, cannot be guaranteed by a Pareto dominance comparison as it defines a partial order among solutions. A secondary metric such as crowding density needs to be introduced to compare two solutions if the dominance comparison ties.

Second, the best-solution information used in a greedy mutation strategy loses its original meaning of providing direction information to the optimum. This situation usually appears after only a few generations, especially when the dimension of the objective space is high and thus most or all solutions in the population quickly become non-dominated from one another. In this case, the best solutions are identified mainly according to their crowding density, instead of the dominance ranking. As a result, a usual greedy mutation strategy is useful to move the population towards the sparsest regions which however are not necessarily closer to the true Pareto front.

A. Pareto Dominance and Crowding Density

In an M-objective minimization problem, it is defined that an objective vector $\mathbf{f} = (f_1, f_2, \ldots, f_M)$ dominates another vector $\mathbf{g} = (g_1, g_2, \ldots, g_M)$ if and only if

$$\forall i \in \{1,2,...,M\} : f_i \le g_i \text{ and } \exists i \in \{1,2,...,M\} : f_i < g_i \,. \quad (12)$$

If it dominates, the objective vector $\mathbf{f}$ (and its corresponding decision vector) is considered better than the other vector $\mathbf{g}$ (and its corresponding decision vector). It is clear that two objective vectors may be incomparable (i.e., not dominated by each other) according to the definition of Pareto dominance.

A solution is considered as Pareto optimal if it is not dominated by any other vectors in the objective space. The set of these non-dominated solutions composes the Pareto-optimal front. In general, there are a huge, if not infinite, number of Pareto optimal solutions, while an evolutionary algorithm can only search for a small set of representative ones that are expected to be well diversified and to spread over the Pareto-optimal front. In the literature, this is usually achieved by different techniques of estimating the crowding density of solutions. More crowded solutions are more likely to be pruned from the population to maintain the diversity.

B. Selection Operation

In the selection operation of JADE2, solutions are compared according to the Pareto dominance and crowding density estimation at a lexicographic order. That is, a solution is considered better than another if it dominates or if it has a lower crowding density when the dominance comparison ties. To be specific, JADE2 adopts a three-step comparison to select NP vectors from the pool P' of $2NP$ parent and trial vectors. The selected ones form the parent population in the next generation.

In the first step, a one-to-one dominance comparison is conducted between each pair of parent and trial vectors. If one dominates, the other one is immediately removed from P'. This is similar to the original one-to-one comparison of DE in single-objective optimization. After this step, the size of P' ranges from NP to $2NP$.

The second step is to further reduce the size of P' based on the dominance relationship among all solution. First, all non-dominated solutions in P' are assigned rank 1, indicating the best fitness in the population. Then, the remaining solutions are considered and the non-dominated ones among them are assigned rank 2. This process repeats until no less than NP vectors have been assigned rank values. All other vectors are immediately removed from P'.

The third step is based on the crowding density estimation on all solutions that have the worst rank value. The most crowded among them is pruned one by one, with the crowding density of remaining solutions updated after each removal until the size of P' is reduced to NP. The density estimation is based on the consideration that the crowdedness of a solution is lower if its distances to nearest neighbors are (1) larger and (2) closer to be uniform. For this purpose, the fairness measure that is usually used in network engineering [26] to maintain the fairness (uniformity) of competing variables (e.g., network flow rates) while maximizing their summation can serve as a good method of estimating the crowding density. Consider the (p, α)-proportionally fairness measure in [26]. With a usual $p = 1$ and $\alpha \ge 0$, we can calculate the crowding distance d_i of a solution i as

$$d_i = \begin{cases} \sum_{i=1}^{k} \log d_{ij} & \text{if } \alpha = 1 \\ (1-\alpha)^{-1} \sum_{i=1}^{k} d_{ij}^{1-\alpha} & \text{otherwise} \end{cases} . \quad (13)$$

where d_{ij} is the Euclidean distance of solution i to its j-th nearest neighbor in P'. It is easy to show that if $\alpha = 0$, d_i is the sum distance. If $\alpha = 1$, it is *equivalent* to the product distance,

$$d_i = \prod_{i=1}^{k} d_{ij} \,; \quad (14)$$

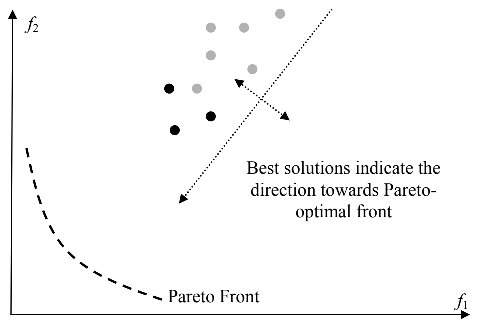
if $\alpha = 2$, the harmonic distance

$$d_i = \frac{1}{\frac{1}{d_{i,1}} + \frac{1}{d_{i,2}} + \cdots + \frac{1}{d_{i,2(M-1)}}} \,; \quad (15)$$

and if $\alpha \to \infty$, the max-min distance

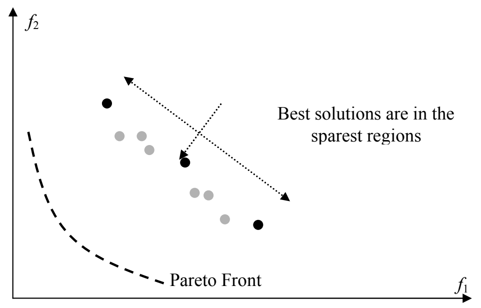
$$d_i = \max(\min(d_{i,1}, d_{i,2}, ..., d_{i,k})) \,. \quad (16)$$

