Complex & Intelligent Systems (2022) 8:4277–4299
https://doi.org/10.1007/s40747-022-00706-9

ORIGINAL ARTICLE

A knee point-driven many-objective pigeon-inspired optimization algorithm

Lihong Zhao[1] · Yeqing Ren[2] · Youqian Zeng[1] · Zhihua Cui[1] · Wensheng Zhang[3]

Received: 17 August 2021 / Accepted: 27 February 2022 / Published online: 31 March 2022

Abstract

The number of solutions obtained is too large to provide a set of solutions with good performance in the nearby area of the true Pareto front when problem-specific preferences are unavailable. Therefore, this paper proposes a knee point-driven many-objective pigeon-inspired optimization algorithm (KnMAPIO). An environmental selection strategy based on knee-oriented dominance is proposed to improve selection pressure and population diversity. In addition, a new velocity updating equation with Gaussian distribution, Cauchy distribution and Levy distribution is proposed in this paper to provide new search directions and reduce the possibility of falling into local optima. Two types of experiments are carried out in this paper: one is to compare the proposed method with four other algorithms on the knee-oriented benchmark PMOPs to verify the algorithm's performance in detecting the knee points and the knee region; another is to compare the proposed method with eight other state-of-the-art algorithms on the classic benchmark DTLZ and WFG. The results of both experiments verify the effectiveness of the proposed algorithm and the ability to approximate to the true Pareto front.

Keywords Knee point · Knee-oriented dominance · Many-objective optimization · Pigeon-inspired algorithm · Preference

Introduction

Multi-objective optimization problems (MOPs) that have multiple conflicting objectives are widely used in real-world applications in the areas of production management, economic planning, engineering design, and system control [1–3]. To solve these MOPs, researchers have proposed a number of multi-objective swarm intelligence optimizations and multi-objective evolutionary algorithms (MOEAs). As the complexity of an MOP increased, the concept of many-objective optimization problems (MaOPs), which have more than three objectives, is put forward.

Over the past few years, research on MOEAs for solving MaOPs has made significant advances. Some of this research is based on the Pareto-dominance relationship, such as the NSGA-III [4] and SPEA2 [5], which have good performance in low-dimensional search spaces. Some of this research is based on performance indicators, such as HypE [6], assigning corresponding fitness value to each solution according to the hypervolume indicator value. Some algorithms are based on a decomposition approach, such as MOEA/D [7] and MOEA/DD [8], decomposing the multi-objective optimization problems into multiple optimization subproblems and solving them separately. Others are based on a preference approach, such as in [9, 10]. Although MOEAs have been widely studied, their insufficient performance in solving MaOPs has prompted researchers to devote their attention to the field of swarm intelligence algorithms [11].

Swarm intelligence algorithms are optimization algorithms that simulate the intelligent behavior of certain insects

✉ Zhihua Cui
cuizhihua@tyust.edu.cn

Lihong Zhao
zhaolihong0312@163.com

Yeqing Ren
18435155956@163.com

Youqian Zeng
youqianzeng997@163.com

Wensheng Zhang
wensheng.zhang@ia.ac.cn

[1] School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, China

[2] Beijing University of Posts and Telecommunications, Beijing, China

[3] The State Key Laboratory of Intelligent Control and Management of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China

and animals in nature, and that display superior performance in solving complex problems. Swarm intelligence optimization algorithms include the classical ant colony algorithm (ACO) [12, 13], particle swarm algorithm (PSO) [14, 15], cuckoo search algorithm (CS) [16], bat algorithm (BA) [17, 18], and the pigeon-inspired algorithm (PIO) [19], the last of which is studied in this paper. In addition, several hybrid bio-inspired optimization approaches have been proposed, such as in the literature [20–22].

The PIO algorithm, which simulates the spontaneous homing behavior of pigeons, was proposed by Duan et al. [23] in 2014, to solve complex problems, such as the air robot path planning problem.

The PIO algorithm has been the subject of much research due to its advantages, such as its simple principle, its robustness and its requiring few parameters to be adjusted. Researchers have turned to the multi-objective pigeon-inspired optimization (MPIO) algorithm to improve the algorithm's performance in solving complex problems and to apply it in a variety of fields [24–26]. However, with the advent of MaOPs, the performance of MPIO algorithms was deemed insufficient, leading to the proposal of the many-objective pigeon-inspired algorithm (MAPIO), taking into consideration four, six, eight, and ten objectives [27].

Although some measures have been taken to improve the performance of the Pareto-dominance mechanism, the selection of individuals and the convergence are not guaranteed, and the limitations of non-dominance solutions produced by MAPIO become more prominent in MaOPs. Therefore, this paper proposes the knee point-driven many-objective pigeon-inspired optimization algorithm (KnMAPIO). The knee point is one of the Pareto optima solutions, where a slight improvement of one objective will lead to serious degradation of at least one other objective [28] in multi-objective optimization problem. It provides a set of solutions closer to the true Pareto front for individual selection when problem-specific preferences are unavailable [29].

The two main contributions of this article are as follows:

(1) A novel many-objective pigeon-inspired optimization algorithm based on knee point is proposed. An environmental selection strategy based on knee-oriented dominance is proposed for the individual selection, and the selection pressure and the diversity of the population are improved through full use of extreme points and boundary points. The proposed algorithm provides a new scheme for the swarm intelligence algorithm to solve high-dimensional problems.

(2) Also a new velocity updating equation has been proposed with Gaussian distribution [30, 31], Cauchy distribution [32], and Levy distribution [33] to reduce the possibility of falling into local optimality. With the change in the iteration stage, the distribution strategies used were adjusted to improve the search performance of the algorithm.

The rest of this paper is organized as follows. In the next section, the related work of the pigeon-inspired optimization algorithm and knee-oriented algorithms is described. The details of the proposed KnMAPIO algorithm are given in **"**The proposed algorithm for many-objective optimization". In the next section, the performance of the proposed KnMAPIO algorithm and four other algorithms is tested on the knee-oriented benchmark PMOPs, and the comparison results of the eight algorithms on the WFG and DTLZ test functions are discussed. The conclusions and directions for future work are summarized in the last section.

Related work

The PIO algorithm [23] simulates the behavior of pigeons' spontaneous homing. It is composed of two independent loops. In the first loop, the map and compass operator are used to search the target space globally, and in the secend loop, the landmark operator is used to search the solution space locally. Many scholars have modified the PIO and applied it to different fields [34, 35]. Li et al. [36] proposed a novel pigeon-inspired optimization algorithm with edge potential function and simulated annealing for the task of detecting targets in unmanned aerial vehicles (UAVs). Duan et al. [37] proposed a collaborative control method with a predation and escape pigeon-inspired algorithm for UAVs, which uses the inner and outer ring controller to solve the problem of tight formation cooperative control. Hai et al. [38] introduced a double strategic evolutionary game into PIO to improve the coordination and the search efficiency.

As the scale of the problem increases, the MPIO is proposed accordingly. Qiu et al. [39] proposed a variant of PIO, called multi-objective PIO, which is used in the parameter design of brushless direct current motors, and it uses the Pareto sorting scheme [40] and consolidation operator to enhance the selection pressure of the individuals. Liang et al. [41] proposed an MPIO algorithm with self-organizing multimodal properties and improved the space division of the solution set by the special crowding distance, which makes solving multimodal problems more efficiently. Duan et al. [42] proposed a novel MPIO with a limit cycle-based mutant mechanism to produce new solutions and search directions, and it has good performance in terms of the diversity and accuracy of solutions. Shang et al. [43] proposed a multi-objective pigeon-inspired optimization algorithm for community detection with objective functions of negative ratio association and ratio cut and adjusted the representation and update of pigeons to an adaptive process by introducing genetic operators.

With the emergence of MaOPs, the existing MPIO algorithm proved unable to provide sufficient selection pressure and solving accuracy. Cui et al. [27] proposed a pigeon-inspired optimization algorithm for many-objective optimization problems (MAPIO), which uses the BFE approach [15] and the external archive approach [15] to improve the selection capacity of the individuals.

However, the performance of MAPIO proved insufficient when user or problem-specific preferences were not available. To solve this problem, this paper introduces a knee point mechanism to supply Pareto dominance, and proposes a novel many-objective pigeon-inspired optimization algorithm based on knee point. In many-objective optimization problems, the knee points are a subset of the Pareto optimal solutions, in which the improvement of one objective will result in serious degradation of at least one other objective [44]. Therefore, the solutions in the knee area of the Pareto front are selected preferentially. Zhang et al. [29] proposed an adaptive strategy to identify knee points in a small neighborhood for solving MaOPs. Their research proved that a large hypervolume will be obtained if the knee solutions are selected preferentially in many-objective optimization problems. This means that the convergence and the diversity of the algorithm are guaranteed by locating the local knee point of the non-dominant front. Yue et al. [45] proposed a knee point-driven multi-objective particle swarm optimization algorithm. Their research indicated that the knee point mechanism can select the local and global optimal particles effectively, and the proposed algorithm was used to solve sparse reconstruction in compressed sensing. Zou et al. [46] proposed a new prediction strategy with center point and knee points to solve dynamic MOPs, and the location and distribution of the Pareto front after environmental change can be predicted accurately by introducing the knee set into the predicted population. Yu et al. [47] put forward a priori knee identification multi-objective evolutionary algorithm with α-dominance, which can speed up the convergence rate and reduce misleading search processes by eliminating the dominance-resistant solutions (DRSs) in the search for knee solutions. Yu et al. [48] proposed a new MOEA for locating the knee point area by using two local dominance relationships, α-dominance and knee-oriented dominance. The α-dominance guided the search of different potential knee regions and eliminated DRSs. The knee-oriented dominance provided the precise identification of the concave knee region and identified as many knee points as possible. In this paper, according to the knee-oriented dominance strategy, the final solution set is obtained by selecting the non-dominant individuals in the critical layer.

Inspired by these studies, we introduce the environment selection based on knee-oriented dominance into the many-objective pigeon-inspired algorithm and modify the individual velocity update equation to improve the convergence of the proposed algorithm.

The proposed algorithm for many-objective optimization

In this section, the proposed KnMAPIO algorithm is described in detail. First, the general framework of KnMAPIO is presented. Next, the knee point-driven environment selection, the novel velocity update equation, and the archive update are introduced. Finally, the computational complexity of the proposed algorithm is discussed.

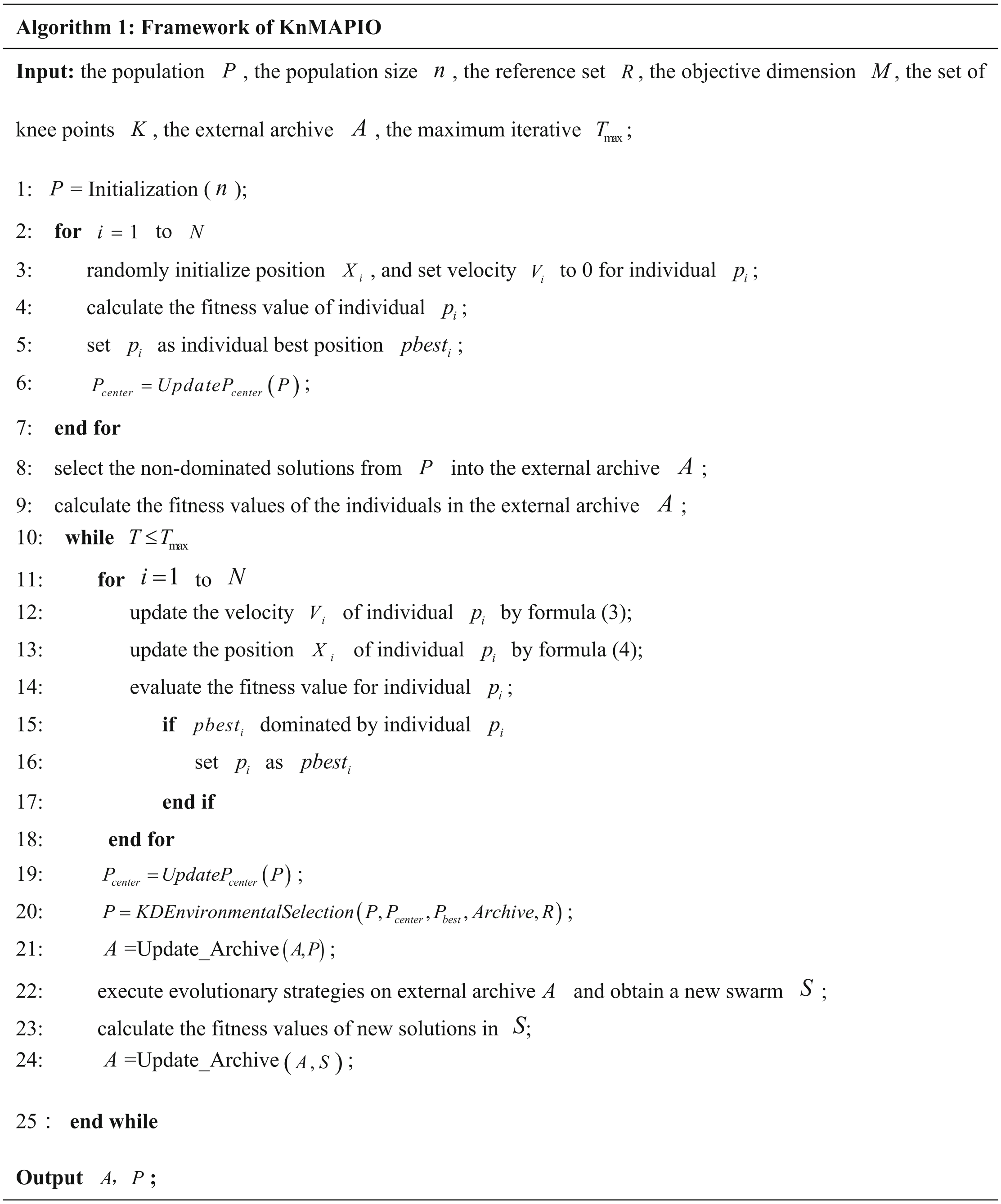
The general framework of the proposed algorithm

The general framework of the proposed KnMAPIO is presented in Algorithm 1. First, the initialization process of the population is performed, where population P is randomly initialized, and the external archive set A is set to null. For each individual p_i in the population P, the position X_i is randomly initialized, the velocity V_i is set to 0, and the fitness value of each individual p_i is calculated. At the same time, the local center point P_{center} of the population is calculated. Then, as shown in lines 8–9, the non-dominated individuals in population P are placed as the elite individuals in the external archive set A, and the corresponding fitness values are calculated. Next, the main evolutionary process is carried out. The position X_i and the velocity V_i of individual p_i are updated by formula (3) and formula (4) in "Novel velocity update equation". In so doing, a new population is obtained, and the fitness values of all individuals in the new population are calculated. Next, the local best individual $pbest_i$ and local center points P_{center} of the population are obtained through the individual dominance relationship and the corresponding formula. In line 20, the knee-oriented dominance environment selection strategy is implemented for the parent population (P, P_{center}, P_{best}, $Archive$, R) to generate good performance solutions. Next, a new swarm S is obtained by executing two evolutionary strategies—simulated binary crossover (SBX) and polynomial-based mutation (PM) on external archive A. The elite individuals are retained in the archive A again through the update mechanism in "Archive update". This entire evolutionary process is repeated until the maximum iteration is reached. Finally, the final population P and the external archive A are obtained.

The knee point-driven environmental selection

In this section, the proposed knee point-driven environmental selection strategy is introduced. More specifically, this solution adopts the knee-oriented dominance strategy in the critical layer of Pareto dominance. Figure 1 depicts the process of the environmental selection with knee-oriented dominance. The entire solution space is divided into two separate subspaces, S_1 and S_2, and the reference vectors in the subspaces are R_1 and R_2, respectively. L_1, L_2, and L_3

are the stratifications of individuals in the population after the Pareto dominance sorting. The aim of this strategy is to select the individuals in the critical layer L_3 into offspring. Assume that A, B, C, and D are the knee points on layer L_3 (for the definition of knee point, refer to "Knee point"), knee point A dominates knee point B in subspace S_1, while knee point C and knee point D do not dominate each other in subspace S_2 according to the knee-oriented dominance described in "Knee-oriented dominance relationship". Thus, knee points A, C, and D as non-dominant individuals are selected into the offspring. A new population is generated by the entire above-mentioned evolutionary process. The strategy also avoids premature convergence and improves the insufficient selection pressure caused by Pareto dominance strategy. Algorithms 2 and 3 in Sect. 3.2.3 introduce the procedure of environment selection in detail.



Algorithm 1: Framework of KnMAPIO

Input: the population P, the population size n, the reference set R, the objective dimension M, the set of knee points K, the external archive A, the maximum iterative $T_{\max}$;

```
P = Initialization ( n );
for  i = 1  to  N
    randomly initialize position  $X_i$ , and set velocity  $V_i$  to 0 for individual  $p_i$ ;
    calculate the fitness value of individual  $p_i$ ;
    set  $p_i$  as individual best position  $pbest_i$ ;
    $P_{center} = UpdateP_{center}(P)$ ;
end for
select the non-dominated solutions from  P  into the external archive  A ;
calculate the fitness values of the individuals in the external archive  A ;
while  $T \le T_{\max}$
    for  $i = 1$  to  N
        update the velocity  $V_i$  of individual  $p_i$  by formula (3);
        update the position  $X_i$  of individual  $p_i$  by formula (4);
        evaluate the fitness value for individual  $p_i$ ;
            if  $pbest_i$  dominated by individual  $p_i$
                set  $p_i$  as  $pbest_i$
            end if
    end for
    $P_{center} = UpdateP_{center}(P)$ ;
    $P = KDEnvironmentalSelection(P, P_{center}, P_{best}, Archive, R)$ ;
    A =Update_Archive (A,P) ;
    execute evolutionary strategies on external archive A  and obtain a new swarm  S ;
    calculate the fitness values of new solutions in  S;
    A =Update_Archive ( A , S ) ;
end while
```

Output A, P **;**

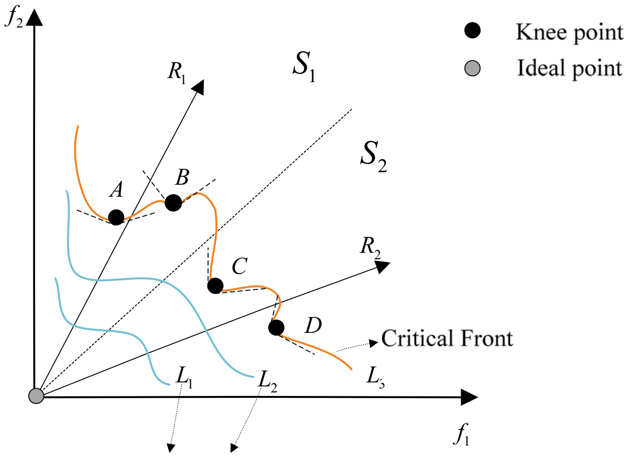


Fig. 1 Environment selection proposed in KnMAPIO

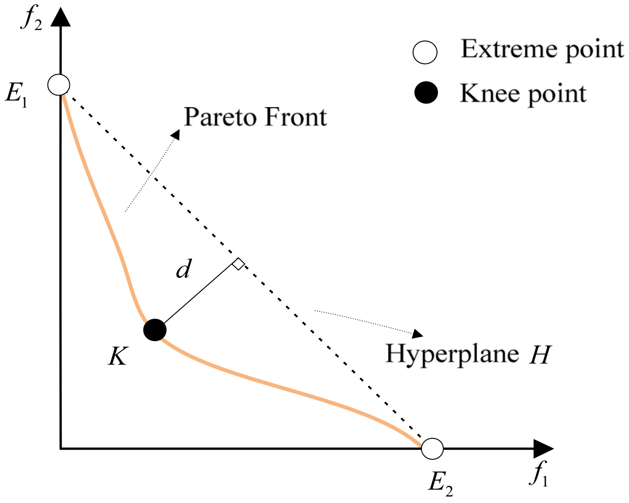


Fig. 2 Example of the knee point

Knee point

The knee point is one point in the convex hull of individual minima far from the hyperplane [48]. Figure 2 illustrates the knee point in the Pareto front. A and C are the extreme points, and they construct the hyperplane S. d is the maximum distance from point K (the knee points) on the Pareto front to the hyperplane S.

Knee-oriented dominance relationship

Pareto dominance: For any two different solutions $x,\ y \in S$, it holds that $x$dominates y, denoted by $x \prec y$, if and only if

$$\forall i \in \{1, 2, \ldots, m\},\ f_i(x) \le f_i(y) \wedge \exists\ j \in \{1, 2, \ldots, m\} f_j(x) < f_j(y). \tag{1}$$

With the increase in the number of objectives, the Pareto dominance mechanism will gradually lose its effect and all the solutions will become non-dominant to each other. At this time, the selection pressure of the algorithm will drop sharply. The knee points in knee-oriented dominance are considered as the points with better properties. The knee points and their surrounding points were taken as the first selection criterion for individuals, and the distance between the point and the hyperplane formed by the extreme point was used to measure the performance of individuals, which was converted into angle information. The appropriate dominance relationship can improve the selection pressure in the potential knee point region. As a result, the knee-oriented dominance relationship proposed in [48] is used in the environment selection of KnMAPIO. The dominance relationship is defined in formula 1.

Suppose there are two solutions M and N in the region of the convex hull of individual minima, and Mdominates $N$when the following conditions are met:

$$\begin{cases} \mu(M, N) = \left\langle \overrightarrow{N_{id}M}, \overrightarrow{MN} \right\rangle \\ \quad - \tau\left(\max\limits_{i=1.....m}\{\delta_i(M)\} + \min\limits_{i=1.....m}\{\delta_i(M)\} \right) \\ \delta_i(M) = \arctan\left(\dfrac{\sqrt{\sum_{j=1,\, j\ne i}^{m} \left(f_j(M) - f_j(N_{id})\right)^2}}{|f_i(M) - \max f_i(E) - \varepsilon|} \right), \end{cases} \tag{2}$$

where $\mu(M, N) < 0$ means that solution Mknee-oriented dominates solution N. $\left\langle \overrightarrow{N_{id}M}, \overrightarrow{MN} \right\rangle$ represents the acute angle between the two vectors $\overrightarrow{N_{id}M}$ and $\overrightarrow{MN}$. τcontrols the size of the knee region, and $\tau \in [1/2, 1]^2$. $\max\{\delta_i(M)\} +_{i=1.....m} \min\{\delta_i(M)\}_{i=1.....m}$ means that the region size is dominated by solution $M$with the help of extreme points. $\delta_i(M)$ is an acute angle determined by the ithobjective value of solution M. $f_i(M)$ is the ithobjective fitness value of individual M. $\varepsilon$is a positive constant to guarantee that the denominator is not 0. $N_{id}$is the ideal point and can be calculated as follows:

$$f_j(N_{id}) = \min f_j(E) - \varepsilon, \tag{3}$$

where $E = \{E_i | i = 1, 2, ...m\}$ is the set of extreme points, and $\varepsilon$is a positive constant.

An example of the knee-oriented dominance relationship can be seen in Fig. 3. M and N are two individuals, N_{id} is the ideal point, ϕ is the acute angle between the two vectors $\overrightarrow{N_{id}M}$ and $\overrightarrow{MN}$, and $\delta_2 = \max\{\delta_i(M)\}_{i=1.....m}$ and $\delta_3 = \min\{\delta_i(M)\}_{i=1.....m}$ represent that the solution M knee-oriented dominates solution N when ϕ is smaller than $\tau(\delta_2 + \delta_3)$. In addition, the dominance regions of individuals

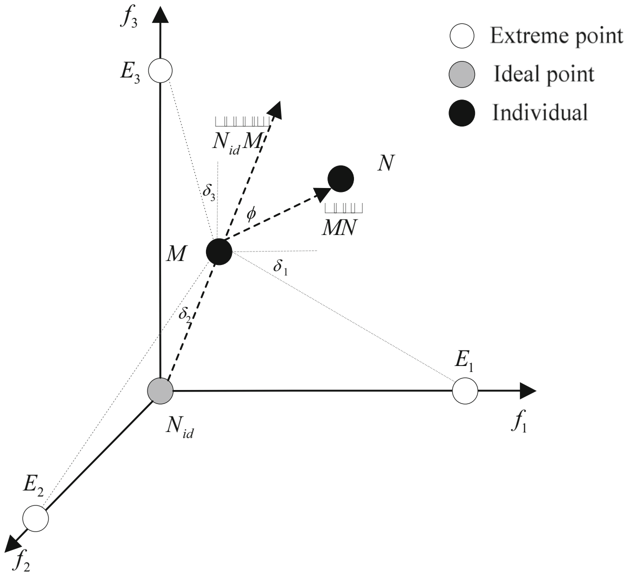


Fig. 3 Example of the knee-oriented dominance relationship

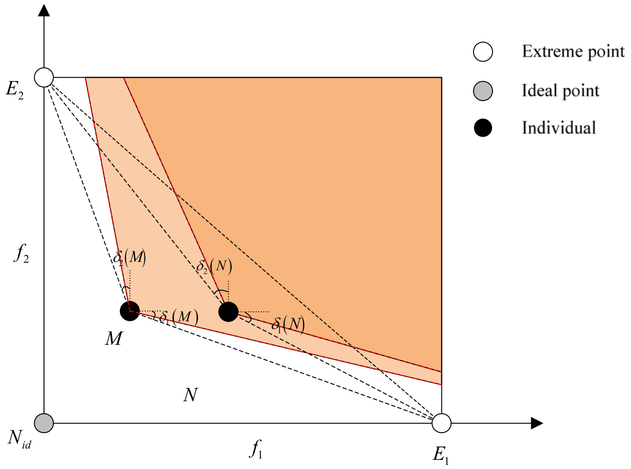


Fig. 4 Dominance region of individuals M and N

in a two-dimensional space can be seen in Fig. 4, where individual M dominates individual N, and shadows and dashed lines represent the corresponding dominance region of different individuals.

Processs of environmental selection

The process of knee point-driven environmental selection is shown in Algorithm 2 . First, the offspring P is initialized as an empty set, and the parent population Q is sorted as $L_1,\ L_2,\ \cdots L_l$ according to the Pareto domination relationship. The individuals in all layers $L_1,\ L_2,\ \cdots L_{l-1}$ before the critical layer L_l are placed into offspring P according to the layer number in ascending order. Next, only a portion of the individuals in the critical layer L_l can be selected into the offspring P according to the knee-oriented dominance selection strategy (refer to Algorithm 3), and the number of individuals in this part is $n - |P|$, where n is the size of the output offspring.

Algorithm 2: *KDEnvironmentalSelection*

```
Input: the population $Q$, the population size $n$, the reference set $R_r$, the extreme point set $E$;
$P = \varnothing$;
//* Non-dominance sorting. *//
$non-dominanceSorting(Q, R_r) = \{L_1, L_2, \cdots\}$;
for each $L_l \in \{L_1, L_2, \cdots\}$
    if $|P| + |L_l| \le n$ then
        $P = P \cup L_l$
    else
//* Knee-oriented dominance selection on critical layer *//
        $P = P \cup KDSelection(L_l, n - |P|, E, R_r)$
    end if
end for
Output $P$;
```

Algorithm 3 introduces the knee-oriented dominance selection strategy used in Algorithm 2 in detail. First, the initialization procedure is performed. Next, the solution space is divided into different subspaces ℓ_i, and each subspace is initialized to empty. Next, the solutions in the critical layer are associated with the reference vector R_r in the nearest subspace according to the grouping function, where C_1, $C_2 \cdots C_k$ stand for C_1 subspace, C_2 subspace, and C_k subspace. In lines 5–8, the individuals in subspace C_i are ordered according to the knee-oriented dominance relationship, and the front number is assigned to each individual. In line 7, the solutions in different subpopulations are recombined in a set u. Next, several sub-layers are grouped according to the knee-oriented dominance front number. The selection procedure in lines 10–14 is similar to the non-dominated sorting process in NSGA-II [40]. All solutions in the critical layer are sorted in ascending order according to the front number, and the solutions with the same front number are sorted in descending order by the crowding distance.

Algorithm 3: *KDSelection*

```
Input: the population $W$, the population size $w$, the reference set $R_r$, the extreme point set $E$;
1:  $P = \varnothing$;
2: //* Knee-oriented dominance sorting. *//
3:  $u = \{\mathrm{l}_1, \mathrm{l}_2, \mathrm{L}\} \wedge \mathrm{l}_1 = \varnothing, \mathrm{l}_2 = \varnothing, \mathrm{L}$  //*A set of empty lists in $u$.*//
4:  $C : \{C_1, C_2 \mathrm{L}\ C_k\} = Grouping(R_r)$
5: for each $C_i \in C$ do
6:        $KDdominanceSorting(C_i) = \{S^i_1, S^i_2, \mathrm{L}\}$
7:        $u = \{S^i_1 \mathrm{U}\, \mathrm{l}_1, S^i_2 \mathrm{U}\, \mathrm{l}_2, \mathrm{L}\}$
8: end for
9: for each $\mathrm{l}_i \in u \wedge |P| < l$ do
10:      if $|P| + |\ell| \le l$ then
11:            $P = P\, \mathrm{U}\, \mathrm{l}_l$
12:      else
13:            $P = P\, \mathrm{U}\, CrowdingDistance(\mathrm{l}_l, l - |P|)$
14:      end if
15: end for
Output $P$;
```

Novel velocity update equation

In this section, a novel velocity update equation is introduced, and Gaussian distribution [30, 31], Cauchy distribution [32], and Levy distribution [33] are used to improve search ability. Specifically, the Cauchy distribution tends to global search in the early iteration. The Levy distribution is used to update the individuals of the population in the late iteration, so that a few individuals can carry out local search and others can carry out global search to reduce the possibility of falling into local optimum to a certain extent. The introduction of Gaussian distribution improves the centralized search ability, which improves the convergence speed of the algorithm. These distributions are used at different stages to affect the updating of velocity, in order to provide new search directions and reduce the possibility of the algorithm falling into the local optimal.

The proposed update equation is as follows:

$$\begin{aligned} V_i(t_{now}) = {} & e^{-Rt_{now}} \cdot V_i(t_{now}-1) \\ & + r_1 \cdot Cauchy \cdot tr \cdot \left(1 - \log_T^{t_{now}}\right) \\ & \cdot \left(X_{glo} - X_i(t_{now})\right) \\ & + r_3 \cdot Levy \cdot tr \cdot \log_T^{t_{now}} \cdot (X_{cen} - X_i(t_{now})) \\ & + r_5 \cdot G \cdot \left(X_{glo} - X_{cen}\right), \end{aligned} \tag{4}$$

$$X_i(t_{now}) = X_i(t_{now}-1) + V_i(t_{now}), \tag{5}$$

where t_{now} represents the current iteration number, T represents the maximum iteration number, R denotes the map and compass operator, and tr is the migration factor to ensure a smooth transition between map and compass operator and landmark operator. $Cauchy$, $Levy$, and G represent the Cauchy distribution, Levy distribution, and Gaussian distribution, respectively.

The search combined with the Levy distribution has a relatively high probability of large stride in the process of random walk, thereby ensuring that the walk is not limited to a small local area, which can increase the diversity of the population and expand the search range. This provides a high probability of other search directions in the later iterations of the population. Gaussian distribution can search for large probability and small range variation in individual local area. Therefore, Gaussian distribution is introduced into the search process to improve the local search ability of the algorithm, and the velocity updating strategy of this part also affects the entire algorithm iteration process. Cauchy distribution has thicker and longer tails, the offspring generated in the algorithm will be more dispersed, and the diversity of offspring population will be better, which is more suitable for global optimization. Therefore, we introduce Cauchy distribution into the search process to improve the global search ability of the algorithm in the early iteration process.

X_{glo} is the position information about the global best in all individuals. The meaning of X_{cen} is the center position information of some individuals in the current iteration, and it can be calculated according to the formula (6).

$$X_{cen} = \frac{\sum_{j=1}^{n_1^x} S_{1j}^X}{n_1^x} S_1^X n_1^X, \tag{6}$$

where $\sum_{j=1}^{n_1^x} S_{1j}^X$ represents the sum of all solutions of individuals in the non-dominated set S_1^X, and n_1^X is the number of the solutions in the set S_1^X.

The r_1, r_3 and r_5 represent three learning factors and are defined as follows:

$$r_i = \begin{cases} 0, \ 0 < rand() \le \frac{1}{M} \\ 1, \ \frac{1}{M} < rand() \le 1 \end{cases}, \tag{7}$$

where $rand()$ is a random number between [0,1], and M presents the number of objectives. The part r_i located in formula (4) will not influence the $V_i(t_{now})$ if r_i is 0. With the change of parameter M, $V_i(t_{now})$ will update dynamically when r_i is 1.

Archive update

Simulated binary crossover (SBX) [49] and polynomial-based mutation (PM) [49] as another search pattern are used to provide an additional search direction in the process of the archive update, and then, the new solutions S are produced. In KnMAPIO, the archive update strategy [15, 50] is also adopted to retain the elite solutions. Meanwhile, elite individuals are selected and eventually retained in the external archive by using the BFE method [15], which not only ensures the convergence of the population but also guarantees the diversity. These two methods can guide the solutions approximate to the true PF. By comparing the Pareto-dominance relationships between S and the external archive A, archive A is updated until it reaches the terminal condition.

Computational complexity

The computational complexity of the proposed algorithm is mainly derived from the knee point-driven environment selection and the archive update. The knee point-driven environment selection includes non-dominance sorting and the knee-oriented dominance sorting. The computational complexity of the non-dominance sorting is $O\left(N^2 M\right)$ when the population size is N and the objective vector dimension is M. The knee-oriented dominance sorting is applied to the critical layer of non-dominance sorting. For any two solutions, the computational complexity is $O(M)$ when the angle of two solutions is calculated. The worst case is that all the solutions are at the same layer, and the computational complexity of knee-oriented dominance sorting is $O\left(N^2 M\right)$ at this time. The computational complexity of the crowding distance is $O(M \times N \log N)$, which is the same as that of NSGA-II [40].

The computational complexity of the archive update is mainly obtained by comparing the individuals in the population with the elite individuals in the archive set. Since the archive size is usually selected in proportion to the size of the population, the complexity is $O\left(N^2 M\right)$ in the worst case. Overall, the computational complexity of the proposed KnMAPIO is $O\left(N^2 M\right)$.

Experimental results

The proposed algorithm KnMAPIO and other four algorithms were tested on the knee−oriented benchmark PMOPs to verify the performance of detecting the knee points and the knee regions. KnMAPIO was also compared with five state−of−the−art many−objective algorithms and three algorithms for the last 3 years on standard benchmarks DTLZ and WFG to measure the degree to wihch the solutions obtained by these algorithms dominate the true Pareto Front. The experimental results were analyzed numerically, a process that has been applied in many other fields [51–53].

All the comparative experiments in this paper were run on PlatEMO proposed by Tian et al. [54] in MATLAB 2016b, using the Intel Xeon Gold 5222CPU @ 3.80 GHz and 64 GB RAM. The operating system used in this paper is 64−bit Microsoft 10 and based on the × 64 processor.

Table 1 Parameter setting for the PMOPs test functions

Test function	Number of objectives (M)	Number of decision variables (N)	Parameters in $k(x)(A, B, s, p, l)$	Number of convex knees
PMOP 1	3,5,8,10	$m+9$	(4, 1, -1, 1)	$[A/2]^{m-1}$
PMOP 2,3,8,11,12	3,5,8,10	$m+9$	(4, 1, 2, 1)	$[A/2]^{m-1}$
PMOP 5	3,5,8,10	$m+9$	(1, 1, 2, 1, 12)	$[2*A]^{m-1}$
PMOP 6,9	3,5,8,10	$m+9$	(2,1,2,1)	$[A-1]^{m-1}$
PMOP 13	3,5,8,10	$m+9$	(2,1,-2,1)	∞
PMOP 14	3,5,8,10	$m+9$	(2,1,-1,1)	∞

Verification of the performance of KnMAPIO to identify knee points

Benchmark problems and experimental settings

PMOPs are a classic test suite with complex knee regions for assessing the accurate identification capability of the knee point [55]. The PMOPs PMOP1−PMOP3, PMOP5−PMOP6, PMOP8−PMOP9, and PMOP11−PMOP14 were selected as the test examples to measure the performance of the algorithm. All the parameters in these test problems are listed in Table 1. M denotes the number of objectives, which is set as 3, 5, 8, and 10 in this experiment, and the corresponding population number is 105, 126, 156, and 275, respectively. N represents the number of decision variables. $k(x)$ is the function of knee regions, and (A, B, s, p, l) is a set of parameters controlling the shape and the number of knee regions. The algorithms were run 30 times independently for each test function.

In this experiment, five algorithms including KnEA [29], LA-MOEA [47], LBD-MOEA [48], MAPIO [27], and the proposed KnMAPIO were compared to verify the performance in identifying knee points. All parameters were set as recommended in the original papers. Among them, the rate of knee points was 0.5 in KnEA [29]. In LA-MOEA, the parameter about the localized α-dominance was set to $\alpha = 0.75$ [34]. In LBD-MOEA, the (H_1, H_2) was set to (1, 5), (1, 3), (1, 2), and (1, 3) for the generation of the reference vector with 3, 5, 8, and 10 objectives, and for other parameters, refer to [48]. For KnMAPIO, the setting of parameter τ in knee-oriented dominance was similar to that of LBD-MOEA. For a fair comparison, we set the same parameters as MAPIO [23], including the transition factor tr as 1, and the map and compass factor R as 0.3.

Performance measures

To measure the performance of the proposed algorithm KnMAPIO and other algorithms on benchmark PMOPs, two indicators were used: knee-driven generational distance (KGD) [55] and knee- driven inverted generational distance (KIGD) [55].

(1) KGD

KGD indicates the proximity of the solutions to the reference points in the Pareto Front knee region, which can evaluate the convergence performance of the proposed algorithm. The smaller the KGD value, the better is the convergence performance. KGD can be calculated as follows:

$$KGD = \frac{1}{|\mathbb{Q}|}\sum_{i=1}^{|\mathbb{Q}|} d(v_i, \mathbb{R}), \tag{8}$$

where $\mathbb{Q}$ is the approximation solution set obtained by the algorithm, $\mathbb{R}$ represents a reference points set that is distributed in the knee region uniformly, and $d(v_i, \mathbb{R})$ denotes the Euclidean distance between point v_i belongs to approximation solution set $\mathbb{Q}$ and the closest reference point in set $\mathbb{R}$.

(2) KIGD

KIGD indicates the extent to which the solution obtained by the algorithm covers the knee region. The smaller the KIGD value, the better is the diversity performance. This means that the solutions obtained by the algorithm can cover the knee region extensively. KIGD can be calculated by the formula (9):

$$KIGD = \frac{1}{|\mathbb{R}|}\sum_{i=1}^{|\mathbb{R}|} d(v_i, \mathbb{Q}), \tag{9}$$

where $\mathbb{Q}$ is the approximation solution set obtained by the algorithm. $\mathbb{R}$ represents a reference points set that is uniformly distributed in the knee region, and $d(v_i, \mathbb{Q})$ denotes the Euclidean distance between point v_i belongs to approximation solution set $\mathbb{R}$ and the closest reference point in set $\mathbb{Q}$.

Table 2 The KGD value of five algorithms for different objectives in the PMOP test problems

Problem	M	D	KnEA	LA_MOEA	LBD_MOEA	MAPIO	KnMAPIO
PMOP1	3	12	1.8469e−3 (2.44e−4) =	1.4322e−3 (1.35e−4) +	**1.4016e−3 (1.19e−4) +**	1.9981e−3 (3.36e−4) =	1.9213e−3 (2.06e−4)
	5	14	2.2393e−2 (2.79e−3) +	**1.6326e−2 (1.38e−3) +**	1.6990e−2 (1.33e−3) +	4.4279e−2 (7.95e−3) −	3.3608e−2 (3.59e−3)
	8	17	7.7401e−2 (2.00e−2) +	**5.3710e−2 (4.16e−3) +**	5.5873e−2 (4.30e−3) +	1.8405e−1 (4.14e−2) =	1.8334e−1 (1.78e−2)
	10	19	9.6444e−2 (6.09e−2) +	**6.1521e−2 (5.47e−3) +**	6.2354e−2 (6.76e−3) +	1.6543e−1 (8.11e−2) +	2.3085e−1 (3.96e−2)
PMOP2	3	12	1.9565e−3 (2.90e−3) −	3.1685e−3 (9.44e−4) −	2.9804e−3 (8.07e−4) −	**7.0534e−4 (4.91e−5) +**	7.6479e−4 (3.33e−5)
	5	14	1.8891e−2 (9.17e−3) −	5.7350e−3 (5.81e−4) −	6.1427e−3 (7.61e−4) −	1.1061e−1 (4.73e−1) −	**5.2127e−3 (1.35e−4)**
	8	17	2.8814e+0 (1.26e+0) −	1.2864e−2 (2.02e−3) −	1.2421e−2 (2.69e−3) =	**1.1515e−2 (1.62e−3) =**	1.1769e−2 (9.64e−4)
	10	19	3.2156e+0 (2.12e+0) −	2.0847e−2 (9.18e−3) −	1.8909e−2 (5.68e−3) −	1.3402e−2 (1.77e−3) =	**1.2415e−2 (1.81e−3)**
PMOP3	3	12	9.3679e−2 (1.39e−1) −	1.4563e−2 (1.49e−2) −	1.3153e−2 (1.07e−2) −	2.8556e−3 (5.07e−3) −	**2.0614e−3 (2.36e−3)**
	5	14	3.0432e+0 (1.07e+0) −	1.5511e−2 (1.05e−2) −	1.6246e−2 (8.44e−3) −	6.3091e−2 (2.58e−1) −	**6.3105e−3 (1.41e−3)**
	8	17	7.1034e+0 (1.90e+0) −	1.3408e−2 (7.58e−3) =	1.2494e−2 (4.78e−3) =	1.3903e+0 (2.96e+0) −	**1.1617e−2 (3.31e−3)**
	10	19	5.0461e+0 (2.71e+0) −	1.0155e−1 (4.52e−2) −	1.0363e−1 (4.98e−2) −	1.3361e+0 (2.10e+0) −	**7.0070e−2 (1.30e−1)**
PMOP5	3	12	3.3007e+0 (8.58e+0) =	**7.2112e−1 (9.60e−1) =**	1.0358e+0 (8.92e−1) =	3.8080e+3 (1.41e+4) =	1.1629e+0 (1.90e+0)
	5	14	1.3326e+1 (2.28e+1) −	9.3691e−1 (5.21e−1) =	**9.2124e−1 (9.94e−1) =**	1.7550e+4 (3.47e+4) =	1.2684e+0 (2.09e+0)
	8	17	1.4201e+3 (1.43e+3) −	2.1125e+0 (1.93e+0) −	1.5023e+0 (1.23e+0) =	1.6264e+4 (5.02e+4) −	**9.1905e−1 (4.46e−2)**
	10	19	3.2018e+4 (9.13e+4) −	2.2011e+1 (1.15e+1) −	2.4439e+1 (1.10e+1) −	5.8693e+4 (6.72e+4) −	**1.7880e+0 (3.44e+0)**
PMOP6	3	12	1.3213e−1 (1.82e−1) −	2.3207e−2 (4.64e−2) =	4.3035e−3 (1.21e−2) =	**1.3016e−3 (2.73e−4) =**	1.5043e−3 (2.97e−4)
	5	14	3.5210e+0 (2.35e+0) −	1.1492e−1 (1.38e−1) −	5.9528e−2 (7.52e−2) −	5.5949e−1 (2.46e+0) −	**1.0083e−2 (8.70e−4)**
	8	17	5.5865e+2 (2.08e+2) −	1.0850e+0 (1.31e+0) −	5.9012e−1 (4.69e−1) =	3.7760e+1 (5.93e+1) =	**3.6419e−1 (8.06e−2)**
	10	19	8.2870e+3 (8.01e+3) −	**2.3691e+2 (3.48e+2) +**	2.7826e+2 (3.94e+2) +	1.8033e+3 (1.95e+3) =	1.6698e+3 (1.23e+3)
PMOP8	3	12	2.7807e−3 (3.44e−3) −	1.1518e−3 (1.57e−4) −	1.1338e−3 (2.18e−4) −	**5.6264e−4 (4.57e−5) =**	5.8972e−4 (2.91e−5)
	5	14	2.6941e−2 (7.77e−3) −	3.1981e−3 (3.56e−4) +	3.2131e−3 (3.49e−4) +	**3.0957e−3 (4.65e−4) +**	3.6720e−3 (1.36e−4)
	8	17	1.9409e−1 (4.40e−2) −	3.2794e−3 (7.56e−4) +	**3.1164e−3 (4.23e−4) +**	6.3988e−3 (1.28e−3) +	8.3614e−3 (3.57e−4)
	10	19	1.4787e−1 (2.45e−2) −	2.1042e−2 (8.72e−3) −	2.0419e−2 (7.14e−3) −	1.2530e−2 (6.13e−3) =	**9.6205e−3 (1.59e−3)**
PMOP9	3	12	1.9194e−2 (8.43e−3) −	1.3688e−2 (8.53e−3) −	1.1580e−2 (6.50e−3) −	4.5841e−2 (6.16e−2) −	**9.9630e−4 (7.80e−4)**
	5	14	2.8807e−1 (7.58e−2) −	1.5171e−2 (8.83e−3) −	1.6958e−2 (7.94e−3) −	2.4622e−2 (2.68e−2) −	**7.3974e−3 (5.26e−3)**

Table 2 (continued)

Problem	M	D	KnEA	LA_MOEA	LBD_MOEA	MAPIO	KnMAPIO
	8	17	9.5797e−1 (2.50e−1) −	**2.6375e−2 (9.77e−3) +**	2.8023e−2 (1.66e−2) +	8.7855e−2 (1.35e−1) −	6.1833e−2 (3.97e−2)
	10	19	1.5176e+0 (2.89e−1) −	8.3021e−2 (3.72e−2) +	**7.6761e−2 (2.57e−2) +**	2.8071e−1 (2.81e−1) +	5.3258e−1 (3.03e−1)
PMOP11	3	12	4.0225e−3 (4.92e−3) −	6.6403e−3 (6.89e−3) −	1.2664e−2 (1.33e−2) −	**7.3591e−4 (2.46e−4) =**	8.1330e−4 (5.06e−5)
	5	14	6.8024e−2 (3.37e−2) −	1.3109e−1 (1.03e−1) −	2.8282e−1 (3.11e−1) −	**1.1223e−2 (2.35e−3) +**	1.3067e−2 (4.65e−4)
	8	17	9.0247e+0 (3.39e+0) −	3.5381e−1 (1.68e−1) −	3.5049e−1 (1.50e−1) −	4.5320e−2 (6.83e−3) =	**4.4666e−2 (1.31e−3)**
	10	19	8.3257e+0 (4.67e+0) −	8.7800e−1 (4.76e−1) −	7.0604e−1 (3.59e−1) −	7.5371e−2 (1.76e−2) −	**5.0304e−2 (3.50e−3)**
PMOP12	3	12	1.7102e−2 (2.89e−2) −	1.9103e−4 (1.03e−4) =	7.7803e−4 (1.94e−3) =	**1.3622e−4 (4.20e−5) =**	1.7267e−4 (6.93e−5)
	5	14	4.8386e−1 (1.13e−1) −	3.2192e−3 (6.61e−3) =	2.5359e−3 (4.66e−3) −	1.1588e−1 (3.80e−1) =	**3.5954e−4 (1.35e−4)**
	8	17	4.7259e−1 (1.93e−1) −	**2.8734e−4 (1.37e−4) +**	3.6082e−4 (1.85e−4) +	3.6956e−1 (5.34e−1) =	1.3624e−3 (3.13e−3)
	10	19	1.7012e−1 (1.93e−1) −	**1.3372e−3 (4.90e−4) +**	1.3964e−3 (4.79e−4) +	1.2453e−1 (2.73e−1) =	4.1743e−2 (4.13e−2)
PMOP13	3	12	4.2588e−2 (2.72e−2) −	2.3506e−2 (1.23e−2) −	2.7068e−2 (1.83e−2) −	4.9823e−2 (1.06e−1) −	**1.7025e−3 (3.19e−4)**
	5	14	1.3874e+0 (4.57e−1) −	3.0477e−1 (2.06e−1) −	2.9364e−1 (2.21e−1) −	4.3872e−1 (5.27e−1) −	**9.7955e−2 (9.70e−3)**
	8	17	4.4765e+1 (3.14e+1) −	**1.9912e+0 (1.93e+0) +**	2.1635e+0 (1.45e+0) +	2.5893e+1 (1.89e+1) −	1.0305e+1 (1.03e+1)
	10	19	1.0412e+2 (1.61e+2) +	1.0088e+1 (4.96e+0) +	**8.9263e+0 (5.66e+0) +**	8.7474e+1 (6.42e+1) +	2.3184e+2 (1.38e+2)
PMOP14	3	12	2.6551e−2 (6.55e−2) −	3.4955e−2 (5.01e−2) −	1.7731e−2 (2.12e−2) −	1.0437e−3 (3.77e−4) =	**1.0194e−3 (2.84e−4)**
	5	14	2.2893e+0 (1.51e+0) −	1.8049e−1 (1.46e−1) −	1.6238e−1 (1.92e−1) −	1.4398e+0 (6.31e+0) −	**2.4340e−2 (3.11e−3)**
	8	17	4.2772e+1 (2.48e+1) −	2.1939e−1 (1.11e−1) +	**1.9790e−1 (1.64e−1) +**	6.8881e+1 (5.90e+1) −	4.1384e−1 (7.81e−2)
	10	19	4.6106e+1 (5.45e+1) =	9.5097e−1 (6.28e−1) +	**8.8581e−1 (4.01e−1) +**	1.2108e+2 (1.28e+2) =	2.4970e+1 (2.31e+1)
± / =			4/37/3	15/23/6	15/21/8	7/18/19	

Table 3 The KIGD value of five algorithms for different objectives in the PMOP test problems

Problem	M	D	KnEA	LA_MOEA	LBD_MOEA	MAPIO	KnMAPIO
PMOP1	3	12	**2.6853e−1 (1.18e−1) =**	5.9505e−1 (1.26e−1) −	5.4260e−1 (2.34e−2) −	6.0434e−1 (1.83e−1) −	2.7254e−1 (6.58e−2)
	5	14	1.0157e+0 (3.23e−1) =	1.1116e+0 (7.93e−2) −	1.2159e+0 (1.90e−1) −	1.8858e+0 (5.11e−1) −	**1.0083e+0 (2.01e−1)**
	8	17	2.6883e+0 (5.57e−1) =	**2.0252e+0 (2.47e−1) +**	2.0805e+0 (2.47e−1) +	3.8210e+0 (1.05e+0) −	2.5159e+0 (5.98e−1)
	10	19	3.3227e+0 (8.43e−1) =	**2.4065e+0 (5.82e−1) +**	2.4333e+0 (6.86e−1) +	3.8954e+0 (1.49e+0) −	3.1598e+0 (5.70e−1)
PMOP2	3	12	8.2361e−2 (3.27e−2) −	1.9643e−1 (3.91e−2) −	1.8918e−1 (2.89e−2) −	1.2218e−1 (8.63e−2) −	**5.1144e−2 (2.77e−3)**
	5	14	2.1464e−1 (3.66e−2) −	2.5161e−1 (1.95e−2) −	2.6696e−1 (9.84e−3) −	2.7517e−1 (6.38e−2) −	**1.2601e−1 (3.83e−3)**
	8	17	1.0408e+0 (5.95e−1) −	2.9526e−1 (1.57e−2) −	2.8723e−1 (2.72e−2) −	2.5802e−1 (4.53e−2) −	**1.7112e−1 (8.86e−3)**
	10	19	4.2602e+0 (1.36e+1) −	2.5368e−1 (6.42e−2) −	2.3400e−1 (3.19e−2) −	2.4022e−1 (3.94e−2) −	**1.6852e−1 (1.82e−2)**
PMOP3	3	12	**2.5274e−1 (1.05e−1) +**	5.8548e−1 (1.85e−1) −	5.8268e−1 (1.48e−1) −	8.7264e−1 (4.08e−1) −	3.6911e−1 (9.86e−2)
	5	14	3.7703e+0 (1.77e+0) −	**4.1051e−1 (7.52e−2) =**	4.1513e−1 (5.60e−2) =	7.1453e−1 (3.25e−1) −	4.5518e−1 (1.38e−1)
	8	17	5.2670e+0 (2.46e+0) −	8.0183e−1 (1.05e−1) −	7.6525e−1 (1.22e−1) −	9.3674e−1 (2.51e−1) −	**6.3808e−1 (1.92e−1)**
	10	19	2.2139e+0 (6.92e−1) −	7.3272e−1 (1.70e−1) −	7.5130e−1 (1.58e−1) −	9.2550e−1 (3.80e−1) −	**4.6871e−1 (1.52e−1)**
PMOP5	3	12	**2.8170e+0 (2.13e+0) +**	3.3324e+0 (4.19e+0) +	4.7727e+0 (3.75e+0) +	4.2699e+0 (2.29e+0) =	9.5969e+0 (1.49e+1)
	5	14	1.4829e+1 (1.11e+1) −	**4.3223e+0 (2.12e+0) +**	4.6777e+0 (4.05e+0) +	7.0438e+0 (3.45e+0) =	1.0108e+1 (1.56e+1)
	8	17	3.3853e+2 (3.25e+2) −	1.5657e+1 (1.85e+1) −	8.5184e+0 (7.50e+0) =	3.9376e+1 (4.42e+1) −	**7.4689e+0 (4.63e−1)**
	10	19	6.8345e+2 (1.12e+3) −	8.8091e+1 (4.85e+1) −	1.0208e+2 (5.66e+1) −	9.3403e+1 (1.19e+2) −	**1.0832e+1 (1.14e+0)**
PMOP6	3	12	**1.7601e−1 (1.60e−1) +**	5.4568e−1 (1.47e−1) −	4.6391e−1 (1.20e−1) −	6.2970e−1 (1.49e−1) −	2.7046e−1 (5.56e−2)
	5	14	4.3831e+0 (3.47e+0) −	7.6308e−1 (2.75e−1) +	**7.3838e−1 (9.21e−2) +**	1.1036e+0 (4.67e−1) −	8.8021e−1 (1.21e−1)
	8	17	7.5155e+1 (3.37e+1) −	**1.7853e+1 (1.73e+0) +**	1.8490e+1 (1.59e+0) +	2.1558e+1 (1.77e+0) =	2.1015e+1 (2.21e+0)
	10	19	2.5178e+3 (6.14e+2) −	4.4665e+3 (8.82e+1) −	4.4723e+3 (9.92e+1) −	4.6534e+3 (1.26e+2) −	**1.9794e+3 (3.05e+2)**
PMOP8	3	12	7.0425e−2 (1.13e−2) −	1.4075e−1 (1.58e−2) −	1.4191e−1 (1.35e−2) −	8.2930e−2 (4.10e−2) −	**4.8745e−2 (3.21e−3)**
	5	14	1.3441e−1 (4.09e−2) −	1.0892e−1 (1.08e−2) −	1.1311e−1 (1.21e−2) −	1.2163e−1 (2.07e−2) −	**8.2575e−2 (4.88e−3)**
	8	17	8.8452e−1 (2.18e−1) −	9.0989e−2 (7.80e−3) +	**8.9956e−2 (8.29e−3) +**	1.0620e−1 (9.73e−3) =	1.0402e−1 (6.56e−3)
	10	19	7.6039e−1 (1.50e−1) −	2.4150e−1 (9.75e−2) −	2.4283e−1 (7.85e−2) −	1.4406e−1 (4.79e−2) −	**9.2146e−2 (7.88e−3)**
PMOP9	3	12	**1.3906e−1 (4.50e−2) +**	2.0702e−1 (1.61e−2) −	2.2717e−1 (3.69e−2) −	2.2026e−1 (8.45e−2) −	1.6904e−1 (3.37e−2)
	5	14	5.8132e−1 (1.71e−1) −	3.1323e−1 (3.14e−2) +	**3.0794e−1 (3.15e−2) +**	3.6916e−1 (1.07e−1) =	3.8989e−1 (7.01e−2)

Table 3 (continued)

Problem	M	D	KnEA	LA_MOEA	LBD_MOEA	MAPIO	KnMAPIO
	8	17	1.9717e+0 (5.31e−1) −	1.2528e+0 (1.56e−1) −	1.1900e+0 (1.28e−1) −	1.2311e+0 (4.38e−1) =	**1.0381e+0 (1.91e−1)**
	10	19	2.7666e+0 (7.77e−1) =	**2.1324e+0 (8.83e−1) +**	2.2090e+0 (8.58e−1) +	2.5009e+0 (8.30e−1) =	3.0383e+0 (8.21e−1)
PMOP11	3	12	1.3908e−1 (8.77e−2) −	4.3339e−1 (2.27e−1) −	4.3479e−1 (1.63e−1) −	2.8026e−1 (1.97e−1) −	**7.7811e−2 (5.72e−3)**
	5	14	5.2547e−1 (1.04e−1) −	1.1296e+0 (3.85e−1) −	1.4137e+0 (5.62e−1) −	5.1416e−1 (1.38e−1) −	**3.1333e−1 (8.19e−3)**
	8	17	3.9618e+0 (9.52e−1) −	2.5680e+0 (6.86e−1) −	2.2941e+0 (6.27e−1) −	7.8853e−1 (1.37e−1) −	**5.4073e−1 (1.93e−2)**
	10	19	4.7537e+0 (2.20e+0) −	3.0703e+0 (1.01e+0) −	3.0707e+0 (1.07e+0) −	1.0975e+0 (2.18e−1) −	**5.7539e−1 (4.82e−2)**
PMOP12	3	12	**2.4202e−2 (3.05e−2) +**	7.5334e−2 (1.25e−2) −	7.1418e−2 (9.85e−3) −	6.6945e−2 (2.24e−2) −	3.9892e−2 (6.91e−3)
	5	14	1.9901e−1 (6.55e−2) −	2.2920e−2 (6.16e−3) =	2.3061e−2 (4.56e−3) −	4.4371e−2 (2.65e−2) −	**1.9186e−2 (4.65e−3)**
	8	17	1.1159e−1 (3.93e−2) −	1.4485e−2 (2.07e−3) =	**1.3576e−2 (1.71e−3) =**	1.8497e−2 (2.18e−2) =	1.4013e−2 (3.24e−3)
	10	19	3.7249e−2 (1.58e−2) −	1.0771e−2 (2.31e−3) =	**1.0078e−2 (2.02e−3) =**	2.5190e−2 (9.69e−3) −	1.4129e−2 (9.13e−3)
PMOP13	3	12	**3.0682e−1 (1.41e−1) +**	5.7992e−1 (6.52e−2) −	6.0788e−1 (8.69e−2) −	5.6996e−1 (1.19e−1) −	4.1210e−1 (1.29e−1)
	5	14	3.0814e+0 (9.15e−1) −	2.1149e+0 (2.81e−1) =	**2.0965e+0 (3.49e−1) =**	2.7325e+0 (9.71e−1) =	2.1606e+0 (3.61e−1)
	8	17	2.8672e+1 (9.63e+0) =	2.3196e+1 (1.97e+0) =	**2.2053e+1 (2.06e+0) =**	2.4960e+1 (6.15e+0) =	2.4559e+1 (7.42e+0)
	10	19	8.9153e+1 (3.43e+1) +	7.4262e+1 (1.87e+1) +	**7.3543e+1 (1.53e+1) +**	9.6421e+1 (1.48e+1) +	1.1515e+2 (3.09e+1)
PMOP14	3	12	**1.7942e−1 (3.12e−1) +**	4.1954e−1 (9.96e−2) −	4.0260e−1 (6.51e−2) −	3.6961e−1 (9.20e−2) −	2.6477e−1 (7.99e−2)
	5	14	2.5583e+0 (1.75e+0) −	5.2732e−1 (1.64e−1) +	**4.9098e−1 (1.28e−1) +**	7.5403e−1 (5.47e−1) =	5.5255e−1 (6.78e−2)
	8	17	1.2453e+1 (6.48e+0) −	1.5843e+0 (3.46e−1) =	**1.3726e+0 (2.94e−1) +**	1.8290e+0 (7.60e−1) =	1.4863e+0 (1.95e−1)
	10	19	2.1500e+1 (1.49e+1) −	5.7148e+0 (1.23e+0) −	5.9720e+0 (1.45e+0) −	4.6215e+0 (3.41e+0) −	**2.2220e+0 (7.37e−1)**
± / =			8/30/6	11/26/7	12/26/6	1/31/12	

Experimental results and analysis

In this section, Tables 2 and 3 show the comparison results on KnEA [29], LA-MOEA [47], LBD-MOEA [48], MAPIO [27], and the proposed KnMAPIO. Table 2 presents the KGD values of five algorithms on the PMOP test suite, allowing for an evaluation of the convergence performance of the algorithms. Table 3 presents the KIGD values of five algorithms, allowing for an evaluation of the diversity performance of algorithms. In these tables, the highlight values represent the best results, and ‘ + ,’ ‘−’, and ‘ = ’ denote the results from other algorithms that are respectively higher than, lower than, or equal to the results from the proposed KnMAPIO.

From Table 2, the number of best results—0, 9, 7, 8, 20—is produced by KnEA, LA_MOEA, LBD_MOEA, MAPIO, and KnMAPIO, respectively. As observed from the last row of Table 2, compared with KnEA, KnMAPIO has a significant advantage on the 37 test functions. The performance of LA_MOEA is similar to that of LBD_MOEA, in which 15 items are better than the proposed KnMAPIO. However, LA_MOEA has 21 items that are worse than KnMAPIO, and LBD_MOEA has 23 items that are worse than the proposed algorithm. In the comparison results with MAPIO, seven items of MAPIO are better than KnMAPIO, while 18 items are worse than the proposed algorithm. These comparisons show that our algorithm has certain advantage in KGD.

From the different PMOP problems presented in Table 2, it can be seen that the proposed algorithm has good performance on PMOP 2, 3, 5, 6, 9, 11, 13, 14 test functions, indicating the superior performance of the proposed algorithm in solving these problems, such as the concave basic shape, multimodal, and non-separable problems. For PMOP3, our algorithm achieves the good preference in all four objectives. For PMOP1, although the proposed algorithm is not optimal in each objective, it also preforms better than some algorithms in some objectives. Further research will be carried out in solving linear problems, fundamental problems and shape problems in the future. In the comparison results with MAPIO, the proposed algorithm has a slight advantage. This is because the proposed algorithm not only uses BFE strategy but also uses knee point-driven environment selection to increase the probability of knee region identification.

As shown in Table 3, KnMAPIO obtained 20 of the best results, LBD_MOEA obtained 10 of the KnEA obtained eight, LA_MOEA obtained six, and MAPIO failed to obtain any. As observed from the last row of Table 3, compared with KnEA, KnMAPIO has a significant advantage on 30 test functions. The performance of LA_MOEA is similar to that of LBD_MOEA, in which 26 items are worse than the proposed KnMAPIO, while the performance of MAPIO has 31 items worse than KnMAPIO. These comparison results show that the proposed algorithm has significant advantages in performance.

From different PMOP problems in Table 3, the proposed algorithm has superior performance on PMOP 2, 3, 5, 8, and 11 test functions. For PMOP2 and PMOP11, our algorithm achieves the good performance in all four objectives, indicating its robust diversity in solving the problems with the characteristics of concave, basic, shape and complex PoF. However, our algorithm does not perform satisfactorily in solving the PMOP13 problem. The comparison results make it obvious that the knee-driven environment selection strategy can significantly improve the identifying effect of the proposed algorithm. Overall, Tables 2 and Table 3 show that the proposed algorithm has excellent overall performance, which is attributed to the improvement of individual selection pressure by the knee point-driven environmental selection.

Overall, Tables 2 and 3 show that the proposed algorithm has excellent overall performance, which is attributed to the improvement of individual selection pressure by the knee point-driven environmental selection.

Table 4 Parameter setting for the DTLZ and WFG test functions

Test function	Number of decision variables (n)	Parameters	Maxgen
DTLZ 1	$M-1+k$	$k=5$	700
DTLZ 2	$M-1+k$	$k=10$	250
DTLZ 3	$M-1+k$	$k=10$	1000
DTLZ 4-DTLZ 6	$M-1+k$	$k=10$	250
DTLZ 7	$M-1+k$	$k=20$	250
WFG 1	$k+l$	$k=M-1$, $l=20$	1000
WFG 2	$k+l$	$k=M-1$, $l=20$	700
WFG3–WFG9	$k+l$	$k=M-1$, $l=20$	250

Approximation degree of KnMAPIO to the true Pareto front

Benchmark problems and experimental settings

DTLZ [56] and WFG [57] are standard benchmarks for assessing the capability of the many-objective optimization algorithm, and these functions have obvious attributes for solving linear, multi-modal, disconnected, mixed convex/concave, and deceptive problems. DTLZ1-DTLZ7 were selected as the test examples for DTLZ problems, and WFG1–WFG9 were chosen for the WFG problems. The settings of parameters for the DTLZ and WFG are shown in Table 4. The population size in the DTLZ and WFG test suites are 120, 132, 156, and 275 with the number of objectives at four, six, eight, and ten. The algorithms were run 30 times independently for each test function.

The proposed algorithm was tested on the standard benchmarks DTLZ and WFG with five other state-of-the-art algorithms: NSGA-III [4], GrEA [58], MOEA/D [7], RVEA [59], and VaEA [60]. The proposed algorithm was also compared with three algorithms for the last 3 years on the standard benchmark DTLZ and WFG test functions to verify the performance of our proposed algorithm, including CSEA [61], hpaEA [62], and MaPSO-MC [63]. CSEA [61] was proposed in 2019 by Pan et al., who used an artificial neural network to predict the dominance relationship between candidate solutions and reference solutions instead of approximating the objective values separately. In 2020, Tian et al. proposed hpaEA [62], which was first algorithm to differentiate the non-dominated solutions by exhibiting tendencies toward the Pareto-optimal front as prominent solutions and using the hyperplane formed by their neighboring solutions. MaPSO-MC [63] was proposed by Hu et al. in 2021 to solve the hybrid recommendation model by using a generation-based fitness evaluation strategy coupled with diversity enhancement (GBFE–DE) and ISDE+to evaluate individual performance.

All parameters were set as recommended in the original papers. Simulated binary crossover [4] and polynomial mutation [4] were used in some algorithms [59–61] to produce offspring, the distribution indices of crossover and mutation were set to $n_c = 20$, and $n_m = 20$. The probabilities of crossover and mutation were set to $p_c = 1.0$ and $p_m = 1/D$, where D represents the number of decision variables. For GrEA, the setting of the parameter div was taken from [58], which denotes the number of divisions in each dimension. The parameters about the range of neighborhood in MOEA/D were set to $N/10$ for all test problems. Other relevant parameters were set by referring to [7]. In [59], the change rate of the penalty function α was set to 2, and the frequency of reference vector adaptation was set to $f_r = 0.1$. For CSEA, the setting of maximum epochs for training the FNN T was 500, the hidden neurons H was 10, and the number of parameters for reference solutions was 6. The number of prominent solutions K in hpaEA [62] was set to 6, and other parameters were set to the same as NSGA-III. In [63], the setting of learning parameters c_1, c_2, c_3 belonged to $[1.5, 2.5]$, and the number of parameters of generation influence θ in MaPSO-MC was 2.

Performance measures

To measure the approximation degree of the algorithm to the true Pareto front, the coverage [7] was used in this experiment. C-metric indicates the dominant relationship between the solution set and the Pareto front, which can measure the coverage performance of a solution set. The C-metric can be calculated as follows:

$$C(Pop, PF) = \frac{|\{u \in PF | \exists v \in Pop : v \text{ dominance} \, u\}|}{|PF|}, \tag{10}$$

where the PF is the Pareto front, and Pop represents the objective values of all the individuals in the population. The numerator is the number of solutions in PF that are dominated by at least one solution in Pop, and the denominator represents the total number of solutions in PF.

Experimental results and analysis

In this section, Tables 5 and 6 show the comparison results on GrEA [58], MOEA/D [7], NSGA-III [4], RVEA [59], VaEA [60], CSEA [61], hpaEA [62], MaPSO-MC [63], and the proposed KnMAPIO, and the performance of the coverage is demonstrated in the DTLZ test functions. Tables 7 and 8 show the comparison results of the algorithms on the WFG test functions, and the performance of the coverage is demonstrated in these test functions. In these tables, the highlighted numbers represent the best results, and ' + ,' '-,' ' = ' denote the results from the other algorithms that are respectively higher than, lower than, or equal to the results from the proposed KnMAPIO.

As can be seen from Table 5, the number of best results—5, 5, 1, 0, 0, and 14—is produced by GrEA, MOEA/D, NSGA-III, RVEA, VaEA, and the proposed KnMAPIO, respectively. In the six, eight, and ten objectives of DTLZ4, our algorithm achieves the same result as MOEA/D. Looking at the last row in Table 5, compared with RVEA, KnMAPIO has a significant advantage on 28 test functions. NSGA-III has only 3 items better than the proposed algorithm, GrEA and VaEA have four items better, and MOEA/D has six items better than the KnMAPIO. As can be seen from Table 6, the number of best results—5, 1, 4, and 18—is produced by CSEA, hpaEA, MaPSO-MC, and the proposed KnMAPIO, respectively. Looking at the last row of Table 6, KnMAPIO has 11 items better than CSEA, while hpaEA has only 1 item better than KnMAPIO, and MaPSO-MC has 3 items better. KnMAPIO has 16 items better than MaPSO-MC and has significant advantages over the other three algorithms on DTLZ 5 and DTLZ 6.

Table 5 shows that the proposed algorithm has good performance on DTLZ 2, 5, and 6 test functions, and it again proves the superior performance of the proposed algorithm in solving concave and multimodal functions. However, the table also shows that KnMAPIO does not perform well on DTLZ7, because of the multiple modal characteristics of the test functions and the difficulty in obtaining the convergence solutions. Table 6 shows that the performance of the proposed algorithm is better than that of the other three algorithms. The

Table 5 The coverage value of six algorithms for different objectives in the DTLZ test problems

Problem	M	D	GrEA	MOEAD	NSGAIII	RVEA	VaEA	KnMAPIO
DTLZ1	4	8	4.5879e−1 (2.47e−1) −	7.3624e−1 (3.82e−1) −	4.2250e−1 (2.24e−1) −	5.8777e−1 (3.10e−1) −	5.9782e−1 (9.49e−2) −	**4.0408e−1 (7.60e−2)**
	6	10	**1.0480e−1 (6.87e−2) +**	4.4028e−1 (4.51e−1) =	2.9848e−1 (1.78e−1) +	6.2092e−1 (3.24e−1) −	2.9040e−1 (7.16e−2) +	4.5758e−1 (9.46e−2)
	8	12	**8.1838e−2 (8.28e−2) +**	2.7124e−1 (4.47e−1) +	2.1132e−1 (1.52e−1) +	6.0749e−1 (3.15e−1) −	1.2778e−1 (3.20e−2) +	2.8715e−1 (7.22e−2)
	10	14	2.2461e−1 (1.20e−1) −	4.6631e−1 (5.06e−1) =	1.8267e−1 (1.03e−1) −	5.4869e−1 (2.85e−1) −	1.7188e−1 (2.77e−2) −	**1.0364e−1 (3.77e−2)**
DTLZ2	4	13	**2.7778e−2 (1.96e−2) =**	1.4866e−1 (8.81e−2) −	1.2389e−1 (7.35e−2) −	1.3906e−1 (8.42e−2) −	3.0722e−1 (8.93e−2) −	4.0556e−2 (2.11e−2)
	6	15	2.3737e−2 (1.80e−2) −	7.6515e−2 (5.05e−2) −	7.9293e−2 (5.58e−2) −	4.4275e−2 (2.91e−2) −	2.0505e−1 (5.50e−2) −	**7.5758e−3 (8.44e−3)**
	8	17	2.7244e−1 (1.61e−1) −	5.4060e−2 (3.86e−2) −	1.5000e−1 (9.74e−2) −	3.9978e−2 (2.94e−2) −	2.0385e−1 (6.50e−2) −	**8.9744e−3 (8.52e−3)**
	10	19	1.2412e−1 (6.98e−2) −	2.6935e−2 (2.17e−2) −	2.1867e−1 (1.59e−1) −	1.6254e−1 (8.95e−2) −	3.4267e−1 (4.24e−2) −	**1.5152e−2 (1.04e−2)**
DTLZ3	4	13	1.9675e−1 (1.05e−1) =	6.9837e−1 (3.45e−1) −	**1.0463e−1 (6.17e−2) +**	4.7460e−1 (2.56e−1) −	1.3375e−1 (6.81e−2) +	2.3634e−1 (1.27e−1)
	6	15	**2.1212e−2 (2.42e−2) +**	6.0294e−1 (3.76e−1) −	1.2980e−1 (7.83e−2) =	4.7319e−1 (2.54e−1) −	1.2828e−1 (4.84e−2) +	1.7407e−1 (8.36e−2)
	8	17	**4.1239e−2 (3.22e−2) +**	4.4444e−1 (4.53e−1) =	9.1026e−2 (7.93e−2) =	4.6503e−1 (2.48e−1) −	6.3034e−2 (3.51e−2) =	7.9915e−2 (4.77e−2)
	10	19	3.4873e−1 (1.85e−1) −	3.5065e−1 (4.16e−1) −	9.9394e−2 (7.29e−2) −	4.6986e−1 (2.04e−1) −	1.2703e−1 (2.98e−2) −	**1.0667e−2 (1.02e−2)**
DTLZ4	4	13	2.0556e−2 (1.97e−2) =	**3.9313e−3 (1.56e−2) +**	7.7500e−2 (5.89e−2) −	1.1776e−1 (8.04e−2) −	2.7000e−1 (9.80e−2) −	2.8333e−2 (1.88e−2)
	6	15	5.5556e−3 (6.87e−3) −	0.0000e+0 (0.00e+0) =	3.7879e−3 (4.77e−3) −	8.9628e−3 (9.70e−3) −	1.0480e−1 (4.90e−2) −	**0.0000e+0 (0.00e+0)**
	8	17	9.8291e−3 (9.33e−3) −	0.0000e+0 (0.00e+0) =	1.4957e−3 (2.76e−3) −	5.8110e−3 (6.64e−3) −	1.6453e−2 (1.38e−2) −	**0.0000e+0 (0.00e+0)**
	10	19	1.2121e−4 (6.64e−4) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	2.4287e−4 (9.24e−4) =	1.9394e−3 (2.82e−3) −	**0.0000e+0 (0.00e+0)**
DTLZ5	4	13	2.2583e−1 (3.47e−2) −	3.4251e−1 (1.13e−1) −	5.5667e−1 (1.20e−1) −	5.8905e−1 (8.79e−2) −	2.6556e−1 (2.88e−2) −	**1.9111e−1 (3.38e−2)**
	6	15	1.3359e−1 (4.08e−2) −	3.8518e−1 (9.84e−2) −	3.6869e−1 (8.50e−2) −	3.1238e−1 (1.00e−1) −	2.7778e−1 (3.54e−2) −	**1.9444e−2 (1.03e−2)**
	8	17	1.6346e−1 (5.65e−2) −	4.4867e−1 (9.84e−2) −	3.2650e−1 (9.61e−2) −	4.9464e−1 (1.52e−1) −	4.1303e−1 (5.79e−2) −	**1.3889e−2 (8.76e−3)**
	10	19	2.0279e−1 (7.03e−2) −	5.1906e−1 (1.03e−1) −	3.7382e−1 (7.12e−2) −	5.6948e−1 (1.39e−1) −	5.1479e−1 (8.46e−2) −	**4.7273e−3 (2.17e−3)**
DTLZ6	4	13	1.6361e−1 (8.29e−2) −	2.2552e−1 (2.69e−1) −	3.8056e−1 (2.08e−1) −	4.6218e−1 (1.91e−1) −	3.8667e−1 (1.67e−1) −	**3.5278e−2 (2.49e−2)**
	6	15	3.1061e−1 (1.40e−1) −	4.6105e−1 (2.73e−1) −	7.2020e−1 (2.42e−1) −	4.0089e−1 (1.45e−1) −	8.9672e−1 (2.11e−2) −	**1.3889e−2 (8.92e−3)**
	8	17	7.7885e−1 (2.31e−1) −	5.1588e−1 (2.58e−1) −	6.2329e−1 (1.55e−1) −	6.0568e−1 (1.26e−1) −	8.7073e−1 (1.50e−2) −	**6.6239e−3 (1.17e−3)**
	10	19	7.8085e−1 (1.28e−1) −	5.4339e−1 (2.16e−1) −	5.6812e−1 (1.27e−1) −	6.8867e−1 (1.44e−1) −	9.1576e−1 (1.62e−2) −	**3.6364e−3 (1.32e−18)**
DTLZ7	4	23	7.3917e−1 (3.14e−1) −	**1.0312e−1 (6.12e−2) +**	9.5750e−1 (9.92e−2) −	9.3947e−1 (1.41e−1) −	1.0000e+0 (0.00e+0) −	1.4083e−1 (5.86e−2)
	6	25	6.5227e−1 (2.72e−1) −	**4.6063e−3 (4.78e−3) +**	9.2626e−1 (1.71e−1) −	8.5340e−1 (3.10e−1) −	1.0000e+0 (0.00e+0) −	1.2500e−1 (7.00e−2)
	8	27	6.3718e−1 (3.02e−1) −	**4.2735e−4 (1.63e−3) +**	8.7842e−1 (2.78e−1) −	8.3217e−1 (2.87e−1) −	1.0000e+0 (0.00e+0) −	1.2158e−1 (4.92e−2)
	10	29	8.3709e−1 (3.71e−1) −	**0.0000e+0 (0.00e+0) +**	8.7455e−1 (2.86e−1) −	8.3392e−1 (2.92e−1) −	1.0000e+0 (0.00e+0) −	3.7188e−1 (1.74e−1)
± / =			4/20/4	6/16/6	3/22/3	0/27/1	4/23/1	

Table 6 The coverage value of four algorithms for the last 3 years on the DTLZ test problems

Problem	M	D	CSEA	hpaEA	MaPSO−MC	KnMAPIO
DTLZ1	4	8	2.0738e−1 (8.19e−2) =	5.6677e−1 (9.87e−2) =	**1.9483e−1 (4.16e−2) +**	3.5385e−1 (1.55e−1)
	6	10	**1.2194e−1 (9.48e−2) +**	2.6597e−1 (5.13e−2) +	2.2651e−1 (8.42e−2) +	4.3182e−1 (1.13e−1)
	8	12	**1.3484e−1 (9.03e−2) +**	3.4003e−1 (4.40e−2) =	6.0640e−1 (2.34e−1) =	2.7949e−1 (7.91e−2)
	10	14	1.2299e−1 (8.45e−2) =	3.2379e−1 (5.84e−2) −	7.3257e−1 (6.57e−2) −	**1.0691e−1 (1.88e−2)**
DTLZ2	4	13	4.3457e−2 (4.55e−3) =	2.3866e−1 (4.99e−2) −	**2.1864e−2 (1.14e−2) =**	3.3333e−2 (8.33e−3)
	6	15	1.4294e−2 (2.44e−3) =	1.4643e−1 (2.05e−2) −	**0.0000e+0 (0.00e+0) =**	9.0909e−3 (8.30e−3)
	8	17	6.9210e−3 (2.48e−3) =	3.3982e−1 (4.71e−2) −	2.5641e−2 (1.76e−2) =	**6.4103e−3 (6.41e−3)**
	10	19	**6.2743e−3 (3.62e−3) =**	4.9282e−1 (1.03e−1) −	6.6406e−2 (5.61e−3) −	1.7455e−2 (1.13e−2)
DTLZ3	4	13	1.6000e−1 (1.02e−1) =	**1.1421e−1 (1.21e−1) =**	1.1682e−1 (6.00e−2) +	2.3337e−1 (5.68e−2)
	6	15	7.7102e−2 (2.25e−2) =	6.5152e−2 (6.78e−3) =	**2.0237e−2 (1.50e−2) =**	1.3333e−1 (7.64e−2)
	8	17	**3.0030e−2 (1.76e−2) =**	1.6785e−1 (2.52e−2) =	3.7703e−1 (1.09e−1) −	8.0769e−2 (6.50e−2)
	10	19	3.6201e−2 (2.04e−2) =	2.7883e−1 (7.72e−2) −	4.6611e−1 (9.24e−2) −	**2.0364e−2 (1.38e−2)**
DTLZ4	4	13	4.0185e−2 (6.46e−3) =	7.6634e−2 (7.43e−2) =	1.8333e−2 (2.24e−2) =	**1.5000e−2 (2.07e−2)**
	6	15	4.5612e−3 (3.49e−3) −	1.8417e−2 (1.95e−2) −	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
	8	17	8.9403e−4 (8.68e−4) −	7.8836e−3 (7.69e−3) =	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
	10	19	1.2099e−4 (1.49e−4) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
DTLZ5	4	13	**1.5942e−1 (3.69e−2) =**	3.0333e−1 (4.51e−2) −	4.1731e−1 (3.99e−2) −	1.8667e−1 (3.75e−2)
	6	15	1.4225e−1 (3.11e−2) −	3.5606e−1 (4.25e−2) −	4.2605e−1 (6.48e−2) −	**2.1212e−2 (6.34e−3)**
	8	17	1.3577e−1 (3.13e−2) −	3.1923e−1 (2.85e−2) −	2.6525e−1 (1.11e−1) −	**8.9744e−3 (5.73e−3)**
	10	19	1.3757e−1 (2.50e−2) −	3.4473e−1 (3.81e−2) −	3.4531e−1 (1.16e−1) −	**4.3636e−3 (1.63e−3)**
DTLZ6	4	13	2.1035e−1 (2.81e−2) −	4.3833e−1 (1.85e−1) −	3.2251e−1 (8.11e−2) −	**4.3333e−2 (4.10e−2)**
	6	15	2.8056e−1 (3.47e−2) −	7.5748e−1 (2.16e−2) −	5.5960e−1 (1.25e−1) −	**1.0606e−2 (4.15e−3)**
	8	17	1.5116e−1 (1.84e−2) −	7.6537e−1 (4.40e−2) −	5.6244e−1 (1.60e−1) −	**6.4103e−3 (0.00e+0)**
	10	19	1.3233e−1 (2.91e−2) −	8.3200e−1 (2.92e−2) −	7.0163e−1 (7.34e−2) −	**3.6364e−3 (0.00e+0)**
DTLZ7	4	23	4.1363e−1 (6.31e−2) −	9.8833e−1 (1.83e−2) −	9.6988e−1 (4.70e−2) −	**1.2000e−1 (3.89e−2)**
	6	25	2.6336e−1 (6.63e−2) =	9.9394e−1 (9.88e−3) −	9.7871e−1 (4.26e−2) −	**2.3485e−1 (1.16e−1)**
	8	27	2.4612e−1 (5.60e−2) −	1.0000e+0 (0.00e+0) −	9.7183e−1 (5.55e−2) −	**1.0256e−1 (2.94e−2)**
	10	29	4.4943e−1 (1.51e−1) =	1.0000e+0 (0.00e+0) −	1.0000e+0 (0.00e+0) −	**4.4873e−1 (8.48e−2)**
± / =			2/11/15	1/19/8	3/16/9	

coverage values of some functions are 0.00e+0 in this table, indicating that no solutions in PF are dominated by the solutions in the final population. The coverage values of VaEA on DTLZ 7 are all 1.0000e+0, which indicates that all solutions in PF are dominated by some solutions in the final population.

Table 7 reveals the comparison results of KnMAPIO with five advanced algorithms on WFG test functions, with four to ten objectives using the indicator of coverage. The number of best results—9, 4, 0, 0, 0, and 19—is produced by GrEA, MOEA/D, NSGA-III, RVEA, VaEA, and the proposed KnMAPIO, respectively.

Our algorithm achieves the same result as other algorithms in all objectives of WFG1, and the coverage values are all 1.0000e+0, which indicates that represents all solutions in PF are dominated by some solutions in the population. It also shows that these algorithms have the same performance in solving the bias functions. Compared with NSGA-III, RVEA and VaEA, KnMAPIO has an obvious advantage over the 36 test functions. Among the comparison results, GrEA has only seven items better than KnMAPIO, but 20 items worse than KnMAPIO. MOEA/D has 5 items better than KnMAPIO, but almost two thirds of the items are worse than the proposed algorithm. Table 7 shows that the proposed algorithm has good performance on WFG 4, 5, 6, and 9 test functions, especially on WFG 5. The good performance of the proposed algorithm on WFG 5 indicates the best coverage on the concave–basic–shape functions.

Table 8 reveals the comparison results of KnMAPIO with three advanced algorithms on WFG test functions with –four to ten objectives using the indicator of coverage. The number of best results—4, 1, 14, and 17—was respectively produced by CSEA, hpaEA, MaPSO-MC, and the proposed

Table 7 The coverage value of six algorithms for different objectives in the WFG test problems

Problem	M	D	GrEA	MOEAD	RVEA	NSGAIII	VaEA	KnMAPIO
WFG1	4	13	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	6	15	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	8	17	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	10	19	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
WFG2	4	13	**6.6375e−1 (1.48e−1) +**	9.5170e−1 (8.27e−2) =	8.6233e−1 (8.80e−2) +	9.0833e−1 (6.43e−2) +	9.2458e−1 (4.67e−2) +	9.6625e−1 (1.80e−2)
	6	15	2.0833e−1 (1.08e−1) -	3.9986e−1 (3.53e−1) -	3.6283e−1 (1.37e−1) -	3.4508e−1 (1.34e−1) -	7.0492e−1 (1.12e−1) -	**1.6288e−1 (1.45e−1)**
	8	17	**1.4103e−2 (1.56e−2) =**	7.0152e−2 (1.59e−1) =	8.3573e−2 (1.06e−1) -	7.2756e−2 (5.60e−2) -	4.2051e−1 (1.05e−1) -	1.6346e−2 (2.52e−2)
	10	19	2.2982e−1 (8.50e−2) +	**7.9488e−2 (2.22e−1) +**	2.8001e−1 (1.21e−1) +	2.9600e−1 (9.08e−2) +	5.4618e−1 (1.34e−1) -	3.6727e−1 (1.13e−1)
WFG3	4	13	2.0833e−1 (4.42e−2) -	9.7711e−2 (3.76e−2) =	2.9012e−1 (4.78e−2) -	2.1667e−1 (3.05e−2) -	1.7708e−1 (3.05e−2) -	**8.0833e−2 (2.21e−2)**
	6	15	6.4773e−2 (1.52e−2) -	**5.6818e−3 (5.43e−3) +**	5.9463e−2 (2.44e−2) -	5.7576e−2 (6.22e−2) -	8.5227e−2 (3.18e−2) -	2.0076e−2 (7.88e−3)
	8	17	6.2500e−2 (1.50e−2) -	**0.0000e+0 (0.00e+0) +**	2.7750e−2 (1.95e−2) -	7.7244e−2 (3.38e−2) -	5.9615e−2 (2.37e−2) -	1.2500e−2 (5.29e−3)
	10	19	1.9636e−2 (4.78e−3) -	**0.0000e+0 (0.00e+0) +**	1.8235e−2 (9.73e−3) -	6.5636e−2 (2.36e−2) -	3.7636e−2 (1.15e−2) -	8.0000e−3 (5.73e−3)
WFG4	4	13	6.4500e−1 (5.67e−2) =	9.8609e−1 (2.66e−2) -	9.4731e−1 (2.79e−2) -	9.3625e−1 (3.26e−2) -	9.5708e−1 (2.25e−2) -	**6.3625e−1 (6.78e−2)**
	6	15	3.0720e−1 (3.87e−2) -	4.8399e−1 (8.44e−2) -	4.6930e−1 (6.09e−2) -	4.7652e−1 (1.24e−1) -	4.4659e−1 (6.51e−2) -	**2.0341e−1 (4.36e−2)**
	8	17	1.7051e−1 (3.23e−2) -	6.2188e−1 (1.00e−1) -	3.3488e−1 (6.29e−2) -	2.7692e−1 (6.80e−2) -	1.9712e−1 (2.93e−2) -	**1.2853e−1 (2.18e−2)**
	10	19	**4.5636e−2 (2.06e−2) +**	7.2709e−1 (1.34e−1) -	3.1210e−1 (7.07e−2) -	9.1273e−2 (3.61e−2) =	1.1309e−1 (2.20e−2) =	1.0345e−1 (2.01e−2)
WFG5	4	13	8.4042e−1 (3.20e−2) -	9.8041e−1 (1.90e−2) -	9.7353e−1 (1.53e−2) -	9.7833e−1 (1.56e−2) -	9.7750e−1 (1.51e−2) -	**7.3167e−1 (3.18e−2)**
	6	15	2.9924e−1 (5.21e−2) -	8.0567e−1 (5.30e−2) -	5.0910e−1 (6.64e−2) -	4.7727e−1 (6.26e−2) -	4.8485e−1 (4.66e−2) -	**1.3712e−1 (2.30e−2)**
	8	17	1.7821e−1 (2.95e−2) -	9.0509e−1 (2.39e−2) -	4.5119e−1 (8.53e−2) -	3.3718e−1 (5.29e−2) -	2.4519e−1 (2.40e−2) -	**7.3397e−2 (1.11e−2)**
	10	19	1.3327e−1 (1.75e−2) -	9.1205e−1 (3.68e−2) -	5.1249e−1 (9.85e−2) -	2.9145e−1 (6.95e−2) -	1.9418e−1 (2.17e−2) -	**3.9636e−2 (1.07e−2)**
WFG6	4	13	**9.7458e−1 (3.90e−2) +**	9.9680e−1 (7.04e−3) +	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	9.9917e−1 (3.73e−3) =	1.0000e+0 (0.00e+0)
	6	15	5.9280e−1 (7.74e−2) =	9.6534e−1 (4.40e−2) -	6.8979e−1 (1.21e−1) -	7.2538e−1 (7.50e−2) -	7.0909e−1 (7.81e−2) -	**5.5530e−1 (3.62e−2)**
	8	17	3.5449e−1 (5.43e−2) -	9.6251e−1 (3.44e−2) -	6.5698e−1 (1.19e−1) -	5.5994e−1 (5.22e−2) -	3.7115e−1 (5.57e−2) -	**2.4231e−1 (3.04e−2)**
	10	19	2.2491e−1 (3.50e−2) -	9.7176e−1 (2.97e−2) -	6.0188e−1 (1.02e−1) -	4.3164e−1 (8.20e−2) -	3.2073e−1 (3.76e−2) -	**1.1436e−1 (1.52e−2)**
WFG7	4	13	**3.4333e−1 (3.40e−2) +**	9.5430e−1 (5.36e−2) -	9.3964e−1 (3.31e−2) -	9.5125e−1 (3.26e−2) -	9.6292e−1 (2.03e−2) -	3.9292e−1 (7.26e−2)
	6	15	2.2462e−1 (4.44e−2) -	5.2022e−1 (1.24e−1) -	4.5206e−1 (7.56e−2) -	5.8030e−1 (6.29e−2) -	4.9697e−1 (8.59e−2) -	**1.6856e−1 (2.84e−2)**
	8	17	1.9359e−1 (3.13e−2) -	5.9153e−1 (1.35e−1) -	3.7343e−1 (9.09e−2) -	3.4135e−1 (1.02e−1) -	2.7821e−1 (5.89e−2) -	**1.2436e−1 (2.37e−2)**
	10	19	**4.4000e−2 (7.82e−3) +**	7.0603e−1 (1.21e−1) -	3.9081e−1 (7.28e−2) -	1.7545e−1 (6.85e−2) =	1.8218e−1 (2.98e−2) -	1.5636e−1 (3.83e−2)
WFG8	4	13	**9.6208e−1 (1.68e−2) =**	9.9957e−1 (1.93e−3) -	9.9640e−1 (6.79e−3) -	9.9500e−1 (7.84e−3) -	9.9625e−1 (5.04e−3) -	9.7042e−1 (1.52e−2)
	6	15	8.3068e−1 (3.65e−2) -	9.9955e−1 (2.01e−3) -	9.3790e−1 (3.45e−2) -	8.6250e−1 (7.71e−2) -	8.6894e−1 (3.36e−2) -	**6.2652e−1 (5.08e−2)**
	8	17	7.4712e−1 (3.23e−2) -	1.0000e+0 (0.00e+0) -	8.8322e−1 (1.10e−1) -	5.5481e−1 (2.11e−1) -	7.0737e−1 (4.17e−2) -	**3.6218e−1 (4.56e−2)**
	10	19	**6.5455e−2 (4.27e−2) +**	1.0000e+0 (0.00e+0) -	7.6137e−1 (6.90e−2) -	4.8055e−1 (1.02e−1) -	6.3436e−1 (3.65e−2) -	3.9364e−1 (4.09e−2)
WFG9	4	13	**6.4167e−1 (6.78e−2) =**	9.9246e−1 (1.50e−2) -	9.8242e−1 (1.87e−2) -	9.8917e−1 (1.51e−2) -	9.7250e−1 (2.89e−2) -	7.4917e−1 (1.86e−1)
	6	15	3.3068e−1 (1.03e−1) -	9.7253e−1 (4.65e−2) -	7.0578e−1 (1.22e−1) -	6.7727e−1 (1.27e−1) -	7.3712e−1 (1.11e−1) -	**2.3295e−1 (1.53e−1)**
	8	17	2.0801e−1 (6.03e−2) -	9.7050e−1 (4.67e−2) -	5.0030e−1 (1.40e−1) -	4.3590e−1 (1.63e−1) -	3.7340e−1 (1.05e−1) -	**8.7500e−2 (3.12e−2)**
	10	19	5.3091e−2 (2.20e−2) -	9.8128e−1 (3.52e−2) -	3.6511e−1 (1.03e−1) -	2.3382e−1 (1.00e−1) -	2.1491e−1 (3.55e−2) -	**3.1818e−2 (1.33e−2)**
± / =			7/20/9	5/24/7	2/29/5	2/27/7	1/29/6	

Table 8 The coverage value of four algorithms for the last 3 years on the WFG test problems

Problem	*M*	*D*	CSEA	hpaEA	MaPSO-MC	KnMAPIO
WFG1	4	13	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	6	15	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	8	17	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	10	19	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
WFG2	4	13	7.9324e−1 (9.24e−2) +	9.3517e−1 (4.13e−2) =	**5.9935e−1 (2.09e−1) +**	9.6141e−1 (3.50e−2)
	6	15	3.8777e−1 (1.90e−1) -	4.9106e−1 (1.18e−1) -	**8.8876e−2 (4.53e−2) =**	1.0303e−1 (1.17e−1)
	8	17	1.1402e−1 (4.52e−2) -	1.6848e−1 (6.10e−2) -	**4.0831e−3 (6.14e−3) =**	1.1538e−2 (1.60e−2)
	10	19	1.6548e−1 (1.03e−1) +	4.4655e−1 (5.25e−2) =	**4.3490e−2 (2.57e−2) +**	3.9055e−1 (1.30e−1)
WFG3	4	13	8.7882e−2 (3.44e−2) =	2.4833e−1 (2.66e−2) -	4.1569e−1 (8.80e−2) -	**7.8333e−2 (1.73e−2)**
	6	15	1.3177e−1 (7.58e−2) -	1.3030e−1 (9.88e−3) -	2.9846e−1 (5.95e−2) -	**1.8182e−2 (6.78e−3)**
	8	17	7.3760e−2 (2.48e−2) -	6.9231e−2 (3.64e−2) -	3.4138e−1 (1.17e−1) -	**8.9744e−3 (7.31e−3)**
	10	19	5.8935e−2 (3.18e−2) -	5.0909e−2 (1.23e−2) -	7.0664e−2 (3.82e−2) -	**1.3091e−2 (7.54e−3)**
WFG4	4	13	7.9583e−1 (2.39e−2) -	9.7651e−1 (1.16e−2) -	**1.6537e−1 (5.07e−2) +**	6.6000e−1 (6.70e−2)
	6	15	3.4200e−1 (6.39e−2) -	3.4739e−1 (7.20e−2) -	**1.4041e−1 (4.03e−2) +**	2.3030e−1 (4.72e−2)
	8	17	8.1938e−2 (1.02e−2) +	9.1026e−2 (4.09e−2) +	**7.7381e−2 (2.13e−2) +**	1.3974e−1 (1.39e−2)
	10	19	3.8910e−2 (9.01e−3) +	**2.0364e−2 (1.35e−2) +**	6.8109e−2 (2.21e−2) +	1.2218e−1 (2.43e−2)
WFG5	4	13	8.0637e−1 (1.64e−2) -	9.8632e−1 (5.51e−3) -	7.9901e−1 (5.26e−2) -	**7.1667e−1 (5.03e−2)**
	6	15	4.4579e−1 (3.09e−2) -	5.1614e−1 (7.01e−2) -	2.8720e−1 (3.12e−2) -	**1.5455e−1 (2.05e−2)**
	8	17	2.7664e−1 (2.88e−2) -	2.5960e−1 (5.32e−2) -	1.3378e−1 (2.73e−2) -	**6.6667e−2 (9.72e−3)**
	10	19	1.6444e−1 (3.20e−2) -	1.1055e−1 (1.51e−2) -	8.8968e−2 (2.37e−2) -	**3.7818e−2 (7.54e−3)**
WFG6	4	13	9.9621e−1 (7.27e−3) =	1.0000e+0 (0.00e+0) =	**9.8457e−1 (2.15e−2) =**	9.9667e−1 (7.45e−3)
	6	15	6.3115e−1 (7.14e−2) =	8.2101e−1 (4.09e−2) -	**3.2693e−1 (6.52e−2) +**	5.7121e−1 (2.76e−2)
	8	17	**2.6822e−1 (3.38e−2) =**	4.3341e−1 (9.76e−2) -	2.9338e−1 (6.54e−2) =	2.7308e−1 (4.29e−2)
	10	19	1.7479e−1 (4.73e−2) -	2.2764e−1 (5.20e−2) -	2.1954e−1 (4.41e−2) -	**1.0764e−1 (1.83e−2)**
WFG7	4	13	7.0001e−1 (2.78e−2) -	9.4556e−1 (3.67e−2) -	**1.1432e−1 (2.58e−2) +**	3.7000e−1 (3.61e−2)
	6	15	3.0822e−1 (2.74e−2) -	5.0667e−1 (3.09e−2) -	**9.6447e−2 (2.91e−2) +**	1.6818e−1 (3.00e−2)
	8	17	**9.0444e−2 (1.62e−2) =**	2.1282e−1 (4.91e−2) -	1.0548e−1 (2.95e−2) =	1.2564e−1 (3.41e−2)
	10	19	**3.5486e−2 (6.21e−3) +**	7.2000e−2 (2.47e−2) +	5.4875e−2 (2.00e−2) +	1.4691e−1 (1.11e−2)
WFG8	4	13	9.8938e−1 (4.95e−3) =	9.9708e−1 (6.53e−3) =	**8.8770e−1 (2.27e−2) +**	9.9333e−1 (6.97e−3)
	6	15	8.1679e−1 (1.41e−1) -	8.1250e−1 (3.22e−2) -	5.5898e−1 (2.22e−2) =	**5.4091e−1 (5.07e−2)**
	8	17	4.0045e−1 (8.99e−2) =	5.6923e−1 (7.26e−2) -	**3.9748e−1 (6.40e−2) =**	4.1795e−1 (3.75e−2)
	10	19	**1.4196e−1 (5.10e−2) +**	3.7745e−1 (6.53e−2) =	2.8114e−1 (4.52e−2) =	3.3964e−1 (3.87e−2)
WFG9	4	13	1.0000e+0 (0.00e+0) -	9.3195e−1 (5.65e−2) =	**5.9301e−1 (8.81e−2) =**	6.4667e−1 (2.14e−1)
	6	15	6.3146e−1 (1.88e−1) -	6.7048e−1 (6.09e−2) -	2.3974e−1 (2.80e−2) =	**1.9242e−1 (5.45e−2)**
	8	17	4.5997e−1 (5.55e−2) -	3.5734e−1 (8.59e−2) -	2.3556e−1 (1.66e−1) -	**6.2821e−2 (2.10e−2)**
	10	19	2.0897e−1 (1.41e−1) -	1.2145e−1 (2.94e−2) -	1.0091e−1 (2.41e−2) -	**2.7636e−2 (7.97e−3)**
± / =			6/19/11	3/23/10	11/11/14	

KnMAPIO. Our algorithm achieves the same result as other algorithms in all objectives of WFG1 and the coverage values are all 1.0000e+0. KnMAPIO has 19 items better than CSEA and 23 items better than hpaEa, with only three items worse. KnMAPIO has similar performance to MaPSO-MC on the WFG test problem. Although MaPSO-MC performs better on WFG 2 and WFG 4, the KnMAPIO has much better performance on WFG 3, WFG 5, and WFG 9.

The results in the tables show that the proposed algorithm has superior performance to that of the other algorithms, which can be attributed to the knee-oriented dominance and the distribution-based individual update strategy. These strategies also improve the selection pressure of the solution set toward the true PF.

In addition, the performance of using only one distribution, two distributions, and three distributions in our proposed

Table 9 The coverage value of algorithms for different objectives in the DTLZ test problems

Problem	M	MAPIO	MAPIOC	MAPIOG	MAPIOLC	MAPIOGC	KnMAPIO
DTLZ1	4	5.4567e−1 (1.50e−1) =	4.5275e−1 (2.63e−1) =	5.8506e−1 (1.92e−1) =	**3.9092e−1 (1.61e−1) =**	5.7206e−1 (1.74e−1) =	4.5156e−1 (1.06e−1)
	6	**5.6970e−1 (6.63e−2) =**	5.6970e−1 (5.75e−2) =	6.6818e−1 (1.10e−1) =	7.0455e−1 (1.54e−1) =	5.9394e−1 (1.45e−1) =	6.3030e−1 (1.78e−1)
	8	6.0333e−1 (9.14e−2) =	6.3333e−1 (1.20e−1) =	5.9103e−1 (2.00e−1) =	**4.8718e−1 (1.88e−1) =**	5.1410e−1 (1.40e−1) =	5.2607e−1 (6.96e−2)
	10	6.2109e−1 (4.86e−2) -	4.5309e−1 (1.17e−1) =	4.5382e−1 (5.85e−2) =	4.6327e−1 (6.92e−2) =	4.6182e−1 (1.04e−1) =	**3.7382e−1 (8.74e−2)**
DTLZ2	4	9.5000e−2 (3.21e−2) -	**3.1667e−2 (1.09e−2) =**	3.6667e−2 (2.25e−2) =	4.5000e−2 (2.54e−2) =	3.3333e−2 (8.33e−3) =	3.1667e−2 (2.07e−2)
	6	4.6970e−2 (2.10e−2) -	1.9697e−2 (8.64e−3) =	1.0606e−2 (4.15e−3) =	7.5758e−3 (7.58e−3) =	**3.0303e−3 (4.15e−3) =**	1.0606e−2 (8.64e−3)
	8	3.3333e−2 (9.51e−3) =	2.3077e−2 (1.85e−2) =	2.5641e−2 (1.63e−2) =	1.9231e−2 (9.07e−3) =	2.0513e−2 (1.05e−2) =	**1.9231e−2 (1.20e−2)**
	10	6.9818e−2 (2.11e−2) =	8.1455e−2 (1.58e−2) -	7.6364e−2 (2.37e−2) -	6.2545e−2 (2.46e−2) =	5.0182e−2 (2.20e−2) =	**4.6545e−2 (1.90e−2)**
DTLZ3	4	2.0199e−1 (6.44e−2) =	8.3043e−2 (6.64e−2) =	1.1681e−1 (8.69e−2) =	1.2603e−1 (7.89e−2) =	**7.2601e−2 (4.89e−2) =**	1.2543e−1 (1.18e−1)
	6	4.0222e−1 (9.77e−2) =	3.8485e−1 (1.14e−1) =	3.4848e−1 (1.31e−1) =	4.2121e−1 (1.25e−1) =	**2.7917e−1 (1.12e−1) =**	3.1667e−1 (8.66e−2)
	8	**3.6597e−1 (9.46e−2) =**	4.0000e−1 (1.18e−1) =	3.8462e−1 (8.64e−2) =	3.9159e−1 (1.86e−1) =	3.9872e−1 (1.31e−1) =	4.0000e−1 (1.10e−1)
	10	4.2691e−1 (8.06e−2) -	4.2182e−1 (1.00e−1) -	3.8182e−1 (8.88e−2) -	3.5636e−1 (7.56e−2) -	4.0727e−1 (7.47e−2) -	**1.9636e−1 (9.58e−2)**
DTLZ4	4	7.1667e−2 (2.80e−2) -	4.0000e−2 (3.25e−2) =	**1.1667e−2 (2.61e−2) =**	3.3333e−2 (1.02e−2) =	2.5000e−2 (3.73e−2) =	2.8333e−2 (1.73e−2)
	6	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
	8	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
	10	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	**0.0000e+0 (0.00e+0)**
DTLZ5	4	**2.6667e−1 (8.25e−2) =**	3.1833e−1 (1.16e−1) =	3.6167e−1 (3.36e−2) =	3.7833e−1 (4.66e−2) =	3.0167e−1 (6.91e−2) =	3.4000e−1 (1.13e−1)
	6	**6.9697e−2 (3.14e−2) =**	1.3333e−1 (3.92e−2) -	1.7424e−1 (6.15e−2) -	1.1364e−1 (7.74e−2) =	1.2273e−1 (8.81e−2) =	7.4242e−2 (4.09e−2)
	8	1.9231e−2 (4.53e−3) =	2.1795e−2 (1.16e−2) =	**1.4103e−2 (8.36e−3) =**	4.2308e−2 (4.36e−2) =	3.0769e−2 (1.05e−2) =	2.6923e−2 (2.19e−2)
	10	1.0182e−2 (3.98e−3) =	**4.3636e−3 (1.63e−3) +**	1.6727e−2 (7.09e−3) =	1.2364e−2 (4.15e−3) =	1.3818e−2 (5.39e−3) =	1.5273e−2 (1.22e−2)
DTLZ6	4	2.7333e−1 (1.49e−2) -	2.1167e−1 (1.54e−1) =	2.3667e−1 (1.37e−1) -	3.8167e−1 (3.54e−1) =	2.5333e−1 (2.30e−1) =	**6.1667e−2 (4.74e−2)**
	6	3.7424e−1 (1.85e−1) -	**3.6364e−2 (3.32e−2) =**	1.1970e−1 (1.34e−1) =	1.6970e−1 (1.50e−1) =	4.6970e−2 (3.45e−2) =	9.8485e−2 (7.93e−2)
	8	5.4744e−1 (1.66e−1) -	2.0000e−1 (1.13e−1) -	1.8718e−1 (1.35e−1) -	4.0128e−1 (1.07e−1) -	3.6282e−1 (2.66e−1) -	**3.2051e−2 (4.44e−2)**
	10	7.0255e−1 (9.28e−2) -	8.1091e−1 (6.69e−2) -	7.8109e−1 (2.96e−2) -	7.2655e−1 (8.72e−2) -	8.1891e−1 (7.34e−2) -	**2.3564e−1 (7.25e−2)**
DTLZ7	4	**9.8500e−1 (3.35e−2) =**	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	9.9667e−1 (7.45e−3)
	6	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	8	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
	10	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	1.0000e+0 (0.00e+0) =	**1.0000e+0 (0.00e+0)**
± / =		0/9/19	1/5/22	0/6/22	0/3/25	0/3/25	

algorithm was compared on DTLZ test functions. The experimental results are shown in Table 9. MAPIO algorithm does not uses the velocity position update formula proposed by us. MAPIOC is the algorithm only uses the Cauchy distribution. MAPIOG algorithm uses the Gaussian distribution only. MAPIOLC is the algorithm uses Levy and Cauchy distributions simultaneously, and MAPIOGC algorithm uses Gaussian and Cauchy distributions at the same time.

The Coverage value of algorithms for different objectives in the DTLZ test problems has shown in Table 9. KnMAPIO has 13 best results, MAPIOC has 3 best results. MAPIOG, MAPIOLC, and MAPIOGC have only 2 best results. As can be seen from the last row, MAPIO has nine results worse than our algorithm, but zero result better than KnMAPIO. MAPIOC has only one item better than the proposed method, but five items worse than ours. MAPIOLC and MAPIOGC have neither result better than ours, but worse in three. Although the proposed algorithm is slightly inadequate in some test problems, its overall performance is better than other comparison algorithms. The performance of the algorithm using only one distribution is slightly worse than the performance of the algorithm using two distributions, and the performance of the algorithm using two distributions is worse than the performance of the algorithm combining the three distributions. So the strategy we proposed is effective.

Conclusion

This paper proposes a knee point-driven many-objective pigeon-inspired algorithm (KnMAPIO) to solve the problem of selection of individuals in the case of non-decision maker preference. The proposed algorithm uses an environmental selection strategy based on knee-oriented dominance to enhance the selection pressure of the individuals, and the diversity of the population and the identification ability of the knee point region have been promoted through the full use of extreme points, boundary points, and knee sorting mechanisms. Next, a new velocity update equation with Gaussian distribution, Cauchy distribution, and Levy distribution is proposed, and the use of these three distributions at the appropriate stage effectively improves the search performance. In most test problems, our algorithm has excellent performance in identifying knee points and their regions in comparison experiments with four other algorithms on the knee-oriented benchmark PMOPs. In addition, compared with five state-of-the-art many-objective algorithms and three algorithms for the last 3 years on the standard benchmark DTLZ and WFG suites, our algorithm shows significantly better performance.

In future work, we will apply the proposed KnMAPIO algorithm to solve practical engineering problems without decision-maker preference and provide a set of efficient solutions.

Acknowledgements This work was supported by the National Key Research and Development Program of China under Grant No.2018YFC1604000, the National Natural Science Foundation of China under Grant Nos. 61806138, 61772478, U1636220, 61961160707 and 61976212, Key R&D program of Shanxi Province (High Technology) under Grant No. 201903D121119, Key R&D program of Shanxi Province (International Cooperation) under Grant No.201903D421048, and the Key R&D program (international science and technology cooperation project) of Shanxi Province (No. 201903D421003).

Author contributions LZ: conceptualization, methodology, writing—original draft. YZ: software. YR: writing revising. ZC: methodology. WZ: supervision.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Cui ZH, Zhao YR, Cao Y, Cai XJ, Zhang WS, Chen JJ (2021) Malicious code detection under 5G HetNets based on multi-objective RBM model. IEEE Netw 35(2):82–87
2. Cui ZH, Xu XH, Xue F, Cai XJ et al (2020) Personalized recommendation system based on collaborative filtering for IoT scenarios. IEEE Trans Serv Comput 13(4):685–695
3. Cui ZH, Xue F, Zhang SQ et al (2020) A hybrid blockchain-based identity authentication scheme for multi-WSN. IEEE Trans Serv Comput 13(2):241–251
4. Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans Evol Comput 18(4):577–601
5. Zitzler E, Laumanns M, and Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In: Proc. Fifth Conference on Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems (pp. 95–100)
6. Bader J, Zitzler E (2011) HypE: an algorithm for fast hypervolume-based many-objective optimization. Evol Comput 19(1):45–76
7. Zhang Q, Li H (2007) MOEA/D: a multi-objective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731

8. Li K, Deb K, Zhang QF, Kwong S (2015) An evolutionary any-objective optimization algorithm based on dominance and decomposition. IEEE Trans Evol Comput 19(5):694–716
9. Wang WJ, Wang H, Li CY (2020) Decision preference-based artificial bee colony algorithm for many-objective optimal allocation of water resources. Int J Comput Sci Math 12(4):364–373
10. Wang H, Olhofer M, Jin Y (2017) A mini-review on preference modeling and articulation in multi-objective optimization: current status and challenges. Complex Intell Syst 3(4):233–245
11. Wang JY (2020) A hybrid grid-based many-objective optimisation algorithm for software defect prediction. Int J Comput Sci Math 12(4):374–384
12. Dorigo M, Maniezzo V (1996) Ant system: optimization by a colony of cooperating agents. IEEE Trans SMC-Part B 26(1):29
13. Wu D (2020) An effective improved co-evolution ant colony optimization algorithm with multi-strategies and its application. Int J Bio Inspired Comput 16(3):158–170
14. Cui ZH, Zhang JJ, Wu D et al (2020) Hybrid many-objective particle swarm optimization algorithm for green coal production problem. Inf Sci 518:256–271
15. Lin QZ, Liu SB, Zhu QL et al (2018) Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems. IEEE Trans Evol Comput 22(1):32–46
16. Cui ZH, Zhang MQ, Wang H, Cai XJ, Zhang WS, Chen JJ (2020) Hybrid many-objective cuckoo search algorithm with lévy and exponential distributions. Memet Comput 20(3):251–265
17. Hafiz TR, Muhammad H, Abdur R (2020) Bat algorithm with Weibull walk for solving global optimisation and classification problems. Int J Bio Inspired Comput 15(3):159–170
18. Yang XS, He X (2013) Bat algorithm: literature review and applications. Int J Bio Inspired Comput 5(3):141–149
19. Zhang B, Duan HB (2014) Predator-prey pigeon-inspired optimization for UAV three-dimensional path planning. Int Conf Swarm Intell 96–105
20. Zhao J, Tang JJ, Shi AY, Fan TH, Xu LZ (2020) Improved density peaks clustering based on firefly algorithm. Int J Bio-Inspired Comput 15(1):24–42
21. Jon AH, Jørgen S, Dylan T, Ali A, Ghavameddin N (2020) An enhanced breeding swarms algorithm for high dimensional optimisations. Int J Bio Inspired Comput 15(3):181–193
22. Premalatha B, Maheswari SU (2020) A hybrid bio-inspired optimisation approach for wirelength minimisation of hardware tasks placement in field programmable gate array devices. Int J Bio-Inspired Comput 15(2):125–134
23. Duan HB, Qiao P (2014) Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. Int J Intell Comput Cybern 7(1):24–37
24. Fu XY, Chan FTS, Niu B et al (2019) A multi-objective pigeon inspired optimization algorithm for fuzzy production scheduling problem considering mould maintenance. Sci China (Inform Sci) 062(007):11–28
25. Duan HB, Zhao JX, Deng YM, Shi YH, Ding XL (2021) Dynamic discrete pigeon-inspired optimization for multi-UAV cooperative search-attack mission planning. IEEE Trans Aerosp Electron Syst. https://doi.org/10.1109/TAES.2020.3029624
26. Qiu HX, Duan HB (2020) A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. Inf Sci 509:515–529
27. Cui ZH, Zhang JJ, Wang YC et al (2019) A pigeon-inspired optimization algorithm for many-objective optimization problems. Sci China Inf Sci 62:70212
28. Deb K, Gupta S (2011) Understanding knee points in bicriteria problems and their implications as preferred solution principles. Eng Optim 43(11):1175–1204
29. Zhang X, Tian Y, Jin Y (2015) A knee point-driven evolutionary algorithm for many-objective optimization. IEEE Trans Evol Comput 19(6):761–776
30. Sun H, Deng ZC, Zhao J, Xie HH (2020) Particle swarm optimisation by adding Gaussian disturbance item guided by hybrid narrow centre. Int J Comput Sci Math 11(4):327–337
31. Kumar N, Rahman MS, Duary A, Mahato SK, Bhunia AK (2020) A new QPSO based hybrid algorithm for bound-constrained optimisation problem and its application in engineering design problems. Int J Comput Sci Math 12(4):385–412
32. Duan HB, Yang ZY (2018) Large civil aircraft receding horizon control based on Cauthy mutation pigeon inspired optimization (in Chinese). Sci Sin Tech 48:277–288
33. Yang ZY, Duan HB, Fan YM (2018) Unmanned aerial vehicle formation controller design via the behavior mechanism in wild geese based on Levy flight pigeon-inspired optimization (in Chinese). Sci Sin Tech 48:161–169
34. Arshad H, Batool S, Amjad Z et al (2017) Pigeon inspired optimization and enhanced differential evolution using time of use tariff in smart grid. Int Conf Intell Netw Collab Syst: 563–575
35. Sun XX, Pan JS, Chu SC et al (2020) A novel pigeon-inspired optimization with QUasi-Affine TRansformation evolutionary algorithm for DV-Hop in wireless sensor networks. Int J Distrib Sens Netw 16(6):155014772093274
36. Li C, Duan HB (2014) Target detection approach for UAVs via improved pigeon-inspired optimization and edge potential function. Aerosp Sci Technol 39:352–360
37. Duan HB, Qiu HX, Fan YM (2015) Unmanned aerial vehicle close formation cooperative control based on predatory escaping pigeon-inspired optimization (in Chinese). Sci Sin Tech 45:559–572
38. Hai XS, Wang ZL et al (2021) A novel adaptive pigeon-inspired optimization algorithm based on evolutionary game theory. Sci China Inf Sci 64(3):139203
39. Qiu HX, Duan HB (2015) Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. Sci China Tech Sci 58:1915–1923
40. Deb K, Pratap A, Agarwal S et al (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
41. Yi HU, Wang J, Liang J et al (2019) A self-organizing multimodal multi-objective pigeon-inspired optimization algorithm. