A Grid Weighted Sum Pareto Local Search for Combinatorial Multi and Many-Objective Optimization

Xinye Cai, *Member, IEEE*, Haoran Sun, Qingfu Zhang, *Fellow, IEEE*, and Yuhua Huang

***Abstract*—Combinatorial multiobjective optimization problems (CMOPs) are very popular due to their widespread applications in the real world. One common method for CMOPs is Pareto local search (PLS), a natural extension of single-objective local search (LS). However, classical PLS tends to reserve all of the nondominated solutions for LS, which causes the inefficient LS, as well as unbearable computational and space cost. Due to the aforementioned reasons, most PLS approaches can only handle CMOPs with no more than two objectives. In this paper, by combining the Pareto dominance and weighted sum (WS) approach in a grid system, the grid weighted sum dominance (gws-dominance) is proposed and integrated into PLS for CMOPs with multiple objectives. In the grid system, at most one representative solution is maintained in each grid for more efficient LS, thus largely reducing the computational and space complexity. The grid-based WS approach can further guide the LS in different grids for maintaining more widely and uniformly distributed Pareto front approximations. In the experimental studies, the grid WS PLS is compared with the classical PLS, three decomposition-based LS approaches [multiobjective evolutionary algorithm based on decomposition-LS (WS, Tchebycheff, and penalty-based boundary intersection)], a grid-based algorithm (ϵ-MOEA), and a state-of-the-art hybrid approach (multiobjective memetic algorithm based on decomposition) on two sets of benchmark CMOPs. The experimental results show that the grid weighted sum Pareto local search significantly outperforms the compared algorithms and remains effective and efficient on combinatorial multiobjective and even many-objective optimization problems.**

Manuscript received October 10, 2017; revised March 28, 2018; accepted June 5, 2018. Date of publication July 23, 2018; date of current version June 6, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61300159 and Grant 61732006, in part by the National Key Basic Research Program of China (973 Program) under Grant 2014CB744901, Grant 2014CB744903, and Grant 2014CB744904, in part by the Natural Science Foundation of Jiangsu Province of China under Grant SBK2018022017, in part by ANR/RGC Joint Research Scheme under Grant A-CityU101/16, in part by the China Post-Doctoral Science Foundation under Grant 2015M571751, in part by the City University of Hong Kong under Internal Research Grant 7200386, and in part by the Fundamental Research Funds for the Central Universities of China under Grant NS2017070. This paper was recommended by Associate Editor S. Mostaghim. *(Corresponding author: Xinye Cai.)*

X. Cai, H. Sun, and Y. Huang are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China, and also with the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210023, China (e-mail: xinye@nuaa.edu.cn; nuist_sevn@yeah.net; hyuhua2k@nuaa.edu.cn).

Q. Zhang is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: qingfu.zhang@cityu.edu.hk).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2018.2849403

***Index Terms*—Combinatorial many-objective optimization, combinatorial multiobjective optimization, grid weighted sum dominance (gws-dominance), Pareto local search (PLS).**

I. Introduction

COMBINATORIAL multiobjective optimization problems (CMOPs), such as the multiobjective traveling salesman problem (MOTSP) [1]–[3], multiobjective knapsack problem (MOKP) [4], multiobjective next release problem [5]–[7], multiobjective flowshop scheduling problem [8], and multiobjective vehicle routing problem [9], [10], have attracted a great amount of attention over the past decades, due to their wide applications.

However, obtaining the exact Pareto front (PFs) of a real-world CMOP is often NP-hard by nature. For this reason, the local search (LS) heuristics and/or meta-heuristics (e.g., iterative LS [11], guided LS [12], tabu search [13], variable neighborhood search [14], ant colony optimization [15], [16], and simulated annealing [17]) have been widely adopted to approximate the PFs of CMOPs.

The Pareto LS (PLS) can be considered as a natural extension of single-objective LS [18]–[20]. It works by exploring the neighbors of a nondominated solution set and update the population by newly generated efficient solutions. Several variants of PLS have been proposed to tackle the different CMOPs [21], [22].

However, classical PLS stores all the nondominated solutions for LS, which may bring unbearable time and space complexity. It is well-known that, the number of Pareto optimal solutions is likely to increase exponentially with the increase of number of objectives. Due to this reason, most of the applications of PLS are limited to the combinatorial bi-objective optimization. Alternatively, it may be more practical to approximate PF with a good representative nondominated set of a reasonable size. Usually, convergence can be measured as the distance of the representative solutions toward the PF, which should be as small as possible. Diversity can be measured as the spread of solutions along the PF, which should be as uniform as possible [23]–[26].

As one of different ways to deal with CMOPs, decomposition methods (e.g., [27]–[34]) decompose an multiobjective optimization problem (MOP) into a set of single-objective subproblems and each subproblem can be solved by a single-objective heuristic simultaneously. A representative of such approaches is multiobjective evolutionary algorithm

based on decomposition (MOEA/D) [28]. In MOEA/D, three commonly used decomposition approaches are weighted sum (WS), Tchebycheff (TCH), and penalty-based boundary intersection (PBI) [35].

In this paper, the grid-weighted sum dominance (gws-dominance) is proposed and integrated into PLS for maintaining a number of diversely populated nondominated solutions, which is very beneficial for LS in CMOPs. The proposed PLS variant, named grid WS PLS [grid weighted sum Pareto local search (GWS-PLS)], maintains a set of gws-nondominated solutions in a grid system. Each grid is allowed to record at most one representative solution inside to reduce the overall computational and space cost; and the grid-based WS approach can further guide the LS inside different grids for maintaining better diversity. As a result, GWS-PLS is able to maintain a number of diversely populated solution set for more efficient LS; and it significantly reduces the computational and space cost compared with the original PLS, which enables it to be extended for CMOPs with more than two objectives.

The rest of this paper is organized as follows. Section II gives some preliminary knowledge on CMOPs. The gws-dominance is also proposed in this section. In Section III, the GWS-PLS is proposed based on gws-dominance. The experimental setups are in Section IV, where two benchmark CMOPs—MOTSP and MOKP are introduced. Experimental studies and discussion are presented in Section V, where we compare GWS-PLS with three decomposition-based LS approaches [MOEA/D-LS (WS, TCH, and PBI)], a grid-based approach (ϵ-MOEA-LS), the classical PLS, and one state-of-the-art multiobjective memetic algorithm based on decomposition (MOMAD) [36], which hybrids MOEA/D-LS (WS) with PLS. Finally, Section VI concludes this paper.

II. Preliminaries

A. Basic Definitions

An MOP can be stated as follows:

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

where Ω is the *decision space* and $F : \Omega \to R^m$ consists of m real-valued objective functions. The *attainable objective set* is $\{F(x)|x \in \Omega\}$. In the case when Ω is a finite set, (1) is called a combinatorial MOP (CMOP).[1]

Let $u, v \in R^m$, u is said to *dominate* v, denoted by $u \prec v$, if and only if $u_i \leq v_i$ for every $i \in \{1, \ldots, m\}$ and $u_j < v_j$ for at least one index $j \in \{1, \ldots, m\}$.[2] Given a set S in R^m, a solution in it is called nondominated in S if no other solution in S can dominate it. A solution $x^* \in \Omega$ is *Pareto-optimal* if $F(x^*)$ is nondominated in the attainable objective set. $F(x^*)$ is then called a *Pareto-optimal (objective) vector*. In other words, any improvement in one objective of a Pareto optimal solution must lead to deterioration to at least another objective. The set of all the Pareto-optimal solution is called the Pareto set (*PS*) and the set of all the Pareto-optimal objective vectors is the PF [35].

The *ideal* and *nadir objective vectors* can be used to define the ranges of PFs as follows. The *ideal objective vector* z^* is a vector $z^* = \{z_1^*, \ldots, z_m^*\}^T$, which can be computed by

$$z_j^* = \min_{x \in \Omega} f_j(x), j \in \{1, \ldots, m\}. \tag{2}$$

The *nadir objective vector* z^{nad} is a vector $z^{\text{nad}} = (z_1^{\text{nad}}, \ldots, z_m^{\text{nad}})^T$, which can be computed by

$$z_j^{\text{nad}} = \max_{x \in PS} f_j(x), j \in \{1, \ldots, m\}. \tag{3}$$

B. Weighted Sum Approach

One commonly used aggregation function is WS, which is introduced as follows.

Let $\lambda = (\lambda_1, \ldots, \lambda_m)^T$ be a direction vector, where $\lambda_j \geq 0$, $j \in 1, \ldots, m$ and $\sum_{j=1}^m \lambda_j = 1$.

1) *WS Approach:* An aggregated subproblem is defined as

$$\begin{aligned} &\text{minimize} \quad h^{\text{ws}}(x|\lambda) = \sum_{j=1}^{m} \lambda_j f_j(x) \\ &\text{subject to} \quad x \in \Omega. \end{aligned} \tag{4}$$

III. Grid Weighted Sum Dominance

In this section, some basic concepts in the grid system are first introduced. After that, we propose the gws-dominance, which can be used as a key component in GWS-PLS.

A. Basic Concepts in the Grid System

The grid system can be set up as follows. Each objective from ideal to nadir point has been divided into L equal intervals, where L is a preset parameter. The grid interval vector D for m objectives can be calculated by

$$\begin{aligned} &D = (d_1, \ldots, d_m)^T \\ &\text{where} \quad d_i = \left(z_i^{\text{nad}} - z_i^*\right)/L. \end{aligned} \tag{5}$$

Definition 1 (Grid Index): For a solution x, if $k_i = \lfloor (f_i(x) - z_i^*)/d_i \rfloor, \forall i \in \{1, \ldots, m\}$, then $K = (k_1, \ldots, k_m) \in \{0, \ldots, (L-1)\}$ is called the grid index of x.

Definition 2 (Grid Corner Point): For a solution x whose grid index is $K = (k_1, \ldots, k_m)$, its grid corner point z^{gcp} can be defined as follows:

$$\begin{aligned} &z^{\text{gcp}} = \left(z_1^{\text{gcp}}, \ldots, z_m^{\text{gcp}}\right)^T \\ &\text{where} \quad z_i^{\text{gcp}} = z_i^* + d_i \times k_i \end{aligned} \tag{6}$$

where d_i is the grid interval on the ith objective.

In the grid system, the Pareto dominance relation can be redefined as follows.

Definition 3 [Grid-Dominance (i.e., ϵ-Dominance [37], [38])]: Let $x^a, x^b \in \Omega$, $K^a = (k_1^a, \ldots, k_m^a)$, and $K^b = (k_1^b, \ldots, k_m^b)$ are their grid index, respectively. $x^a \prec_\epsilon x^b \Leftrightarrow$

$$\begin{aligned} &\forall i \in (1, \ldots, m), k_i^a \leq k_i^b \wedge \\ &\exists j \in (1, \ldots, m), k_j^a < k_j^b \end{aligned} \tag{7}$$

where $x^a \prec_\epsilon x^b$ denotes x^a grid-dominates x^b, m is the number of objectives.

[1] When m is larger than 3, it is usually called a combinatorial many-objective optimization problem.

[2] In the case of maximization, the inequality signs should be reversed.

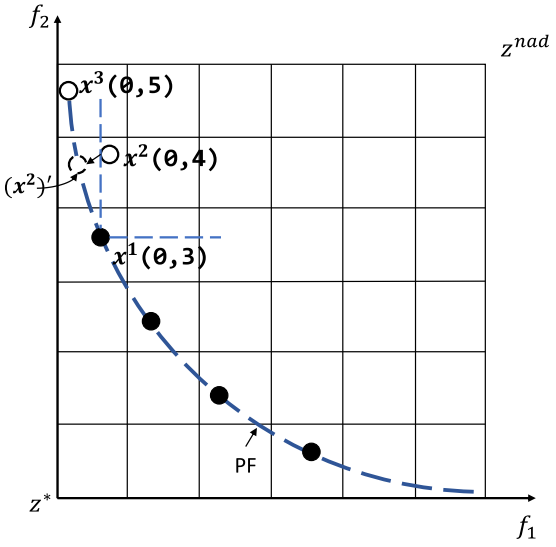


Fig. 1. Illustration of the selected solutions by sg-dominance or grid-dominance in a bi-objective optimization problem. The solid black circles (e.g., x^1) represent the existing solutions and the hollow circles (e.g., x^2 and x^3) denote the important solutions that will not be selected by grid-dominance. However, they will be selected by sg-dominance.

It is worth noting that ϵ in [37] and [38] is equivalent to the grid interval vector D in (5).

Definition 4 [Strong Grid-Dominance (sg-dominance)]: Let $x^a, x^b \in \Omega$, $K^a = (k_1^a, \ldots, k_m^a)$, and $K^b = (k_1^b, \ldots, k_m^b)$ are their grid index, respectively. $x^a \prec_{sg} x^b \Leftrightarrow$

$$\forall i \in (1, \ldots, m), k_i^a < k_i^b \tag{8}$$

where $x^a \prec_{sg} x^b$ denotes that x^a strongly grid-dominates x^b.

Definition 5 (Weakly Grid-Nondominated Set): Among a set of solutions P, the weakly grid-nondominated set of solutions P' are those that are not strongly grid-dominated by any other member of the set P.

B. Grid Weighted Sum Dominance

The use of the grid system potentially can have two benefits as follows. First, by recording at most one representative nondominated solution in each grid, the diversity of the population can be naturally achieved in the grid system. Second, by using sg-dominance, some slightly dominated, but diversely populated solutions may be reserved in the grid system. Such solutions can help guiding the LS more efficiently.

Fig. 1 gives an example, as follows. x^2 and x^3 are important solutions for maintaining the diversity of the population, in which x^3 is a boundary solution and x^2 is very likely to obtain the Pareto optimal solution $(x^2)'$ by the LS. However, x^2 and x^3 will not be kept by the grid-dominance as they are grid-dominated by x^1. On the contrary, both x^2 and x^3 will be kept with x^1 as the weakly grid-nondominated solutions by using sg-dominance, which can largely improve the diversity of the population and the efficiency of the LS. For this reason, unlike the existing grid-based approaches [37]–[40] where grid-dominance is usually used, we adopt sg-dominance with LS.

An underlying assumption of the LS is that the neighborhood of a solution in the search space is also close to this solution in the objective space. Therefore, the LS of a solution is usually conducted around the grid. Under this circumstance, it would be very desirable to further provide different search directions on different solutions for a more widely and uniformly distributed PF approximation.

However, the existing approach in the grid system may fail to achieve such a goal. A commonly adopted method is to push the representative solution in each grid toward the grid corner point [38], [40]. This method, called grid coordinate point distance (GCPD) in [40], can be very sensitive to the shape of PF, which may lead to unsatisfactory diversity. Fig. 2(a) shows the selection of solutions (marked by "•") by GCPD in a bi-objective optimization problem with a concave PF. It can be observed that, the GCPD is able to obtain the diversely distributed solutions when the PF is concave. However, when the shape of PF is convex, the use of GCPD leads to the loss of the boundary approximation as well as the unevenly distributed solution set, as shown in Fig. 2(b). In this paper, we propose the gws-dominance to address the aforementioned issue in the grid system. This is based on our following considerations.

1) GCPD can be regarded as a WS approach with a specific weight vector (direction vector) $\lambda = (1, \ldots, 1)^T$, when Manhattan distance is used. However, such a fixed direction vector for every grid apparently leads to deteriorated performance on CMOPs with the convex PFs.
2) To maintain a widely and uniformly distributed PF approximation for CMOPs with different shapes, the search direction for solutions inside a grid should consider their own grid index.

Definition 6 (Grid Direction Vector): For a solution x whose grid index is $K = (k_1, \ldots, k_m)$, its grid direction vector λ is defined based on its grid index as follows:

$$\lambda = (\lambda_1, \ldots, \lambda_m)^T$$
$$\text{where} \quad \lambda_i = 1.0/|k_i + \varepsilon| \tag{9}$$

where ε is a very small positive number to avoid the division by zero.

Definition 7 (Grid Weighted Sum (GWS)): For a solution x whose grid direction vector is λ and grid corner point is z^{gcp}, its h^{gws} can be defined as follows:

$$\text{minimize} \quad h^{\text{gws}}\left(x|\lambda, z^{\text{gcp}}\right) = \sum_{j=1}^{m} \lambda_j \left(f_j(x) - z^{\text{gcp}}\right)$$
$$\text{subject to} \quad x \in \Omega. \tag{10}$$

Different from GCPD whose search direction is fixed toward the grid corner point, the search direction of a grid (grid direction vector) for GWS is based on its grid index, defined by (9), which tends to select the boundary solutions of the nondominated set. For the same examples in Fig. 2, the solution sets obtained by the GWS are both widely and uniformly distributed for MOPs with either concave [Fig. 2(c)] or convex [Fig. 2(d)] PFs. Based on the GWS in (10) and the sg-dominance in (4), the gws-dominance can be further defined as follows.

Definition 8 (gws-dominance): Let $x^a, x^b \in \Omega$, K^a, and K^b are their grid index, respectively. $x^a \prec_{\text{gws}} x^b \Leftrightarrow$

$$\begin{cases} h^{\text{gws}}(x^a) < h^{\text{gws}}\left(x^b\right), & K^a = K^b \\ x^a \prec_{sg} x^b, & \text{otherwise} \end{cases} \tag{11}$$

where $x^a \prec_{\text{gws}} x^b$ denotes that x^a GWS dominates x^b.

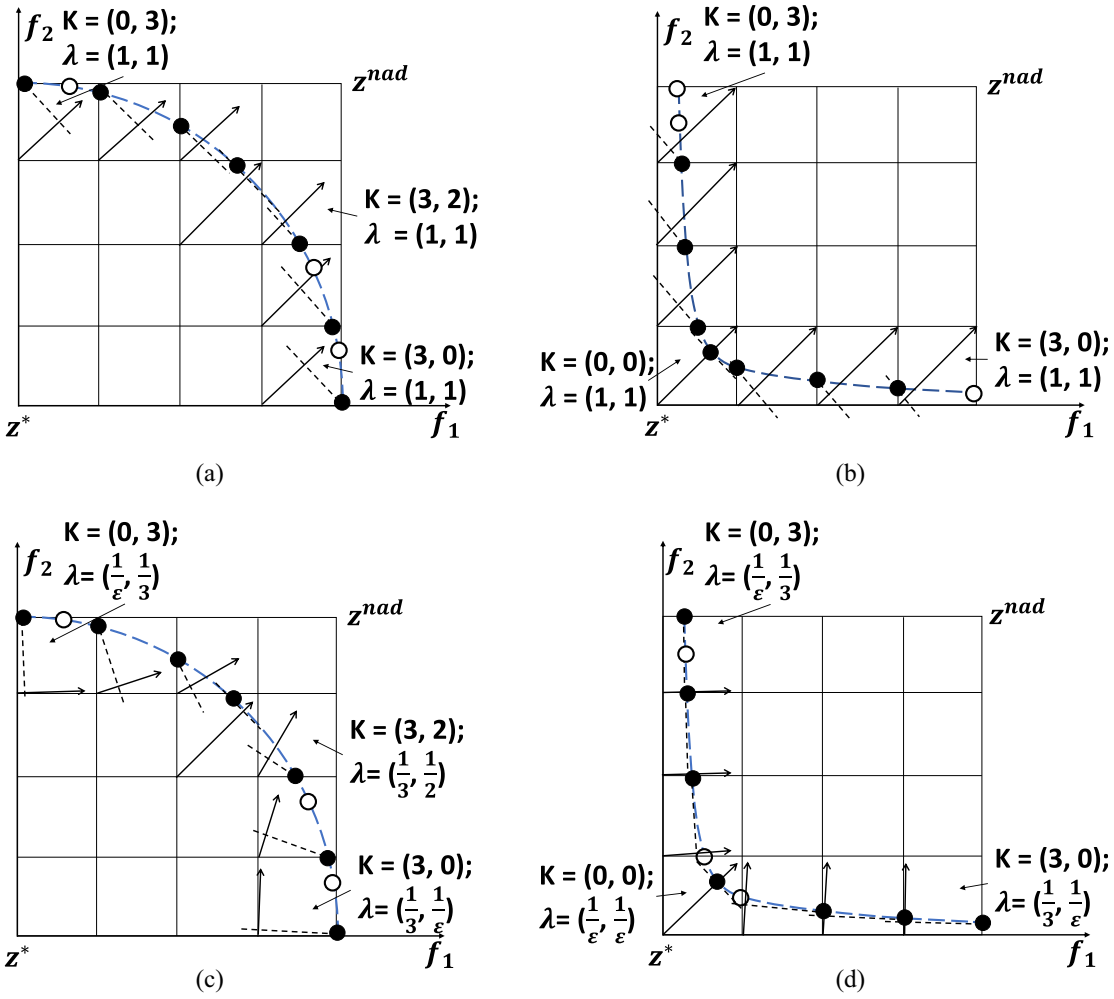


Fig. 2. Selected solutions ("•") and unselected solutions ("○") using GCPD [40] or GWS among a set of grid nondominated solutions for bi-objective optimization problems with both concave and convex PFs. The search direction vector in each grid is denoted by an arrow ("↗") and its contour line is denoted by a dashed line ("−−"). Selection of solutions by GCPD on an MOP with the (a) concave PF and (b) convex PF. Selection of solutions by GWS on an MOP with the (c) concave PF and (d) convex PF.

