An Efficient Solution Strategy for Bilevel Multiobjective Optimization Problems Using Multiobjective Evolutionary Algorithms

Hong Li[a,*], Li Zhang[a], Qingfu Zhang[b], Qin Chen[c]

[a]*School of Mathematics and Statistics, Xidian University, Xi'an 710071, Shaanxi, China*
[b]*School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK*
[c]*Facility Design and Instrument Institute, China Aerodynamic Research & Development Center, Mianyang 621000, Sichuan, China*

Abstract

An efficient solution strategy is proposed for bilevel multiobjective optimization problem (BLMOP) with multiple objectives at both levels when multiobjective optimization problem (MOP) at the lower level satisfies the convexity and differentiability for the lower level variables. In the proposed strategy, the MOP at the lower level is first transformed into a single objective optimization problem by adopting weighted sum scalarization, in which the lower level weight vector is adjusted adaptively with iteration. Using the Karush-Kuhn-Tucker (KKT) optimality conditions to the lower level single objective optimization problem, the original bilevel multiobjective formulation can be converted into a single level MOP with the complementarity constraints. Then a smoothing technique is suggested to cope with the complementarity constraints. In such a way, the BLMOP is formalized as a single level nonlinear constrained MOP. Constrained MOEA/D-based approach and NSGA-II are applied to solve this transformed problem respectively and some instances are tested to demonstrate the feasibility and effectiveness of the solution methodology.

Keywords:
Bilevel optimization problem, multiobjective optimization, Pareto optimality, NSGA-II, MOEA/D

1. Introduction

A bilevel optimization problem models a hierarchical decision process with two levels in a system. When multiple objectives arise in a certain level or both levels, a bilevel multiobjective optimization problem (BLMOP) can be formed. Without loss of generality, the BLMOP with multiple objectives at both levels can be defined as

$$
\begin{array}{ll}
\min\limits_{x,y} & F(x,y) = \big(F_1(x,y), F_2(x,y), \cdots, F_k(x,y)\big) \\
\text{s. t.} & G(x,y) \leq 0 \\
 & \text{where } \ y \ \text{ solves} \\
\min\limits_{y} & f(x,y) = \big(f_1(x,y), f_2(x,y), \cdots, f_l(x,y)\big) \\
\text{s. t.} & g(x,y) \leq 0
\end{array}
\tag{1}
$$

where $x \in R^n$ is an upper level decision variable, called a leader, $y \in R^m$ is a lower level decision variable, called a follower. $F : R^{n+m} \longrightarrow R^k$, $f : R^{n+m} \longrightarrow R^l$, $G : R^{n+m} \longrightarrow R^p$, $g : R^{n+m} \longrightarrow R^q$. F and G are upper level objective and constraint vector-valued functions, which constitute an upper level multiobjective optimization problem (MOP), while f and g are lower level objective and constraint vector-valued functions, which form a lower level MOP. Therefore, problem (1) possesses a two-level structure with each MOP at each level, in which the decision space of the upper level MOP is implicitly determined by the lower level MOP. This leads to the difficulty of computation.

*Corresponding author
Email address: lihong@mail.xidian.edu.cn (Hong Li)

The BLMOPs can be classified into three categories: 1) Multiple objectives at the upper level and a single objective at the lower level [1][2][8][32][48], 2) A single objective at the upper level and multiple objectives at the lower level [4][6][9][20], and 3) Multiple objectives at both levels [15][16][17][21][23][25][35][47][49][50][53]. The third category is the most complicated and difficult to solve due to their nested structure and multiple objectives at both levels. The main challenge is that every upper level decision variable leads to a set of Pareto optimal solutions of the lower level MOP. The solution strategies for the third category can be modified simply to solve the first two categories, so it is extremely important to study how to solve the BLMOPs in the third category.

Many efforts have been made for dealing with single objective bilevel optimization problems [3][24][27][29][30][33][36][45][46], and some monographs and reviews on the algorithms and applications of such single objective bilevel problems can refer to [5][11][12][18][19][44]. However, in contrast with single objective bilevel optimization problems, BLMOPs have not been well investigated, especially for BLMOP in the third category. The aim of this paper is to develop a solution strategy for the problems in the third category.

With regards to the BLMOPs in the third category, some solution algorithms have been proposed. Yin [49] studied the transportation system management problem using BLMOP models. Multiobjective optimization technique must be utilized accordingly to generate Pareto optimal solutions. Consequently a solution procedure was developed for solving the BLMOP models using genetic algorithms. Wang et al. [47] developed a bilevel multiobjective road network toll model, in which three objectives arise in the upper level, while two objectives is applied in the lower level. The model was solved using a combination of a metaheuristic and a classical optimization algorithm. Zhang et al. [50] proposed a fuzzy multi-objective linear bilevel programming model, and then developed an approximation branch-and-bound algorithm to solve multi-objective bilevel decision problems with fuzzy demands. Jia and Wang [25] proposed a genetic algorithm called Ga- BCPP for a class of multiobjective bilevel convex optimization problem. The original problem was converted into a scalar bilevel programming with one objective at each level, in which the upper level was optimized by the genetic algorithm and the lower level was solved by traditional methods. Deb and Sinha [15] analyzed the difficulties which a bilevel evolutionary multiobjective optimization algorithm may meet in dealing with BLMOP and presented a systematic rule for constructing test instances of BLMOP. Deb and Sinha [16] developed an idea of co-evolutionary multiobjective optimization algorithm for solving BLMOPs with several objectives at both levels. In the proposed procedure, both upper and lower level multiobjective optimization problems can be evolved simultaneously through iterations, in order to keep two interacting populations in the process. Deb and Sinha [17] presented a set of difficult test problems by using an extended version of their early proposed test problem construction procedure, and proposed a hybrid evolutionary multiobjective bilevel programming algorithm with local search, which was an interactive algorithm using NSGA-II at both levels respectively. Nishizaki and Sakawa [35] have studied the linear BLMOP with multiple objectives at both levels. In the solution algorithm, a reference point scalarizing program was solved interactively, and the reference point was updated via asking the leader. After a reference point was provided to the upper level objectives, the optimistic (pessimistic) anticipation approach supposed that the follower returned the Pareto optimal solution of the lower level problem such that the best (worst) fitted the reference point of the leader. Eichfelder [21] addressed the nonlinear BLMOP with upper level constraints separate from the lower level variables. A refinement-based approach was proposed for BLMOPs with double objectives at each level and one upper level variable, in which an exhaustive search was used to the upper level optimization. Zhang et al. [53] proposed a hybrid particle swarm optimization with crossover operator (C-PSO), and then used an interactive co-evolutionary procedure to solve BLMOP, in which multiobjective optimization tasks in the upper level and the lower level were pursued interactively by adopting C-PSO algorithm. A set of Pareto optimal solutions in each iteration was got using the elite strategy in NSGA-II [14]. Gupta and Ong [23] proposed an adaptively scalarized bilevel MOEA (AS-BMOEA) to solve BLMOPs with multiple objectives at both levels. Subsequently, a surrogate-assistance technique was incorporated into the AS-BMOEA to lower the computational cost.

Generally, there are three methodologies for solving the BLMOPs with multiple objectives at both levels. The first route is to keep two-level multiobjective optimization formation and then evolve both upper and lower level MOPs simultaneously through interactive approach. For example, this route was used in [16][17][53]. The second route is to convert the lower level MOP into a single objective optimization problem by adopting scalarization technique, yet keep two-level structure, then use multiobjective evolutionary algorithms (MOEAs) to solve the upper level multiobjective formation, at the same time, use classical optimization methods or evolutionary algorithms to solve interactively the lower level single objective optimization for a given upper level variable. For example, this route was used in [23]. The third route is to transform the two-level MOP into a single-level multiobjective optimization formation by adopting scalarization technique and optimality conditions to lower level MOP under some satisfactory conditions, and then use MOEAs to solve the single-level multiobjective formation. Among three methodologies, the first two routes are computationally expensive, because the original problem keeps two-level structure in considering solution algorithm, but the lower level programming is not required special properties. In the third route the lower level programming needs to satisfy certain optimal conditions such that it can be transformed into single level optimization problem. Thus the computational efficiency can be improved by reducing the complexity of original problem. In this paper, we consider the lower level programming has special features such as convexity and differentiability, so the third route is adopted in our approach.

No matter which methodology is used in solving BLMOP, one has to solve a MOP ultimately. MOEAs are effective heuristic tools to cope with MOPs, which are able to find a set of approximate Pareto optimal solutions in a single run. Many different MOEAs have been proposed and successfully applied to MOPs [7][10][14][26][28][34][38][40][41][42][43][51][52][54][55][56]. Among these MOEAs, NSGA-II [14] is a representative algorithm based on nondomination, and MOEA/D [51] is an algorithm based on decomposition. These two algorithms have been successfully applied to a wide range of MOPs. In this paper, a constrained MOEA/D and NSGA-II are used to solve the single-level transformation model, respectively.

Motivated by these considerations, a BLMOP with multiple objectives at both levels is considered in this paper, which belongs to BLMOPs in the third category. A reformulation model is proposed for simplifying and solving this BLMOP. MOEAs can be used to solve the reformulation model. Our approach also uses adaptive scalarization technique to lower level MOP, which is similar to Gupta and Ong's method [23], but different in the solution model and strategy. Firstly, the weighted sum scalarization is used to transform the lower level MOP into a single-objective optimization problem associated with an adaptive lower level weight vector. Secondly, by using the Karush-Kuhn-Tucker (KKT) optimality conditions to the lower level single-objective optimization problem, the original bilevel multiobjective formulation can be converted into a single level MOP with the complementarity constraints. Thirdly, an existing smoothing technique is utilized to handle the complementarity constraints. Finally, a single level nonlinear constrained MOP with the lower level weight vector is established. To solve the single level constrained MOP, a combined coding strategy is designed, and then a constrained multiobjective differential evolution based on decomposition (CMODE/D) according to the framework of MOEA/D in [51] is developed to find a set of Pareto optimal solutions. In addition, NSGA-II [14] is also utilized to solve this constrained multiobjective optimization model. The main contribution of this paper is that we present a reformulation model by transforming BLMOP into a single-level constrained MOP, then use different MOEAs to solve the reformulation model. CMODE/D can reach a good approximation to the entire Pareto front of all test cases.

The remainder of this paper is organized as follows. Some definitions and transformations of BLMOP with multiple objectives at both levels are presented in Section 2. A constrained multiobjective differential evolution based on decomposition (CMODE/D) and NSGA-II are provided for solving the transformed model in Sections 3 and 4, respectively. Experimental studies are provided in Section 5. Section 6 presents the sensitivity of parameters in CMODE/D. The conclusions are drawn in Section 7.

2. Definitions and reformulations of BLMOP

Problem (1) considered in the paper is a minimization problem with inequality constraints. The other formulations can be easily transformed to the discussed problem. The following sets are associated with BLMOP.

The constraint region of problem (1) is defined as

$$\Omega = \{(x, y) : G(x, y) \leq 0, \;\; g(x, y) \leq 0\}.$$

The projection of Ω onto the leader's decision space is denoted by

$$I = \{x : \exists y, \text{ such that } (x, y) \in \Omega\}.$$

Let $\Psi(x)$ be a set of Pareto optimal solutions of the lower level MOP for every $x \in I$. Also, $\Psi(x)$ is the follower's rational reaction set for $x \in I$. Denote the inducible region of BLMOP by

$$IR = \{(x, y) : (x, y) \in \Omega, \; y \in \Psi(x)\}.$$

The decision-making process of BLMOP can be described as follows. When the leader makes a decision, the follower makes his rational response and reflect his result from a set of lower level Pareto optimal solutions to the upper level. The leader must take account of multiple rational responses of the follower. Thus, the leader adjusts his strategies repeatedly according to the set of rational responses of follower, until the upper level Pareto optimal set is reached.

For reformulating problem (1) as a single level MOP, we assume that $F_i(x,y)$ $(i = 1,2,\cdots,k)$ and $G(x,y)$ may be discontinuous, nonconvex, even nondifferentiable, while $f_j(x,y)(j = 1,2,\cdots,l)$ and $g(x,y)$ are differentiable and convex in y for x fixed.

Problem (1) is transformed into the following formulation by using weighted sum scalarization to the lower level optimization problem:

$$\begin{array}{ll} \min\limits_{x} & F(x,y) = \big(F_1(x,y), F_2(x,y), \cdots, F_k(x,y)\big) \\ \text{s. t.} & G(x,y) \le 0 \\ & 0 \le \omega_j \le 1, \ \ j = 1,2,\cdots,l, \ \ \sum\limits_{j=1}^{l} \omega_j = 1 \\ \min\limits_{y} & \mathcal{S}(x,y,\omega) = \sum\limits_{j=1}^{l} \omega_j f_j(x,y) \\ \text{s. t.} & g(x,y) \le 0 \end{array} \tag{2}$$

where $\omega = (\omega_1, \omega_2, \cdots, \omega_l)$ is a lower level weight vector. The lower level weight vector ω can be regarded as a newly introduced variable, which controls and affects the lower level programming of problem (2). Actually, other scalarization techniques, for example Tchebycheff approach can be also used to the lower level programming problem if the differentiability and convexity of lower level programming problem are not considered. This transformation idea has been used in [23].

Now, the lower level programming of problem (2) is not only affected by the upper level variable x, but also affected by the lower level weight vector $\omega = (\omega_1, \omega_2, \cdots, \omega_l)$. According to assumption, the function $\mathcal{S}(x,y,\omega)$ is also differentiable and convex in y for x and ω fixed. For the sake of simplifying computation, problem (2) is converted into the following single-level MOP with the complementarity constraints by using the KKT conditions to the lower level optimization problem.

$$\begin{array}{ll} \min\limits_{x,y,\omega,\gamma} & \mathcal{F}(x,y,\omega,\gamma) = \big(F_1(x,y), F_2(x,y), \cdots, F_k(x,y)\big) \\ \text{s. t.} & G(x,y) \le 0 \\ & \nabla_y L(x,y,\omega,\gamma) = 0 \\ & \gamma^\top g(x,y) = 0 \\ & g(x,y) \le 0 \\ & \gamma \ge 0 \\ & 0 \le \omega_j \le 1, \ \ j = 1,2,\cdots,l, \ \ \sum\limits_{j=1}^{l} \omega_j = 1 \end{array} \tag{3}$$

where $L(x,y,\omega,\gamma)$ is Lagrange function, $\gamma \in R^q$ are Lagrange multipliers, i.e.

$$L(x,y,\omega,\gamma) = \mathcal{S}(x,y,\omega) + \gamma^\top g(x,y). \tag{4}$$

Therefore,

$$\begin{aligned} \nabla_y L(x,y,\omega,\gamma) &= \nabla_y \mathcal{S}(x,y,\omega) + \gamma^\top \nabla_y g(x,y) \\ &= \sum_{j=1}^{l} \omega_j \nabla_y f_j(x,y) + \gamma^\top \nabla_y g(x,y). \end{aligned} \tag{5}$$

In [22], there exists such smoothing function $\phi_\mu(a,b) = a+b-\sqrt{(a-b)^2 + 4\mu^2}$, which has the following properties:

- $\phi_\mu(a,b) = 0 \Longleftrightarrow a \ge 0, b \ge 0, ab = \mu^2$ for every μ;
- For $\mu = 0$, $\phi_\mu(a,b) = 2\min(a,b)$, while, for every $\mu \ne 0$, $\phi_\mu(a,b)$ is a C^∞ function;
- For every (a,b), $\lim\limits_{\mu \to 0} \phi_\mu(a,b) = 2\min(a,b)$.

In our previous work [31, 32], this smoothing function was used successfully to handle the complementarity constraints. Now this smoothing function is applied to problem (3) for dealing with its complementarity constraints. Consequently, problem (3) is approximated as follows.

$$
\begin{array}{ll}
\min\limits_{x,y,\omega,\gamma} & \mathcal{F}(x,y,\omega,\gamma)=\Big(F_1(x,y),F_2(x,y),\cdots,F_k(x,y)\Big)\\
\text{s. t.} & G(x,y)\leq 0,\\
& \nabla_y L(x,y,\omega,\gamma)=0,\\
& \phi_\mu(\gamma_i,-g_i(x,y))=0,\ \ i=1,2,\cdots,q,\\
& 0\leq\omega_j\leq 1,\ \sum\limits_{j=1}^{l}\omega_j=1,\ \ j=1,2,\cdots,l
\end{array}
\tag{6}
$$

where

$$
\gamma=(\gamma_1,\gamma_2,\cdots,\gamma_q)^\top,
$$

$$
g(x,y)=\Big(g_1(x,y),g_2(x,y),\cdots,g_q(x,y)\Big)^\top,
$$

and

$$
\phi_\mu(\gamma_i,-g_i(x,y))=\gamma_i-g_i(x,y)-\sqrt{(\gamma_i+g_i(x,y))^2+4\mu^2}. \tag{7}
$$

The pre-set parameter $\mu\in R$ is a small positive real number.

Problem (6) is equal to the following problem formation:

$$
\begin{array}{ll}
\min\limits_{x,y,\omega,\gamma} & \mathcal{F}(x,y,\omega,\gamma)=\Big(F_1(x,y),F_2(x,y),\cdots,F_k(x,y)\Big)\\
\text{s. t.} & G(x,y)\leq 0,\\
& \nabla_y L(x,y,\omega,\gamma)=0,\\
& \phi_\mu(\gamma_i,-g_i(x,y))=0,\ \ i=1,2,\cdots,q,\\
& 0\leq\omega_j\leq 1, j=1,2,\cdots,l-1,\\
& 0\leq 1-\sum\limits_{j=1}^{l-1}\omega_j\leq 1
\end{array}
\tag{8}
$$

Let $\xi=(x,\omega_1,\omega_2,\cdots,\omega_{l-1})\in R^{n+l-1}$, $\chi=(y,\gamma)\in R^{m+q}$, and introduce the function $\mathcal{H}:R^{n+m+q+l-1}\to R^{m+q}$, defined as

$$
\mathcal{H}(\xi,\chi):=\begin{bmatrix}\nabla_y L(x,y,\omega,\gamma)\\ \phi_\mu(\gamma_1,-g_1(x,y))\\ \phi_\mu(\gamma_2,-g_2(x,y))\\ \vdots\\ \phi_\mu(\gamma_q,-g_q(x,y))\end{bmatrix}, \tag{9}
$$

and let

$$
\mathcal{G}(\xi,\chi)=\begin{bmatrix}G(x,y)\\ -\omega_1\\ \omega_1-1\\ -\omega_2\\ \omega_2-1\\ \vdots\\ -\omega_{l-1}\\ \omega_{l-1}-1\\ -\sum\limits_{j=1}^{l-1}\omega_j\\ \sum\limits_{j=1}^{l-1}\omega_j-1\end{bmatrix}\in R^{p+2l}, \tag{10}
$$

$$\begin{aligned}\mathcal{F}_1(\xi,\chi) &= F_1(x,y),\\ \mathcal{F}_2(\xi,\chi) &= F_2(x,y),\\ &\vdots\\ \mathcal{F}_k(\xi,\chi) &= F_k(x,y).\end{aligned} \tag{11}$$

Problem (8) is expressed more compactly as:

$$\begin{aligned}\min_{\xi,\chi}\quad & \mathcal{F}(\xi,\chi) = (\mathcal{F}_1(\xi,\chi),\mathcal{F}_2(\xi,\chi),\cdots,\mathcal{F}_k(\xi,\chi))\\ \text{s. t.}\quad & \mathcal{G}(\xi,\chi) \le 0,\\ & \mathcal{H}(\xi,\chi) = 0\end{aligned} \tag{12}$$

The MOP (12) with equality and inequality constraints is difficult to solve directly, therefore the following interactive approach is implemented to simplify and solve constrained MOP (12).

For a given vector $\xi = (x, \omega_1, \omega_2, \cdots, \omega_{l-1})$, where $x \in I$ and $\omega_j \in [0, 1]$ $(j = 1, 2, \cdots, l-1)$, a solution $\chi = (y, \gamma)$ is first obtained by solving the following system of equations:

$$\mathcal{H}(\xi,\chi) = 0, \tag{13}$$

and then optimize the following MOP only with inequality constraints on the variable ξ by using constrained MOEAs:

$$\begin{aligned}\min_{\xi}\quad & \mathcal{F}(\xi,\chi) = (\mathcal{F}_1(\xi,\chi),\mathcal{F}_2(\xi,\chi),\cdots,\mathcal{F}_k(\xi,\chi))\\ \text{s. t.}\quad & \mathcal{G}(\xi,\chi) \le 0\end{aligned} \tag{14}$$

The advantages of the interactive approach are reflected in the following aspects. 1) The number of optimization variables is reduced, because only variable ξ in problem (14) is needed to optimize, while χ is obtained by solving the system of equations (13); 2) Equality constraints are not considered in problem (14). Thus, the efficiency of MOEAs and quality of Pareto optimal solutions can be improved.

In recent decades, significant progress has been made in the development of MOEAs [7][10][14][26][28][34][38][40][41][42][43][51][52][54][55][56]. MOEAs aim at exploring a set of representative Pareto optimal solutions in a single run. In this paper, two distinct evolutionary multiobjective frameworks are adopted to solve problem (14) respectively, which are MOEA based on decomposition and MOEA based on nondomination. Their representative algorithms are MOEA/D [51] and NSGA-II [14], respectively.

Some definitions relative to MOP are introduced [14, 51]. Let u and v be two objective vectors with dimension k, u is said to dominate v if and only if $u_i \le v_i$ for every $i = 1, 2, \cdots, k$ and $u_j < v_j$ for at least one index $j \in \{1, 2, \cdots, k\}$. A solution ξ is called Pareto optimal if there is no other solution η such that the objective vector of η dominates the objective vector of ξ. The set of all the Pareto optimal solutions is called the Pareto set (PS) and the set of all objective vectors corresponding to the Pareto optimal solutions is called the Pareto front (PF) [51].

3. Constrained multiobjective differential evolution based on decomposition (CMODE/D) for problem (14)

MOEA/D [51] is a popular MOEA using decomposition, which simultaneously optimizes a number of single objective optimization subproblems. Differential evolution (DE) is a relatively effective heuristic algorithm, which has been found to be quite successful in a wide variety of optimization tasks [13][39]. DE operators taken as main reproduction operator are used in MOEA/D to generate new solutions.

For constrained MOP (14), we use Tchebycheff approach for converting approximation of the Pareto front of problem (14) into a number of single objective optimization problems [51].

A single objective optimization subproblem in Tchebycheff approach is provided as follows

$$\begin{aligned}\min_{\xi}\quad & \mathcal{T}\Big((\xi,\chi)|\lambda, z^*\Big) = \max_{1\le i\le k}\Big\{\lambda_i|\mathcal{F}_i(\xi,\chi) - z_i^*|\Big\}\\ \text{s. t.}\quad & \mathcal{G}(\xi,\chi) \le 0\end{aligned} \tag{15}$$

where $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_k)$ is an upper level weight vector, i.e., $0 \le \lambda_i \le 1$ for all $i = 1, 2, \cdots, k$ and $\sum_{i=1}^{k} \lambda_i = 1$. $z^* = (z_1^*, z_2^*, \cdots, z_k^*)$ is the reference point, i.e., $z_i^* = \min\Big\{\mathcal{F}_i(\xi,\chi) : \mathcal{G}(\xi,\chi) \le 0\Big\}$ for each $i = 1, 2, \cdots, k$.

3.1. A combined coding strategy and initial population

For solving the optimization subproblem (15), an individual is composed of an upper level variable $x = (x_1, x_2, \cdots, x_n)$ and the first $l-1$ components $(\omega_1, \omega_2, \cdots, \omega_{l-1})$ of a lower level weight vector $\omega = (\omega_1, \omega_2, \cdots, \omega_{l-1}, 1 - \sum_{j=1}^{l-1} \omega_j)$. That is to say, a vector $\xi = (x_1, x_2, \cdots, x_n, \omega_1, \cdots, \omega_{l-1})$ is regarded as an individual, where $x_i \in [lb_i, ub_i]$, $i = 1, 2, \cdots, n$, and $\omega_j \in [0, 1]$, $j = 1, 2, \cdots, l-1$. For expressing simply, the individual is also denoted by $\xi = (\xi_1, \xi_2, \cdots, \xi_{n+l-1})$, where $\xi_i \in [lb_i, ub_i]$, $i = 1, 2, \cdots, n$, and $\xi_i \in [0, 1]$, $i = n+1, n+2, \cdots, n+l-1$.

If the bounds of each x_i $(i = 1, 2, \cdots, n)$ of upper level variable x are not given, the following method is used to establish the bounds of upper level variable x, i.e., $lb_i \le x_i \le ub_i$, $i = 1, 2, \cdots, n$.

$$lb_i = \min\{x_i : x_i \in I_i\}, \quad ub_i = \max\{x_i : x_i \in I_i\} \tag{16}$$

where I_i is the projection of I onto the i-th coordinate axis, i.e. $I = I_1 \times I_2 \times \cdots \times I_n$.

Generate randomly N individuals to constitute an initial population $\xi^1, \xi^2, \cdots, \xi^N$, where $\xi^t = (\xi^t_1, \xi^t_2, \cdots, \xi^t_{n+l-1})$, $t = 1, 2, \cdots, N$, such that $\xi^t_i \in [lb_i, ub_i]$ $(i = 1, 2, \cdots, n)$ and $\xi^t_i \in [0, 1]$ $(i = n+1, n+2, \cdots, n+l-1)$. Herein, N is the number of optimization subproblems, and also denotes population size.

3.2. Fitness function

For every individual ξ^t $(t = 1, 2, \cdots, N)$ in the population, the solution $\chi^t = (y^t, \gamma^t)$ is obtained by solving the system of equations (13). After the lower level variable y^t is determined, the fitness function $fit(\xi^t, \lambda)$ related to upper level weight vector λ is expressed as:

$$fit(\xi^t, \lambda) = \begin{cases} \mathcal{T}\Big((\xi^t, \chi^t)|\lambda, z^*\Big) + P(\xi^t, \chi^t), & \text{if } \chi^t \text{ exists} \\ +\infty, & \text{otherwise} \end{cases} \tag{17}$$

where $P(\xi^t, \chi^t) = \Theta \cdot \Big(\sum_{i=1}^{p+2l} \max\Big\{0, \mathcal{G}_i(\xi^t, \chi^t)\Big\}\Big)$, $\mathcal{G}_i(\xi^t, \chi^t)$ is i-th component of the vector function $\mathcal{G}(\xi^t, \chi^t)$, and $\Theta > 0$ denotes a penalty parameter.

