Engineering Optimization
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/geno20

MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition

Yan-Yan Tan [a], Yong-Chang Jiao [a], Hong Li [b] & Xin-Kuan Wang [a]
[a] National Key Laboratory of Antennas and Microwave Technology, Xidian University, Xi'an, P.R. China
[b] School of Science, Xidian University, Xi'an, P.R. China
Published online: 31 Jan 2012.

To cite this article: Yan-Yan Tan, Yong-Chang Jiao, Hong Li & Xin-Kuan Wang (2012) MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition, Engineering Optimization, 44:9, 1095-1115, DOI: 10.1080/0305215X.2011.632008

To link to this article: http://dx.doi.org/10.1080/0305215X.2011.632008

Engineering Optimization
Vol. 44, No. 9, September 2012, 1095–1115

Taylor & Francis
Taylor & Francis Group

MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition

Yan-Yan Tan[a]*, Yong-Chang Jiao[a], Hong Li[b] and Xin-Kuan Wang[a]

[a]*National Key Laboratory of Antennas and Microwave Technology, Xidian University, Xi'an, P.R. China;* [b]*School of Science, Xidian University, Xi'an, P.R. China*

(*Received 14 May 2011; final version received 9 September 2011*)

A multi-objective memetic algorithm based on decomposition is proposed in this article, in which a simplified quadratic approximation (SQA) is employed as a local search operator for enhancing the performance of a multi-objective evolutionary algorithm based on decomposition (MOEA/D). The SQA is used for a fast local search and the MOEA/D is used as the global optimizer. The multi-objective memetic algorithm based on decomposition, *i.e.* a hybrid of the MOEA/D with the SQA (MOEA/D-SQA), is designed to balance local versus global search strategies so as to obtain a set of diverse non-dominated solutions as quickly as possible. The emphasis of this article is placed on demonstrating how this local search scheme can improve the performance of MOEA/D for multi-objective optimization. MOEA/D-SQA has been tested on a wide set of benchmark problems with complicated Pareto set shapes. Experimental results indicate that the proposed approach performs better than MOEA/D. In addition, the results obtained are very competitive when comparing MOEA/D-SQA with other state-of-the-art techniques.

Keywords: multi-objective memetic algorithm; multi-objective optimization; quadratic approximation; decomposition

1. Introduction

Handling multi-objective optimization problems (MOPs) is a very important issue for real-world applications, because, in real-world applications, there are usually two or more objectives that conflict with each other. These conflicting objectives pose a challenge for optimization algorithm developers because there is no general rule for appropriately combining these objectives into a single one and decision makers may wish to know all the possible tradeoffs that they can have. Traditionally in mathematics, the procedure for solving a multi-objective problem is firstly to transform it into a single-objective problem by using weights on the objectives. This method makes the problem solvable by many existing, well-developed tools based on mathematics or heuristics. However, such weights often cannot be pre-determined, especially when the domain knowledge for the problem is unavailable. Furthermore, the best solution to the transformed

*Corresponding author. Email: yytan@mail.xidian.edu.cn

ISSN 0305-215X print/ISSN 1029-0273 online

http://dx.doi.org/10.1080/0305215X.2011.632008
http://www.tandfonline.com

single-objective problem is merely one solution on the Pareto front (PF) of the MOP. Hence, better optimization frameworks must be developed to fulfill the need for solving MOPs.

Owing to the limitation of traditional mathematical methods for MOPs, more and more researchers are trying to solve MOPs in a direct way and to approximate the PF as completely as possible. Their goal is to provide a set of solutions that are partially optimal. Many advanced multi-objective algorithms have been proposed in the literature. Some of them try to approximate the PF by using mathematical models (Wiecek *et al.* 2001, Ruzika and Wiecek 2005), and some are developed based on evolutionary algorithms (Zitzler and Thiele 1999, Zitzler *et al.* 2001, Deb *et al.* 2002, Coello *et al.* 2004, Beume *et al.* 2007, Du *et al.* 2007, Ghomsheh *et al.* 2007, Ismail *et al.* 2011). A hybrid framework making use of decomposition methods in mathematics and the optimization paradigm in evolutionary computation was proposed and called MOEA/D (Zhang and Li 2007). MOEA/D and its new versions have solved a number of multi-objective optimization problems efficiently (Zhang and Li 2007, Chang *et al.* 2008, Ishibuchi *et al.* 2009, Li and Zhang 2009, Zhang *et al.* 2009). MOEA/D explicitly decomposes the given MOP into a number of scalar optimization subproblems and solves these subproblems simultaneously by evolving a population of solutions. At each generation, the population is composed of the best solution found so far for each subproblem. It has been proved that MOEA/D has a lower complexity than NSGA-II (Deb *et al.* 2002), the most popular Pareto dominance based MOEA at each generation (Zhang and Li 2007, Li and Zhang 2009). The new version of MOEA/D (Zhang *et al.* 2009) has been ranked first among 13 entries in the unconstrained MOEA competition in CEC2009.

However, all the population-based optimization algorithms suffer from heavy computational burden in general, not excepting MOEA/D, because of their evolutionary or stochastic nature (Rahnamayan *et al.* 2008). In single-objective optimization, several studies have shown that incorporating some form of domain knowledge can greatly improve the search capability of evolutionary algorithms (Kim and Myung 1997, Chakraborty *et al.* 2006, Takahama and Sakai 2006, Wang *et al.* 2007, Noman and Iba 2008, Rahnamayan *et al.* 2008). Many problem dependent heuristics, such as approximation algorithms, local search techniques, and specialized recombination operators, have been tried in many different ways to accomplish this task (Noman and Iba 2008). In particular, the hybridization of evolutionary algorithms with local searches has proven to be very promising (Goldberg and Voessner 1999, Jaszkiewicz 2002, Chakraborty *et al.* 2006, Takahama and Sakai 2006, Noman and Iba 2008). Such a hybrid algorithm is referred to as a memetic algorithm (Moscato 1999). The high search ability of memetic algorithms has been reported in the literature (Jaszkiewicz 2002, Knowles and Corne 2005, Krasnogor and Smith 2005, Ong *et al.* 2006, Smith 2007, Samanlioglu *et al.* 2008). Motivated by these observations, a multi-objective memetic algorithm based on decomposition, called the MOEA/D-SQA (multi-objective evolutionary algorithm based on decomposition in which the simplified quadratic approximation – SQA – is adopted as a local search technique for enhancing the performance of MOEA/D), is proposed in this article for solving multi-objective optimization problems.

The remainder of this article is organized as follows. Section 2 describes multi-objective optimization problems. Section 3 briefly reviews the framework of MOEA/D. Section 4 describes the SQA. The proposed approach is then given in Section 5. Experimental results and a discussion are presented in Section 6. Section 7 concludes this article.

2. Multi-objective problems

Most real-world problems are multi-objective optimization problems (MOPs), of which single-objective problems are a special case. For example, in many engineering problems there are usually

at least two conflicting objectives, performance and cost. Formally, an MOP can be stated as:

$$\begin{aligned} &\text{minimize } F(x) = (f_1(x), \ldots, f_m(x))^{\mathrm{T}} \\ &\text{subject to } x \in \Omega, \end{aligned} \tag{1}$$

where $x = (x_1, \ldots, x_n)^{\mathrm{T}}$ is called the decision vector, Ω is called the decision space or the variable space, and R^m is the objective space. $F : \Omega \to R^m$ consists of m real-valued objective functions. If Ω is a closed and connected region in R^n and all the objective functions are continuous, the problem is a continuous MOP.

In order to consider the tradeoff between objectives, the concept of domination between solutions is introduced. Let $u = (u_1, \ldots, u_m)^{\mathrm{T}}, v = (v_1, \ldots, v_m)^{\mathrm{T}} \in R^m$ be two vectors; u is said to dominate v if $u_i \leq v_i$ for all $i = 1, \ldots, m$, and $u \neq v$. A point $x^* \in \Omega$ is Pareto optimal to (1) if there is no $x \in \Omega$ such that $F(x)$ dominates $F(x^*)$. The set of all the Pareto optimal points is called the Pareto set (PS), and the set of all the objective vectors corresponding to the PS is called the Pareto front (PF), where $\mathrm{PF} = \{F(x^*) = (f_1(x^*), \ldots, f_m(x^*))^{\mathrm{T}} | x^* \in \mathrm{PS}\}$ (Miettinen 1999).

Instead of searching for a single or just a few optimal solutions as in solving single-objective problems, the goal of handling multi-objective problems is to find the Pareto front as well as the Pareto set of the problem. Given limited computational resource, including time and storage, how to provide good solutions in terms of both quality and spread is the key and challenging task for multi-objective optimization.

3. Multi-objective evolutionary algorithm based on decomposition

One of the key issues of MOEA/D is the use of a decomposition method to transform an MOP into a number of scalar optimization subproblems. MOEA/D attempts to optimize these subproblems collectively and simultaneously instead of trying to approximate the Pareto front directly, as do many other evolutionary algorithms because each optimal solution to these scalar subproblems is a Pareto optimal solution to the given MOP. The collection of these optimal solutions is an approximation of the Pareto front. The weighted sum, the Tchebycheff approach, boundary intersection, and any other decomposition approaches can serve this purpose. In the present work, the Tchebycheff approach (Miettinen 1999) is adopted. A scalar optimization problem obtained by decomposing the given MOP can be represented as

$$\begin{aligned} &\text{minimize } g^{te}(x|\lambda, z^*) = \max_{1 \leq i \leq m} \{\lambda_i |f_i(x) - z_i^*|\}, \\ &\text{subject to } x \in \Omega, \end{aligned} \tag{2}$$

where $\lambda = (\lambda_1, \ldots, \lambda_m)^{\mathrm{T}}$ is a vector of weights, *i.e.* $\lambda_i \geq 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^{m} \lambda_i = 1$. $z^* = (z_1^*, \ldots, z_m^*)^{\mathrm{T}}$ is the reference point, *i.e.* $z_i^* = \min\{f_i(x) | x \in \Omega\}$ for each $i = 1, \ldots, m$. Let $\lambda^1, \ldots, \lambda^N$ be a set of even spread weight vectors. If N is large enough and the set of weight vectors is selected properly, all the optimal solutions to the scalar subproblems from decomposition will well approximate the Pareto front. Moreover, MOEA/D defines a neighbourhood relationship for each of the subproblems by computing Euclidean distances between weight vectors. Scalar subproblems which are considered neighbours will have similar fitness landscapes and their optimal solutions should be close in the decision space. MOEA/D exploits information sharing among subproblems which are neighbours to accomplish the optimization task effectively and efficiently.

Algorithm 1 – The framework of the MOEA/D algorithm
Step 1. Initialization

Step 1.1. Compute the Euclidean distances between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.
Step 1.2. Generate an initial population $x^1, \ldots, x^N$ by randomly sampling from Ω. Set $FV^i = F(x^i)$.
Step 1.3. Initialize $z = (z_1, \ldots, z_m)^{\mathrm{T}}$ by setting $z_j = \min_{1 \le i \le N} f_j(x^i), j = 1, \ldots, m$.

Step 2. Update
For $i = 1, \ldots, N$, do:

Step 2.1. Reproduction: Randomly select two indexes k, l from $B(i)$, and then generate a new solution $\bar{y}$ from x^i, x^k and x^l by using genetic operators.
Step 2.2. Improvement: Apply a problem-specific repair/improvement heuristic on $\bar{y}$ to produce y.
Step 2.3. Update of z: For each $j = 1, \ldots, m$, if $z_j > f_j(y)$, then set $z_j = f_j(y)$.
Step 2.4. Update of neighbouring solutions: For each index $j \in B(i)$, if $g^{te}(y|\lambda^j, z) \le g^{te}(x^j|\lambda^j, z)$, then set $x^j = y$ and $FV^j = F(y)$.

Step 3. Stopping criteria
If the stopping criteria are satisfied, then stop and output $\{x^1, \ldots, x^N\}$ and $\{F(x^1), \ldots, F(x^N)\}$. Otherwise, go to **Step 2**.

4. Simplified quadratic approximation (SQA)

The SQA method is a direct search technique and purely heuristic. It was first used in the modified controlled random search algorithms for unconstrained global optimization problems (Ali *et al.* 1997, Jiao *et al.* 2006), then it was combined with real-coded genetic algorithms for solving constrained optimization problems (Li *et al.* 2005, 2011). Experimental results indicated that the SQA is a powerful heuristic tool for single-objective optimization, it is effective in improving the local search ability and the accuracy of the minimum function value, and in greatly reducing the computational overhead of the algorithms (Ali *et al.* 1997, Li *et al.* 2005, 2011, Jiao *et al.* 2006). However, it has not yet been widely used in multi-objective evolutionary optimization. In this article, the SQA is integrated into MOEA/D for solving multi-objective optimization problems.

Denote three best points corresponding to the smallest objective function values of f by $x^{i_a} = (x_1^{i_a}, \ldots, x_n^{i_a})^{\mathrm{T}}$, $x^{i_b} = (x_1^{i_b}, \ldots, x_n^{i_b})^{\mathrm{T}}$, $x^{i_c} = (x_1^{i_c}, \ldots, x_n^{i_c})^{\mathrm{T}}$. Then, the approximate minimal point $x^{i'} = (x_1^{i'}, \ldots, x_n^{i'})^{\mathrm{T}}$ is calculated according to

$$x_k^{i'} = \frac{1}{2}\frac{\mathcal{A}_k}{\mathcal{B}_k}, \quad k = 1, \ldots, n, \tag{3}$$

where $\mathcal{A}_k = [(x_k^{i_b})^2 - (x_k^{i_c})^2]f(x^{i_a}) + [(x_k^{i_c})^2 - (x_k^{i_a})^2]f(x^{i_b}) + [(x_k^{i_a})^2 - (x_k^{i_b})^2]f(x^{i_c})$, and $\mathcal{B}_k = (x_k^{i_b} - x_k^{i_c})f(x^{i_a}) + (x_k^{i_c} - x_k^{i_a})f(x^{i_b}) + (x_k^{i_a} - x_k^{i_b})f(x^{i_c})$.

In contrast to a gradient-based local optimizer, the SQA needs no gradient computation, which broadens its use in optimization problems. In contrast to other quadratic approximation models needing many unknowns for building the quadratic model of the objective function (see for example Brachetti *et al.* 1997), the SQA is a three-point quadratic approximation, and it is convenient

to compute and easy to use. The simplified quadratic approximation tries to make good use of the values of the objective function already evaluated – it is solved at each iteration to yield a step toward the solution of the problem.

5. The proposed approach

In order to enhance the performance of the MOEA/D for multi-objective optimization, it is necessary to take the exploration advantages of both MOEA/D and the SQA by combining them in a well-balanced manner; hence a hybrid of MOEA/D with a simplified quadratic approximation (MOEA/D-SQA) is proposed in this article, in which the MOEA/D is used as the main algorithm to implement global exploration in order to maintain the diversity of the population, while the SQA is used as a local search scheme and integrated into the MOEA/D to accelerate its convergence speed and improve the accuracy of the resulting solution.

In the global search part, let $\lambda^1, \ldots, \lambda^N$ be a set of evenly spread weight vectors and z^* be the reference point. The problem of approximation of the PF in MOEA/D-SQA is decomposed into N scalar optimization subproblems. The objective in each of these subproblems is an aggregation of all the objectives. With the Tchebycheff decomposition approach, the objective function of the ith subproblem is of the form

$$g^{te}(x|\lambda^i, z^*) = \max_{1\le j\le m} \{\lambda_j^i |f_j(x) - z_j^*|\}, \tag{4}$$

where $\lambda^i = (\lambda_1^i, \ldots, \lambda_m^i)^{\mathrm{T}}$ and $i = 1, \ldots, N$. MOEA/D-SQA minimizes all these N subproblems (*i.e.* scalar aggregation functions) simultaneously in a single run. Each subproblem is optimized by using information mainly from its neighbouring subproblems. Neighbourhood relations among subproblems are defined based on the distances between their aggregation coefficient vectors. The optimal solutions to two neighbouring subproblems should be very similar.

In the local search part, SQA is applied on each subproblem to improve the local search ability. To adopt the SQA effectively to improve the local search and to speed up the convergence of MOEA/D, the following questions need to be considered.

- How to choose three points (individuals) $x^{i_a}, x^{i_b}, x^{i_c}$ from the neighbourhood of each subproblem i for the SQA?
- If $\mathcal{B}_k = 0$ for some $k \in \{1, 2, \ldots, n\}$, how to deal with this situation?
- How to control the approximate minimum yielded from the SQA?

The answers to these three questions are helpful for the application of the SQA.

The choice of these three points is critical for the performance of the SQA. The similar heuristic choice in Ali *et al.* (1997) is adopted here, except that, different from the use of the SQA in single-objective optimization, SQA as a local search part of the MOEA/D-SQA needs to be executed in each subproblem. For each subproblem i ($i = 1, \ldots, N$), compare the scalar aggregation function values of all individuals in the neighbourhood, order ascendingly and relabel them. Choose the best individual x^{i_a} with minimal scalar aggregation function value $g^{te}(x^{i_a}|\lambda^i, z)$, and randomly choose two distinct individuals x^{i_b}, x^{i_c} from the whole neighbourhood such that $x^{i_a} \neq x^{i_b} \neq x^{i_c}$. Instead of the whole neighbourhood, the choice range of these two individuals can be retricted to a certain part of the neighbourhood to balance the diversity and convergence. The range of the part changes dynamically based on the variation of the current solution x^i during the last two generations. The scalar aggregation function values of these two random individuals are denoted by $g^{te}(x^{i_b}|\lambda^i, z)$, $g^{te}(x^{i_c}|\lambda^i, z)$, respectively.

Consider the extreme case, namely, $\mathcal{B}_k = 0$ for some $k \in \{1, 2, \ldots, n\}$. This extreme case may occur in two scenarios: (1) the kth components of $x^{i_a}, x^{i_b}, x^{i_c}$ happen to be equal; and (2) the sum

of these three terms is equal to zero. During the evolutionary process, this situation likely occurs when the diversity of the population in the same neighbourhood decreases gradually. Therefore, the effect of the SQA on the MOEA/D is limited at the last stage of evolution. If $\mathcal{B}_k = 0$ for some $k \in \{1, 2, \ldots, n\}$, the value of $x_k^{i'}$ is set to be equal to $x_k^{i_a}$ for improving the accuracy of the local search in experiments.

For the bound-constrained multi-objective optimization, an approximate minimum derived from the SQA is either infeasible or feasible, which is helpful for improving the diversity of the population. However, for the infeasible approximate minimum out of the boundary, it may slow the convergence of MOEA/D-SQA. In this case, it will be adjusted in advance. The process of controlling the approximate minimum $x^{i'} = (x_1^{i'}, x_2^{i'}, \ldots, x_n^{i'})^{\mathrm{T}}$ is described as follows.

If $\mathcal{B}_k \neq 0$ for $k \in \{1, 2, \ldots, n\}$, calculate $x_k^{i'}$ by using Equation (3), otherwise, let $x_k^{i'} = x_k^{i_a}$. Then, adjust the value of the trial approximate minimum: if $x_k^{i'} < L_k$, let $x_k^{i'} = L_k$; if $x_k^{i'} > U_k$, let $x_k^{i'} = U_k$. L_k and U_k are the lower bound and upper bound of the kth decision variable, respectively.

In order to improve the accuracy of the scalar aggregation function value, the following replacement strategy is adopted for each subproblem i. If $g^{te}(x^{i'}|\lambda^i, z) \leq g^{te}(x^i|\lambda^i, z)$, then replace the current solution to the ith subproblem with $x^{i'}$. If $g^{te}(x^{i'}|\lambda^i, z) > g^{te}(x^i|\lambda^i, z)$, maintain the current solution x^i unchanged.

Now consider the computational cost of the proposed MOEA/D-SQA within one iteration, which is measured by the number of function evaluations (FEs). Note that MOEA/D-SQA converts an MOP into N subproblems and optimizes them simultaneously in a single run; the SQA is performed once for each subproblem in each run. Then the approximate minimum is evaluated at most N times in each run. Thus the number of FEs required by MOEA/D-SQA in one iteration is approximately equal to the number of FEs required by MOEA/D in one iteration plus N.

Along with the new version of MOEA/D (Zhang *et al.* 2009) in the CEC2009 MOEA competition, the MOEA/D-SQA proposed in this article adopts a dynamical resource allocation strategy for conquering complicated multi-objective problems. Each subproblem i is distributed to a utility π^i. In each generation, which subproblems are involved in the evolution are based on their utilities. During the search, MOEA/D-SQA with the Tchebycheff approach maintains:

- A population of N points $x^1, \ldots, x^N \in \Omega$, where x^i is the current solution to the ith subproblem;
- $FV^1, \ldots, FV^N$: FV^i is the F-value of x^i, *i.e.* $FV^i = F(x^i)$ for each $i = 1, \ldots, N$;
- $z = (z_1, \ldots, z_m)^{\mathrm{T}}$: z_i is the best value found so far for objective f_i;
- $\pi^1, \ldots, \pi^N$: π^i is the utility of subproblem i;
- *gen*: the current generation number.

Algorithm 2 – The MOEA/D-SQA algorithm
Input:

- MOP(1);
- stopping criteria;
- N: the number of subproblems considered in MOEA/D-SQA;
- $\lambda^1, \ldots, \lambda^N$: a uniform spread of N weight vectors;
- T: the number of the weight vectors in the neighbourhood of each weight vector.

Output:

- $\{x^1, \ldots, x^N\}$ and $\{F(x^1), \ldots, F(x^N)\}$.

Step 1. Initialization

Step 1.1: Compute the Euclidean distances between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.

Step 1.2: Generate an initial population $x^1, \ldots, x^N$ by uniformly sampling from the search space. Set $FV^i = F(x^i)$.
Step 1.3: Initialize $z = (z_1, \ldots, z_m)^{\mathrm{T}}$ by setting $z_j = \min_{1 \le i \le N} f_j(x^i), j = 1, \ldots, m$.
Step 1.4: Set $gen = 0$ and $\pi^i = 1$ for all $i = 1, \ldots, N$.

Step 2. Selection of subproblems for search

By using 10-tournament selection based on π^i, select $\left[\frac{N}{5}\right]$ indexes to form I.

Step 3. Update

For each $i \in I$, do:

Step 3.1. Selection of mating/update range: Generate a number *rand* from [0, 1] randomly. Then set

$$P = \begin{cases} B(i) & \text{if } rand < \delta, \\ \{1, \ldots, N\} & \text{otherwise.} \end{cases}$$

δ is the probability that parent solutions are selected from the neighbourhood.
Step 3.2. Reproduction: Set $r_1 = i$ and randomly select two indexes r_2, r_3 from P, then generate a solution $\bar{y}$ from x^{r_1}, x^{r_2} and x^{r_3} by a DE operator, and then perform a mutation operator on $\bar{y}$ with probability p_m to produce a new solution y.
Step 3.3. Repair: If an element of y is out of the boundary, its value is reset to be a randomly selected value inside the boundary.
Step 3.4. Update of z: For each $j = 1, \ldots, m$, if $z_j > f_j(y)$, then set $z_j = f_j(y)$.
Step 3.5. Update of solutions: Set $c = 0$ and then do the following:
(1) If $c = \eta_r$ or P is empty, go to **Step 3.6**. Otherwise, randomly pick an index j from P.
(2) If $g^{te}(y|\lambda^j, z) < g^{te}(x^j|\lambda^j, z)$, then set $x^j = y, FV^j = F(y)$ and $c = c + 1$.
(3) Remove j from P and go to (1).
Step 3.6. Local search: Compute $g^{te}(x^{i_k}|\lambda^i, z)$ for each $k = 1, \ldots, T$, where $i_k \in B(i)$. Set $G = \{g^{te}(x^{i_k}|\lambda^i, z), k = 1, \ldots, T\}$. Select the three best points x^{i_a}, x^{i_b} and x^{i_c} corresponding to the minimal scalar aggregation function values in G, where

$$g^{te}(x^{i_a}|\lambda^i, z) < g^{te}(x^{i_b}|\lambda^i, z) < g^{te}(x^{i_c}|\lambda^i, z).$$

$g^{te}(x^{i_k}|\lambda^i, z)$ is abbreviated hereafter as $g^{te}_{i_k}$. Let

$$x^{i_a} = (x_1^{i_a}, \ldots, x_n^{i_a})^{\mathrm{T}}, \quad x^{i_b} = (x_1^{i_b}, \ldots, x_n^{i_b})^{\mathrm{T}} \quad \text{and} \quad x^{i_c} = (x_1^{i_c}, \ldots, x_n^{i_c})^{\mathrm{T}},$$

n is the number of variables. Compute the trial vector $x^{i'} = (x_1^{i'}, x_2^{i'}, \ldots, x_n^{i'})^{\mathrm{T}}$ in the following.
If $(x_j^{i_a} - x_j^{i_c})g^{te}_{i_b} + (x_j^{i_c} - x_j^{i_b})g^{te}_{i_a} + (x_j^{i_b} - x_j^{i_a})g^{te}_{i_c} < \epsilon$, set

$$x_j^{i'} = x_j^i,$$

else

$$x_j^{i'} = \frac{1}{2} \frac{[(x_j^{i_a})^2 - (x_j^{i_c})^2]g^{te}_{i_b} + [(x_j^{i_c})^2 - (x_j^{i_b})^2]g^{te}_{i_a} + [(x_j^{i_b})^2 - (x_j^{i_a})^2]g^{te}_{i_c}}{(x_j^{i_a} - x_j^{i_c})g^{te}_{i_b} + (x_j^{i_c} - x_j^{i_b})g^{te}_{i_a} + (x_j^{i_b} - x_j^{i_a})g^{te}_{i_c}},$$

$j = 1, \ldots, n$, where ϵ is a suitable small positive value (*e.g.* 10^{-6}). Then, if

$$g^{te}(x^{i'}|\lambda^i, z) \le g^{te}(x^i|\lambda^i, z),$$

set

$$x^i = x^{i'}.$$

Step 4. Stopping criteria: If the stopping criteria are satisfied, then stop and output $\{x^1, \ldots, x^N\}$ and $\{F(x^1), \ldots, F(x^N)\}$.

Step 5. *gen* = *gen* + 1: If *gen* is a multiple of 50, then compute Δ^i, the relative decrease of the objective for each subproblem i during the last 50 generations, update

$$\pi^i = \begin{cases} 1 & \text{if } \Delta^i > 0.001, \\ \left(0.95 + 0.05\dfrac{\Delta^i}{0.001}\right)\pi^i & \text{otherwise.} \end{cases}$$

Go to **Step 2**.

In 10-tournament selection in **Step 2**, the index with the highest π^i value from 10 uniformly randomly selected indexes is chosen to enter I. This selection should be done $\left\lceil \frac{N}{5} \right\rceil$ times.

In **Step 5**, the relative decrease is defined as

$$\Delta^i = \frac{g^{te}(x^i_{\text{old}}|\lambda^i, z) - g^{te}(x^i_{\text{new}}|\lambda^i, z)}{g^{te}(x^i_{\text{old}}|\lambda^i, z)}; \tag{5}$$

if Δ^i is smaller than 0.001, the value of π^i will be reduced.

In **Step 3.6**, two distinct individuals x^{i_b}, x^{i_c} are randomly selected from the whole neighbourhood. In fact, as presented in the beginning of Section 5, the selection range can be reduced to a certain extent in experiments for balancing the diversity and efficiency. The two distinct individuals are selected from part of the neighbourhood, and the range of the part changes dynamically based on the variation of the current solution during the last two generations. The greater is the variation of the current solution x^i, the smaller the selection range will be, and, vice versa, when the solution x^i changes very little, the selection range will be broadened to the entire neighbourhood.

In the DE operator used in **Step 3.2**, the variant DE/best/1 is used:

$$\text{DE/best/1}: v_k = x_k^{r_1} + F \times (x_k^{r_2} - x_k^{r_3}). \tag{6}$$

Each element $\bar{y}_k$ in $\bar{y} = (\bar{y}_1, \ldots, \bar{y}_n)$ is generated as follows:

$$\bar{y}_k = \begin{cases} v_k & \text{with probability } CR, \\ x_k^{r_1} & \text{with probability } 1 - CR, \end{cases} \tag{7}$$

where CR and F are two control parameters.

6. Experimental study

To evaluate the performance of the proposed MOEA/D-SQA, 13 benchmark problems with complicated PS shapes UF1–UF13 (Zhang *et al.* 2008) are tested, and the results obtained by MOEA/D-SQA are compared with those obtained by MOEA/D (Zhang *et al.* 2009), as well as the results provided by Chen *et al.* (2009), Gao *et al.* (2009), Huang *et al.* (2009), Kukkonen and Lampinen (2009), Liu and Li (2009), Liu *et al.* (2009), Qu and Suganthan (2009), Sindhya *et al.* (2009), Tiwari *et al.* (2009), Tseng and Chen (2009), Wang *et al.* (2009) and Zamuda *et al.* (2009). Among these 13 problems, the first 7 are 2-objective problems and the next 3 are 3-objective while the last 3 are 5-objective problems. The number of decision variables is 30 for UF1–UF13. All these test instances are minimizations of the objectives. The experiments are implemented on a personal computer (AMD Athlon (tm) 64 × 2 Dual Core Processor 3800+ 2.0 GHz, 1 GB RAM). The programming languages are MATLAB™ and C++.

6.1. *Performance metric*

In order to compare the performance of the different algorithms quantitatively, performance metrics are needed. There are two goals in a multi-objective optimization: (1) convergence to the Pareto optimal set, and (2) diversity of solutions in the Pareto optimal set. The performance metrics used in this article are described below (Fan and Chang 2009).

- Inverted generational distance (IGD) (Zitzler *et al.* 2003): Let P^* be a set of uniformly distributed points in the objective space along the PF. Let A be an approximation to the PF, the inverted generational distance from P^* to A is defined as

$$\mathrm{IGD}(P^*, A) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}, \tag{8}$$

  where $d(v, A)$ is the minimum Euclidean distance between v and the points in A. If $|P^*|$ is large enough to represent the PF very well, $\mathrm{IGD}(P^*, A)$ could measure both the diversity and convergence of A in a sense. To have a low value of $\mathrm{IGD}(P^*, A)$, A must be very close to the PF and cannot miss any part of the whole PF.
- Generational distance (GD) (Van Veldhuizen and Lamont 1998, 1999): Let P^* be a set of uniformly distributed points in the objective space along the PF and A be an approximation to the PF; the generational distance measures the distance from A to P^*.

$$\begin{aligned} \mathrm{GD}(A, P^*) &= \frac{\sqrt{\sum_{v \in A} (d(v, P^*))^2}}{|A|} \\ d(v, P^*) &= \min_{x^* \in P^*} \left\{ \sqrt{\sum_{i=1}^{m} (f_i(x) - f_i(x^*))^2} \right\}, \end{aligned} \tag{9}$$

  where $d(v, P^*)$ is the minimum Euclidean distance between v and points in P^*. The generational distance measurement indicates how close a non-dominated solutions set is to the Pareto-optimal front. In other words, this metric indicates the 'degree' of convergence of a non-dominated solution set to any particular part of the Pareto-optimal front. The lower the value of the generational distance, the closer the non-dominated solution set is to a particular part of the true Pareto front indicating better performance of the algorithm. The value $\mathrm{GD}(A, P^*) = 0$ indicates that all solutions in A are Pareto-optimal solutions.
- Spacing (S): Schott (1995) proposed the spacing metric that measures the variance of the distance of each solution in P from its closest neighbour:

$$\begin{aligned} S(P) &= \sqrt{\frac{1}{|P| - 1} \sum_{x \in P} (\bar{d} - d_x)^2} \\ d_x &= \min_{x^* \neq x}^{x^* \in P} \left\{ \sum_{l=1}^{m} |f_l(x) - f_l(x^*)| \right\} \\ \bar{d} &= \frac{1}{|P|} \sum_{x \in P} d_x. \end{aligned} \tag{10}$$

  A lower variance is preferred as this indicates a better distribution of solutions in the Pareto set. The ideal value is 0 as this indicates that the distances from one solution to its closest neighbour is the same for every solution in the Pareto set, which means a uniform distribution of solutions in the Pareto set.

- Hypervolume (HV) (Zitzler and Thiele 1999): The hypervolume metric measures the size of the region which is dominated by the obtained Pareto front. Therefore a higher value of the HV-metric is preferred. In low-dimension, 2- and 3-objective spaces, it is known as area and volume, respectively. Mathematically, the HV-metric is described as

$$HV(P) = \left\{ \bigcup vol(v) | v \in P \right\}. \tag{11}$$

6.2. *Parameter setting*

The parameters in this article are listed below:

- N: 600 for 2-objective, 1000 for 3-objective, and 1500 for 5-objective problems;
- $\lambda^1, \ldots, \lambda^N$: obtained through the uniform design method (Fang and Lin 2003);
- $T = 0.1N$ and $\eta_r = 0.01N$;
- $\delta = 0.9$;
- in DE and mutation operators: $CR = 1.0$, $F = 0.5$ and $p_m = 1/n$;
- in local search: $\epsilon = 10^{-6}$;
- stopping condition: the algorithm stops after 300,000 function evaluations for each test instance.

In addition, the maximal number of solutions for computing the performance metric of MOEA/D-SQA is: 100 for 2-objective problems, 150 for 3-objective problems and 800 for 5-objective problems. Each of the test problems is run 30 times independently.

6.3. *Experimental results and analysis*

To see how the proposed MOEA/D-SQA strategy prevails in its superiority over MOEA/D, a comprehensive numerical comparison is carried out. Under the same stopping condition, MOEA/D-SQA is compared with MOEA/D in terms of the quality of the performance metric values and the efficiency of these two algorithms.

All the parameter settings are the same as those mentioned in Section 6.2. The statistical results of MOEA/D and the proposed MOEA/D-SQA are summarized in Tables 1–4. These tables present the obtained results for each benchmark problem and the statistics for 30 independent runs, including the best, mean, worst and standard deviation (std) of the performance metrics values. In

Table 1. IGD statistics of the MOEA/D-SQA and MOEA/D based on 30 independent runs.

	MOEA/D-SQA				MOEA/D				
Problems	mean	std	best	worst	mean	std	best	worst	SS
UF1	0.00431	0.00009	0.00416	0.00464	0.00435	0.00029	0.00399	0.00519	1
UF2	0.00565	0.00140	0.00443	0.01014	0.00679	0.00182	0.00481	0.01087	1
UF3	0.00468	0.00097	0.00410	0.00783	0.00742	0.00589	0.00394	0.02433	1
UF4	0.04384	0.00175	0.04050	0.04716	0.06385	0.00534	0.05687	0.08135	1
UF5	0.14506	0.12142	0.04310	0.55308	0.18071	0.06811	0.08028	0.30621	1
UF6	0.00317	0.00079	0.00213	0.00483	0.00587	0.00171	0.00342	0.01005	1
UF7	0.0150	0.05681	0.00409	0.31574	0.00444	0.00117	0.00405	0.01058	−1
UF8	0.05768	0.00857	0.04846	0.08646	0.05840	0.00321	0.05071	0.06556	1
UF9	0.09576	0.05706	0.03051	0.14585	0.07896	0.05316	0.03504	0.14985	−1
UF10	0.28908	0.10916	0.10737	0.54315	0.47415	0.07360	0.36405	0.64948	1
UF11	0.10642	0.01550	0.10093	0.18787	0.11032	0.00233	0.10692	0.11519	1
UF12	119.81665	33.59665	74.17540	206.0560	146.7813	41.8281	66.1690	214.2261	1
UF13	1.84765	0.00387	1.84150	1.85502	1.8489	0.0198	1.8346	1.8993	1

Table 2. GD statistics of the MOEA/D-SQA and MOEA/D based on 30 independent runs.

	MOEA/D-SQA				MOEA/D				
Problems	mean	std	best	worst	mean	std	best	worst	SS
UF1	3.14E−04	3.59E−04	1.18E−04	1.59E−03	2.51E−04	5.67E−05	1.73E−04	4.31E−04	−1
UF2	3.23E−04	1.69E−04	1.47E−04	9.70E−04	5.85E−04	2.32E−04	2.43E−04	1.14E−03	1
UF3	2.70E−04	2.61E−04	9.84E−05	1.12E−03	9.58E−04	0.001426	1.61E−04	7.30E−03	1
UF4	0.004762	2.45E−04	0.004379	0.005279	0.006872	5.65E−04	0.005717	0.007931	1
UF5	0.011802	0.014608	3.25E−04	0.063179	0.022788	0.018924	0.009077	0.094012	1
UF6	5.10E−04	2.41E−04	1.87E−04	1.26E−03	0.003311	0.001164	0.001305	0.006258	1
UF7	4.22E−04	0.001127	1.15E−04	6.37E−03	3.06E−04	9.94E−05	1.90E−04	7.20E−04	−1
UF8	0.002798	9.64E−04	0.001526	0.00534	0.004344	0.00215	0.002208	0.011577	1
UF9	0.018061	0.012467	0.003623	0.042936	0.011137	0.007049	0.004126	0.029052	−1
UF10	0.058179	0.050654	0.001855	0.195766	0.119853	0.031433	0.054126	0.194871	1
UF11	0.003887	1.08E−04	0.003451	0.004161	0.004213	9.40E−05	0.004088	0.004435	1
UF12	10.79164	2.528618	6.604107	15.11032	11.52052	2.073689	8.032672	17.76085	1
UF13	0.064774	1.60E−04	0.064479	0.065031	0.064324	1.94E−04	0.063861	0.064618	−1

Table 3. Spacing-metric statistics of the MOEA/D-SQA and MOEA/D based on 30 independent runs.

	MOEA/D-SQA				MOEA/D				
Problems	mean	std	best	worst	mean	std	best	worst	SS
UF1	0.00187	0.00083	0.001316	0.004792	0.0016	0.00041	8.92E−04	0.00268	−1
UF2	0.00163	0.00019	0.00143	0.002212	0.00165	0.00025	0.001193	0.002088	1
UF3	0.00147	0.00034	9.25E−04	0.002124	0.00188	0.00192	9.34E−04	0.011051	1
UF4	0.00158	0.00027	0.001367	0.002866	0.00175	0.00042	0.001473	0.003756	1
UF5	0.00169	0.0019	1.97E−05	0.009586	0.0048	0.00368	0.001201	0.01414	1
UF6	0.00186	0.00058	0.001486	0.003611	0.00244	0.0013	0.001455	0.006417	1
UF7	0.00103	0.00047	6.30E−04	0.003125	0.00106	0.00016	8.23E−04	0.001449	1
UF8	0.02491	0.00204	0.022063	0.031586	0.02509	0.00458	0.021425	0.046155	1
UF9	0.02666	0.01031	0.014705	0.060583	0.02114	0.00743	0.015116	0.05041	−1
UF10	0.05064	0.029	0.006044	0.121205	0.08648	0.03665	0.05072	0.250225	1
UF11	0.08812	0.00709	0.05135	0.091975	0.09845	0.00128	0.095744	0.100148	1
UF12	22.19167	13.45289	7.748953	63.88682	21.49553	11.06829	3.584912	46.57203	−1
UF13	0.14216	0.00309	0.13604	0.149694	0.13306	0.00444	0.126176	0.14297	−1

Table 4. HV-metric statistics of the MOEA/D-SQA and MOEA/D based on 30 independent runs.

	MOEA/D-SQA				MOEA/D				
Problems	mean	std	best	worst	mean	std	best	worst	SS
UF1	0.869729	0.000262	0.870022	0.868817	0.864819	0.016266	0.868923	0.778848	1
UF2	0.867623	0.001899	0.869434	0.862587	0.86445	0.002354	0.867525	0.858773	1
UF3	0.868396	0.003998	0.870683	0.855543	0.855288	0.027065	0.869887	0.7711	1
UF4	0.466894	0.003159	0.471387	0.459857	0.43311	0.008475	0.449267	0.418499	1
UF5	0.714896	0.136501	0.855269	0.39689	0.567042	0.149353	0.771583	0.176278	1
UF6	0.644661	0.000311	0.645138	0.643797	0.636407	0.008109	0.643869	0.615022	1
UF7	0.691282	0.055557	0.702994	0.397588	0.700654	0.000849	0.70162	0.698028	−1
UF8	1.916588	0.020388	1.932842	1.84431	1.898264	0.014962	1.913773	1.838528	1
UF9	2.171913	0.099342	2.285809	2.08281	2.220542	0.080294	2.275151	2.069364	−1
UF10	16.74861	2.835092	20.86084	13.10127	14.40157	1.385256	18.28202	11.70854	1
UF11	73.62631	0.091295	73.6542	73.14429	73.61587	0.008359	73.6337	73.60343	1
UF12	1.50E + 12	7.00E + 11	2.24E + 12	1.23E + 11	1.70E + 12	3.20E + 11	2.12E + 12	9.84E + 11	−1
UF13	1.41E + 03	2.231919	1417.031	1406.125	1.42E + 03	2.918914	1422.836	1410.811	−1

the last columns of Tables 1–4, the statistical significance (SS) of the advantage of the MOEA/D-SQA in the mean IGD-metric, GD-metric, spacing-metric and HV-metric value, respectively, is reported. The numerical values $-1, 0, 1$ indicate that the MOEA/D-SQA is statistically inferior to, equal to and superior to the MOEA/D in terms of mean performance metric value. As described in Tables 1–2, for almost all test problems, the mean, best and worst GD and IGD-metric values obtained by MOEA/D-SQA are smaller, respectively, than those obtained by MOEA/D, which indicates that MOEA/D-SQA performs better than MOEA/D in approximating the PF on both the diversity and convergence.

The spacing-metric numerically describes the spread of the vectors in PF. It is clear from Table 3 that the spacing-metric values obtained by MOEA/D-SQA and MOEA/D are very small except for UF12. MOEA/D-SQA performs better than MOEA/D in terms of mean spacing-metric value for all the test problems except UF1, UF9, UF12 and UF13, which indicates the solutions obtained by MOEA/D-SQA are spaced more evenly than those obtained by MOEA/D in general.

The hypervolume metric measures the size of the region dominated by the obtained Pareto front, *i.e.* the region of coverage of the obtained PF. Therefore the higher value of the HV-metric is preferred. As described in Table 4, MOEA/D-SQA has, generally speaking, better performance than MOEA/D in terms of the HV-metric.

In order to compare the convergence speed of the proposed algorithm, the convergence graphs of MOEA/D-SQA and MOEA/D on all the test problems are shown in Figure 1. It can be observed from Figure 1 that MOEA/D-SQA converges, in terms of function evaluations, much faster than MOEA/D in minimizing the IGD-metric values for all the problems except UF7 and UF9, which indicates that in most cases the simplified quadratic model is effective in accelerating the convergence and in improving the accuracy of the minimum scalar aggregation function value.

From Tables 1–4, it can be noted that the standard deviations of the performance metrics found by MOEA/D-SQA are small except for UF12, and the results obtained by MOEA/D-SQA are not greater than MOEA/D for almost all test instances. These results imply that MOEA/D-SQA is robust and stable in solving this kind of multi-objective problem.

Figure 2 plots the distribution in the objective space of the final population with the lowest IGD-metric value obtained in 30 runs on UF1–UF10. It is clear from the figure that MOEA/D-SQA is able to find a good approximation to UF1–UF3 and UF6–UF9. However, it fails, within the given number of function evaluations, to approximate satisfactorily the PFs of the problems UF4, UF5 and UF10, perhaps for the reason that incorporating a local search operator in an MOEA would, in some sense, spoil the diversity of the algorithm. However, as evidenced from Table 1 and Figure 1, in general the SQA as a local search operator can enhance the performance of MOEA/D.

Under the same number of iterations, the minimum, mean, maximum and standard deviation (std) of the running time used by MOEA/D-SQA and MOEA/D in 30 independent runs are summarized in Table 5. It is evident from Table 5 that, under the same number of iterations, incorporating SQA in MOEA/D has the effect of slowing down the algorithm. As analysed at the beginning of Section 5, the number of function evaluations required by MOEA/D-SQA in one iteration is approximately equal to those required by MOEA/D in one iteration plus N. Fortunately, the SQA is a simple local search strategy and it is convenient to compute under the same number of function evaluations. As shown in Tables 1–4 and Figure 1, the results obtained by MOEA/D-SQA are very competitive when compared with those of MOEA/D.

MOEA/D-SQA is also compared with several state-of-the-art MOEAs in terms of mean IGD-metric value based on 30 independent runs. Under the same stopping criteria, the results are shown in Tables 6 and 7. These compared MOEAs include GDE3 (Kukkonen and Lampinen 2009), MOEADGM (Chen *et al.* 2009), MTS (Tseng and Chen 2009), the LiuLi Algorithm (Liu and Li 2009), DMOEADD (Liu *et al.* 2009), NSGAIILS (Sindhya *et al.* 2009), OWMOSaDE (Huang *et al.* 2009), ClusteringMOEA (Wang *et al.* 2009), AMGA (Tiwari *et al.* 2009), MOEP (Qu and Suganthan 2009), DECMOSA-SQP (Zamuda *et al.* 2009) and OMOEAII (Gao *et al.*

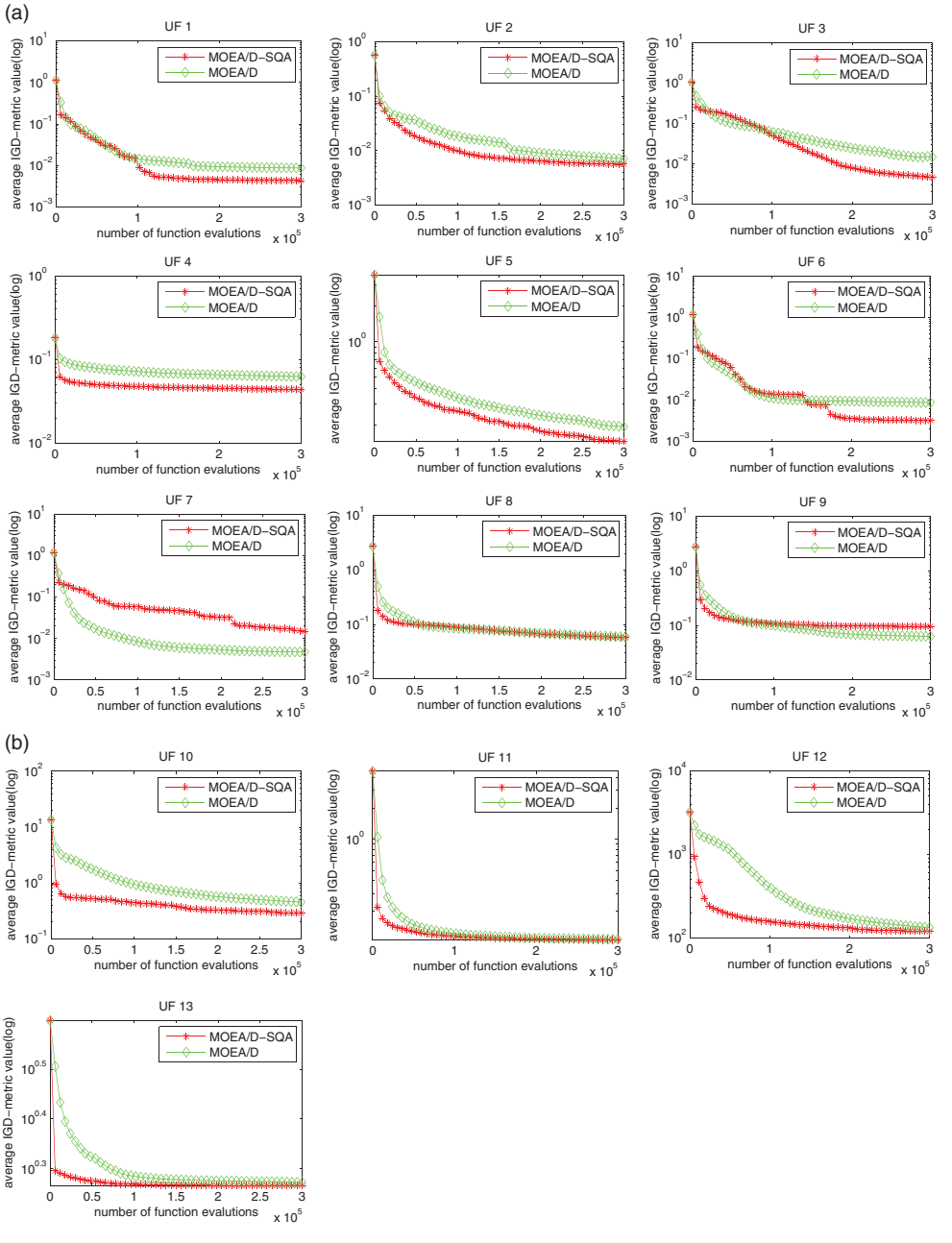

Figure 1. The convergence graphs of UF1–UF13.

2009), in which MOEADGM, AMGA and OMOEAII have not solved problems UF11–UF13 with five objectives in the related references; in these cases '–' is used to indicate the unobtained IGD-metric values in Tables 6 and 7. As shown in the right-most column in Table 7, the overall rank of MOEA/D is improved from 54 to 33 because of the SQA, and MOEA/D-SQA ranks first among these 14 algorithms. All these results demonstrate the competitive performance of MOEA/D-SQA.

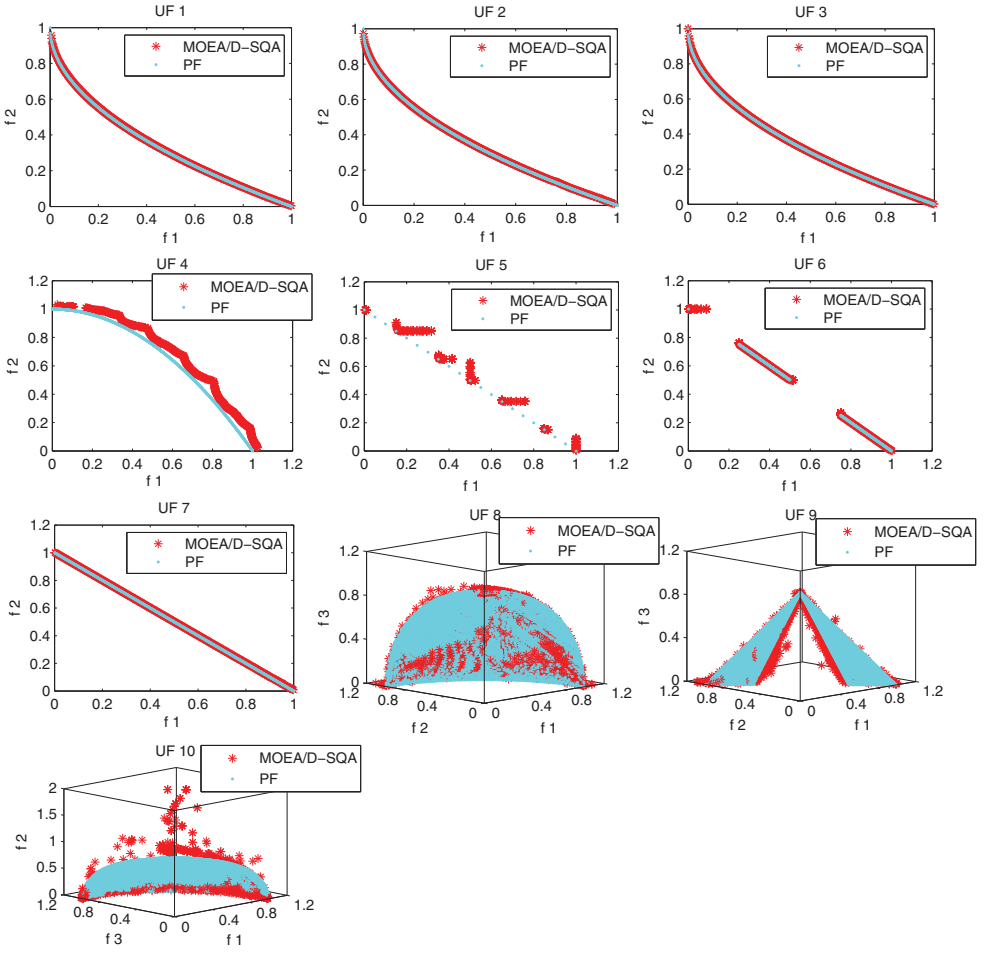

Figure 2. The best approximation to UF1–UF10 obtained by MOEA/D-SQA.

Table 5. Running time statistics of the MOEA/D-SQA and MOEA/D based on 30 independent runs.

		MOEA/D-SQA				MOEA/D			
Problems	Iterations	mean	std	min	max	mean	std	min	max
UF1	2500	133.6768	0.866284	131.962	135.482	106.4901	1.103806	104.469	110.281
UF2	2500	134.6433	1.119939	133.0375	137.499	109.0568	0.583938	107.359	110.5
UF3	2500	130.1258	1.875836	126.5155	135.1095	104.6026	1.194171	101.734	106.625
UF4	2500	139.4524	5.552145	133.6015	162.258	108.7151	0.834213	106.906	109.984
UF5	2500	113.9904	2.416524	110.078	118.539	89.82603	1.836333	84.531	92.656
UF6	2500	133.7125	3.754755	129.547	141.141	100.5667	1.294295	98.172	103.422
UF7	2500	136.2971	6.931642	127.672	156.57	107.0885	0.475432	106.203	108.078
UF8	1500	392.8701	2.494837	389.3985	398.875	326.5146	1.143894	324.141	328.766
UF9	1500	395.6438	12.68123	385.18	436.406	346.8833	22.09957	329.5	401.906
UF10	1500	365.6422	12.82532	348	411.4765	309.3286	23.94505	288.75	359.812
UF11	1000	1107.185	51.65228	1066.135	1283.39	1096.969	120.3371	1020.8	1462.13
UF12	1000	888.9278	22.08981	847.51	954.365	792.5484	26.10414	752.219	847.047
UF13	1000	1126.186	95.59277	1052.235	1409.54	1072.608	43.06959	1049.36	1288.45

Table 6. The rank of mean IGD-metric values on test problems UF1–UF7.

Algorithm	UF1	UF2	UF3	UF4	UF5	UF6	UF7	Rank
MOEA/D-SQA	0.00431	0.00565	0.00468	0.04384	0.14506	0.00317	0.01500	21
	1	1	1	8	4	1	5	
MOEA/D	0.00435	0.00679	0.00742	0.06385	0.18071	0.00587	0.00444	33
	2	4	2	14	8	2	1	
GDE3	0.00534	0.01195	0.10639	0.0265	0.03928	0.25091	0.02522	49
	3	7	13	2	2	12	10	
MOEADGM	0.0062	0.0064	0.049	0.0476	1.7919	0.5563	0.0076	53
	4	3	5	10	14	14	3	
MTS	0.00646	0.00615	0.0531	0.02356	0.01489	0.05917	0.04079	30
	5	2	6	1	1	3	12	
LiuLiAlgorithm	0.00785	0.0123	0.01497	0.0435	0.16186	0.17555	0.0073	41
	6	8	3	7	5	10	2	
DMOEADD	0.01038	0.00679	0.03337	0.04268	0.31454	0.06673	0.01032	39
	7	4	4	5	11	4	4	
NSGAIILS	0.01153	0.01237	0.10603	0.0584	0.5657	0.31032	0.02132	74
	8	9	12	12	13	13	7	
OWMOSaDE	0.0122	0.0081	0.103	0.0513	0.4303	0.1918	0.0585	74
	9	6	11	11	12	11	14	
ClusteringMOEA	0.0299	0.0228	0.0549	0.0585	0.2473	0.0871	0.0223	66
	10	12	7	13	10	6	8	
AMGA	0.03588	0.01623	0.06998	0.04062	0.09405	0.12942	0.05707	58
	11	10	8	4	3	9	13	
MOEP	0.0596	0.0189	0.099	0.0427	0.2245	0.1031	0.0197	61
	12	11	10	6	9	7	6	
DECMOSA-SQP	0.07702	0.02834	0.0935	0.03392	0.16713	0.12604	0.02416	61
	13	13	9	3	6	8	9	
OMOEAII	0.08564	0.03057	0.27141	0.04624	0.1692	0.07338	0.03354	74
	14	14	14	9	7	5	11	

6.4. *Parameter sensitivity study*

To investigate the impact of parameter settings on the performances of MOEA/D-SQA proposed in this article, different settings of N (population size), T (size of neighbourhood), CR (crossover control parameter) and F (mutation factor) have been tested in the algorithm.

(1) Setting of population size N: Population size N is an important control parameter in MOEA. Decision makers may not want to have a huge number of Pareto optimal solutions at high computational cost. They are often interested in obtaining a small number of evenly distributed solutions at low computational cost. To investigate the impact of population size on the performances of MOEA/D-SQA, different settings of N in the algorithm on UF1 with two objectives, UF8 with three objectives, and UF13 with five objectives have been tested. The algorithm stops after a given number of generations, the maximal number of generations is set to be 2500, 1500 and 1000 for 2-objective UF1, 3-objective UF8 and 5-objective UF13, respectively. All the other parameters remain the same as in Section 6.2. Figure 3 reveals the IGD-metric values in the algorithm with different settings of N. It is evident that MOEA/D-SQA is not very sensitive to the setting of N under the ranges considered. It also suggests that the proposed MOEA/D-SQA is adequately stable and robust.

(2) Sensitivity of T: T is a major control parameter in MOEA/D-SQA. To study the sensitivity of the performance to T, different settings of T in the implementation of MOEA/D-SQA for 2-objective UF2, 3-objective UF10 and 5-objective UF11 have been tested. All the other parameter settings are the same as in Section 6.2 except the setting of T. As clearly shown in Figure 4, MOEA/D-SQA performs well with T from $0.04N$ to $0.5N$ (N is the population size, *i.e.* the number of subproblems) on UF10 and UF11, meanwhile it works very well for all

Table 7. The rank of mean IGD-metric values on test problems UF8–UF13.

Algorithm	UF8	UF9	UF10	UF11	UF12	UF13	Rank	Overall rank
MOEA/D-SQA	0.05768 1	0.09576 6	0.28908 2	0.10642 1	119.81665 1	1.84765 1	12	33(1)
MOEA/D	0.05840 2	0.07896 3	0.47415 10	0.11032 2	146.7813 2	1.8489 2	21	54(2)
GDE3	0.24855 13	0.08248 4	0.43326 8	0.23425 5	202.12 4	3.2057 8	42	91(6)
MOEADGM	0.2446 12	0.1878 10	0.5646 11	– 12	– 12	– 12	69	122(11)
MTS	0.11251 7	0.11442 8	0.15306 1	0.45505 9	305.2 5	1.9079 3	33	63(3)
LiuLiAlgorithm	0.08235 4	0.09391 5	0.44691 9	0.13254 3	444.82 6	2.2884 7	34	75(5)
DMOEADD	0.06841 3	0.04896 1	0.32211 3	1.20328 10	477.65 7	1.9971 5	29	68(4)
NSGAIILS	0.0863 5	0.0719 2	0.84468 14	0.1752 4	158.05 3	3.2323 9	37	111(8)
OWMOSaDE	0.0945 6	0.0983 7	0.743 13	0.3951 7	734.56 8	3.2573 10	51	125(12)
ClusteringMOEA	0.2383 11	0.2934 13	0.4111 7	1.2401 11	1039.36 11	3.4043 11	64	130(13)
AMGA	0.17125 8	0.18861 11	0.32418 4	– 12	– 12	– 12	59	117(9)
MOEP	0.423 14	0.342 14	0.3621 5	0.4337 8	885.89 9	2.0145 6	56	117(9)
DECMOSA-SQP	0.21583 10	0.14111 9	0.36985 6	0.38304 6	943.35 10	1.9178 4	45	106(7)
OMOEAII	0.192 9	0.23179 12	0.62754 12	– 12	– 12	– 12	69	143(14)

the values of T except very small ones on UF2. Thus, it can be claimed that MOEA/D-SQA is not very sensitive to the setting of T, at least for MOPs that are fairly similar to these test instances. The reason that MOEA/D-SQA does not work well when T is very small could be because MOEA/D-SQA with a T that is too small is poor at exploration. Figure 4 also reveals that MOEA/D-SQA with a large T works poorly on UF10 and UF11. This could be because the solutions to two subproblems with very different weight vectors are far different in the two problems.

(3) Setting of CR and F: CR and F are two important control parameters in the DE operator used in MOEA/D-SQA. Both CR and F remain fixed during the entire execution of the algorithm. Parameter $CR \in [0, 1]$, which controls the crossover operation, represents the probability that an element for the trial vector is chosen from a linear combination of three randomly chosen vectors and not from the old vector. Parameter F is a scaling factor for mutation and its value range is $(0, 1]$. In practice, CR controls rotational invariance of the search, and its small value (*e.g.* 0.1) is practicable with separable problems while larger values (*e.g.* 0.9) are for non-separable problems. Parameter F controls the speed and robustness of the search, *i.e.* a lower value for F increases the convergence rate but it also increases the risk of getting stuck into a local optimum. Parameters CR and N have an effect on the convergence rate similar to that of F. To investigate the impact of CR and F on the performances of MOEA/D-SQA, different combinations of CR and F in the algorithm on UF8 have been tested. All the other parameters remain the same as in Section 6.2. Figure 5 clearly shows the IGD-metric values obtained by the algorithm for different CR and F values. It can be noticed that, according to the IGD-metric values, larger values of CR and smaller values of F provide much better results in MOEA/D-SQA.

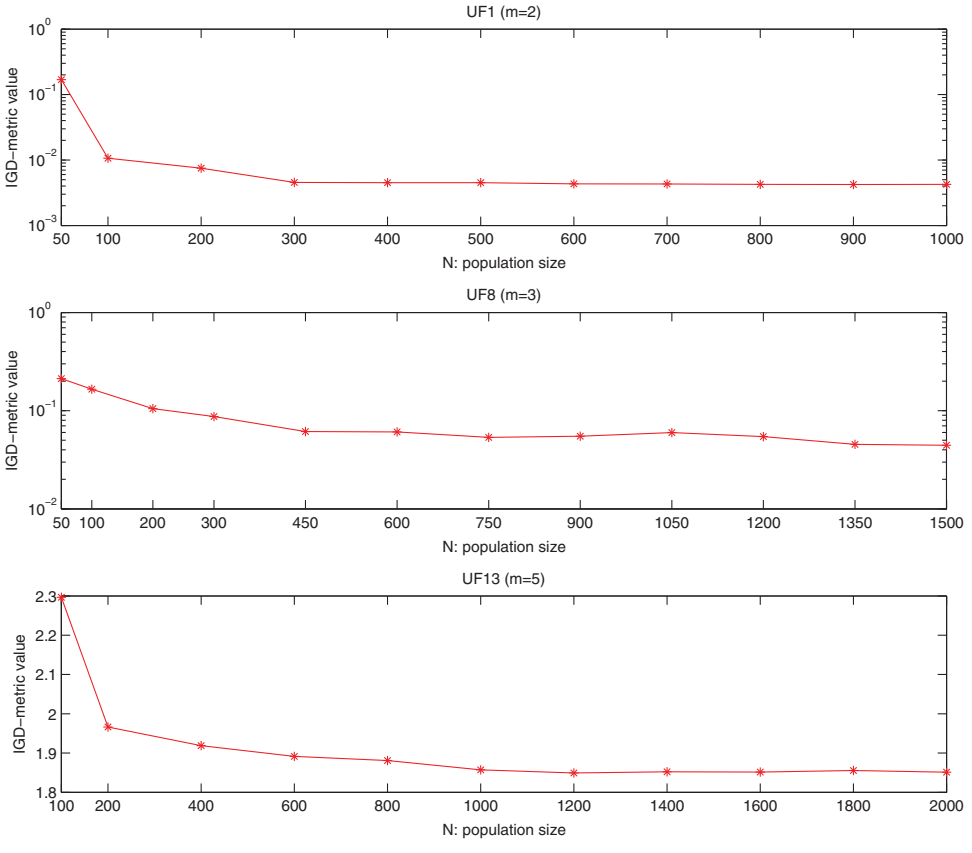

Figure 3. Performance evaluation with varying N values.

7. Conclusion

This article presents a multi-objective memetic algorithm based on decomposition named MOEA/D-SQA in order to accelerate the MOEA/D and also to improve its performance. MOEA/D-SQA is a new version of the hybrid MOEA/D with local search, in which MOEA/D is used as a main algorithm to implement the global exploration in order to maintain the diversity of the population, while the SQA used as a local search scheme is integrated in MOEA/D to accelerate its convergence speed and also to improve the accuracy of the resulting solution.

MOEA/D-SQA has been tested on 13 benchmark problems with complicated PS shapes. In the experiments, MOEA/D-SQA uses the same parameter suite and stopping criterion as the MOEA/D. The experimental results show that the SQA can enhance the convergence performance of the MOEA/D and improve the local search ability of the algorithm. The close integration of the MOEA/D with the SQA makes the proposed algorithm converge fast and adapt itself to the problem being solved. Moreover, the proposed MOEA/D-SQA performs the best among 14 different algorithms reported in the literature on the benchmark test problems used in this work.

The SQA is a simple technique that can be hybridized with other new MOEA/D variants without increasing the computational burden too much. How to hybridize the SQA with some other MOEA/D variants and improve their efficiency remains to be studied further.

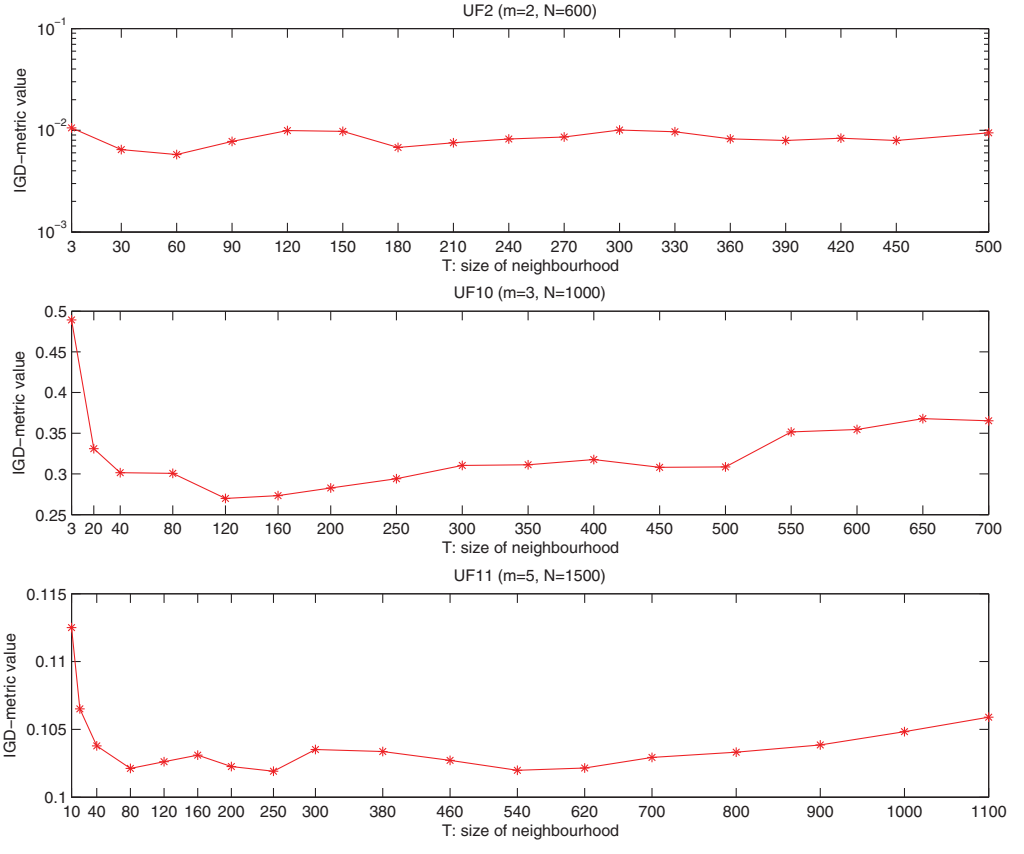

Figure 4. Performance evaluation with varying T values.

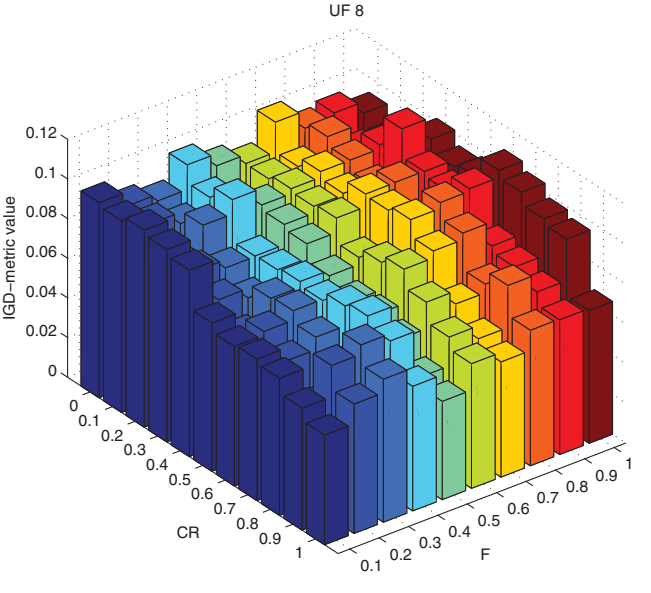

Figure 5. Performance evaluation with varying CR and F values.

Acknowledgements

The authors gratefully thank Professor Qingfu Zhang for his constructive suggestions on this work.

References

Ali, M.M., Törn, A., and Vitanen, S., 1997. A numerical comparison of some modified controlled random search algorithms. *Journal of Global Optimization*, 11(4), 377–385.

Beume, N., Naujoks, B., and Emmerich, M., 2007. SMS-EMOA: multi-objective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3), 1653–1669.

Brachetti, P., *et al.*, 1997. A new version of the Price's algorithm for global optimization. *Journal of Global Optimization*, 10 (2), 165–184.

Chakraborty, U.K., Das, S., and Konar, A., 2006. Differential evolution with local neighborhood. *In*: *IEEE congress on evolutionary computation (CEC'06)*, 16–21 July, Vancouver, BC, Canada. Piscataway, NJ: IEEE Press, 2042–2049.

Chang, P.C., *et al.*, 2008. MOEA/D for flowshop scheduling problems. *In*: *IEEE congress on evolutionary computation (CEC'08)*, 1–6 June, Hong Kong, PR China. Piscataway, NJ: IEEE Press, 1433–1438.

Chen, C.M., Chen, Y.P., and Zhang, Q., 2009. Enhancing MOEA/D with guided mutation and priority update for multi-objective optimization. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 209–216.

Coello, C.A.C., Pulido, G.T., and Lechuga, M.S., 2004. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8 (3), 256–279.

Deb, K., *et al.*, 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6 (2), 182–197.

Du, J., Cai, Z.H., and Chen, Y.L., 2007. A sorting based algorithm for finding non-dominated set in multi-objective optimization. *In*: *Third international conference on natural computation (ICNC 2007)*, 24–27 August, Hainan, China. Washington, DC: IEEE Computer Society, 417–421.

Fan, S.K.S. and Chang, J.M., 2009. A parallel particle swarm optimization algorithm for multi-objective optimization problems. *Engineering Optimization*, 41 (7), 673–697.

Fang, K.T. and Lin, D.K.J., 2003. Uniform designs and their application in industry. *In*: R. Khattree and C.R. Rao, eds. *Handbook of Statistics 22: Statistics in Industry*. Amsterdam, The Netherlands: Elsevier/North-Holland, 131–170.

Gao, S., *et al.*, 2009. An orthogonal multi-objective evolutionary algorithm with lower-dimensional crossover. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 1959–1964.

Ghomsheh, V.S., Khanehsar, M.A., and Teshnehlab, M., 2007. Improving the non-dominate sorting genetic algorithm for multi-objective optimization. *In*: *Proceedings of the IEEE international conference on micro electro mechanical systems (MEMS)*, 21–25 January, Kobe, Japan. Washington, DC: IEEE Computer Society, 89–92.

Goldberg, D. and Voessner, S., 1999. Optimizing global–local search hybrids. *In*: *Proceedings of the genetic and evolutionary computation conference (GECCO'99)*, 13–17 July, Orlando, Florida, USA. San Francisco, CA: Morgan Kaufmann Publishers, 220–228.

Huang, V.L., *et al.*, 2009. Multi-objective optimization using self-adaptive differential evolution algorithm. *In*: *IEEE congress on evolutionary computation (CEC'09)*, Trondheim, Norway. Piscataway, NJ: IEEE Press, 190–194.

Ishibuchi, H., *et al.*, 2009. Adaptation of scalarizing functions in MOEA/D: an adaptive scalarizing function based multiobjective evolutionary algorithm. *In*: *Proceedings of the 5th international conference devoted to evolutionary multi-criterion optimization (EMO'09)*, 7–10 April, Nantes, France. Berlin: Springer-Verlag, 438–452.

Ismail, F.S., Yusof, R., and Khalid, M., 2011. Self organizing multi-objective optimization problem. *International Journal of Innovative Computing, Information and Control*, 7 (1), 301–314.

Jaszkiewicz, A., 2002. Discrete optimization, genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137 (1), 50–71.

Jiao, Y.C., *et al.*, 2006. A modification to the new version of the Price's algorithm for continuous global optimization problems. *Journal of Global Optimization*, 36 (4), 609–626.

Kim, J.H. and Myung, H., 1997. Evolutionary programming techniques for constrained optimization problems. *IEEE Transactions on Evolutionary Computation*, 1 (2), 129–140.

Knowles, J. and Corne, D., 2005. Memetic algorithms for multi-objective optimization: issues, methods and prospects. *In*: W. Hart, J. Smith and N. Krasnogor, eds. *Recent advances in memetic algorithms*. Studies in fuzziness and soft computing Vol. 166. Berlin: Springer-Verlag, 313–352.

Krasnogor, N. and Smith, J.E., 2005. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, 9 (5), 474–488.

Kukkonen, S. and Lampinen, J., 2009. Performance assessment of generalized differential evolution 3 with a given set of constrained multi-objective test problems. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 3593–3600.

Li, H., Jiao, Y.C., and Wang, Y., 2005. Integrating the simplified interpolation into the genetic algorithm for constrained optimization problems. *In*: *International conference on computational intelligence and security (CIS2005)*, Part 1, Lecture notes in artificial intelligence 3801, 15–19 December, Xi'an, China. Berlin: Springer-Verlag, 247–254.

Li, H., Jiao, Y.C., and Zhang, L., 2011. Hybrid differential evolution with a simplified quadratic approximation for constrained optimization problems. *Engineering Optimization*, 43 (2), 115–134.
Li, H. and Zhang, Q., 2009. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 12 (2), 284–302.
Liu, H.L. and Li, X.Q., 2009. The multiobjective evolutionary algorithm based on determined weight and sub-regional search. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 1928–1934.
Liu, M.Z., *et al.*, 2009. Performance assessment of DMOEA-DD with CEC 2009 MOEA competition test instances. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 2913–2918.
Miettinen, K., 1999. *Nonlinear multiobjective optimization*. Boston, MA: Kluwer Academic.
Moscato, P., 1999. Memetic algorithms: a short introduction. *In*: D. Corne, M. Dorigo, and F. Glover, eds. *New ideas in optimization*. Maidenhead, UK: McGraw-Hill, 219–234.
Noman, N. and Iba, H., 2008. Accelerating differential evolution using an adaptive local search. *IEEE Transactions on Evolutionary Computation*, 12 (1), 107–125.
Ong, Y.S., *et al.*, 2006. Classification of adaptive memetic algorithms: a comparative study. *IEEE Transactions on Systems, Man and Cybernetics – Part B, Cybernetics*, 36 (1), 141–152.
Qu, B.Y. and Suganthan, P.N., 2009. Multi-objective evolutionary programming without non-domination sorting is up to twenty times faster. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 2934–2939.
Rahnamayan, S., Tizhoosh, H.R., and Salama, M.M.A., 2008. Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation*, 12 (1), 64–79.
Ruzika, S. and Wiecek, M., 2005. Approximation methods in multiobjective programming. *Journal of Optimization Theory and Applications*, 126 (3), 473–501.
Samanlioglu, F., Ferrell, W.G., and Kurz, M.E., 2008. A memetic random-key genetic algorithm for a symmetric multi-objective traveling salesman problem. *Computers & Industrial Engineering*, 55 (2), 439–449.
Schott, J., 1995. Fault tolerant design using single and multicriteria genetic algorithm optimization. Thesis (Master's). Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA.
Sindhya, K., *et al.*, 2009. Local search based evolutionary multi-objective optimization algorithm for constrained and unconstrained problems. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 2919–2926.
Smith, J.E., 2007. Coevolving memetic algorithms: a review and progress report. *IEEE Transactions on Systems, Man and Cybernetics – Part B, Cybernetics*, 37 (1), 6–17.
Takahama, T. and Sakai, S., 2006. Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites. *In*: *IEEE congress on evolutionary computation (CEC'06)*, 16–21 July, Vancouver, BC, Canada. Piscataway, NJ: IEEE Press, 1–8.
Tiwari, S., *et al.*, 2009. Performance assessment of the hybrid archive-based micro genetic algorithm (AMGA) on the CEC09 test problems. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 1935–1942.
Tseng, L.Y. and Chen, C., 2009. Multiple trajectory search for unconstrained/constrained multi-objective optimization. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 1951–1958.
Van Veldhuizen, D.A. and Lamont, G.B., 1998. Evolutionary computation and convergence to a Pareto front. *In*: J.R. Koza, ed. *Late breaking papers at the 1998 genetic programming conference*, 22–25 July, University of Wisconsin, Madison, WI, USA. Stanford, CA: Stanford University Bookstore, 221–228.
Van Veldhuizen, D.A. and Lamont, G.B., 1999. Multiobjective evolutionary algorithm test suites. *In*: J. Carroll, *et al.*, eds. *Proceedings of the 1999 ACM symposium on applied computing*, 28 February–2 March, San Antonio, TX. New York: ACM Press, 351–357.
Wang, Y., *et al.*, 2007. Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems. *IEEE Transactions on Systems, Man and Cybernetics – Part B, Cybernetics*, 37 (3), 560–575.
Wang, Y.P., *et al.*, 2009. A clustering multi-objective evolutionary algorithm based on orthogonal and uniform design. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 2927–2933.
Wiecek, M.M., Chen, W., and Zhang, J., 2001. Piecewise quadratic approximation of the non-dominated set for bi-criteria programs. *Journal of Multi-Criteria Decision Analysis*, 10 (1), 35–47.
Zamuda, A., *et al.*, 2009. Differential evolution with self-adaptation and local search for constrained multiobjective optimization. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 195–202.
Zhang, Q. and Li, H., 2007. MOEA/D: a multi-objective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11 (6), 712–731.
Zhang, Q., Liu, W., and Li, H., 2009. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. *In*: *IEEE congress on evolutionary computation (CEC'09)*, 18–21 May, Trondheim, Norway. Piscataway, NJ: IEEE Press, 203–208.
Zhang, Q., *et al.*, 2008. Multiobjective optimization test instances for the CEC 2009 special session and competition. Technical Report CES-487. School of Computer Science and Electronic Engineering, University of Essex, UK.

Zhang, Q., *et al.*, 2009. Expensive multiobjective optimization by MOEA/D with gaussian process model. Technical Report CES-489. School of Computer Science and Electronic Engineering, University of Essex, UK.

Zitzler, E., Laumanns, M., and Thiele, L., 2001. SPEA2: improving the strength pareto evolutionary algorithm for multiobjective optimization. *In*: K.C. Giannakoglou, *et al.*, eds. *Evolutionary methods for design, optimisation and control with applications to industrial problems (EUROGEN 2001)*, 19–21 September, Athens, Greece. Berlin: Springer-Verlag, 95–100.

Zitzler, E. and Thiele, L., 1999. Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3 (4), 257–271.

Zitzler, E., *et al.*, 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7 (2), 117–132.