Definition 9 [GWS Nondominated Set (gws-nondominated set)]: Among a set of solutions P, the gws-nondominated set of solutions P' are those that are not GWS dominated by any other member of the set P.

The gws-dominance and the corresponding gws-nondominated set have the following interesting properties, which are very desirable for PLS. The proof of both two theorems can be found in the supplementary material.

Theorem 1: Suppose R is the gws-dominance relationship on a solution set P, then R is irreflexive, antisymmetric, and transitive; thus R is a strict partial order.

This property indicates that, similar to the original Pareto dominance, gws-dominance remains to be a strict partial order.

Theorem 2: For a m-objective optimization problem, given a gws-nondominated solution set P^* and a division parameter L, $|P^*|$ satisfies

$$\left|P^*\right| \leq L^m - (L-1)^m. \tag{12}$$

This property indicates that, for a m-objective problem, the upper bound for the size of the gws-nondominated solution set can be controlled by the grid division parameter L. Thus, gws-dominance can largely reduce the time and space complexity of the original PLS, which can be used to address MOPs with more than two objectives.

IV. GWS-PLS

A. Framework of GWS-PLS

In this section, the gws-dominance is integrated into the PLS framework. This variant of PLS, called GWS-PLS, is presented in Algorithm 1.

The following three populations are maintained during the optimization process.

1) A working population (WP).
2) An external population (EP).
3) A temporary population (TP) that stores the solutions for LS in the next iteration.

The initial population size N and the division parameter L are given as two inputs. It is worth noting that, as the division parameter L is fixed and only one solution is allowed in each

Algorithm 1: Framework of GWS-PLS

Input : N : the initial population size,
L : the division parameter.
Output: A solution set.

```
[WP, EP] = INITIALIZATION(N, L);
while termination criterion is not fulfilled do
    TP = φ;
    foreach x ∈ WP do
        [EP, TP] = LS(x, TP, EP);
    end
    WP = TP;
    [EP, WP] = UPDATE(WP, EP, L);
end
/* Return the gws-nondominated
   solutions in EP.                    */
return GWS-NONDOMINATED-SELECTION(EP);
```

Algorithm 2: Initialization Procedure (INITIALIZATION)

Input : N : the initial population size,
L : the division parameter.
Output: WP: the working population,
EP : the external population.

```
/* Initialize the population randomly
   or by a heuristic.                  */
P = INITIALIZE-POPULATION(N);
/* Select nondominated solutions from
   P.                                  */
WP = GWS-NONDOMINATED-SELECTION(P);
EP = WP;
```

EP is used to update the ideal and nadir points based on Eq. (2) and (3);

Setup grid system according to z^*, z^{nad} and L;

return WP, EP;

Algorithm 3: LS

Input : x : a solution,
TP : A temporary population,
EP : the external population.
Output: TP, EP

```
foreach y ∈ N(x) do
    if x ⊀_gws y then
        [ EP, isAdded ] = UPDATEEP(y, EP);
        if isAdded then
            TP = UPDATETP(y, TP);
        end
    end
end
return EP, TP;
```

Algorithm 4: Updating (UPDATE)

Input : WP : the working population,
EP : the external population,
L : the division parameter.
Output: WP, EP.

EP is used to update ideal and nadir points based on Eq. (2) and (3);

Update grid system according to z^*, z^{nad} and L;

```
/* Get the gws-nondominated set from EP
   and WP.                             */
EP = GWS-NODOMINATED-SELECTION(EP);
WP = GWS-NODOMINATED-SELECTION(WP);
return WP, EP;
```

grid in GWS-PLS, the population size varies adaptively during the optimization process.

The output is the gws-nondominated solutions of EP. They are obtained by calling GWS-NONDOMINATED-SELECTION, which is to obtain the first level of gws-nondominated solutions by the nondominated sorting [41], except that Pareto dominance is replaced with GWS-dominance.

The whole framework of GWS-PLS can be further divided into three major steps: 1) initialization; 2) gws-dominance-based LS; and 3) updating. In Algorithm 1, after the initialization step, the LS and the updating steps are iterated until some predefined stopping criterion is satisfied. In the following sections, each step is specified in detail.

B. Initialization

In the initialization procedure (Algorithm 2), a population P is either generated randomly or by a heuristic. The nondominated solutions of P are used to initialize the EP and the WP. The ideal and nadir point are initialized based on (2) and (3). The grid system is set up based on the approximation of ideal and nadir points as described in Section III-A.

C. Local Search

The procedure of the LS in GWS-PLS is given in Algorithm 3. The whole procedure is similar to the PLS except that the Pareto dominance is replaced with the gws-dominance. For each neighbor solution y of x [denoted by $N(x)$], if x cannot gws-dominate y, y is used to update EP based on gws-dominance as follows.

1) If y is gws-dominated by any solution in EP, y cannot be added to EP.
2) Otherwise y is added to EP and all the solutions gws-dominated by y in EP are removed.

If y has successfully entered into EP (denoted by a Boolean variable *isAdded*), y is also used to update TP based on gws-dominance.

D. Updating

The updating procedure is presented in Algorithm 4. The ideal point, nadir point and the grid system are first updated. After that, the gws-nondominated set in EP is selected to update EP and the gws-nondominated set in WP is selected to update WP.

E. Comparisons With ϵ-MOEA [38]

ϵ-MOEA is also a grid-based algorithm that maintains two populations (i.e., a work population and an EP). In ϵ-MOEA,

the solutions are selected based on the ϵ-dominance and distances to the grid corner points. At most a solution is maintained in a grid for maintaining the diversity of the EP. The fundamental differences between ϵ-MOEA with GWS-PLS can be summarized as follows.

1) When updating the EP, GWS-PLS adopts sg-dominance, while ϵ-MOEA adopts grid-dominance (ϵ-dominance) to select solutions located in the different grids.
2) To distinguish solutions located in the same grid, GWS-PLS uses GWS and ϵ-MOEA uses distances to the grid corner points.
3) In GWS-PLS, the WP is dynamically sized and the LS is only conducted on the solutions newly added to the EP. On the contrary, the WP in ϵ-MOEA is fixed-sized. A solution may exist in the EP for many generations in the WP and the LS may be conducted on the same solution for many times, which may lower the efficiency of ϵ-MOEA.
4) The grid interval ϵ in ϵ-MOEA is a preset parameter and the number of grids cannot be known in advance. Thus it is difficult to control the size of the obtained PF approximation for ϵ-MOEA. On the contrary, the grid division parameter (i.e., the number of the grid intervals along each objective) L is a parameter in GWS-PLS, which can be used to control the upper bound for the size of the obtained PF approximation (see Theorem 2).

V. Experimental Setups

In this section, two combinatorial MOPs, MOTSP and MOKP, are used to verify the performance of GWS-PLS. These CMOPs are introduced as follows.

A. Multiobjective Traveling Salesman Problems

The traveling salesman problem can be modeled as a graph $G(V, E)$, where $V = \{1, \ldots, n\}$ is the set of vertices (cities) and $E = \{e_{i,j}\}_{n\times n}$ is the set of edges (connections between cities). The task is to find a Hamiltonian cycle of minimal length, which visits each vertex exactly once. In the case of the MOTSP, each edge $e_{i,j}$ is associated with multiple values such as cost, length, traveling time, etc. Each of them corresponds to one criterion. Mathematically, the MOTSP can be formulated as

$$\begin{aligned} &\text{minimize } f_i(\pi) = \sum_{j=1}^{n-1} c^{(i)}_{\pi(j),\pi(j+1)} + c^{(i)}_{\pi(1),\pi(n)} \\ &\text{subject to } i = 1, \ldots, m \end{aligned} \tag{13}$$

where $\pi = (\pi(1), \ldots, \pi(n))$ is a permutation of cities; $c^{(i)}_{s,t}$ is the cost of the edge between city s and city t regarding criterion i; and m is the number of objectives to be optimized. In our case, m is equal to 2 or 3.

B. Multiobjective Knapsack Problem

Give a set of n items and a knapsack. Each item j has m weight indexes $w_{i,j} \geq 0$ and m profit indexes $p_{i,j} \geq 0 (i = 1, \ldots, m)$. The capacity of the knapsack for weight index i is c_i. A feasible solution x is an item subset which meets the following m constraints:

$$\sum_{j\in x} w_{i,j} \leq c_i, \ i = 1, \ldots, m. \tag{14}$$

The ith objective to maximize is

$$f_i(x) = \sum_{j\in x} p_{i,j}, \ i = 1, \ldots, m. \tag{15}$$

This problem is known as MOKP. The MOKP is NP-hard, which has been widely used to test the multiobjective heuristics [28], [42], [43].

C. Test Instances

In this paper, bi- and tri-objective MOTSPs called "kro," "euclid," and "cluster" are collected from [19], [44], and [45] as the test instances.[3]

These MOTSP instances are named as follows. For instance, "kroAB100" indicates the instance is "kro" with two objectives ("AB") and 100 variables.

Bi-, tri-, and four-objective MOKPs used in this paper can be referred to in [42]. Five- and six-objective MOKPs [42], [43] are randomly generated by following the suggestions given in [46], where $p_{i,j}$ and $w_{i,j}$ are random integers ranging in [10, 100]. The knapsack capacities are set to half of the total weights regarding the corresponding knapsack

$$c_i = 0.5 \times \sum_{j=1}^{n} w_{i,j} \tag{16}$$

where n is the number of items.

MOKP instances are named as follows. "knapsack250_2" means that the knapsack problem has two objectives with 250 variables.

D. Performance Metrics

1) *Set Coverage (C-Metric) [47]:* Let A and B be two solution sets of an MOP, $C(A, B)$ indicates the percentage of the solutions in B that are dominated by at least one solution in A

$$C(A, B) = \frac{|u \in B | \exists v \in A : v \prec u|}{|B|} \times 100\%. \tag{17}$$

$C(A, B) = 0$ means that no solution in B is dominated by A and $C(A, B) = 100$ means that all the solutions in B are dominated by A.

2) *Hypervolume (HV) [47]:* Let $r^* = (r^*_1, r^*_2, \ldots, r^*_m)^T$ be a reference point in the objective space that is dominated by all solutions in a PF approximation S. The HV value of S (with regard to r^*) measures the volume of region in the objective space dominated by S and bounded by r^*

$$\text{HV}(S) = \text{VOL}\left(\bigcup_{x\in S} \left[f_1(x), r^*_1\right] \times \ldots \left[f_m(x), r^*_m\right]\right) \tag{18}$$

[3]Files are downloaded from http://sites.google.com/site/thibautlust/research/.

TABLE I
GRID DIVISIONS IN DIFFERENT OBJECTIVE PROBLEMS

# of objectives	2	3	4	5	6
L	200	14	7	5	4

TABLE II
ϵ FOR DIFFERENT NUMBER OF ITEMS AND OBJECTIVES IN MOKP

items \ objs.	2	3	4	5	6
250	5	70	150	240	200
500	7	105	240	360	500
750	10	140	300	450	800

where VOL($\bullet$) indicates the Lebesgue measure. HV can measure the approximation in terms of both diversity and convergency. Obviously, the higher the HV value, the better the approximation is. In this paper, the HV value of the normalized S is calculated based on (18) with the reference point $(1.1, \ldots, 1.1)^T$.

C-metric measures the convergence performance based on Pareto dominance. It is well-known that Pareto dominance becomes much less effective on MOPs with more than three objectives. Therefore, C-metric is only used for bi- or tri-objective problems and HV is used for all the test problems.

E. Parameter Settings

1) *Parameters in MOEA/D-LS:* In MOEA/D-LS (WS, PBI, and TCH), the neighborhood size is set to 20. The population size is set to 300 for bi- or tri-objective instances, 364 for four-objective, 495 for five-objective, and 792 for six-objective instances. The penalty parameter θ in PBI is set to 5.0.
2) *Parameters in GWS-PLS:* For different MOTSP and MOKP instances, the values of the division parameter L are presented in Table I.
3) *Parameters in ϵ-MOEA-LS:* To maintain the similar population size with MOEA/D-LS and GWS-PLS, ϵ in ϵ-MOEA-LS is set to 100 for bi-objective MOTSP instances and 1000 for tri-objective MOTSP instances. The values of ϵ for different MOKP instances are presented in Table II.
4) *Parameters in MOGLS:* As suggestion in [48], the TP size is set to 16 and the WP size is set to 60. PMX crossover [49] is adopted for MOTSP and single point crossover is adopted for MOKP.
5) *Stopping Criterion:* Each algorithm was run independently for 30 times on each instance. Each of the compared algorithms is terminated when reaching the maximum number of iterations or there are no new solutions in WP for LS. The maximum number of iterations is set to 100 for MOKPs, 200 for MOTSPs (100 cities), and 500 for MOTSPs (200 or 300 cities).
6) *The Initialization of the Populations:* In MOTSP instances, the initial population is generated randomly. In MOKP instances, a greedy method [36] is adopted to generate the initial population. The initial population sizes for GWS-PLS and ϵ-MOEA-LS are set the same as that in MOEA/D-LS.
7) *Neighborhood $N(x)$ for LS:* In MOTSPs, $N(x)$ is defined by the well-known two-opt neighborhood [50]. In MOKPs, one neighbor solution in $N(x)$ is generated by removing one selected item from starting solution x and adding one unselected item without violating any constraints. It is worth noting that the same $N(x)$ is adopted in all the compared algorithms for a fair comparison.

All the compared algorithms are implemented based on the open source MOEA platform, JMetal [51]. The fundamental data structures are the same for all the compared algorithms. For a fair comparison, the population sizes of all the compared algorithms are set as close to the number of reference vectors in MOEA/D-LS as possible. Except for GWS-PLS, the parameters of all the other compared algorithms are set the same as in the original papers.

VI. EXPERIMENTAL STUDIES AND DISCUSSION

In this section, the following experiments are conducted to test the performance of GWS-PLS.

1) Investigations on the effects of GWS in GWS-PLS.
2) Comparisons with MOEA/D-LS (WS, TCH, and PBI), ϵ-MOEA, MOGLS, and NSGA-II-LS which limit the number of the obtained nondominated solutions.
3) Comparisons with PLS [18]–[20] and a state-of-the-art MOMAD [36] on multiobjective instances (MOMAD and PLS cannot be extended to combinatorial optimization problems with more than two objectives, due to their high computational complexity).
4) Investigations for the effects of grid division parameter L.

A. Effects of Grid Weighted Sum

In GWS-PLS, GWS is adopted to guide the LS for solutions in different grids. An alternative method, called GCPD [38], [40], is to always select the solution closest to its grid corner point. In Section III-B, we have demonstrated that GWS can further improve the diversity of the obtained solution set. In this section, the effects of GWS are verified experimentally by comparing PLS with GWS (i.e., GWS-PLS) and GCPD (i.e., GCPD-PLS).

Table III shows the performance of GWS-PLS and GCPD-PLS, in terms of both C-metric and HV, on combinatorial bi- or tri-objective problems. It can be observed that GWS-PLS significantly outperforms GCPD-PLS on all the instances in terms of C-metric and GWS-PLS significantly outperforms GCPD-PLS on most instances, except for knapsack500-2 and knapsack750-3. In addition, Table IV shows the performance of GWS-PLS and GCPD-PLS, in terms of HV, on combinatorial many-objective problems. It is clear to see that GWS-PLS performs significantly better than GCPD-PLS on all the many-objective instances.

The nondominated sets obtained by GWS-PLS and GCPD-PLS on kroAB100 and kroABC100 in the run with the median HV values are presented in Fig. 3 for better visualization.

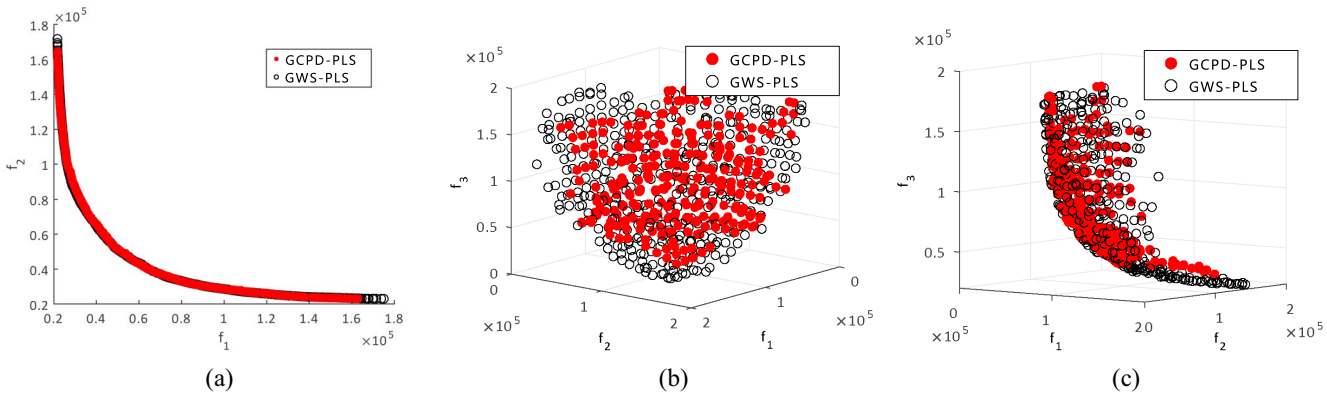


Fig. 3. Final solution sets obtained by GWS-PLS and GCPD-PLS. (a) kroAB100 instance. (b) and (c) kroABC100 instance.

TABLE III
C-METRIC AND HV VALUES OF THE NONDOMINATED SOLUTIONS OBTAINED BY GCPD-PLS AND GWS-PLS ON VARIOUS MULTIOBJECTIVE INSTANCES

Instances	*C*-metric		HV	
	C(A,B)	C(B,A)	GCPD-PLS	GWS-PLS
kroAB100	46.07	35.88	8.864e-01 (1.3e-03)$^{\approx}$	8.870e-01 (1.2e-03)
kroAB200	64.81	17.66	8.985e-01 (1.2e-03)$^{-}$	9.012e-01 (1.0e-03)
kroAB300	84.83	3.95	9.010e-01 (1.1e-03)$^{-}$	9.069e-01 (7.4e-04)
euclidAB100	46.27	36.74	8.535e-01 (1.5e-03)$^{\approx}$	8.540e-01 (1.6e-03)
euclidAB300	80.41	6.40	8.816e-01 (1.1e-03)$^{-}$	8.875e-01 (7.7e-04)
ClusterAB100	54.36	29.27	9.136e-01 (1.3e-03)$^{-}$	9.146e-01 (1.4e-03)
ClusterAB300	85.32	5.47	8.939e-01 (1.7e-01)$^{-}$	9.299e-01 (6.8e-04)
kroABC100	38.52	1.55	6.295e-01 (4.5e-03)$^{-}$	7.017e-01 (2.1e-03)
euclidABC100	38.01	1.30	5.755e-01 (3.6e-03)$^{-}$	6.401e-01 (1.8e-03)
knapsack250_2	14.56	5.68	7.754e-01 (1.1e-05)$^{-}$	7.756e-01 (5.5e-06)
knapsack500_2	22.48	17.49	7.662e-01 (3.7e-05)$^{+}$	7.661e-01 (5.7e-06)
knapsack750_2	30.89	16.35	7.623e-01 (7.3e-05)$^{-}$	7.626e-01 (2.2e-05)
knapsack250_3	13.59	5.28	5.152e-01 (1.6e-04)$^{-}$	5.164e-01 (1.4e-04)
knapsack500_3	22.70	1.07	5.168e-01 (3.1e-04)$^{-}$	5.185e-01 (3.5e-07)
knapsack750_3	16.65	2.07	5.074e-01 (1.2e-04)$^{+}$	5.067e-01 (1.0e-06)

A corresponds to GWS-PLS. B corresponds to GCPD-PLS.
'$+$', '$-$' or '$\approx$' indicate that the HV values obtained by the corresponding algorithm is significantly better, worse or similar to that of GWS-PLS on this test instance, respectively. Wilcoxon's rank sum test at a 0.05 significance level is performed.
N.A: Not available

TABLE IV
HV VALUES OF THE NONDOMINATED SOLUTION SETS OBTAINED BY GCPD-PLS AND GWS-PLS ON MANY-OBJECTIVE INSTANCES

Instances	GCPD-PLS	GWS-PLS
knapsack250_4	4.986e-01 (2.6e-02)$^{-}$	5.482e-01 (1.6e-03)
knapsack500_4	3.448e-01 (5.3e-02)$^{-}$	4.993e-01 (3.6e-03)
knapsack750_4	4.055e-01 (8.3e-02)$^{-}$	5.107e-01 (1.4e-03)
knapsack250_5	2.560e-01 (5.6e-02)$^{-}$	3.669e-01 (2.1e-03)
knapsack500_5	2.067e-01 (6.2e-02)$^{-}$	3.164e-01 (1.9e-03)
knapsack750_5	1.967e-01 (4.6e-02)$^{-}$	2.817e-01 (1.3e-03)
knapsack250_6	9.298e-02 (3.1e-02)$^{-}$	1.761e-01 (1.4e-03)
knapsack500_6	7.579e-02 (1.6e-02)$^{-}$	1.664e-01 (1.7e-03)
knapsack750_6	8.333e-02 (9.7e-03)$^{-}$	1.609e-01 (6.8e-04)

'$+$', '$-$' or '$\approx$' indicate that the HV values obtained by the corresponding algorithm is significantly better, worse or similar to that of GWS-PLS on this test instance, respectively. Wilcoxon's rank sum test at a 0.05 significance level is performed.

It can be observed from these figures that GWS-PLS cannot only obtain widely and uniformly distributed solutions but also solutions with better convergence near the boundary regions, than that obtained by GCPD-PLS. These observations verify the effectiveness of GWS, which is consistent with our motivations in Section III-B.

B. Comparisons With Algorithms That Limit the Population Size

In this section, GWS-PLS is compared with MOEA/D-LS (WS, TCH, and PBI), ϵ-MOEA-LS, MOGLS, and NSGA-II-LS on different CMOPs.

The performance of the nondominated sets obtained by all the compared algorithms, in terms of average *C*-metric, on multiobjective instances are presented in Table V. It can be observed that GWS-PLS outperforms other compared algorithms on most MOKP instances, although MOEA/D-LS (WS) performs slightly better than GWS-PLS on tri-objective MOKP instances. The convergence of MOGLS performs better than GWS-PLS on all the MOTSP instances although GWS-PLS performs better than MOGLS on all the MOKP instances. This can be explained by the facts that, for MOTSP, MOGLS uses both LS and an very efficient crossover operator (PMX) which can greatly enhance its performance while GWS-PLS only uses LS. For MOKP, MOGLS uses a single-point crossover operator which is less efficient. It is worth noting that GWS-PLS is a general framework for CMOPs which can adopt any variation operators, such as crossover and LS.

The comprehensive performance of the nondominated sets, in terms of HV, obtained by all the compared algorithms on various instances are presented in Table VI. The average sizes of nondominated solutions obtained by ϵ-MOEA-LS (denoted by $|P_1|$) and GWS-PLS (denoted by $|P_2|$) on all the instances over 30 runs are also given in the last two columns of Table VI. It can be observed that GWS-PLS significantly outperforms other algorithms, in terms of HV, on four MOTSP instances, although MOGLS performs significantly better than GWS-PLS on two MOTSP instances due to the additional use of PMX. GWS-PLS has the best performance, in terms of HV, on all the MOKP instances, except for knapsack250_2. It is worth noting that the number of nondominated solutions obtained by GWS-PLS is deliberately limited to be less than that of other compared algorithms for a fair comparison, as a larger nondominated solution set leads to a better HV value.

TABLE V
COMPARISONS OF GWS-PLS WITH MOEA/D-LS (WS, TCH, AND PBI), NSGA-II-LS, MOGLS, AND ϵ-MOEA-LS, IN TERMS OF C-METRIC

Instances	MOEA/D-LS (WS)		MOEA/D-LS (PBI)		MOEA/D-LS (TCH)		ϵ-MOEA-LS		MOGLS		NSGA-II-LS	
	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)
kroAB100	77.72	4.86	65.78	3.08	72.47	6.71	50.51	18.72	23.73	54.34	33.95	54.29
kroAB200	92.49	0.34	44.42	7.56	37.09	14.86	56.34	8.87	23.64	54.31	56.31	26.13
kroAB300	64.44	2.78	39.95	6.33	24.16	20.31	83.62	1.84	20.78	45.46	100.00	0.00
eculidAB100	80.08	4.40	54.96	10.80	80.66	5.15	43.81	28.09	32.47	44.09	46.75	42.43
eculidAB300	70.84	2.10	34.77	8.83	18.57	27.50	87.30	1.33	24.26	51.43	100.00	0.00
ClusterAB100	71.61	4.48	72.47	3.38	60.45	9.48	58.65	15.40	30.87	41.44	53.57	33.68
ClusterAB300	75.39	1.25	82.39	1.00	54.21	9.54	95.82	0.44	20.40	35.48	100.00	0.00
kroABC100	1.26	12.88	42.95	4.18	14.15	2.67	60.73	0.04	1.10	17.22	N.A.	N.A.
eculidABC100	1.25	13.37	44.33	4.56	16.12	2.28	68.18	0.02	0.43	22.88	N.A.	N.A.
knapsack250-2	19.62	0.20	96.83	0.28	73.40	3.56	4.50	17.28	69.18	0.45	99.11	0.36
knapsack500-2	26.35	2.35	93.15	0.22	75.94	1.63	22.48	18.07	90.43	0.52	99.73	0.00
knapsack750-2	33.41	1.53	87.22	1.03	62.93	4.75	35.45	21.06	93.47	0.48	99.38	0.00
knapsack250-3	0.34	3.04	54.30	0.03	12.28	2.68	7.02	3.71	21.44	1.08	N.A.	N.A.
knapsack500-3	1.00	2.42	46.68	0.00	8.91	1.65	17.24	2.26	12.38	0.42	N.A.	N.A.
knapsack750-3	1.09	2.25	37.31	0.00	8.92	1.78	16.81	0.99	10.23	0.14	N.A.	N.A.

A corresponds to GWS-PLS.
B corresponds to the compared algorithm.

TABLE VI
MEAN AND STANDARD DEVIATION VALUES OF HV OBTAINED BY MOEA/D-LS (WS, TCH, AND PBI), ϵ-MOEA, AND GWS-PLS. THE FINAL AVERAGE NUMBER OF NONDOMINATED SOLUTIONS OBTAINED BY ϵ-MOEA (P_1) AND GWS-PLS (P_2) ARE LISTED IN THE LAST TWO COLUMNS

Instances	MOEA/D-LS (TCH)	MOEA/D-LS (WS)	MOEA/D-LS (PBI)	ϵ-MOEA-LS	MOGLS	NSGA-II-LS	GWS-PLS	$\lvert P_1\rvert$	$\lvert P_2\rvert$
kroAB100	1.052e+00 (5.6e-03)$^-$	1.044e+00 (1.4e-02)$^-$	1.002e+00 (1.3e-02)	1.049e+00 (1.4e-02)$^-$	1.089e+00 (7.2e-04)$^\approx$	1.091e+00 (7.6e-03)$^+$	1.089e+00 (1.3e-03)	359.5	313.2
kroAB200	1.068e+00 (4.1e-03)$^-$	1.061e+00 (9.9e-03)$^-$	9.959e-01 (1.1e-02)	9.796e-01 (2.6e-02)$^-$	1.105e+00 (6.6e-04)$^-$	1.098e+00 (4.4e-03)$^-$	1.107e+00 (1.0e-03)	406.9	325.7
kroAB300	1.090e+00 (4.6e-03)$^-$	1.094e+00 (9.2e-03)$^-$	1.002e+00 (1.7e-02)	9.552e-01 (3.1e-02)$^-$	1.122e+00 (5.0e-04)$^\approx$	8.764e-01 (3.7e-01)$^-$	1.121e+00 (9.7e-04)	399.0	290.5
euclidAB100	9.901e-01 (7.1e-03)$^-$	1.005e+00 (1.1e-02)$^-$	9.558e-01 (1.0e-02)	9.959e-01 (1.2e-02)$^-$	1.036e+00 (1.1e-03)$^\approx$	1.027e+00 (7.7e-03)$^-$	1.036e+00 (1.4e-03)	386.5	314.2
euclidAB300	1.054e+00 (3.5e-03)$^-$	1.062e+00 (1.0e-02)$^-$	9.673e-01 (1.8e-02)	8.746e-01 (4.0e-02)$^-$	1.088e+00 (5.7e-04)$^+$	7.846e-01 (4.0e-01)$^-$	1.086e+00 (1.1e-03)	400.9	297.2
ClusterAB100	1.068e+00 (6.0e-03)$^-$	1.060e+00 (1.7e-02)$^-$	9.978e-01 (1.4e-02)	1.057e+00 (1.7e-02)$^-$	1.101e+00 (8.4e-04)$^-$	1.106e+00 (7.5e-03)$^-$	1.103e+00 (1.0e-03)	293.0	294.8
ClusterAB300	1.108e+00 (4.4e-03)$^-$	1.089e+00 (3.2e-02)$^-$	9.991e-01 (1.6e-02)	9.566e-01 (4.2e-02)$^-$	1.124e+00 (4.9e-04)$^-$	9.513e-01 (3.2e-01)$^-$	1.139e+00 (6.4e-04)	306.2	292.2
kroABC100	9.915e-01 (1.4e-03)$^-$	9.698e-01 (2.0e-03)$^-$	9.219e-01 (6.6e-03)	6.154e-01 (3.1e-02)$^-$	1.007e+00 (1.5e-03)$^+$	N.A.	9.986e-01 (1.8e-03)	322.5	311.5
euclidABC100	8.854e-01 (3.5e-03)$^-$	8.669e-01 (2.5e-03)$^-$	8.272e-01 (5.1e-03)	5.197e-01 (4.1e-02)$^-$	9.085e-01 (1.4e-03)$^+$	N.A.	8.943e-01 (2.4e-03)	327.8	287.3
knapsack250_2	9.794e-01 (2.1e-03)$^-$	9.928e-01 (1.6e-03)$^-$	9.055e-01 (4.9e-03)	9.995e-01 (6.9e-05)$^+$	9.830e-01 (5.0e-04)$^-$	9.844e-01 (1.4e-03)$^-$	9.986e-01 (2.8e-05)	259.4	245.3
knapsack500_2	9.520e-01 (4.0e-03)$^-$	9.704e-01 (1.1e-03)$^-$	8.937e-01 (5.7e-03)	9.757e-01 (2.0e-04)$^\approx$	9.650e-01 (7.9e-04)$^-$	9.649e-01 (9.8e-04)$^-$	9.765e-01 (6.8e-05)	319.1	240.1
knapsack750_2	9.563e-01 (2.7e-03)$^-$	9.718e-01 (1.7e-03)$^-$	8.179e-01 (5.6e-03)	9.760e-01 (9.5e-04)$-$	9.743e-01 (4.4e-04)$^-$	9.722e-01 (7.1e-04)$^-$	9.820e-01 (1.0e-04)	387.0	274.8
knapsack250_3	7.499e-01 (3.9e-03)$^-$	7.712e-01 (1.2e-03)$^-$	6.337e-01 (1.4e-02)	7.795e-01 (3.3e-04)$-$	7.677e-01 (9.8e-04)$^-$	N.A.	7.836e-01 (6.1e-04)	284.3	288.3
knapsack500_3	7.122e-01 (4.2e-03)$^-$	7.464e-01 (1.8e-03)$^-$	5.931e-01 (1.1e-02)$^-$	7.381e-01 (6.3e-04)	7.443e-01 (1.4e-03)$^-$	N.A.	7.560e-01 (1.5e-03)	313.5	263.8
knapsack750_3	7.215e-01 (3.1e-03)$^-$	7.495e-01 (1.7e-03)$^-$	5.302e-01 (1.1e-02)$^-$	7.555e-01 (2.9e-04)$^-$	7.504e-01 (9.5e-04)$^-$	N.A.	7.614e-01 (7.6e-04)	316.0	248.6
knapsack250_4	5.498e-01 (3.1e-03)$^-$	5.689e-01 (1.6e-03)$^-$	5.121e-01 (5.5e-03)$^-$	5.776e-01 (5.5e-04)$^-$	5.600e-01 (1.8e-03)$^-$	N.A.	5.844e-01 (1.7e-03)	364.5	365.5
knapsack500_4	5.043e-01 (2.6e-03)$^-$	5.198e-01 (1.5e-03)$^-$	4.578e-01 (5.1e-03)$^-$	5.261e-01 (1.8e-04)$^-$	5.130e-01 (1.2e-03)$^-$	N.A.	5.284e-01 (3.6e-03)	375.0	293.7
knapsack750_4	4.920e-01 (1.7e-03)$^-$	5.163e-01 (1.2e-03)$^-$	4.559e-01 (3.5e-03)$^-$	5.116e-01 (1.3e-04)$^-$	5.066e-01 (1.5e-03)$^-$	N.A.	5.257e-01 (1.2e-03)	365.1	308.1
knapsack250_5	3.847e-01 (2.1e-03)$^-$	3.775e-01 (1.7e-03)$^-$	3.331e-01 (5.4e-03)$^-$	3.601e-01 (4.1e-04)$^-$	3.722e-01 (1.5e-03)$^-$	N.A.	3.965e-01 (2.2e-03)	387.0	519.4
knapsack500_5	3.151e-01 (2.2e-03)$-$	3.110e-01 (1.5e-03)$^-$	2.796e-01 (5.9e-03)$^-$	3.006e-01 (2.8e-04)$^-$	3.217e-01 (1.5e-03)$^-$	N.A.	3.328e-01 (1.7e-03)	508.6	405.3
knapsack750_5	2.687e-01 (2.7e-03)$-$	2.670e-01 (1.5e-03)$^-$	2.455e-01 (4.3e-03)$^-$	2.597e-01 (1.5e-04)$^-$	2.755e-01 (1.3e-03)$^-$	N.A.	2.883e-01 (1.3e-03)	485.6	371.8
knapsack250_6	1.794e-01 (1.6e-03)$^-$	1.651e-01 (6.7e-04)$^-$	1.538e-01 (2.3e-03)$^-$	1.728e-01 (2.4e-04)$^-$	1.730e-01 (9.2e-04)$^-$	N.A.	1.874e-01 (1.5e-03)	1517.4	714.4
knapsack500_6	1.809e-01 (1.2e-03)$^-$	1.668e-01 (8.3e-04)$^-$	1.454e-01 (2.5e-03)$^-$	1.777e-01 (1.2e-04)$^-$	1.779e-01 (8.8e-04)$^-$	N.A.	1.829e-01 (1.9e-03)	755.8	604.7
knapsack750_6	1.751e-01 (1.2e-03)$^-$	1.590e-01 (7.5e-04)$^-$	1.325e-01 (3.1e-03)$^-$	1.664e-01 (1.1e-04)$^-$	1.729e-01 (9.4e-04)$^-$	N.A.	1.828e-01 (6.0e-04)	703.4	634.6