It is worth noting that the harmonic distance and product distance have been adopted in MODE [24] and GDE3 [23] as the crowding distance, respectively. Experimental results show that the sum distance usually leads to less satisfactory results, while both the product and harmonic distance perform well for the test problems studied in this paper.



f_2
Best solutions indicate the direction towards Pareto-optimal front
Pareto Front
f_1

(a) Before most or all solutions become non-dominated



(b) After most or all solutions become non-dominated

Fig. 1: An illustration of the relative positions of the best solutions (dark dots) and other solutions (gray dots) in the objective space. The unidirectional arrows show the progress direction towards the Pareto front, while the bidirectional arrows show the spread of the population orthogonal to the progress direction.

As GDE3 is considered as a benchmark algorithm in this paper, we adopt the product distance (14) in JADE2 for fair comparison. Different from the setting in GDE3 (where $k = M$), however, we consider $k = 2(M–1)$ nearest neighbors in calculating the crowding distance. This number is selected based on the consideration that the dimension of the Pareto front is usually $M – 1$ and each solution is expected to have two nearest neighbors on its either side along each dimension.

C. Mutation Operation

In multi-objective optimization, most or all individuals in the population usually become non-dominated from one another after only a few generations. In this case, the best solutions are mainly identified according to the crowding density. Thus, in a usual greedy strategy, the population is moved towards the sparsest region where the best solutions are located. This is different from the situation of single-objective optimization or in the early stage of multi-objective optimization where the best solutions indicate the direction towards the optimal solutions. A simple illustration is shown in Fig. 1. It is clear that the best solutions are incapable of indicating promising directions when they become non-dominated from many other solutions.

We propose a new approach of utilizing direction information based on the inferior solutions previously explored in the optimization process. An external archive, A, is created to store the parent individuals that are recently removed from the population because they are dominated by other solution in selection process (i.e., the parent solutions that are removed from the population in the first two steps of the selection process). Then, as an improvement on the mutation strategy in (4), the mutant vector is generated as follows

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i[(\mathbf{x}^p_{\text{best},g} - \mathbf{x}_{i,g}) + (\tilde{\mathbf{x}}_{r1,g} - \mathbf{x}_{r2,g})], \qquad (17)$$

where $\tilde{\mathbf{x}}_{r1,g}$ is a vector randomly chosen from the union of the parent population and the archive, while $\mathbf{x}_{r2,g}$ and $\mathbf{x}^p_{\text{best},g}$ are chosen from the parent population in the way as in (4).

It is clear that (17) falls back to (4) if the archive is empty. If the archive is not empty and $\tilde{\mathbf{x}}_{r1,g}$ is chosen from it, the difference between $\tilde{\mathbf{x}}_{r1,g}$ and $\mathbf{x}_{r2,g}$ indicates the direction towards the optimum. Note that this is different from the method proposed in [27], where direction information is obtained by comparing solutions that are in the current population but have different dominance ranks (thus, the direction information becomes unavailable if all solutions in the population become non-dominated). In addition, the operation in (17) is clearly different from many other MOEAs where an archive is maintained to store the *best-so-far* non-dominated solutions. However, similar to these archive-based strategies, our method is also helpful to improve the diversity of solutions, other than its main benefit of providing direction information.

We consider rather simple archive operations to minimize the complexity. The archive is initialized to be empty. After each generation, we add to the archive those parent individuals that fail in the first two steps of the selection process. Then, if the archive size exceeds a threshold, say $2NP$, we randomly removed some solutions to keep the archive size at $2NP$.

V. Performance Comparison

In this section, we describe the performance matrices used for the evaluation of multi-objective optimization algorithms and present the comparison results of JADE2, NSGA-II and GDE3 based on the experimental studies on twenty-five multi-objective test problems.

A. Performance Measures

In multi-objective optimization, different evolutionary algorithms are compared according to the quality of the obtained *approximation set*, i.e., the set of non-dominated objective vectors that is an approximation to the true Pareto-optimal front. Because of the existence of incomparable solutions and therefore incomparable approximation sets, there exist many different approaches to performance assessment of multi-objective optimizers. In this paper, we follow the guidelines from [28] and use dominance ranking, quality indicators and the empirical attainment surface for comparing the approximation sets obtained by different algorithms.