3.3. Reproduction operation

Calculate the Euclidean distances between any two upper level weight vectors (a total of N upper level weight vectors) and then list the T closest upper level weight vectors to each upper level weight vector. For each $t = 1, 2, \cdots, N$, set $B(t) = \{t_1, \cdots, t_T\}$ where $\lambda^{t_1}, \cdots, \lambda^{t_T}$ are the T closest upper level weight vectors to λ^t. Thus $B(t)$ contains the indexes of the T closest vectors of λ^t. If there exists an index $i \in B(t)$, the i-th subproblem can be regarded as a neighbor of the t-th subproblem.

Let r_1, r_2, r_3 are mutually distinct indexes in $B(t)$ not equal to t. An offspring individual $\eta^t = (\eta^t_1, \eta^t_2, \cdots, \eta^t_{n+l-1})$ is generated by the following DE operator:

$$\eta^t_i = \begin{cases} \xi^{r_1}_i + SF \cdot (\xi^{r_2}_i - \xi^{r_3}_i), & \text{if } R_i \le CR \text{ or } i = rn(t) \\ \xi^t_i, & \text{otherwise} \end{cases} \tag{18}$$

where $t = 1, 2, \cdots, N$, $R_i \in (0, 1)$ is a uniform random number, $rn(t) \in \{1, 2, \cdots, n+l-1\}$ is the randomly selected index chosen once for each t, $SF \in (0, 1)$ denotes scaling factor, and $CR \in (0, 1)$ is a real-valued crossover rate constant.

A new individual $\zeta^t = (\zeta^t_1, \zeta^t_2, \cdots, \zeta^t_{n+l-1})$ is generated by the following Gaussian mutation operator:

$$\zeta^t_i = \begin{cases} \eta^t_i + \sigma_i \cdot N_i(0, 1), & \text{if } R_i \le p_m \\ \eta^t_i, & \text{otherwise} \end{cases} \tag{19}$$

where $t = 1, 2, \cdots, N$, $N_i(0, 1)$ is a random number drawn from the standard normal distribution, p_m is a mutation probability. If $R_i < 0.5$, $\sigma_i = \dfrac{ub_i - lb_i}{\delta}$ for $i = 1, 2, \cdots, n$, and $\sigma_i = \dfrac{1}{\delta}$ for $i = n+1, n+2, \cdots, n+l-1$; otherwise, $\sigma_i = 1$ for $i = 1, 2, \cdots, n+l-1$. Herein, δ=20.

3.4. The CMODE/D algorithm based on the framework of MOEA/D in [51]

At each generation, the CMODE/D algorithm maintains the following:

- A population of N points $\xi^1, \cdots, \xi^N$, where ξ^t is the current solution to the t-th subproblem, $t = 1, 2, \cdots, N$;
- $\chi^1, \cdots, \chi^N$, where χ^t is the optimal solution by solving the system of equations (13) for every individual ξ^t, $t = 1, 2, \cdots, N$.
- $FV^1, \cdots, FV^N$, where FV^t is the objective vector value of (ξ^t, χ^t), i.e., $FV^t = \Big(\mathcal{F}_1(\xi^t, \chi^t) + P(\xi^t, \chi^t), \mathcal{F}_2(\xi^t, \chi^t) + P(\xi^t, \chi^t), \cdots, \mathcal{F}_k(\xi^t, \chi^t) + P(\xi^t, \chi^t)\Big)$ for each $i = 1, \cdots, N$;
- $z = (z_1, z_2, \cdots, z_k)$, where z_j is the best value found so far for function $\mathcal{F}_j(\xi^t, \chi^t) + P(\xi^t, \chi^t)$, $j = 1, \cdots, k$.

Now we provide the constrained multiobjective differential evolution based on decomposition (CMODE/D) as follows.

Input:

- Problem(14);
- N: the number of the subproblems;
- $\lambda^1, \cdots, \lambda^N$: a set of N upper level weight vectors;
- T: the number of the upper level weight vectors in the neighborhood of each upper level weight vector;
- μ: a small positive parameter in the smoothing function (7);
- SF: a scaling factor in reproduction operator;
- CR: a real-valued crossover rate constant in reproduction operator;
- p_m: a Gaussian mutation probability in reproduction operator;
- Θ: a penalty parameter in the fitness function;
- $MaxG$: maximum number of generations.

Output:

- Approximation to the PS: $\Big\{(x^1, y^1), \cdots, (x^N, y^N)\Big\}$;
- Approximation to the PF: $\Big\{F(x^1, y^1), \cdots, F(x^N, y^N)\Big\}$.

Step 1 Initialization

Step 1.1 Calculate the Euclidean distances between any two upper level weight vectors and then list the T closest upper level weight vectors to each upper level weight vector. For each $t = 1, \cdots, N$, set $B(t) = \{t_1, \cdots, t_T\}$, where $\lambda^{t_1}, \cdots, \lambda^{t_T}$ are the T closest upper level weight vectors to λ^t.

Step 1.2 Randomly generate an initial population $\xi^1, \cdots, \xi^N$. For every individual ξ^t, solve the system of equations (13), and obtain the optimal solution χ^t. Set $FV^t = \Big(\mathcal{F}_1(\xi^t, \chi^t) + P(\xi^t, \chi^t), \mathcal{F}_2(\xi^t, \chi^t) + P(\xi^t, \chi^t), \cdots, \mathcal{F}_k(\xi^t, \chi^t) + P(\xi^t, \chi^t)\Big)$.

Step 1.3 Initialize $z = (z_1, z_2, \cdots, z_k)$ by setting $z_j = \min\limits_{1 \le t \le N} \Big\{\mathcal{F}_j(\xi^t, \chi^t) + P(\xi^t, \chi^t)\Big\}$.

Step 2 Weight vector adjustment

If the number of iterations is less than $0.9 \times MaxG$, then the set of initial weight vectors keeps unchanged. Otherwise, adjust the set of weight vectors based on the obtained approximation to the PF.

Step 3 Update

For $t = 1, \cdots, N$, do

Step 3.1 Reproduction: Randomly select three distinct indexes r_1, r_2, r_3 from $B(t)$ not equal to t, then generate a solution η^t from $\xi^{r_1}, \xi^{r_2}, \xi^{r_3}$ by a DE operator, and then perform the Gaussian mutation operator on η^t with probability p_m to produce a new solution ζ^t.

Step 3.2 Repair: If an element of ζ^t is out of the box boundary, its value is reset to be a randomly selected value inside the boundary. For every repaired individual ζ^t, solve the system of equations (13), and obtain the optimal solution $\overline{\chi}^t$.

Step 3.3 Update of z: For each $j = 1, 2, \cdots, k$, if $z_j > \mathcal{F}_j(\zeta^t, \overline{\chi}^t) + P(\zeta^t, \overline{\chi}^t)$, then set $z_j = \mathcal{F}_j(\zeta^t, \overline{\chi}^t) + P(\zeta^t, \overline{\chi}^t)$.

Step 3.4 Update of solutions: For each index $j \in B(t)$, if $fit(\zeta^t, \lambda^j) \leq fit(\xi^j, \lambda^j)$, then set $\xi^j = \zeta^t$, $FV^j = \Big(\mathcal{F}_1(\zeta^t, \overline{\chi}^t) + P(\zeta^t, \overline{\chi}^t), \mathcal{F}_2(\zeta^t, \overline{\chi}^t) + P(\zeta^t, \overline{\chi}^t), \cdots, \mathcal{F}_k(\zeta^t, \overline{\chi}^t) + P(\zeta^t, \overline{\chi}^t)\Big)$.

Step 4 Stopping criterion: If the stopping criterion is satisfied, then stop and output approximation to the PS and PF. Otherwise, go to **Step 2**.

4. NSGA-II for problem (14)

NSGA-II developed by Deb et al. [14] is an excellent MOEA based on nondomination. Since it emerged, NSGA-II has had very significant effect on applications of MOPs. In order to use NSGA-II to solve the MOP (14), same penalty function method as CMODE/D is adopted to handle the constraints, so problem (14) is converted into the following unconstrained MOP:

$$\min_{\xi} \mathcal{F}(\xi, \chi) = \Big(\mathcal{F}_1(\xi, \chi) + P(\xi, \chi), \mathcal{F}_2(\xi, \chi) + P(\xi, \chi), \cdots, \mathcal{F}_k(\xi, \chi) + P(\xi, \chi)\Big) \tag{20}$$

where $P(\xi, \chi) = \Theta \cdot \big(\sum_{i=1}^{p+2l} \max\{0, \mathcal{G}_i(\xi, \chi)\}\big)$, and $\Theta > 0$ denotes a penalty parameter.

Same combined coding strategy in CMODE/D is used in NSGA-II, i.e. $\xi = (x_1, x_2, \cdots, x_n, \omega_1, \cdots, \omega_{l-1})$ is regarded as an individual (a member of population), where $x_i \in [lb_i, ub_i]$, $i = 1, 2, \cdots, n$, and $\omega_j \in [0, 1]$, $j = 1, 2, \cdots, l-1$. For each ξ in the population, a solution χ is obtained by solving system of equation (13). Since the χ is determined by ξ, NSGA-II only evolves the variable ξ.

Now we adopt NSGA-II to solve problem (14) as follows.

Input:

- Problem(20);
- N: population size;
- η_c: a distribution index for the simulated binary crossover operator;
- η_m: a distribution index for polynomial mutation operator;
- μ: a small positive parameter in the smoothing function (7);
- p_c: a crossover probability;
- $p_m = 1/(n + l - 1)$: a mutation probability (where $n + l - 1$ is the dimension of variable ξ);
- Θ: a penalty parameter in the fitness function;
- $MaxG$: maximum number of generations.

Output:

- Approximation to the PS: $\{(x^1, y^1), \cdots, (x^N, y^N)\}$;
- Approximation to the PF: $\{F(x^1, y^1), \cdots, F(x^N, y^N)\}$.

Step 1 Initialization

Set $K = 0$. Randomly generate an initial population $P_K = \{\xi_K^1, \cdots, \xi_K^N\}$. For every population member ξ_K^t, solve the system of equations (13), and obtain the optimal solution χ_K^t $(t = 1, 2, \cdots, N)$.

Step 2 Genetic operation

Generate offspring population denoted by $Q_K = \{\bar{\xi}_K^1, \cdots, \bar{\xi}_K^N\}$ by using genetic operation (binary tournament selection based on the crowded-comparison operator, simulated binary crossover, and polynomial mutation) to parent population P_K. For every population member $\bar{\xi}_K^t$, solve the system of equations (13), and obtain the optimal solution $\bar{\chi}_K^t$ $(t = 1, 2, \cdots, N)$.

Step 3 Population combination

Combine parent and offspring population, and obtain a big population with $2N$ members denoted by R_K, i.e. $R_K = P_K \cup Q_K$.

Step 4 Nondominated sorting and selection

Sort the big population R_K according to nondomination, at the same time, select N best solutions in terms of the rank of nondominated set and crowded distance of each solution to form the next population P_{K+1} of size N.

Step 5 Stopping criterion

If the stopping criterion is satisfied, then stop and output the approximation to PS and PF. Otherwise, let $K = K + 1$, go to **Step 2**.

5. Experimental studies

5.1. Test instances and parameter settings

To evaluate the performances of CMODE/D and NSGA-II, we test eleven BLMOPs, where the first six examples are selected from the literature, and the others are the constructed instances with adjustable dimensions ($K = 10$ and $K = 20$ were used in the paper). The size of each instance is denoted by $n - p - m - q$, where n and m are the number of variables at the upper level and at the lower level respectively, p and q are the number of constraints at the upper level and at the lower level respectively. These instances are shown as follows.