'+', '−' and '≈' indicate that the result is significantly better, significantly worse and statistically similar to that of GWS-PLS on this test instance, respectively. Wilcoxon's rank sum test at a 0.05 significance level is performed. N.A. means not available of the algorithm on this instance because of huge computational complexity.

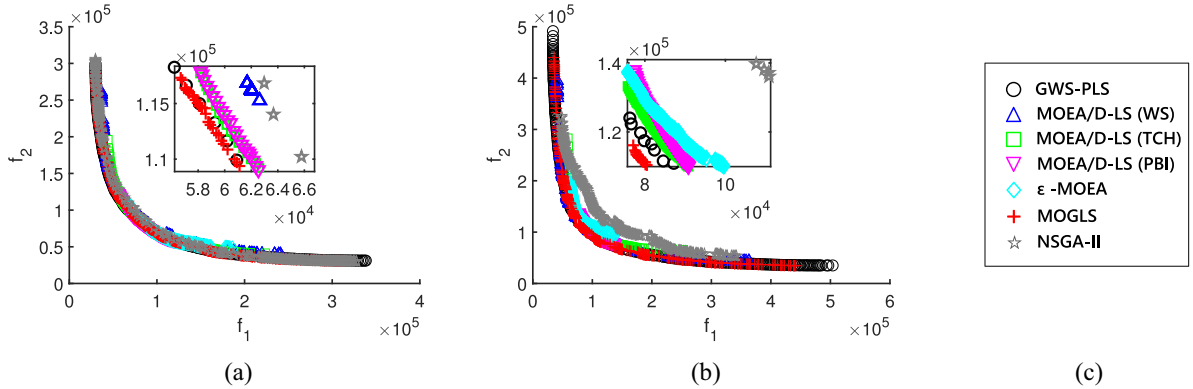


Fig. 4. Final nondominated solutions in the run with the median HV values obtained by five algorithms on kroAB200 and Cluster300. (a) kroAB200. (b) ClusterAB300. (c) Legend.

To visualize the final performance of all the compared algorithms, the nondominated solutions, obtained by various compared algorithms in the run with the median HV values on euclidAB100, kroAB200, and Cluster300 are plotted in Fig. 4. It is clear to see that GWS-PLS ("○") performs significantly better than MOEA/D-LS (WS, "△"), MOEA/D-LS (TCH,"□"), MOEA/D-LS (PBI,"▽"), ϵ-MOEA-LS ("◇"), and NSGA-II-LS ("★") in terms of both convergence and

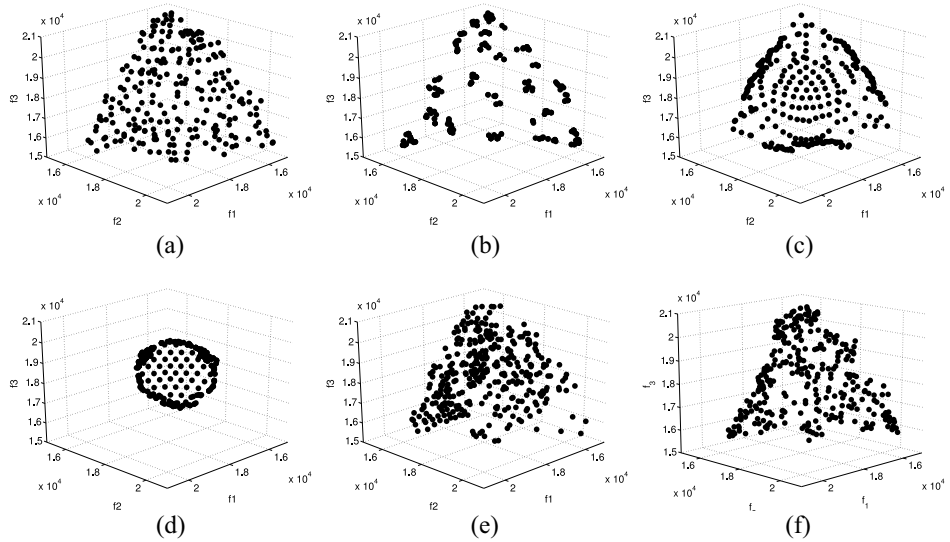


Fig. 5. Nondominated solutions obtained by five algorithms in the run with the median HV values on knapsack500-3. (a) GWS-PLS. (b) MOEA/D-LS (WS). (c) MOEA/D-LS (TCH). (d) MOEA/D-LS (PBI). (e) ϵ-MOEA-LS. (f) MOGLS.

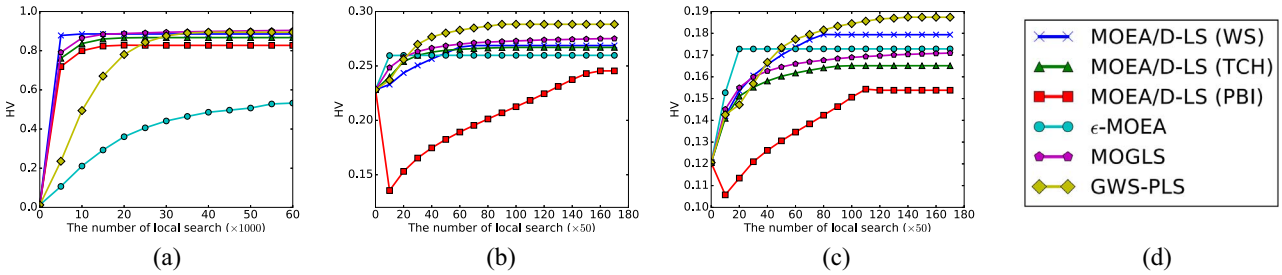


Fig. 6. Mean HV values versus the number of LS times obtained by six algorithms over 30 runs. (a) euclidABC100. (b) knapsack750_5. (c) knapsack250_6. (d) Legend of six compared algorithm.

diversity. MOGLS ("+") performs slightly better in terms of convergence than GWS-PLS but worse in terms of diversity. Fig. 5 shows the nondominated solutions obtained by six algorithms (NSGA-II-LS is unavailable due to huge computational cost by nondominated sorting) on knapsack500-3. It can be observed that the nondominated set obtained by GWS-PLS is more diversely distributed than that of other compared algorithms.

In addition, the convergence plots of the compared algorithms on euclidABC100, knapsack750_5, and knapsack250_6 are presented in Fig. 6. For euclidABC100, it can be observed that, the final performance of GWS-PLS is better than that of MOEA/D-LS (WS, TCH, and PBI) although it converges slower. For kanpsack750_5 and knapsack250_6, the final HV of GWS-PLS is the best among all the compared algorithms. Another interesting observation is that HV values of MOEA/D-LS (PBI) become even worse at the initial phase, which can be explained by the fact that PBI has the difficulty in maintaining a set of diversely distributed solutions.

Table VII illustrates the average CPU time (in seconds) for all the seven compared algorithms on all test instances.[4] It can be observed that MOEA/D-LS (WS) runs fastest among all the compared algorithms on MOTSP instances and bi-objective knapsack instances. GWS-PLS has the best performance in terms of CPU time on all the MOKP instances with more than two objectives, except for knapsack250_5 and knapsack 250_6.

[4]The hardware configurations are: Intel Xeon CPU E5-1600 3.2 GHz (processor), 32.00 GB (RAM).

C. Comparisons With PLS and MOMAD

In this section, GWS-PLS is compared with PLS [18], [20] and MOMAD [36], which tend to preserve as many nondominated solutions as possible.

To achieve the similar final population sizes with that of PLS and MOMAD, the grid division parameter L in GWS-PLS is set to 600 for bi-objective problems and 14 for tri-objective problems. For a fair comparison, the initial populations for all the compared algorithms are generated randomly. All the other parameters are set the same as in Section V-E.

The performance of GWS-PLS, PLS and MOMAD, in terms of both C-metric and HV, is presented in Table VIII. It is clear to see that PLS cannot work on MOTSP instances with 300 cities; both PLS and MOMAD cannot work on tri-objective MOTSP instances due to their high space and time complexity (denoted by N.A.). In addition, GWS-PLS performs the best on all instances in terms of C-metric. As for HV, GWS-PLS performs better than PLS and MOMAD on

TABLE VII
AVERAGE CPU TIME (IN SECONDS) FOR THE SEVEN COMPARED ALGORITHMS

Instances	MOEA/D-LS (WS)	MOEA/D-LS (TCH)	MOEA/D-LS (PBI)	ϵ-MOEA	MOGLS	NSGA-II-LS	GWS-PLS
kroAB100	7.04	18.25	18.92	214.61	137.85	180.43	160.47
kroAB200	71.39	195.26	109.06	765.04	792.42	1125.30	1206.12
kroAB300	120.72	549.86	208.99	1042.27	2437.65	2167.34	1744.75
euclidAB100	7.25	24.98	19.67	271.99	168.48	203.40	169.95
euclidAB300	134.83	536.04	201.13	970.97	2281.49	1678.40	1583.53
ClusterAB100	7.94	18.80	14.44	244.10	171.83	253.54	185.96
ClusterAB300	131.28	532.59	157.29	971.27	1687.47	1794.56	1679.31
kroABC100	44.23	208.54	140.17	198.57	889.21	N.A.	575.84
euclidABC100	40.05	171.64	141.64	268.88	885.59	N.A.	472.87
knapsack250_2	10.81	39.27	46.54	105.85	194.40	63.40	19.95
knapsack500_2	82.37	456.31	466.25	737.88	2014.50	414.34	231.65
knapsack750_2	234.56	485.53	615.80	1977.52	2822.57	873.27	415.42
knapsack250_3	29.39	79.66	77.69	148.59	317.23	N.A.	25.78
knapsack500_3	357.53	1275.64	1522.03	3129.72	3030.70	N.A.	310.11
knapsack750_3	443.69	1230.74	733.55	827.04	3846.36	N.A.	269.22
knapsack250_4	14.56	27.88	42.43	13.24	57.56	N.A.	6.67
knapsack500_4	162.13	272.19	300.59	450.45	527.99	N.A.	53.71
knapsack750_4	155.09	191.06	281.33	71.91	464.38	N.A.	48.74
knapsack250_5	39.24	20.27	28.59	14.03	67.42	N.A.	14.39
knapsack500_5	483.95	586.60	888.81	133.10	636.43	N.A.	100.03
knapsack750_5	482.93	526.06	739.51	99.46	593.69	N.A.	93.54
knapsack250_6	90.22	105.24	193.69	30.82	89.80	N.A.	31.75
knapsack500_6	847.60	726.24	1479.30	235.70	751.05	N.A.	164.65
knapsack750_6	759.20	939.31	1232.77	191.75	710.31	N.A.	148.07

N.A. means not available of the algorithm on this instance because of huge computational complexity.

TABLE VIII
COMPARISONS OF GWS-PLS, PLS, AND MOMAD

Instances	C-metric				HV		
	C(A,B)	C(B,A)	C(A,C)	C(C,A)	PLS	MOMAD	GWS-PLS
kroAB100	57.94	32.32	61.67	37.54	8.860e-01 (1.7e-03)$^-$	8.880e-01 (7.2e-04)$^-$	8.885e-01 (1.1e-03)
kroAB200	44.67	44.98	88.07	17.28	8.990e-01 (1.0e-03)$^\approx$	8.961e-01 (4.3e-04)$^-$	8.994e-01 (8.1e-04)
kroAB300	N.A.	N.A.	96.12	4.08	N.A.	9.113e-01 (2.3e-04)$^-$	9.152e-01 (6.9e-04)
euclidAB100	57.24	37.12	72.98	25.04	8.522e-01 (2.3e-03)$^-$	8.517e-01 (8.5e-04)$^-$	8.543e-01 (1.0e-03)
euclidAB300	N.A.	N.A.	96.08	5.47	N.A.	8.824e-01 (3.2e-04)$^-$	8.876e-01 (7.3e-04)
ClusterAB100	50.92	39.77	49.91	40.40	9.130e-01 (1.7e-03)$^-$	9.139e-01 (6.0e-04)$^\approx$	9.140e-01 (9.9e-04)
ClusterAB300	N.A.	N.A.	92.58	4.21	N.A.	9.285e-01 (3.5e-04)$^-$	9.321e-01 (6.5e-04)
kroABC100	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	7.017e-01 (2.1e-03)
euclidABC100	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	6.401e-01 (1.8e-03)

A corresponds to GWS-PLS. B corresponds to PLS. C corresponds to MOMAD.
'+', '−' or '≈' indicate that the HV values obtained by the corresponding algorithm is significantly better, worse or similar to that of GWS-PLS on this test instance, respectively. Wilcoxon's rank sum test at a 0.05 significance level is performed.
N.A: Not available

TABLE IX
AVERAGE POPULATION SIZES AND THE AVERAGE NUMBER OF LS BY PLS, MOMAD, AND GWS-PLS

Instances	the average population size			the average number of local search		
	PLS	MOMAD	GWS-PLS	PLS	MOMAD	GWS-PLS
kroAB100	4.631e+03	1.570e+03	7.722e+02	6.961e+04	1.909e+06	6.814e+04
kroAB200	1.440e+04	3.163e+03	8.999e+02	5.228e+05	1.945e+07	2.102e+05
kroAB300	N.A	3.214e+03	8.871e+02	N.A	4.303e+07	3.357e+05
euclidAB100	2.882e+03	1.148e+03	7.688e+02	6.410e+04	1.925e+06	6.591e+04
euclidAB300	N.A	3.173e+03	9.051e+02	N.A	4.284e+07	3.328e+05
ClusterAB100	4.672e+03	1.612e+03	8.122e+02	7.964e+04	1.898e+06	7.046e+04
ClusterAB300	N.A	3.194e+03	8.504e+02	N.A	4.008e+07	3.454e+05
kroABC100	N.A	N.A	3.421e+02	N.A	N.A	3.958e+04
euclidABC100	N.A	N.A	3.069e+02	N.A	N.A	3.601e+04

all the instances, even with much smaller population sizes (Table IX).

All these experimental results indicate that GWS-PLS is more efficient than PLS and MOMAD. More importantly, unlike PLS and MOMAD that cannot work on CMOPs with more than two objectives, GWS-PLS is able to address combinatorial tri-objective or even many-objective problems.

D. Effects of the Parameter L

In this section, the effects of the grid division parameter L on the performance of GWS-PLS are investigated. Figs. 7 and 8 present the average HV values, the average run time (in seconds) and the average final population sizes obtained by GWS-PLS on 30 runs with different L (from 50 to 500) on kroAB100 and ClusterAB300, respectively.

It can be observed in the figures that the HV values ("□") increase first but level off with the increase of L. Both run time ("○") and final obtained population size ("*") are roughly linearly correlated to L for bi-objective instances.

The linear relationship between the final population size and the value of L is consistent with Theorem 2, which can be simplified when m is 2, as follows:

$$L^2 - (L-1)^2 \Rightarrow 2L - 1. \tag{19}$$

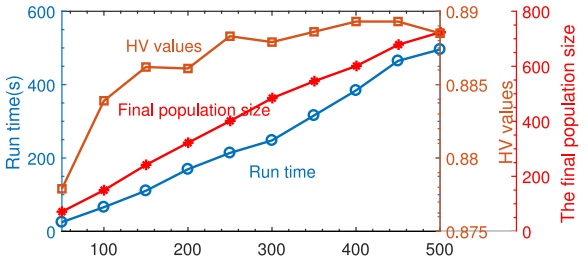


Fig. 7. Average HV values, the average run time (in seconds), and the average final population sizes obtained by GWS-PLS on 30 runs with different L (from 50 to 500) on kroAB100.

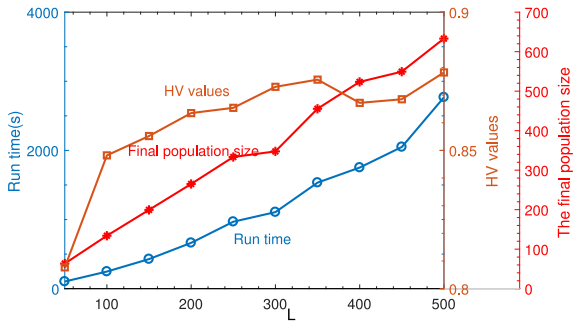


Fig. 8. Average HV values, the average run time (in seconds), and the average final population sizes obtained by GWS-PLS on 30 runs with different L (from 50 to 500) on ClusterAB300.

VII. Conclusion

In this paper, we propose a novel gws-dominance, based on which an improved version of PLS, called GWS-PLS, is further proposed. In GWS-PLS, the diversity can be better maintained by sg-dominance and GWS approach, where each grid contains at most one solution aligned by a grid-based weight vector. Furthermore, we prove that GWS-PLS remains to be a strict partial order (Theorem 1), which is a very desirable property for convergence. The maximum population size of GWS-PLS is also proved to be $L^m - (L-1)^m$ (Theorem 2), where L is the grid division parameter and m is the number of objectives. This means that, compared with the original PLS, the computational and space complexity have been effectively reduced in GWS-PLS/GW, which makes it possible to address CMOPs with more than two objectives.

The systematic experiments have been conducted to verify GWS-PLS. The comparisons of GWS-PLS with six algorithms [MOEA/D-LS (WS, TCH, and PBI), ϵ-MOEA, PLS, and MOMAD] show the efficiency of GWS-PLS. In addition, the effects of GWS are also investigated in detail.

Based on Theorem 2, it can be observed that the maximal population size grows exponentially with the number of objectives m. In other words, when m becomes a large number, given a fixed division parameter L, a very large population size is needed to ensure the effectiveness of gws-dominance. Therefore, how to handle many-objective problems with more than six objectives[5] is a future research direction. In addition, gws-dominance can be considered as a separate component to integrate into other multiobjective and many-objective optimization frameworks.

[5]When $m = 7$ and $L = 3$, $|P^*|$ would be 2059 based on Theorem 2.

References

[1] V. A. Shim, K. C. Tan, and C. Y. Cheong, "A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 5, pp. 682–691, Sep. 2012.

[2] A. Gaspar-Cunha, "A multi-objective evolutionary algorithm for solving traveling salesman problems: Application to the design of polymer extruders," in *Adaptive and Natural Computing Algorithms*. Vienna, Austria: Springer, 2005, pp. 189–193.