Dominance ranking: Suppose the approximate sets generated by an evolutionary algorithm i after n independent runs are $A_1^i, A_2^i, \ldots, A_n^i$. If we consider the collection of approximate sets of all Q algorithms, then we can assign each approximate set a rank on the basis of a dominance relation, e.g., by counting the number of sets by which a specific approximation set is dominated (an approximation set A is said to be dominated by another set B if every objective vector in A is dominated by at least one vector in B). The lower the rank, the better the corresponding approximation set with respect to the entire collec-

	M	JADE v.s. NSGA-II								JADE v.s. GDE3							
		Middle results (after 60% of total generations)				Final results				Middle results (after 60% of total generations)				Final results			
		DR	I_H^-	$I^1_{\varepsilon+}$	I^1_{R2}	DR	I_H^-	$I^1_{\varepsilon+}$	I^1_{R2}	DR	I_H^-	$I^1_{\varepsilon+}$	I^1_{R2}	DR	I_H^-	$I^1_{\varepsilon+}$	I^1_{R2}
ZDT1	2	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
ZDT2	2	▲	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
ZDT3	2	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
ZDT4	2	-	∇	∇	-	-	▲	▲	▲	-	-	∇	-	-	▲	▲	▲
ZDT6	2	-	▲	▲	▲	-	▲	▲	▲	-	▲	-	-	-	▲	▲	-
DTLZ1	3	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
DTLZ2	3	-	▲	▲	▲	-	▲	▲	▲	-	∇	-	∇	-	▲	▲	∇
DTLZ3	3	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
DTLZ4	3	-	▲	▲	▲	-	▲	▲	▲	-	∇	-	∇	-	▲	▲	∇
DTLZ7	3	-	▲	▲	▲	-	▲	▲	▲	-	-	-	∇	-	-	-	∇
WKG1	2	▲	▲	▲	▲	▲	▲	▲	▲	∇	∇	∇	∇	▲	▲	▲	▲
WKG1	3	▲	▲	▲	▲	▲	▲	▲	▲	∇	∇	∇	∇	▲	▲	▲	▲
WKG8	2	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
WKG8	3	-	▲	▲	▲	-	▲	▲	▲	-	▲	-	-	-	▲		▲
WKG9	2	∇	∇	∇	∇	∇	-	∇	∇	∇	∇	∇	∇	∇	∇	∇	∇
WKG9	3	-	∇	-	∇	-	∇	▲	∇	-	∇	∇	∇	-	∇	∇	∇
OKA2	2	-	▲	▲	-	-	▲	▲	▲	-	▲	▲	-	-	▲	-	-
SYMPART	2	▲	▲	▲	▲	▲	▲	▲	▲	-	▲	▲	▲	-	▲	▲	▲
QV	2	-	∇	∇	∇	-	-	∇	∇	-	-	-	▲	-	▲	▲	▲
KUR	2	▲	▲	▲	▲	▲	▲	▲	▲	-	▲	▲	▲	▲	▲	▲	▲
SPH	2	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲	▲
SPH	3	▲	▲	▲	▲	▲	▲	▲	▲	-	∇	-	∇	-	-	▲	∇

Table I: A comparison of the middle and final results obtained by different algorithms with respect to dominance ranking *DR* (Mann-Whitney rank sum test at a significant level 0.05), hyper-volume indicator I_H, unary additive epsilon indicator $I^1_{\varepsilon+}$, and unary R_2 indicator I^1_{R2} (Fisher-independent test at a significant level 0.05). The middle results are obtained after 60% of total number of generations. The symbol '▲' ('∇') means that JADE is significantly better (worse) than the other algorithm regarding to the dominance ranking or other indicators, respectively.

tion. In this way, we obtain a set of ranks $\{r_1^i, r_2^i, \ldots, r_n^i\}$ for the approximation sets obtained by each algorithm. Then, a statistical rank test, such as the Mann-Whitney rank sum test [28, 29], is applied to determine if one algorithm is significantly better than another one in terms of the dominance ranking.

Quality indicators: A quality indicator is a function that assigns a real value to any approximation set by means of some preference information. The difference between two indicator values reveals the superiority of one set over the other one in a certain aspect, even when the two sets are incomparable according to Pareto dominance. This type of information is usually considered secondary to Pareto dominance, and thus the indicators that are compliant to the Pareto relationship are more meaningful than Pareto non-compliant ones. In view of this, we consider the three Pareto compliant quality indicators recommended in [28]: (1) the *hyper-volume indicator* I_H^- that measures the difference between the hyper-volumes of the portion of the objective space that is weakly dominated by an approximation set or by a reference set, (2) the *unary additive epsilon indicator* $I^1_{\varepsilon+}$ that gives the minimum value ε that can be added to each vector in a reference set such that the resulting shifted set is weakly dominated by the approximate set of interest, and (3) the *unary R2 indicator* I^1_{R2} that is used to assess an approximation set on the basis of a set of utility functions. All these indicators are to be minimized.

Similar to the dominance ranking, the indicator values obtained by each algorithm after n independent runs can be collected together and assigned rank values. Then, a statistical rank test, such as the Fisher-independent test [28, 29], can be applied to determine if one algorithm is significantly better than another algorithm in terms of the corresponding indicator.

Attainment surface: The third approach to performance assessment is based on the multi-objective concept of *goal-attainment:* an objective vector is attained when it is weakly dominated by the approximation set obtained by an algorithm. The *k%-approximation surface* of the n approximation sets $A_1, A_2, \ldots, A_n$ obtained by an algorithm consists of the tightest objective vectors that have been attained in at least k percent of the n runs of an algorithm. That is, a k%-attainment surface roughly divides the objective space in two parts: the goals that have been attained and the goals that have not been attained with a frequency of at least k percent. The attainment surface is useful for visualizing an algorithm's performance.

B. Experimental Settings

In order to validate the performance of JADE2, we consider a set of test problems from the literature. These include the frequently used ZDT1 − 4 and ZDT6 in [30], DTLZ1 − 4 and DTLZ7 in [31], and the three WKG functions [32] (WKG1, 8 and 9) that are believed to be hard problems because of the bias, discontinuity, and deception in the problems. Also, we consider three test problems (QV, KUR, and SPH) previously studies in SPEA2 [8] and NSGA-II [10], and two recently proposed test problems SYMPART [33] and OKA2 [34].

We compare the results with respect to NSGA-II and GDE3, as the former is an approach representative of the state-of-the-art, while the latter is a DE based algorithm that has shown competitive performance. For fairness, we adopt the parameter settings in the original paper of NSGA-II [10] (i.e., $p_c = 0.9$, $p_m = 1/D$, $\eta_c = 20$ and $\eta_m = 20$) and GDE3 [19] (i.e., $F = 0.2$, and $CR = 0.2$). The parameters in JADE2 are set to be $c = 1/10$ and $p = 10\%$ in all simulations. The population size is set to be 100 and 300 for two and three dimensional problems, respectively.