Problem 1 [25]

$$\begin{aligned} \min_{x,y} \quad & F(x, y) = \big(-x - y,\ \ x^2 + (y - 10)^2\big) \\ \text{s. t.} \quad & 0 \le x \le 15 \\ \min_{y} \quad & f(x, y) = \big(y^2,\ \ y(x - 30)\big) \\ \text{s. t.} \quad & y - x \le 0,\ 0 \le y \le 15 \end{aligned}$$

Problem 2 [16]

$$\begin{aligned} \min_{x,y} \quad & F(x, y) = \big(y_1 - x,\ \ y_2\big) \\ \text{s. t.} \quad & 1 + y_1 + y_2 \ge 0,\ 0 \le x \le 1 \\ \min_{y} \quad & f(x, y) = \big(y_1,\ \ y_2\big) \\ \text{s. t.} \quad & y_1^2 + y_2^2 \le x^2,\ -1 \le y_1,\ y_2 \le 1 \end{aligned}$$

Problem 3 [16]

$$\begin{aligned} \min_{x,y} \quad & F(x, y) = \big((y_1 - 1)^2 + y_2^2 + x^2,\ \ (y_1 - 1)^2 + y_2^2 + (x - 1)^2\big) \\ \min_{y} \quad & f(x, y) = \big(y_1^2 + y_2^2,\ \ (y_1 - x)^2 + y_2^2\big) \\ \text{s. t.} \quad & -1 \le x, y_1, y_2 \le 2 \end{aligned}$$

Problem 4 [21]

$$\begin{aligned}
\min_{x,y} \quad & F(x,y) = \big(F_1(x,y),\ \ F_2(x,y)\big) \\
\text{s. t.} \quad & 0 \le x \le 10 \\
\min_{y} \quad & f(x,y) = \big(f_1(x,y),\ \ f_2(x,y)\big) \\
\text{s. t.} \quad & y_2 - y_1^2 \ge 0,\ 10 - 5y_1^2 - y_2 \ge 0, \\
& 5 - \frac{x}{6} - y_2 \ge 0,\ 0 \le y_1, y_2 \le 10
\end{aligned}$$

where

$$F_1(x,y) = y_1 + y_2^2 + x + \sin^2(y_1 + x),$$

$$F_2(x,y) = \cos(y_2)(0.1 + x)(\exp(-\frac{y_1}{0.1 + y_2})),$$

$$f_1(x,y) = \frac{(y_1 - 2)^2 + (y_2 - 1)^2}{4} + \frac{y_2 x + (5 - x)^2}{16} + \sin\frac{y_2}{10},$$

$$f_2(x,y) = \frac{y_1^2 + (y_2 - 6)^4 - 2y_1 x - (5 - x)^2}{80}$$

Problem 5 [16]

$$\begin{aligned}
\min_{x,y} \quad & F(x,y) = \big(F_1(x,y),\ \ F_2(x,y)\big) \\
\text{s. t.} \quad & -1 \le x \le 2 \\
\min_{y} \quad & f(x,y) = \left(y_1^2 + \sum_{i=1}^{13} y_{i+1}^2,\ \ (y_1 - x)^2 + \sum_{i=1}^{13} y_{i+1}^2\right) \\
\text{s. t.} \quad & -1 \le y_i \le 2,\ i = 1, 2, \cdots, 14
\end{aligned}$$

where

$$F_1(x,y) = (y_1 - 1)^2 + \sum_{i=1}^{13} y_{i+1}^2 + x^2,$$

$$F_2(x,y) = (y_1 - 1)^2 + \sum_{i=1}^{13} y_{i+1}^2 + (x - 1)^2$$

Problem 6 [17]

$$\begin{aligned}
\min_{x,y} \quad & F(x,y) = \big(F_1(x,y),\ \ F_2(x,y)\big) \\
\text{s. t.} \quad & 0.001 \le x_1 \le K, -K \le x_i \le K,\ i = 2, \cdots, K \\
\min_{y} \quad & f(x,y) = \left(y_1^2 + \sum_{i=2}^{K} (x_i - y_i)^2,\ \ \sum_{i=1}^{K} i(x_i - y_i)^2\right) \\
\text{s. t.} \quad & -K \le y_i \le K,\ i = 1, 2, \cdots, K
\end{aligned}$$

where

$$F_1(x,y) = v_1 + \sum_{i=1}^{K} [x_i^2 + 10 * (1 - \cos(\tfrac{\pi}{K})x_i)] + \sum_{i=2}^{K} (x_i - y_i)^2 - r\cos(\gamma\tfrac{\pi}{2}\tfrac{y_1}{x_1}),$$

$$F_2(x,y) = \quad v_2 + \sum_{i=1}^{K} [x_i^2 + 10 * (1 - \cos(\tfrac{\pi}{K})x_i)] + \sum_{i=2}^{K} (x_i - y_i)^2 - r\sin(\gamma\tfrac{\pi}{2}\tfrac{y_1}{x_1})$$

If $0 \le x_1 \le 1$,

$$v_1 = \cos(0.2\pi)x_1 + \sin(0.2\pi)\sqrt{|0.02\sin(5\pi x_1)|},$$

$$v_2 = -\sin(0.2\pi)x_1 + \cos(0.2\pi)\sqrt{|0.02\sin(5\pi x_1)|},$$

If $x_1 > 1$

$$v_1 = x_1 - (1 - \cos(0.2\pi)),$$

$$v_2 = 0.1(x_1 - 1) - \sin(0.2\pi)$$

Here, $\gamma = 4$, $r = 0.25$. Case (a): $K = 10$, and case (b): $K = 20$.

Problem 7

$$\begin{aligned}
\min_{\mathrm{x,y}} \quad & F(x,y) = \left(\sum_{i=1}^{K} (y_i - 2)^2 + \sum_{i=1}^{K} x_i^2,\ \ \sum_{i=1}^{K} y_i^2 + \sum_{i=1}^{K} (x_i - 2)^2 \right) \\
\text{s. t.} \quad & -2 \le x_i \le 2, i = 1, 2, \cdots, K \\
\min_{\mathrm{y}} \quad & f(x,y) = \left(-\sum_{i=1}^{K} y_i^2,\ \ \sum_{i=1}^{K} (y_i - x_i)^2 \right) \\
\text{s. t.} \quad & -2 \le y_i \le 2,\ i = 1, 2, \cdots, K
\end{aligned}$$

Here, case (a): $K = 10$, and case (b): $K = 20$.

Problem 8

$$\begin{aligned}
\min_{\mathrm{x,y}} \quad & F(x,y) = \left(\sum_{i=1}^{K} \cos(|y_i - 5|) + \sum_{i=1}^{K} \sin(|x_i|),\ \ \sum_{i=1}^{K} \sin(|y_i|) + \sum_{i=1}^{K} \cos(|x_i - 5|) \right) \\
\text{s. t.} \quad & -5 \le x_i \le 5, i = 1, 2, \cdots, K \\
\min_{\mathrm{y}} \quad & f(x,y) = \left(\sum_{i=1}^{K} (x_i + y_i),\ \ \sum_{i=1}^{K} (x_i - y_i)^2 \right) \\
\text{s. t.} \quad & -5 \le y_i \le 5,\ i = 1, 2, \cdots, K
\end{aligned}$$

Here, case (a): $K = 10$, and case (b): $K = 20$.

Problem 9

$$\begin{aligned}
\min_{\mathrm{x,y}} \quad & F(x,y) = \left(\sum_{i=1}^{K} (x_i + y_i),\ \ \sum_{i=1}^{K} (-x_i + y_i) \right) \\
\text{s. t.} \quad & -1 \le x_i \le 1, i = 1, 2, \cdots, K \\
\min_{\mathrm{y}} \quad & f(x,y) = \left(\sum_{i=1}^{K} (-1)^i y_i,\ \ \sum_{i=1}^{K} (-1)^{i+1} y_i \right) \\
\text{s. t.} \quad & -x_i + y_i \le 1, i = 1, 2, \cdots, K \\
& -1 \le y_i \le 1,\ i = 1, 2, \cdots, K
\end{aligned}$$

Here, case (a): $K = 10$, and case (b): $K = 20$.

Problem 10

$$\begin{aligned}
\min_{\mathrm{x,y}} \quad & F(x,y) = \left(\sum_{i=1}^{K} \exp(-\tfrac{x_i}{1+|y_i|}) + \sum_{i=1}^{K} \sin(\tfrac{x_i}{1+|y_i|}),\ \ \sum_{i=1}^{K} \exp(-\tfrac{y_i}{1+|x_i|}) + \sum_{i=1}^{K} \sin(\tfrac{y_i}{1+|x_i|}) \right) \\
\text{s. t.} \quad & -1 \le x_i \le 1, i = 1, 2, \cdots, K \\
\min_{\mathrm{y}} \quad & f(x,y) = \left(\sum_{i=1}^{K} \cos(|x_i| y_i) + \sum_{i=1}^{K} \sin(x_i - y_i),\ \ \sum_{i=1}^{K} \cos(\tfrac{y_i}{1+|x_i|}) + \sum_{i=1}^{K} \sin(x_i + y_i) \right) \\
\text{s. t.} \quad & x_i + y_i \le 1, i = 1, 2, \cdots, K \\
& -1 \le y_i \le 1,\ i = 1, 2, \cdots, K
\end{aligned}$$

Here, case (a): $K = 10$, and case (b): $K = 20$.

Problem 11

$$\begin{aligned}
\min_{\mathrm{x,y}} \quad & F(x,y) = \left(\frac{\sum_{i=1}^{K} \cos(x_i)}{\max(1, \sum_{i=1}^{K} \cos(y_i))},\ \ \frac{\sum_{i=1}^{K} \cos(y_i)}{\max(1, \sum_{i=1}^{K} \cos(x_i))} \right) \\
\text{s. t.} \quad & x_i^2 + y_i^2 \le 1, i = 1, 2, \cdots, K \\
& -1 \le x_i \le 1, i = 1, 2, \cdots, K \\
\min_{\mathrm{y}} \quad & f(x,y) = \left(\sum_{i=1}^{K} (-1)^i \sin(x_i) y_i,\ \ \sum_{i=1}^{K} (-1)^{i+1} \frac{y_i}{\cos(x_i)} \right) \\
\text{s. t.} \quad & x_i + y_i \le 1, i = 1, 2, \cdots, K \\
& -1 \le y_i \le 1,\ i = 1, 2, \cdots, K
\end{aligned}$$

Here, case (a): $K = 10$, and case (b): $K = 20$.

All experiments were performed on a Lenovo-PC with Intel Core i5-4200U CPU 1.60GHz processor and 8.00GB of RAM in MATLAB software. For solving the system of nonlinear equations (13), we directly adopted the function ‘fsolve’ in MATLAB optimization toolbox. In our experiments, each algorithm was run ten times independently for each test instance. The parameter settings in CMODE/D were listed as follows:

- Population size, or the number of the subproblems: $N = 150$;
- The number of the upper level weight vectors in the neighborhood of each upper level weight vector: $T = 20$;
- A small positive parameter in the smoothing function (7): $\mu = 0.0001$;
- Scaling factor: $SF = 0.5$;
- Crossover rate: $CR = 0.6$;
- Gaussian mutation probability: $p_m = 0.6$;
- Penalty factor: $\Theta = 10000$;
- Maximum number of generations: $MaxG = 300$.

The upper level weight vectors $\lambda^1, \lambda^2, \cdots, \lambda^N$ are controlled by a parameter H, where $H = N - 1$, and each individual weight takes a value from $\left\{\frac{0}{H}, \frac{1}{H}, \cdots, \frac{H}{H}\right\}$. The distribution of solutions in objective space cannot be determined only by the Euclidean distance of weights in the weight vector space. To get diversity and uniform distribution of solutions in objective space, it is necessary to adjust the weight vector in the last stage of evolution. Based on the Euclidean distance of the solutions in objective space, we can calculate a set of the amended weight vectors. In the first stage, a set of common weight vectors is used to make the solutions approach to approximate PF. In the last stage, a set of modified weight vectors is adopted to improve the uniform distribution of the solution in objective space. The procedure of weight vectors adjustment can be found in [32].

Some parameters used in NSGA-II were same as those in CMODE/D, and the other control parameters suggested in [14] were adopted. All parameters in NSGA-II were listed as follows:

- Population size: $N = 150$;
- A distribution index for the simulated binary crossover operator: $\eta_c = 20$;
- A distribution index for polynomial mutation operator: $\eta_m = 20$;
- A small positive parameter in the smoothing function (7): $\mu = 0.0001$;
- Crossover probability: $p_c = 0.9$;
- Mutation probability: $p_m = 0.1$;
- Penalty parameter in the fitness function: $\Theta = 10000$;
- Maximum number of generations: $MaxG = 300$.

5.2. Performance metrics

The following two performance metrics are used to assess the advantages of NSGA-II and CMODE/D.

- C-metric (Coverage measure)[55]: Let A and B be two approximations to the Pareto front (PF) of a MOP, $C(A, B)$ is defined as the percentage of the solutions in B that are dominated by at least one solution in A, i.e.,

$$C(A, B) = \frac{|u \in B| \exists v \in A : v \text{ dominates } u|}{|B|}$$

where $|B|$ denotes the number of the elements in the set B. $C(A, B) = 1$ means that all solutions in B are dominated by some solutions in A, while $C(A, B) = 0$ implies that no solution in B is dominated by a solution in A. Generally, $C(A, B) \neq 1 - C(B, A)$. If $C(A, B) > C(B, A)$, the PF of A is better than that of B.

Table 1: The average CPU time (in seconds) used by NSGA-II and CMODE/D.

Problem	Size	NSGA-II	CMODE/D
1	1-2-1-3	319.8721	300.2156
2	1-3-2-5	463.2557	506.9602
3	1-2-2-4	250.2315	206.2603
4	1-2-2-7	615.2630	594.6173
5	1-2-14-28	470.9989	554.5790
6(a)	10-20-10-20	580.1767	453.1113
6(b)	20-40-20-40	2719.0337	1191.8846
7(a)	10-20-10-20	603.0251	582.2160
7(b)	20-40-20-40	3431.7187	2127.6222
8(a)	10-20-10-20	1673.2973	1153.4263
8(b)	20-40-20-40	10131.0917	3467.5431
9(a)	10-20-10-30	2129.9674	2093.1028
9(b)	20-40-20-60	16724.0692	14302.3433
10(a)	10-20-10-30	3720.2693	1369.4874
10(b)	20-40-20-60	33378.1960	15093.2247
11(a)	10-30-10-30	2115.0768	1971.7143
11(b)	20-60-20-60	26453.3754	14182.6722

Table 2: The average C-metric values of the Pareto solutions between NSGA-II (A) and CMODE/D (B). The numbers in parentheses represent the standard deviation.

Problem	Size	$C(A, B)$	$C(B, A)$
1	1-2-1-3	6.60e-4 (2.10e-3)	5.30e-3 (6.90e-3)
2	1-3-2-5	1.26e-2 (6.60e-3)	3.15e-1 (4.38e-2)
3	1-2-2-4	1.85e-2 (6.91e-3)	2.73e-2 (1.24e-2)
4	1-2-2-7	2.25e-2 (1.09e-2)	1.29e-1 (3.19e-2)
5	1-2-14-28	1.12e-2 (7.02e-3)	3.40e-2 (7.31e-3)
6(a)	10-20-10-20	0 (0)	1 (0)
6(b)	20-40-20-40	0 (0)	1 (0)
7(a)	10-20-10-20	8.14e-2 (4.43e-2)	5.03e-1 (9.50e-2)
7(b)	20-40-20-40	2.35e-2 (2.38e-2)	7.84e-1 (7.47e-2)
8(a)	10-20-10-20	2.21e-1 (3.84e-1)	6.95e-1 (4.31e-1)
8(b)	20-40-20-40	6.62e-2 (2.09e-1)	8.63e-1 (3.24e-1)
9(a)	10-20-10-30	3.58e-2 (3.90e-2)	2.51e-1 (1.65e-1)
9(b)	20-40-20-60	1.32e-3 (2.79e-3)	9.47e-1 (4.39e-2)
10(a)	10-20-10-30	4.30e-2 (2.05e-2)	6.43e-1 (4.45e-2)
10(b)	20-40-20-60	3.97e-3 (8.38e-3)	8.82e-1 (1.81e-2)
11(a)	10-30-10-30	8.61e-3 (1.43e-2)	7.00e-1 (4.83e-1)
11(b)	20-60-20-60	0 (0)	1 (0)

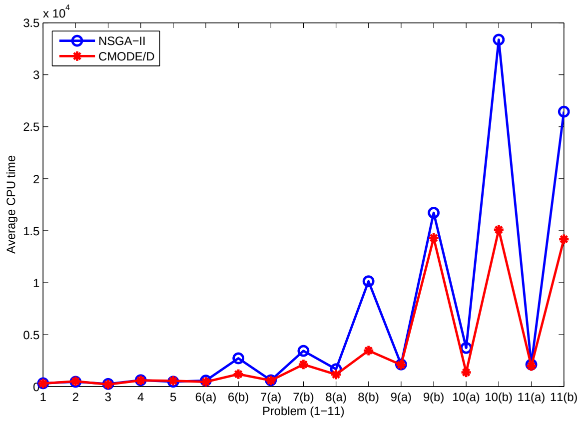


Figure 1: Graphical representation of the average CPU time used by NSGA-II and CMODE/D for problems 1-11.

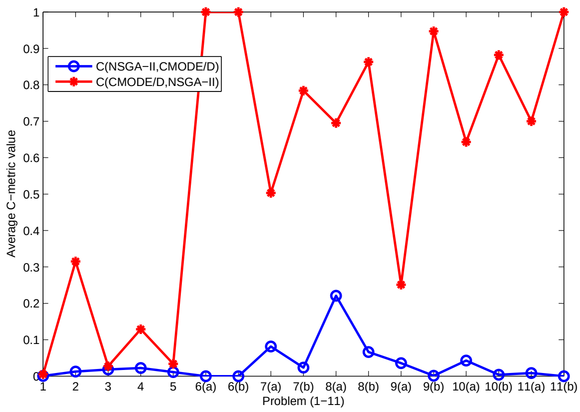


Figure 2: Graphical representation of the average C-metric values between NSGA-II and CMODE/D for problems 1-11.

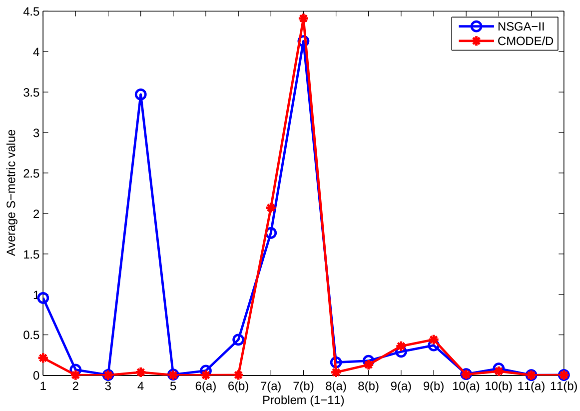


Figure 3: Graphical representation of the average S-metric values obtained by NSGA-II and CMODE/D for problems 1-11.

Table 3: The average S-metric values of the Pareto solutions obtained by NSGA-II and CMODE/D for each instance. The numbers in parentheses represent the standard deviation.

Problem	Size	NSGA-II	CMODE/D
1	1-2-1-3	9.56e-1 (6.75e-2)	2.15e-1 (1.60e-2)
2	1-3-2-5	6.96e-2 (1.06e-4)	3.40e-3 (1.73e-4)
3	1-2-2-4	4.20e-3 (3.41e-4)	1.32e-3 (9.94e-5)
4	1-2-2-7	3.47e+0 (7.17e-1)	3.89e-2 (3.52e-3)
5	1-2-14-28	8.90e-3 (2.27e-4)	1.31e-3 (8.76e-5)
6(a)	10-20-10-20	5.67e-2 (2.99e-2)	3.66e-3 (1.52e-3)
6(b)	20-40-20-40	4.40e-1 (9.48e-2)	3.88e-3 (8.86e-3)
7(a)	10-20-10-20	1.76e+0 (5.26e-1)	2.07e+0 (7.95e-2)
7(b)	20-40-20-40	4.13e+0 (2.60e-1)	4.41e+0 (4.12e-1)
8(a)	10-20-10-20	1.60e-1 (1.21e-1)	3.68e-2 (1.09e-2)
8(b)	20-40-20-40	1.80e-1 (1.16e-1)	1.31e-1 (6.79e-1)
9(a)	10-20-10-30	2.92e-1 (5.82e-2)	3.61e-1 (1.91e-2)
9(b)	20-40-20-60	3.71e-1 (1.09e-1)	4.41e-1 (6.49e-2)
10(a)	10-20-10-30	1.51e-2 (1.81e-3)	1.04e-2 (3.72e-3)
10(b)	20-40-20-60	8.33e-2 (5.10e-3)	5.08e-2 (2.29e-2)
11(a)	10-30-10-30	3.45e-3 (5.22e-4)	2.20e-3 (1.51e-3)
11(b)	20-60-20-60	4.10e-3 (1.86e-4)	2.51e-3 (9.97e-4)

- S-metric [37]: This metric is used to calculate the uniformity in the distribution. For the most uniformly spreadout set of Pareto optimal solutions, the value of S would be zero. Thus a smaller S indicator value is preferable.

$$S(A) = \sqrt{\frac{1}{|A|-1}\sum_{i=1}^{|A|-1}(d_i - \bar{d})^2}$$

where $d_i = \min_{j \neq i}\left\{\sum_{l=1}^{k}|F_l(x^i, y^i) - F_l(x^j, y^j)|\right\}$, $i, j = 1, 2, \cdots, |A| - 1$. $\bar{d}$ is the average of all distances d_i, $i = 1, 2, \cdots, |A| - 1$, assuming that there are $|A|$ solutions on the best Pareto front.

5.3. Experimental results of CMODE/D and NSGA-II

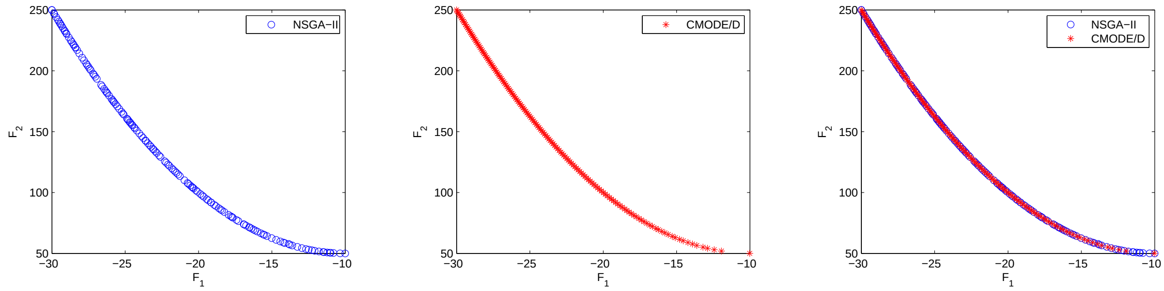
In order to roughly evaluate the efficiency of NSGA-II and CMODE/D, Table 1 presents the average CPU time (in seconds) used by two algorithms for each instance. Moreover, Figure 1 shows the graphical representation of the average CPU time used by NSGA-II and CMODE/D for instances 1-11.

For measuring the convergence of the final solutions obtained by NSGA-II and CMODE/D, Table 2 shows the mean and standard deviation of the C-metric values between two algorithms for each instance. At the same time, Figure 2 displays the graphical representation of the average C-metric values between NSGA-II and CMODE/D for instances 1-11.

For assessing the distribution uniformity of the final Pareto front obtained by two algorithms, Table 3 provides the mean and standard deviation of the S-metric values obtained by NSGA-II and CMODE/D for each instance. Meanwhile, Figure 3 depicts the graphical representation of the average S-metric values obtained by NSGA-II and CMODE/D for instances 1-11.

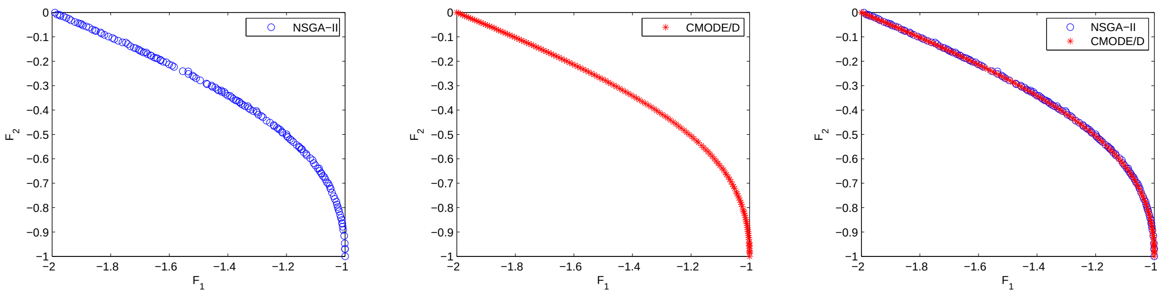
Figures 4-20 display the Pareto front achieved by CMODE/D and NSGA-II in the last generation of a typical run for each instance, respectively.

With same population size and same number of evolutionary generations, it can be observed from Table 1 and Figure 1 that both CMODE/D and NSGA-II need similar computational time for low dimensional instances 1-5, but CMODE/D requires less runtime than NSGA-II for instances 6-11.



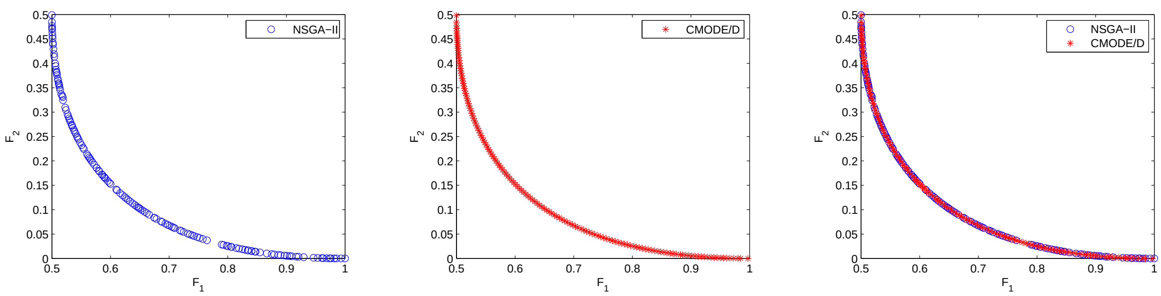
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 4: The PF produced by NSGA-II and CMODE/D for problem 1.



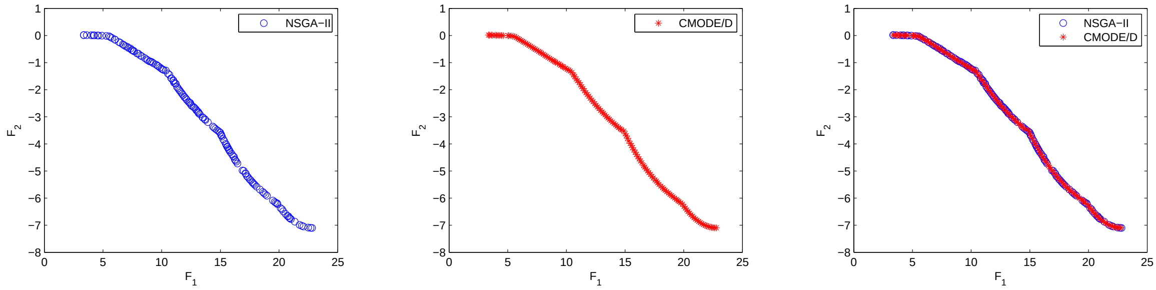
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 5: The PF produced by NSGA-II and CMODE/D for problem 2.



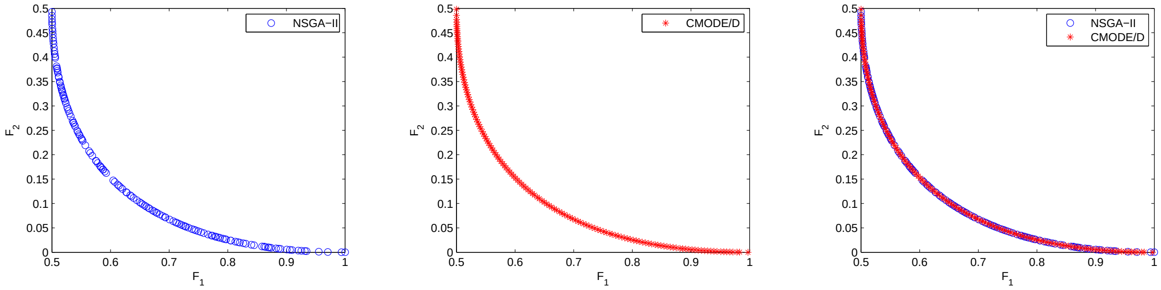
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 6: The PF produced by NSGA-II and CMODE/D for problem 3.



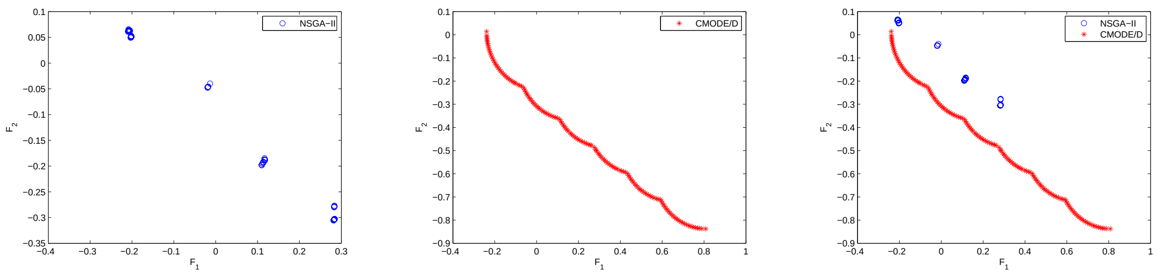
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 7: The PF produced by NSGA-II and CMODE/D for problem 4.



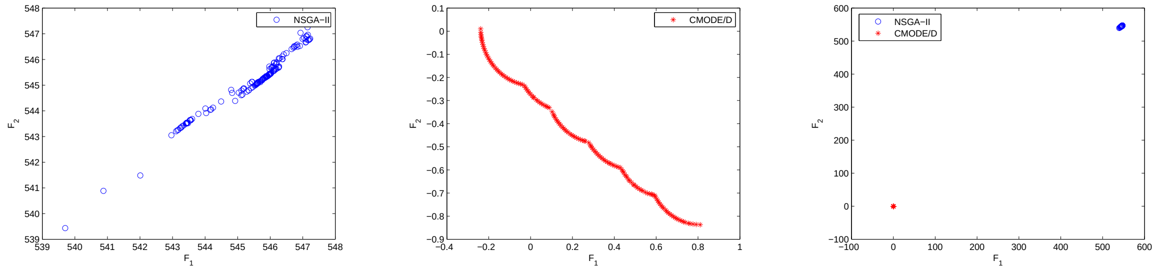
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 8: The PF produced by NSGA-II and CMODE/D for problem 5.



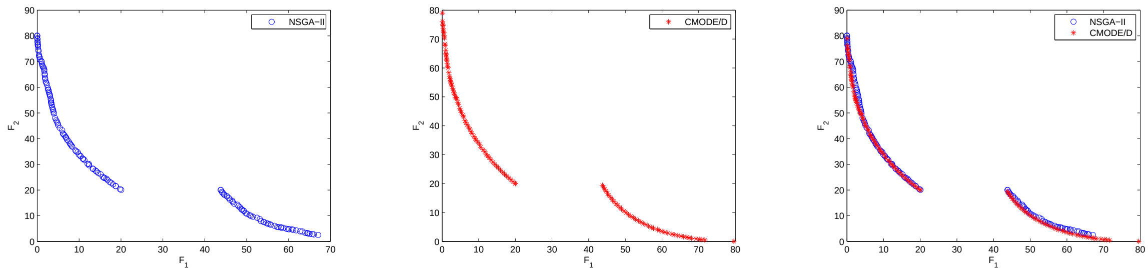
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 9: The PF produced by NSGA-II and CMODE/D for problem 6(a) ($K = 10$).



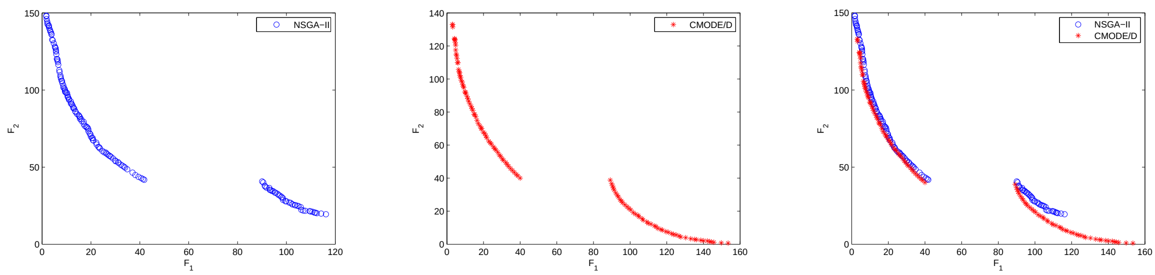
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 10: The PF produced by NSGA-II and CMODE/D for problem 6(b) ($K = 20$).



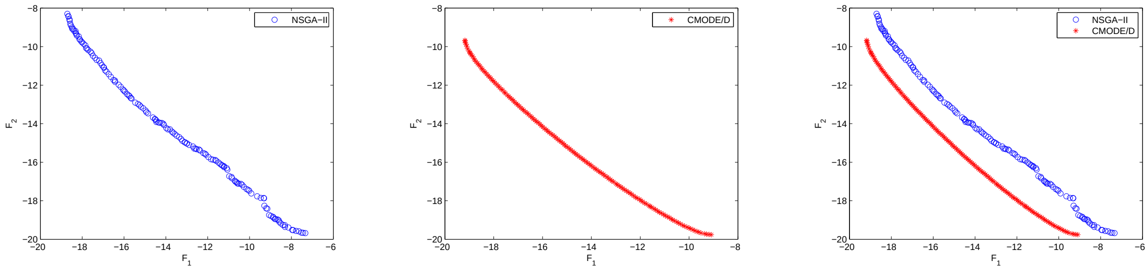
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 11: The PF produced by NSGA-II and CMODE/D for problem 7(a) ($K = 10$).



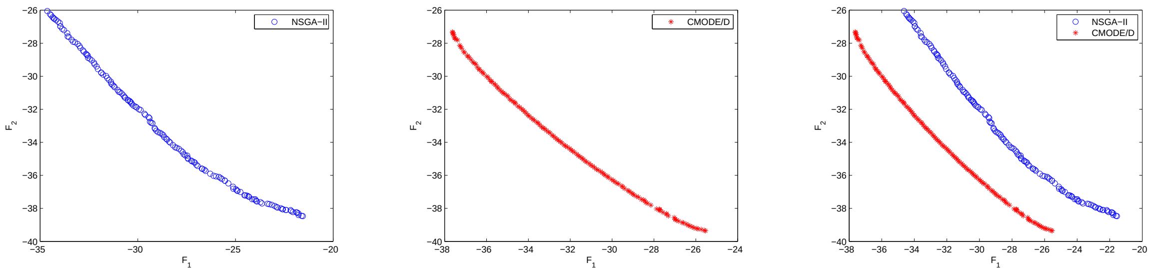
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 12: The PF produced by NSGA-II and CMODE/D for problem 7(b) ($K = 20$).



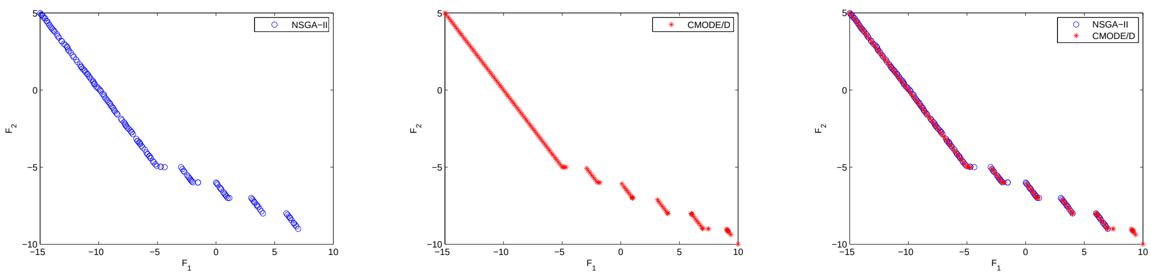
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 13: The PF produced by NSGA-II and CMODE/D for problem 8(a) ($K = 10$).



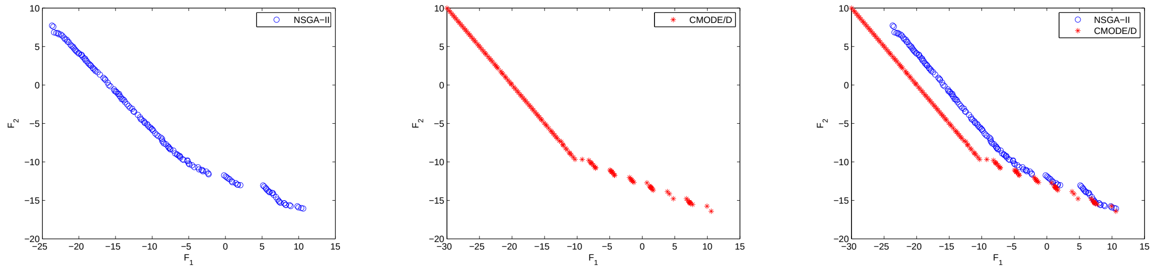
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 14: The PF produced by NSGA-II and CMODE/D for problem 8(b) ($K = 20$).



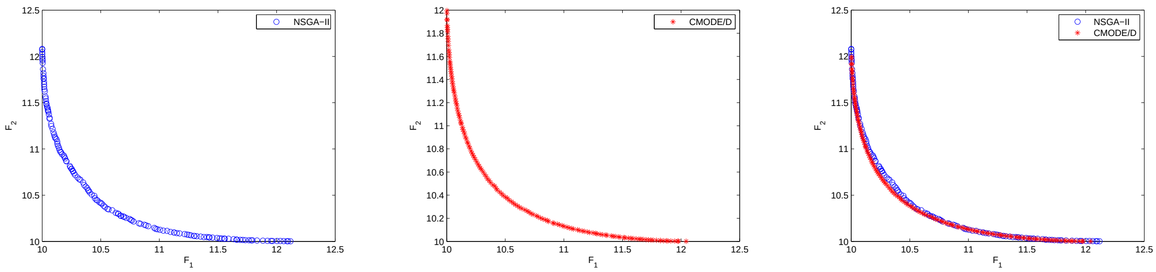
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 15: The PF produced by NSGA-II and CMODE/D for problem 9(a) ($K = 10$).



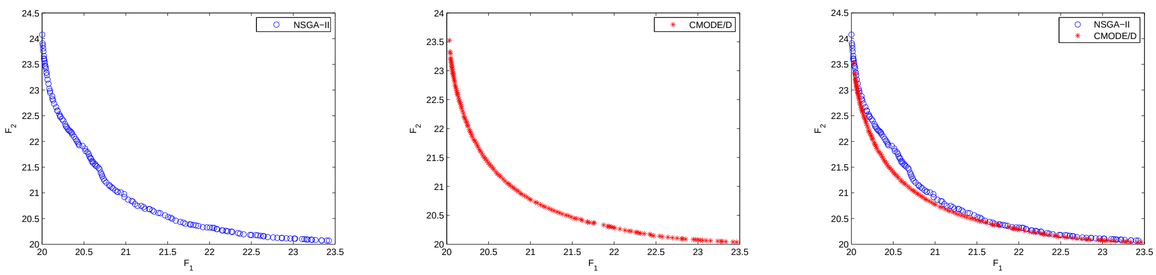
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 16: The PF produced by NSGA-II and CMODE/D for problem 9(b) ($K = 20$).



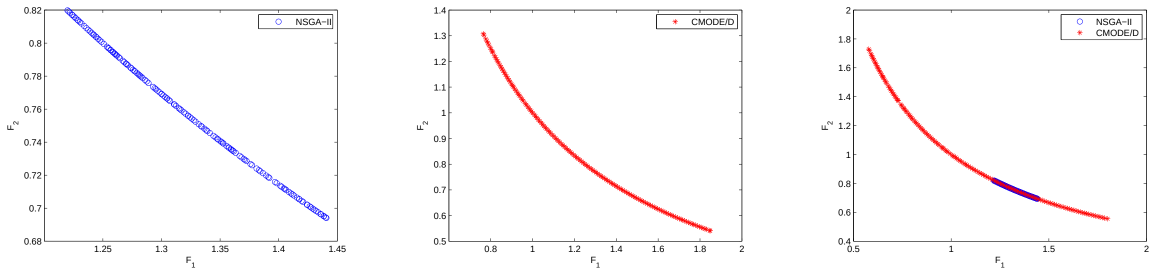
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 17: The PF produced by NSGA-II and CMODE/D for problem 10(a) ($K = 10$).



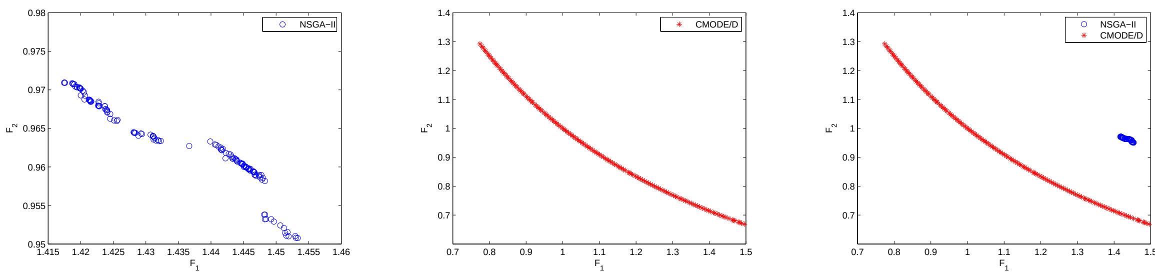
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 18: The PF produced by NSGA-II and CMODE/D for problem 10(b) ($K = 20$).