[3] W. Peng, Q. Zhang, and H. Li, "Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem," in *Multi-Objective Memetic Algorithms*. Berlin, Germany: Springer, 2009, pp. 309–324.

[4] H. Sato, H. E. Aguirre, and K. Tanaka, "Local dominance and local recombination in MOEAs on 0/1 multiobjective knapsack problems," *Eur. J. Oper. Res.*, vol. 181, no. 3, pp. 1708–1723, Sep. 2007.

[5] X. Cai and O. Wei, "A hybrid of decomposition and domination based evolutionary algorithm for multi-objective software next release problem," in *Proc. 10th IEEE Int. Conf. Control Autom.*, 2013, pp. 412–417.

[6] X. Cai, Y. Li, Z. Fan, and Q. Zhang, "An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 508–523, Aug. 2015.

[7] J. J. Durillo, Y. Zhang, E. Alba, M. Harman, and A. J. Nebro, "A study of the bi-objective next release problem," *Empir. Softw. Eng.*, vol. 16, no. 1, pp. 29–60, 2011.

[8] J. Arroyo and V. Armentano, "Genetic local search for multi-objective flowshop scheduling problems," *Eur. J. Oper. Res.*, vol. 167, no. 3, pp. 717–738, Dec. 2005.

[9] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Mineola, NY, USA: Dover, 1998.

[10] K. C. Tan, Y. H. Chew, and L. H. Lee, "A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows," *Comput. Optim. Appl.*, vol. 34, pp. 115–151, May 2006.

[11] L. Paquete and T. Stützle, "Design and analysis of stochastic local search for the multiobjective traveling salesman problem," *Comput. Oper. Res.*, vol. 36, no. 9, pp. 2619–2631, 2009.

[12] A. Alsheddy and E. E. P. K. Tsang, "Guided Pareto local search based frameworks for biobjective optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2010, pp. 1–8.

[13] E. L. Ulungu, J. Teghem, P. H. Fortemps, and D. Tuyttens, "MOSA method: A tool for solving multiobjective combinatorial optimization problems," *J. Multi Criteria Decis. Anal.*, vol. 8, no. 4, pp. 221–236, 1999.

[14] Y.-C. Liang and M.-H. Lo, "Multi-objective redundancy allocation optimization using a variable neighborhood search algorithm," *J. Heuristics*, vol. 16, no. 3, pp. 511–535, 2010.

[15] C. García-Martínez, O. Cordón, and F. Herrera, "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," *Eur. J. Oper. Res.*, vol. 180, no. 1, pp. 116–148, Jul. 2007.

[16] L. Ke, Q. Zhang, and R. Battiti, "MOEA/D-ACO: A multiobjective evolutionary algorithm using decomposition and antcolony," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1845–1859, Dec. 2013.

[17] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb, "A simulated annealing-based multiobjective optimization algorithm: AMOSA," *IEEE Trans. Evol. Comput.*, vol. 12, no. 3, pp. 269–283, Jun. 2008.

[18] L. Paquete and T. Stützle, "A two-phase local search for the biobjective traveling salesman problem," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2003, pp. 479–493.

[19] T. Lust and J. Teghem, "Two-phase Pareto local search for the biobjective traveling salesman problem," *J. Heuristics*, vol. 16, no. 3, pp. 475–510, 2010.

[20] T. Lust and A. Jaszkiewicz, "Speed-up techniques for solving large-scale biobjective TSP," *Comput. Oper. Res.*, vol. 37, no. 3, pp. 521–533, 2010.

[21] E. Angel, E. Bampis, and L. Gourvès, "A dynasearch neighborhood for the bicriteria traveling salesman problem," in *Metaheuristics for Multiobjective Optimisation*. Heidelberg, Germany: Springer, 2004, pp. 153–176.

[22] L. Paquete, M. Chiarandini, and T. Stützle, "Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study," in *Metaheuristics for Multiobjective Optimisation*. Heidelberg: Springer, 2004, pp. 177–199.

[23] M. Li, S. Yang, and X. Liu, "Diversity comparison of Pareto front approximations in many-objective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2568–2584, Dec. 2014.

[24] X. Cai, Z. Yang, Z. Fan, and Q. Zhang, "Decomposition-based-sorting and angle-based-selection for evolutionary multiobjective and many-objective optimization," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2824–2837, Sep. 2017.

[25] C. Bazgan, F. Jamain, and D. Vanderpooten, "Discrete representation of the non-dominated set for multi-objective optimization problems using kernels," *Eur. J. Oper. Res.*, vol. 260, no. 3, pp. 814–827, 2017.

[26] D. Vaz, L. Paquete, C. M. Fonseca, K. Klamroth, and M. Stiglmayr, "Representation of the non-dominated set in biobjective discrete optimization," *Comput. Oper. Res.*, vol. 63, pp. 172–186, Nov. 2015.

[27] T. Murata, H. Ishibuchi, and M. Gen, "Specification of genetic search directions in cellular multi-objective genetic algorithms," in *Proc. 1st Int. Conf. Evol. Multi Criterion Optim.,* 2001, pp. 82–95.

[28] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[29] A. J. Nebro, J. J. Durillo, F. Luna, B. Dorronsoro, and E. Alba, "MOCell: A cellular genetic algorithm for multiobjective optimization," *Int. J. Intell. Syst.*, vol. 24, no. 7, pp. 726–746, Jul. 2009.

[30] J. J. Durillo, A. J. Nebro, C. A. C. Coello, F. Luna, and E. Alba, "A comparative study of the effect of parameter scalability in multi-objective metaheuristics," in *Proc. Congr. Evol. Comput. (CEC)*, Hong Kong, Jun. 2008, pp. 1893–1900.

[31] J. D. Schaffer and J. J. Grefenstette, "Multiobjective learning via genetic algorithms," in *Proc. 9th Int. Joint Conf. Artif. Intell. (IJCAI)*, Los Angeles, CA, USA, 1985, pp. 593–595.

[32] E. J. Hughes, "Multiple single objective Pareto sampling," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 4. Canberra, ACT, Australia, Dec. 2003, pp. 2678–2684.

[33] E. J. Hughes, "MSOPS-II: A general-purpose many-objective optimiser," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Sep. 2007, pp. 3944–3951.

[34] X. Cai, Z. Mei, and Z. Fan, "A decomposition-based many-objective evolutionary algorithm with two types of adjustments for direction vectors," *IEEE Trans. Cybern.*, to be published.

[35] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, MA, USA: Kluwer, 1999.

[36] L. Ke, Q. Zhang, and R. Battiti, "Hybridization of decomposition and local search for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1808–1820, Oct. 2014.

[37] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multi-objective optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 263–282, 2002.

[38] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions," *Evol. Comput.*, vol. 13, no. 4, pp. 501–525, 2005.

[39] M. Laumanns and R. Zenklusen, "Stochastic convergence of random search methods to fixed size Pareto front approximations," *Eur. J. Oper. Res.*, vol. 213, no. 2, pp. 414–421, 2011.

[40] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.

[41] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[42] E. Zitzler and 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, Nov. 1999.

[43] H. Ishibuchi, N. Akedo, and Y. Nojima, "Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 264–283, Apr. 2015.

[44] G. Reinelt, "TSPLIB—A traveling salesman problem library," *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, 1991.

[45] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Stat.*, vol. 18, no. 1, pp. 50–60, 1947.

[46] D. Connolly, "Knapsack problems: Algorithms and computer implementations," *J. Oper. Res. Soc.*, vol. 42, no. 6, p. 513, 1991.

[47] E. Zitzler and 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, Nov. 1999.

[48] A. Jaszkiewicz, "Genetic local search for multi-objective combinatorial optimization," *Eur. J. Oper. Res.*, vol. 137, no. 1, pp. 50–71, 2002.

[49] T. Starkweather, S. McDaniel, K. E. Mathias, L. D. Whitley, and C. Whitley, "A comparison of genetic sequencing operators," in *Proc. ICGA*, 1991, pp. 69–76.

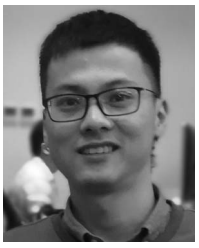
[50] P. C. Borges and M. P. Hansen, "A basis for future successes in multiobjective combinatorial optimization," Dept. Math. Model., Tech. Univ. Denmark, Kongens Lyngby, Denmark, Rep. IMM-REP-1998-8, 1998.

[51] J. J. Durillo and A. J. Nebro, "jMetal: A Java framework for multi-objective optimization," *Adv. Eng. Softw.*, vol. 42, no. 10, pp. 760–771, 2011.

Xinye Cai (M'10) received the B.Sc. degree in information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2004, the M.Sc. degree in electronic engineering from the University of York, York, U.K., in 2006, and the Ph.D. degree in electrical engineering from Kansas State University, Manhattan, KS, USA, in 2009.

He is an Associate Professor with the Department of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, where he is currently leading an Intelligent Optimization Research Group. His current research interests include evolutionary computation, optimization, data mining, and their applications.

Dr. Cai was a recipient of the Evolutionary Many-Objective Optimization Competition at the Congress of Evolutionary Computation 2017. He is an Associate Editor of *Swarm and Evolutionary Computation*.



Haoran Sun received the master's degree in computer science from the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2018.

He is currently an Engineer with MeiTuan, Shanghai, China. His current research interests include evolutionary computation and multiobjective optimization.

Qingfu Zhang (M'01–SM'06–F'17) received the B.Sc. degree in mathematics from Shanxi University, Taiyuan, China, in 1984, and the M.Sc. degree in applied mathematics and the Ph.D. degree in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively.

He is a Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong, where he is currently leading the Metaheuristic Optimization Research Group. He is a Professor on leave from the School of Computer Science and Electronic Engineering, University of Essex, Colchester, U.K., and a Changjiang Visiting Chair Professor with Xidian University. He holds two patents and has authored several research publications. His current research interests include evolutionary computation, optimization, neural networks, data analysis, and their applications.

Dr. Zhang was a recipient of the Unconstrained Multiobjective Optimization Algorithm Competition at the 2009 Congress of Evolutionary Computation for the MOEA/D, a multiobjective optimization algorithm developed in his group, and the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award. He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS. He is also an Editorial Board Member of three other international journals.

Yuhua Huang received the B.Sc. degree in communication engineering from the Harbin Institute of Technology, Harbin, China, in 1996, the M.Sc. degree in communication and information system from Xinjiang University, Ürümqi, China, in 2002, and the Ph.D. degree in signal and information processing from Southeast University, Nanjing, China, in 2006.

He is an Associate Professor with the Center of Information Technology, Nanjing University of Aeronautics and Astronautics, Nanjing. His current research interests include evolutionary computation, optimization, and data mining with their applications.