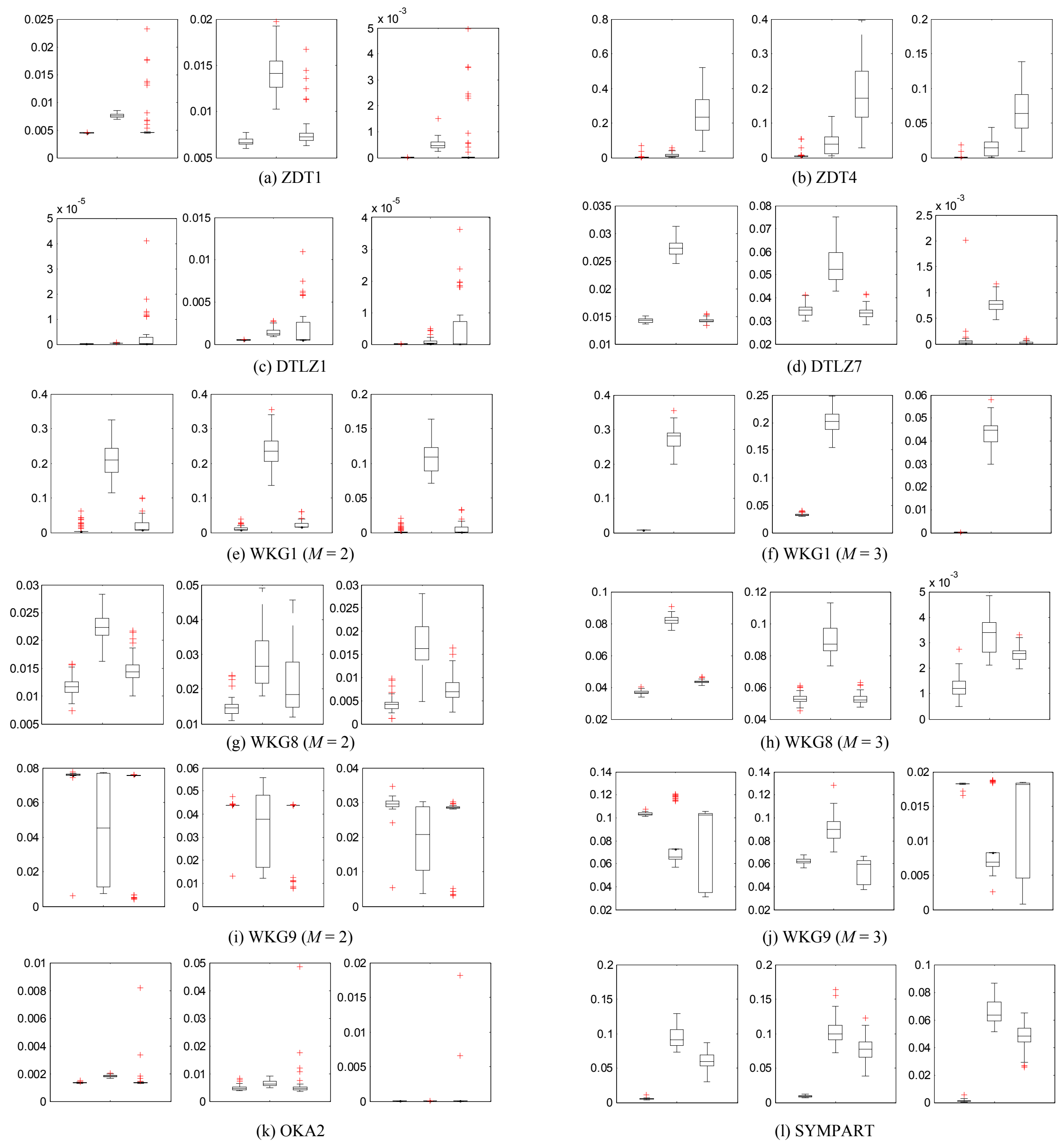


Fig. 2: A comparison of quality indicators. The three box and whisker plots in each figure corresponds to JADE2, NSGA-II, and GDE3, respectively. The three figures for each problem show the results of hyper-volume indicator I_H, unary additive epsilon indicator $I^1_{\epsilon+}$, and unary $R2$ indicator I^1_{R2}, respectively.

The number of generations is set to be 250 for all tested problems, except that 500 for SPH, WFG8 and WFG9, 2000 for QV and KUR, and 3000 for WFG1. The results are obtained based on 50 independent runs of each algorithm. In Table I, we report both the final results and the middle results obtained after 60% of total generations.

C. *Experimental Results*

Table I presents the outcomes of dominance ranking as well as the three quality indicators, hyper-volume indicator I_H^-, unary additive epsilon indicator $I^1_{\epsilon+}$, and unary R2 indicator I^1_{R2}. In Fig. 2, we also draw box and whisker plots for selected problems so as to have a clear comparison between JADE2 and other algorithms.

It is clear that JADE2 shows consistently better performance than NSGA-II for all problems studied in this paper except WKG9 and QV. The dominance rank of JADE2 is significantly better than that of NSGA-II in the case of SYMPART, WKG1, KUR and SPH (In other cases except WKG9 and QV, the comparison results are statistically insignificant but a clear overall trend of Pareto dominance can still be observed as

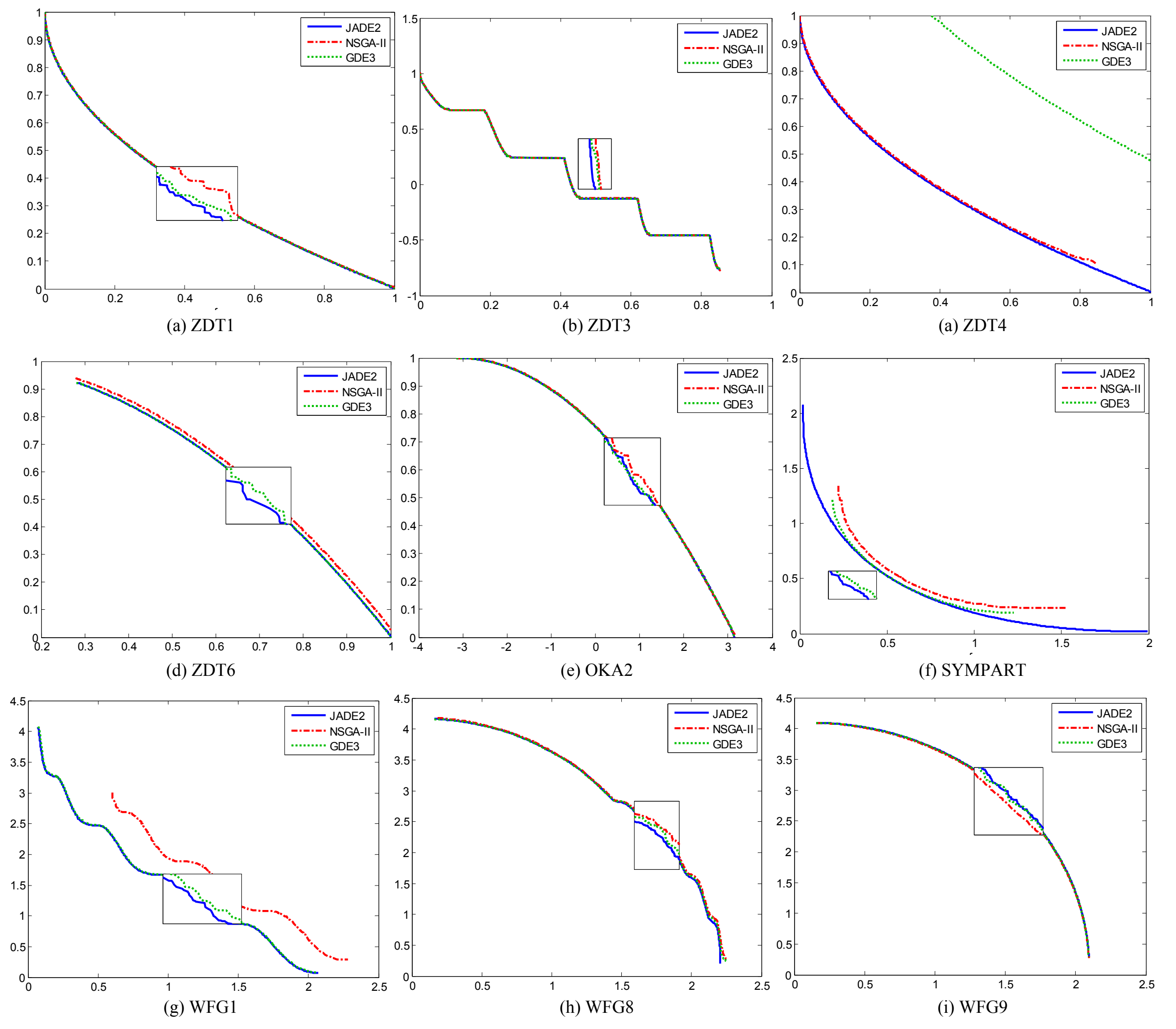


Fig. 3: The 50% attainment surfaces for two-objective problems. The results are obtained based on 50 independent runs of each algorithm.

shown later in Fig. 3). In terms of the three quality indicators, JADE2 shows significantly better performance for all studied problems except WKG9 and QV. In the case of WKG9, JADE2 may cause false convergence while NSGA-II has no difficulty of approaching the true optimal front. In the case of QV, both JADE2 and NSGA-II can easily find the true Pareto front but NSGA-II converges slightly faster.

JADE2 shows on average better final performance than GDE3, especially when the hyper-volume indicator and unary additive epsilon indicator are concerned – JADE2 is better in 18 out of the 22 test problems. In terms of the $R2$ indicator, GDE3 is superior in six problems but is beaten by JADE2 in other 14 problems. Similar to the comparison with NSGA-II, however, JADE shows worse performance than GDE3 in the case of WKG9 because of more frequent false convergence.

Based on the performance comparison of middle and final results in Table I, it is interesting to note that JADE2 gains more competitive advantages over GDE3 after a larger number of generations, especially in the case of ZDT4 ad WDK1. As explained before, the best solutions only direct the evolution search at the beginning of the optimization process, while the archived inferior solutions used in JADE2 are capable of providing direction information even at the later stage of the optimization when most or all solutions in the population become non-dominated. JADE2 is therefore able to keep a relatively fast convergence rate and outperform GDE3 after a larger number of generations.

To visualize the final solutions obtained by different algorithm, we plot in Figs. 3 and 4 the 50% attainment surfaces for some two-objective problems and the best approximation sets (according to the I_H^- indictor) calculated based on the results of 50 independent runs. These graphs are consistent with the statistic results summarized in Table I and Fig. 2, which indicate a generally better performance of JADE2 than NSGA-II in

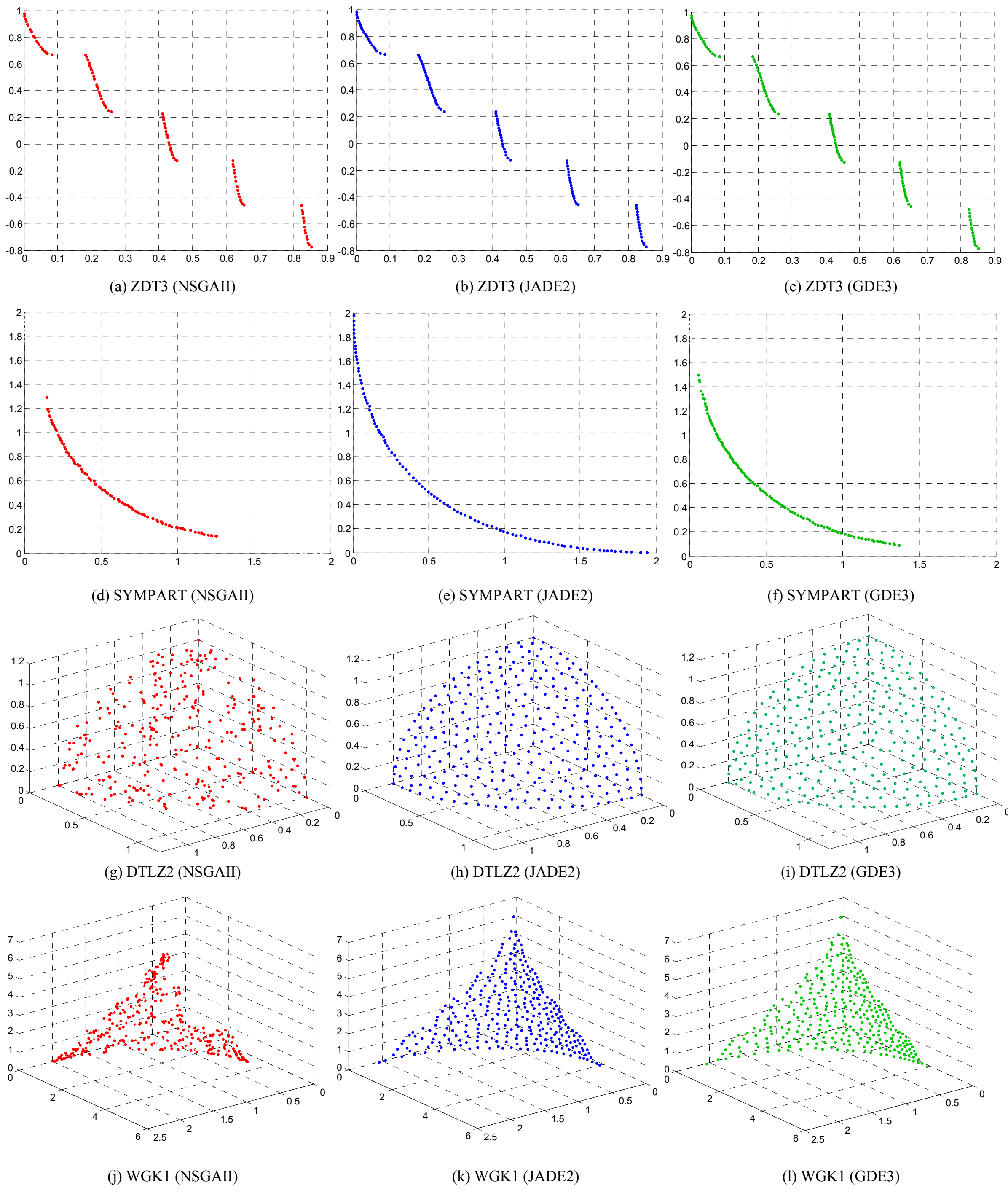


Fig. 4: The best approximation sets (according to I_H^-) found by JADE2, NSGA-II, and GDE3.

terms of both the overall trend of Pareto dominance (as shown in Fig. 3) and the spread of final results (as shown in Fig. 4). The spread of the final results of GDE3 and JADE2 is usually incomparable, mainly because they adopt the similar product distance to estimate the crowding density. However, the comparison in Fig. 3 indicates that in most cases JADE2 is able to attain solutions which are clearly better than or at least competitive to those of GDE3.

VI. Conclusion

We have proposed a new differential evolution algorithm JADE2 for multi-objective optimization by controlling the parameters in a self-adaptive manner and utilizing the direction information provided by archived inferior solutions. Self-adaptive parameter control avoids users' prior knowledge about the fitness characteristics of optimization problems and therefore avoids tedious parameter tuning. The difference between archived inferior solutions and the current population indicates promising directions towards the optimum and thus can be utilized to speed up evolutionary search. JADE2 shows better results than NSGAII for 18 out of 22 test problems in terms of the overall trend of Pareto-dominance and three quality indicators. It outperforms GDE3, especially in terms of the hyper-volume and unary additive epsilon indicators. Also, JADE2 shows more advantages over GDE3 after a larger number of generations. It indicates the relatively fast convergence rate of JADE2 even at a later stage of evolution search when most or all solutions in the population become non-dominated but archived inferior solutions are still able to provide direction information towards the optimum.

A future research direction is to study the proposed algorithm in solving optimization problems with many (>3) objectives and to investigate the effect of parameter α of the fairness measure (used in calculating crowding distances) on achieving diversified solutions. It will be meaningful to compare JADE2 to other latest algorithms such as [35] that have shown promising results in solving many-objective optimization problems.

References

[1] C. A. Coello Coello and G. B. Lamont, *Applications of Multi-Objective Evolutionary Algorithms*, World Scientific, 2004.

[2] C. L. Hwang and A. S. Masud, *Multiple Objective Decision Making — Methods and Applications*. Berlin, Germany: Springer-Verlag, 1979.

[3] C. A. Coello Coello, G. T. Pulido and E. M. Montes. "Current and Future Research Trends in Evolutionary Multiobjective Optimization," *Information Processing with Evolutionary Algorithms: From Industrial Applications to Academic Speculations*, pp. 213-231, Springer-Verlag, 2005.

[4] S. F. Adra , I. Griffin and P. J. Fleming, "A Comparative Study of Progressive Preference Articulation Techniques for Multiobjective Optimisation," *Evolutionary Multi-Criterion Optimization*, Springer, pp. 0302-9743, 2007.

[5] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer; 2nd ed. Edition, 2007.

[6] J. D. Knowles and D.W. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evolutionary Comput*., vol. 8, no. 2, pp. 149–172, 2000.

[7] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evolutionary Computation*, vol. 3, pp. 257–271, 1999.

[8] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," *Computer Engineering and Networks Laboratory (TIK)*, Swiss Federal Inst. Technology, Switzerland, 2001.

[9] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto envelopebased selection algorithm for multiobjective optimization," in *Proc. Parallel Problem Solving Nature VI Conf.*, 2000, pp. 839–848.

[10] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[11] C. A. Coello Coello, "20 Years of Evolutionary Multi-Objective Optimization: What Has Been Done and What Remains to be Done," in Gary Y. Yen and David B. Fogel (editors), *Computational Intelligence: Principles and Practice*, Ch. 4, pp. 73--88, IEEE Comp. Intl. Society, 2006 .