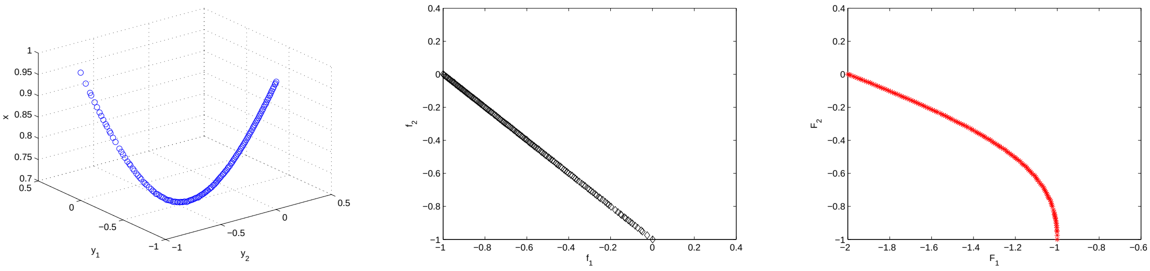
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 19: The PF produced by NSGA-II and CMODE/D for problem 11(a) ($K = 10$).



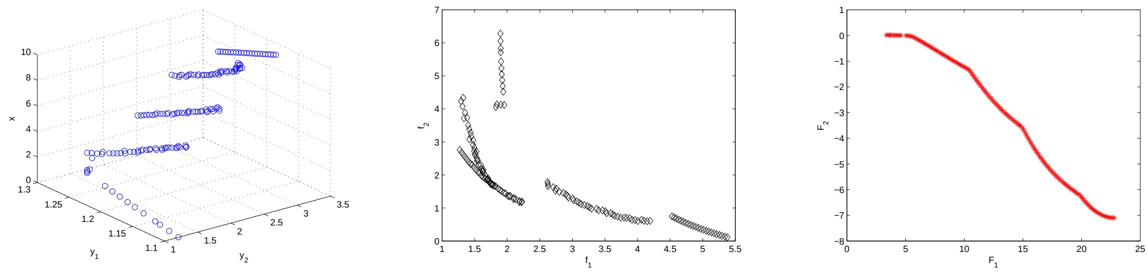
(a) NSGA-II (b) CMODE/D (c) Comparison of two algorithms

Figure 20: The PF produced by NSGA-II and CMODE/D for problem 11(b) ($K = 20$).



(a) PS (b) PF of lower level MOP (c) PF of upper level MOP

Figure 21: The PS and PF produced by CMODE/D for problem 2.



(a) PS (b) PF of lower level MOP (c) PF of upper level MOP

Figure 22: The PS and PF produced by CMODE/D for problem 4.

From Table 2 and Figure 2, it is evident that the final Pareto front obtained by CMODE/D is significantly better than that gotten by NSGA-II for each test instance in terms of the average C-metric values. Especially, CMODE/D performs much better than NSGA-II in the nondomination of final Pareto front for instances 6-11 with $K = 10$ and $K = 20$.

Moreover, it is clear from Table 3 and Figure 3 that CMODE/D performs better than NSGA-II in the uniformity of final Pareto front according to average S-metric values for instances 1-6, 8, 10, 11, except instances 7 and 9.

Figures 4-20 reveal that the final Pareto front approximations obtained by CMODE/D and NSGA-II are similar in shape of the Pareto front except instance 6, but CMODE/D can achieve a better approximation to Pareto front than NSGA-II for all instances, especially for instances 6-11 with $K = 10$ and $K = 20$. It notes that NSGA-II failed in reaching the PFs for instances 6 and 8 with $K = 10$ and $K = 20$, instances 9 and 11 with $K = 20$.

In summary, both CMODE/D and NSGA-II are able to reach the satisfactory Pareto front for low dimensional instances 1-5. With the increase of the size of BLMOP, CMODE/D has obvious difference with NSGA-II in terms of runtime, convergence, and spread of distribution, but there is little difference between two algorithms in uniformity of the distribution.

5.4. The follower's response to Pareto optimal solutions

In BLMOP, the leader makes the first decision and is considered to have complete knowledge of the set of responses available to the follower. Moreover, the leader is aware that the follower observes all its decision. The follower then makes a decision that maximizes its favourable returns in its PS, given the leader's action. In order to observe the follower's response to Pareto optimal solutions produced by CMODE/D, Figures 21-22 display the PS, PF of lower level and upper level MOPs for problem 2 and 4, respectively.

Figures 21-22 reveal visually the follower's response to Pareto optimal solutions. For problem 2, CMODE/D can find a satisfactory PS to make both upper level and lower level MOPs have nice convergence and diversity in objective

Table 4: The results of performance metrics of NSGA-II (A) and NSGA-II-DEGM (D).

No.	NSGA-II $C(A, D)$	NSGA-II S	NSGA-II-DEGM $C(D, A)$	NSGA-II-DEGM S
1	1.46e-2	9.56e-1	1.31e-3	1.51e+0
2	3.24e-1	6.96e-2	1.12e-1	8.31e-3
3	1.87e-2	4.20e-3	8.01e-3	4.15e-3
4	1.32e-1	3.47e+0	9.47e-2	9.17e-2
5	5.33e-3	8.90e-3	8.01e-3	4.43e-3

Table 5: Comparison of the proposed strategy vs Gupta and Ong's strategy.

No.	CMODE/D GD	CMODE/D SP	AS-BMOEA GD	AS-BMOEA SP	SAAS-BMOEA GD	SAAS-BMOEA SP
2	8.35e-4	1.73e-1	3.90e-4	3.05e-1	4.82e-4	3.34e-1
3	4.06e-4	9.24e-2	2.25e-4	4.83e-1	2.13e-4	4.24e-1
5	4.05e-4	9.45e-2	2.29e-4	4.08e-1	2.37e-4	3.66e-1
6(a)	4.60e-3	1.83e-1	1.83e-4	3.45e-1	5.93e-4	9.65e-1

space. For problem 4, CMODE/D can obtain a PS to make upper level MOP have nice convergence and diversity in objective space, but lower level MOP possesses some dominated solutions.

5.5. Comparison of NSGA-II and NSGA-II-DEGM

In order to investigate the effect of reproduction operator in Subsection 3.3 on NSGA-II, we replace the GA operator in NSGA-II with DE operator and Gaussian mutation (GM) operator, called NSGA-II-DEGM. The same parameters in DE and GM operators are used in NSGA-II-DEGM, and other parameters are same as NSGA-II.

Table 4 shows the values of C-metric and S-metric between NSGA-II (A) and NSGA-II-DEGM (D) for problems 1-5.

From Table 4, it can be seen that in the framework of NSGA-II, GA operator performs better than DE and GM operators according to the average C-metric values for problems 1-4. But DE and GM operators obtain more uniform Pareto solutions in objective space than GA operator for problems 2-5 in terms of S-metric values. Hence, DE and GM operators are unsuitable for NSGA-II as a reproduction operator.

5.6. Comparison of the proposed strategy vs Gupta and Ong's strategy

To verify the benefit of the proposed strategy, we compare it with an existing strategy in [23]. Based on the strategy using an adaptive scalarization for BLMOP, Gupta and Ong [23] proposed two solution approach, an adaptively scalarized bilevel MOEA (AS-BMOEA) and a surrogate-assisted AS-BMOEA (SAAS-BMOEA).

In [23], two performance metrics, i.e. generational distance (GD) and spacing (SP) [14] were adopted to evaluate the performance of AS-BMOEA and SAAS-BMOEA. For comparison, we also use these two metrics to assess the performance of CMODE/D. Comparison of the proposed strategy vs Gupta and Ong's strategy are provided in Table 5.

From Table 5, AS-BMOEA and SAAS-BMOEA perform slightly better than CMODE/D in convergence, because the order of magnitude about obtained GD-value is small. However, CMODE/D is better than AS-BMOEA and SAAS-BMOEA in terms of the uniformity of the obtained solution distribution in the objective space. For problem 6(a)(K=10), contrary to a large population size of 400 and 600 evolutionary generations used in AS-BMOEA and SAAS-BMOEA, CMODE/D uses a small population size of 150 and 300 evolutionary generations to achieve the approximate PF.

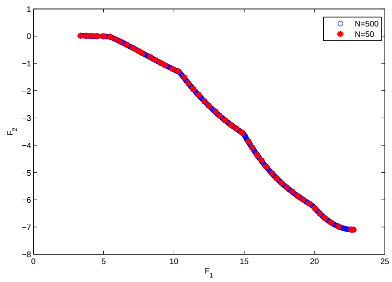


Figure 23: The Pareto fronts found by CMODE/D with $N = 50$ and $N = 500$ for problem 4.

6. Sensitivity of parameters in CMODE/D

Since we took the instance with the known PF as an analytic example in this section, we used the following inverted generational distance (IGD)[56] to assess both the diversity and convergence of the obtained solutions.

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

where P^* is a set of uniformly distributed points in the objective space along the PF. P is an obtained approximation to the PF. $d(v, P)$ is the minimum Euclidean distance between v and the points in P. A smaller value of $\mathrm{IGD}(P^*, P)$ is therefore naturally preferred. In the following experiments, we select $|P^*| = 500$ evenly distributed points in PF.

6.1. Impacts of N on CMODE/D

In CMODE/D, N is the number of optimization subproblem, and also denotes the number of a set of weight vectors. A lager N value will leads to a more expensive computational cost, and a smaller N value will results in a sparser distribution of Pareto optimal solutions. To verify our analysis, we took instance 4 as an example and chose $N = 50$ and $N = 500$, i.e. a set of 50 or 500 weight vectors, and kept all the other parameters as the same as in Section 5.1.

Figure 23 plots the final solutions obtained in a single run for $N = 50$ and $N = 500$. It is evident that CMODE/D found 50 evenly distributed Pareto solutions for $N = 50$, and 500 evenly distributed Pareto solutions for $N = 500$. Clearly, the computational cost of CMODE/D with $N = 500$ is ten times as much as that of CMODE/D with $N = 50$. It is also shown that CMODE/D can obtain a satisfactory Pareto front using a small population. Certainly, this advantage of CMODE/D benefits from its decomposition strategy.

6.2. Impacts of T on CMODE/D

The neighborhood relationship among weight vectors is a characteristic of MOEA/D [51]. To investigate the impact of the neighborhood size T on the performance of CMODE/D, we have tested different settings of T for instance 4. All the parameter settings are the same as Section 5.1, except the settings of T, which are set to be 5, 10, 50, 100, and 150.

Figure 24 shows the mean IGD-metric values in CMODE/D with different settings of T. It is evident that CMODE/D performs well with the setting of $T = 20$. However, when T is small ($T = 5$) or large ($T = 150$), the mean IGD-metric values have no significant difference.

6.3. Impacts of SF and CR on CMODE/D

Scaling factor SF and crossover rate CR are two control parameters in the DE operator for generating new solutions. To investigate the sensitivity of SF and CR in CMODE/D respectively, we have tested $SF = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$ with $CR = 0.6$, and $CR = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$ with $SF = 0.5$ on instance 4. All the parameter settings are the same as Section 5.1, except the settings of SF or CR.

Figure 25 shows the average IGD-metric values found by CMODE/D with different combinations (SF, CR) for instance 4. It is obvious that CMODE/D is less sensitive to the settings of SF and CR.

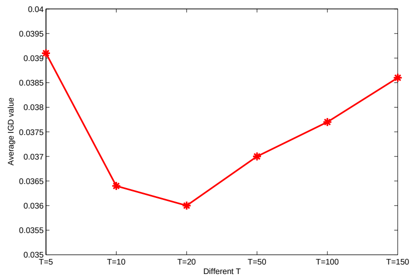


Figure 24: The average IGD-metric value found by CMODE/D with $T = 5, 10, 20, 50, 100, 150$ for problem 4.

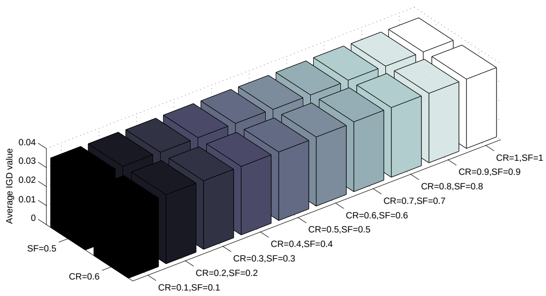


Figure 25: The average IGD-metric value found by CMODE/D with different combinations (SF, CR) for problem 4.

6.4. Impacts of Θ *on CMOEA/D*

Penalty factor Θ is a key parameter in the penalty function method. In comparison with $\Theta = 10000$ used in Section 5.1, we have tested $\Theta = 100, 1000, 100000, 1000000$ on instance 2. The other algorithmic parameters are the same as Section 5.1.

Figure 26 plots the average IGD-metric values found by CMODE/D with different penalty factors Θ for instance 2. It is clear that CMODE/D performs well with the setting of $\Theta = 10000$. However, there is no significant difference in the magnitude of IGD-metric from $\Theta = 100$ to $\Theta = 1000000$.

6.5. Impacts of other parameters on CMOEA/D

In CMODE/D, except aforementioned key parameters, there exist other parameters, i.e. control parameter μ in the smooth function (7), Gaussian mutation probability p_m and control parameter δ of coefficient σ_i in Gaussian mutation (19). To investigate the effect of these parameters on CMODE/D, we do some experiments on only one parameter each time without tuning other parameters using IGD-metric for problem 4. Table 6 provides IGD-metric values of these parameters for problem4.

Table 6: Impacts of other parameters on CMOEA/D.

	p_m		μ				δ				
	0.6	0.9	0.01	0.001	0.0001	0.00001	5	10	20	30	50
IGD	3.60e-2	3.67e-2	3.59e-2	3.58e-2	3.60e-2	3.62e-2	3.65e-2	3.63e-2	3.60e-2	3.64e-2	3.61e-2

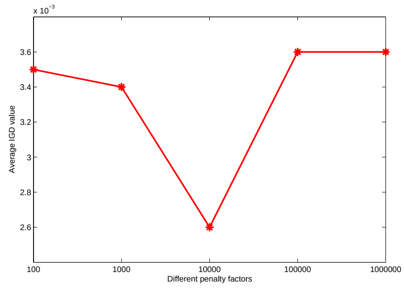


Figure 26: The average IGD-metric value found by CMODE/D with different penalty factors Θ for problem 2.

From Table 6, it is clear that CMODE/D is less sensitive to these parameters, because IGD-metric values are close to 3.60e-2.

7. Conclusion

This paper presents a transformation model for BLMOP with multiple objective functions at both levels. The final transformation model can be easily dealt with by existing MOEAs as long as minor modifications are made for them.

The lower level MOP is transformed into a single objective optimization by adaptive weighted sum scalarization. By using KKT optimality conditions to the lower level single objective optimization problem, the original BLMOP is then converted into a single level MOP with the complementarity constraints. After the smoothing technique is applied to deal with the complementarity constraints, a single-level nonlinear constrained MOP with the lower level weight vector is obtained. To solve this constrained MOP, a combined coding strategy comprised of the upper level variable and the lower level weight vector is adopted such that the upper level decision variable and the lower level weight vector evolve in their own decision space simultaneously, and then two multiobjective evolutionary frameworks are provided.

A constrained multiobjective differential evolution algorithm based on decomposition (CMODE/D) and NSGA-II are utilized to solve this transformation model. Although both CMODE/D and NSGA-II are different algorithmic frameworks, the numerical results show that two algorithms can effectively locate the Pareto optimal solutions for low dimensional instances, and each of two algorithms has own advantages, but CMODE/D performed better than NSGA-II.

The solution approach only for BLMOP with multiple objectives at both levels is proposed, yet the method for the problems in the third category can be modified to solve the BLMOPs in other two categories.

In the future, we will further study the general BLMOP, in which the lower level programming problem has poor properties, such as the nondifferentiability and nonconvexity.

Acknowledgements

The authors would like to thank the anonymous reviewers and the associate editor for their many valuable suggestions and comments, which help us to improve this paper greatly. This work was supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2013JM1022), and the National Natural Science Foundation of China (Grant No. 61401322). The first author gratefully acknowledges the support from China Scholarship Council (Grant No. 201306965006).

References

[1] M. J. Alves, S. Dempe, and J. J. Júdice, Computing the Pareto frontier of a bi-objective bi-level linear problem using a multiobjective mixed-integer programming algorithm, Optimization, vol. 61, no. 3, pp. 335-358, 2012.

[2] M. J. Alves, and J. P. Costa, An algorithm based on particle swarm optimization for multiobjective bilevel linear problems, Applied Mathematics and Computation, vol. 247, pp. 547-561, 2014.
[3] J. S. Angelo, E. Krempser, and H. J. C. Barbosa, Differential evolution for bilevel programming, In 2013 IEEE Congress on Evolutionary Computation (CEC), pp. 470-477, 2013.
[4] Z. Ankhili, A. Mansouri, An exact penalty on bilevel programs with linear vector optimization lower level, European Journal of Operational Research, vol. 197, pp. 36-41, 2009.
[5] J. F. Bard, Practical bilevel optimization: algorithms and applications, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.
[6] H. Bonnel, J. Morgan, Semivectorial bilevel optimization problem: penalty approach, Journal of Optimization Theory and Applications, vol. 131, no. 3, pp. 365-382, 2006.
[7] P. A. N. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans on Evolutionary Computation, vol. 7, no.2, pp. 174-188, 2003.
[8] H. I. Calvete, C. Galé, Linear bilevel programs with multiple objectives at the upper level, Journal of Computational and Applied Mathematics, vol. 234, no. 4, pp. 950-959, 2010.
[9] H. I. Calvete, C. Galé, On linear bilevel problems with multiple objectives at the lower level, Omega, vol. 39, pp. 33-40, 2011.
[10] C. A. C. Coello, An updated survey of GA-based multiobjective optimization techniques, ACM Comput. Surv., vol. 32, no. 2, pp. 109-143, 2000.
[11] B. Colson, P. Marcotte, and G. Savard, Bilevel programming: a survey, 4OR, vol. 3, no. 2, pp. 87-107, 2005.
[12] B. Colson, P. Marcotte, and G. Savard, An overview of bilevel optimization, Annals of Operations Research, vol. 153, no. 1, pp. 235-256, 2007.
[13] S. Das, P. N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Transactions on Evolutionary Computation, vol.15, no.1, pp.4-31, 2011.
[14] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182-197, 2002.
[15] K. Deb, and A. Sinha, Constructing test problems for bilevel evolutionary multi-objective optimization, IEEE Congress on Evolutionary Computation, CEC '09, pp. 1153-1160, 2009.
[16] K. Deb and A. Sinha, Solving bilevel multi-objective optimization problems using evolutionary algorithms, in Evolutionary Multi-criterion Optimization. 5th International Conference, EMO 2009, M. Ehrgott, C.M. Fonseca, X. Gandibleux, J.-K. Hao and M. Sevaux, eds, Lecture Notes in Computer Science, vol. 5467, Springer-Verlag, Berlin, 2009, pp. 110-124.
[17] K. Deb and A. Sinha, An efficient and accurate solution methodology for bilevel multiobjective programming problems using a hybrid evolutionary-local-search algorithm, Evolutionary Computation, vol. 18, no. 3, pp. 403-449, 2010.
[18] S. Dempe, Foundations of bilevel programming, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.
[19] S. Dempe, Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints, Optimization, vol. 52, no. 3, pp. 333-359, 2003.
[20] S. Dempe, N. Gadhi, A.B. Zemkoho, New optimality conditions for the semivectorial bilevel optimization problem, Journal of Optimization Theory and Applications, vol. 157, no. 1, pp. 54-74, 2013.
[21] G. Eichfelder, Multiobjective bilevel optimization, Math. Program., vol. 123, pp. 419-449, 2010.
[22] F. Facchinei, H. Jiang, L. Qi, A smoothing method for mathematical programs with equilibrium constraints, Math. Program, vol.85, pp.107-134, 1999.
[23] A. Gupta and Y. S. Ong, An evolutionary algorithm with adaptive scalarization for multiobjective bilevel programs, IEEE CEC 2015, May 25-28, 2015, Sendai, Japan, pp. 1636-1642.
[24] S. R. Hejazi, A. Memariani, G. Jahanshahloo, M. M. Sepehri, Linear bilevel programming solution by genetic algorithm, Computers & Operations Research, vol.29, pp.1913-1925, 2002.
[25] L. Jia, Y. Wang, A genetic algorithm for multiobjective bilevel convex optimization problems, International Conference on Computational Intelligence and Security, vol. 1, pp. 98-102, 2009.
[26] Y. W. Leung and Y. Wang, Multiobjective programming using uniform design and genetic algorithm, IEEE Trans. Syst., Man, Cybern. C, vol. 30, no. 3, pp. 293-304, 2000.
[27] X. Li, P. Tian, X. Min, A hierarchical particle swarm optimization for solving bilevel programming problems, Artificial Intelligence and Soft Computing-ICAISC 2006, Lecture Notes in Computer Science Volume 4029, pp. 1169-1178, 2006.
[28] H. Li, and Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Trans. on Evolutionary Computation, vol.13, no.2, pp. 284-302, 2009.
[29] H. Li, Y. C. Jiao, F. S. Zhang, L. Zhang, An efficient method for linear bilevel programming problems based on the orthogonal genetic algorithm, International Journal of Innovative Computing, Information and Control, vol.5, pp. 2837-2846, 2009.
[30] H. Li, L. Zhang, A differential evolution with two mutation strategies and a selection based on an improved constraint-handling technique for bilevel programming problems, Mathematical Problems in Engineering, vol. 2014, pp. 1-16, 2014.
[31] H. Li, L. Zhang, and Y. C. Jiao, An interactive approach based on a discrete differential evolution algorithm for a class of integer bilevel programming problems, International Journal of Systems Science, vol.47, no.10, pp.2330-2341, 2016.
[32] H. Li, Q. Zhang, Q. Chen, L. Zhang, Y. C. Jiao, Multiobjective differential evolution algorithm based on decomposition for a type of multiobjective bilevel programming problems, Knowledge-Based Systems, vol. 107, pp. 271-288, 2016.
[33] B. Liu, Stackelberg-nash equilibrium for multilevel programming with multiple followers using genetic algorithms, Computers & Mathematics with Applications, vol. 36, no. 7, pp. 79-89, 1998.
[34] H. Lu and G. G. Yen, Rank-density-based multiobjective genetic algorithm and benchmark test function study, IEEE Trans. Evol. Comput., vol. 7, no. 4, pp. 325-343, 2003.
[35] I. Nishizaki, and M. Sakawa, Stakelberg solutions to multiobjective two-level linear programming problems, Journal of Optimization and Theory Application, vol. 103, pp. 161-182, 1999.
[36] V. Oduguwa, R. Roy, Bi-level optimization using genetic algorithm, in Proc. IEEE Int. Conf. Artificial Intelligence Systems, pp. 123-128,

2002.
[37] J. Schott, Fault tolerant design using single and multicriteria genetic algorithm optimization, Master Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, 1995.
[38] N. Srinivas and K. Deb, Multiobjective optimization using nondominated sorting in genetic algorithms, Evol. Comput., vol. 2, no. 3, pp. 221-248, 1994.
[39] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, vol.11, no.4, pp. 341-359, 1997.
[40] K. C. Tan, T. H. Lee, and E. F. Khor, Evolutionary algorithm with dynamic population size and local exploration for multiobjective optimization, IEEE Trans. Evol. Comput., vol. 5, no. 6, pp. 565-588, 2001.
[41] Y. Y. Tan, Y. C. Jiao, H. Li, X. K. Wang, MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition, Engineering Optimization, vol.44, no.9, pp.1095-1115, 2012.
[42] Y. Y. Tan, Y. C. Jiao, H. Li, X. K. Wang, A modification to MOEA/D-DE for multiobjective optimization problems with complicated Pareto sets, Information Sciences, vol. 213, 14-38, 2012.
[43] Y. Y. Tan, Y. C. Jiao, H. Li, X. K. Wang, MOEA/D+uniform design: A new version of MOEA/D for optimization problems with many objectives, Computers and Operations Research, vol.40, no.6, pp.1648-1660, 2013.
[44] L. N. Vicente, and P. H. Calamai, Bilevel and multilevel programming: a bibliography review, Journal of Global Optimization, vol. 5, no. 3, pp. 291-306, 1994.
[45] Y. Wang, Y. C. Jiao, and H. Li, An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handing scheme, IEEE Transanctions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol. 35, no. 2, pp. 221-232, 2005.
[46] Y. Wang, H. Li, and C. Dang. A new evolutionary algorithm for a class of nonlinear bilevel programming problems and its global convergence. INFORMS Journal on Computing, vol.23, no.4, pp. 618-629, 2011.
[47] J. Y. T. Wang, M. Ehrgott, K. N. Dirks, A. Gupta, A bilevel multi-objective road pricing model for economic, environmental and health sustainability, Transportation Research Procedia, vol. 3, pp. 393-402, 2014.
[48] J. J. Ye, Necessary optimality conditions for multiobjective bilevel programs, Mathematics of Operations Research, vol. 36, no. 1, pp. 165-184, 2011.
[49] Y. Yin, Multiobjective bilevel optimization for transportation planning and management problems, Journal of advanced transportation, vol.36, pp. 93-105, 2002.
[50] G. Zhang, J. Liu, and T. Dillon, Decentralized multi-objective bilevel decision making with fuzzy demands, Knowledge-Based Systems, vol. 20, no. 5, pp. 495-507, 2007.
[51] Q. Zhang, and H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. on Evolutionary Computation, vol.11, no.6, pp. 712-731, 2007.
[52] Q. Zhang, A. Zhou, and Y. Jin, RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm, IEEE Trans. Evol. Comput., vol. 21, no. 1, pp. 41-63, 2008.
[53] T. Zhang, T. Hu, X. Guo, Z. Chen, and Y. Zheng, Solving high dimensional bilevel multiobjective programming problem using a hybrid particle swarm optimization algorithm with crossover operator, Knowledge-Based Systems, vol.53, pp.13-19, 2013.
[54] A. Zhou, B. Y. Qu, H. Li, S. Z. Zhao, P. N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation, vol. 1, pp. 32-49, 2011.
[55] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach, IEEE Trans. Evol. Comput., vol. 3, no. 4, pp. 257-271, 1999.
[56] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Trans. Evol. Comput., vol. 7, no. 2, pp. 117-132, 2003.