[12] Jingqiao Zhang and Arthur C. Sanderson, "JADE: Self-Adaptive Differential Evolution with Fast and Reliable Convergence Performance," *IEEE Congress on Evolutionary Computation*, Sept 2007, Singapore.

[13] B. W. Silverman. *Density estimation for statistics and data analysis. London*, Chapman and Hall, 1986.

[14] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[15] H. Abbass, and R. Sarker, "The Pareto Differential Evolution Algorithm," *International Journal of Artificial Intelligence Tools*, World Scientific Publishing Company, vol. 11, no. 4, pp. 531-552, 2002.

[16] R. Sarker and H. Abbass, "Differential Evolution for Solving Multi-Objective Optimization Problems," *Asia-Pacific Journal of Operations Research*, vol. 21, no. 2, World Scientific, pp. 225-240, 2004.

[17] F. Xue, "Multi-objective differential evolution: Theory and applications," Ph.D. dissertation, Rensselaer Polytechnic Inst., Ney York, USA, 2004.

[18] Tea Tušar, *Design of an Algorithm for Multiobjective Optimization with Differential Evolution*. Master Thesis, U. of Ljubljana, Slovenian, 2007.

[19] S. Kukkonen and J. Lampinen. GDE3: "The third evolution step of Generalized Differential Evolution," *Proc. of the 2005 Congress on Evolutionary Computation*, pp. 443–450, Edinburgh, Scotland, 2005.

[20] A. G. Hernandez-Diaz, L. V. Santana-Quintero, C. Coello Coello, "A New Proposal for Multi-Objective Optimization using Differential Evolution and Rough Sets Theory," *Proc. of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 675 – 682, 2006.

[21] K. Zielinski, R. Laur, "Variants of Differential Evolution for Multi-Objective Optimization," *IEEE Symposium on Computational Intelligence in Multicriteria Decision Making*, pp. 91 – 98, April 2007.

[22] S. Kukkonen. K. Deb, "Improved Pruning of Non-Dominated Solutions Based on Crowding Distance for Bi-objective Optimization Problems," *IEEE Congress on Evolutionary Computation*, 2006.

[23] S. Kukkonen. K. Deb, "A Fast and Effective Method for Pruning of Non-Dominated Solutions in Many-Objective Problems," *Parallel Problem Solving from Nature - PPSN IX*, pp. 553-562, 2006.

[24] V. L. Huang, P. N. Suganthan and S. Baskar, "Multiobjective Differential Evolution with External Archive and Harmonic Distance-Based Diversity Measure," Technical Report, Nanyang Technological University, Singapore, Dec., 2005.

[25] V. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," *Proc. of the 2nd Intl. Conference on Evolutionary Multi-Criterion Optimization*, New York, Edited by Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb and Lothar Thiele, 2003, pp. 376-390.

[26] J. Mo and J. Walrand, "Fair end-to-end windows-based congestion control," *IEEE/ACM Trans. On Networking*, vol. 8, no. 5, pp. 556-567, Oct. 2000.

[27] Iorio, A and Li, X.(2006), "Incorporating Directional Information within a Differential Evolution Algorithm for Multiobjective Optimization", in Proceeding of Genetic and Evolutionary Computation Conference 2006 (GECCO'06), eds. M. Keijzer, et al., p.691 - 697, ACM Press.

[28] J. D. Knowles, L. Thiele, and E. Zitzler. "A tutorial on the performance assessment of stochastic multi-objective optimizers," Technical Report TIK-Report No. 214, Computer Engineering and Networks Laboratory, ETH Zürich, February 2006.

[29] W. J. Conover, *Practical Nonparameteric Statistics*, 3rd edition, New York, NY, Hohn Wiley and Sons, 1999.

[30] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, England, 2001.

[31] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, "Scalable Test Problems for Evolutionary Multiobjective Optimization", *Evolutionary Multiobjective Optimization*. Springer-Verlag, London, pp. 105–145, 2005.

[32] S. Huband, P. Hingston, L. Barone, and L. While, "A Review of Multi-objective Test Problems and a Scalable Test Problem Toolkit," *IEEE Transactions on Evolutionary Computation*, volume 10, no 5, pages 477-506. IEEE, October 2007.

[33] G. Rudolph, B. Naujoks and M. Preuss, "Capabilities of EMOA to Detect and Preserve Equivalent Pareto Subsets," *Evolutionary Multi-Criterion Optimization*, pp. 36- 50. vol. 4403, Springer, Berlin. 2007.

[34] T. Okabe, Y. Jin, M. Olhofer, and B. Sendhoff, "On Test Functions for Evolutionary Multiobjective Optimization," *Parallel Problem Solving from Nature (PPSN VIII)*, pp. 792 ‑ 802, 2004.

[35] S. Bandyopadhyay, S. Saha, U. Maulik and K. Deb, "A Simulated Annealing Based Multi-objective Optimization Algorithm: AMOSA," *IEEE Transaction on Evolutionary Computation*, in press.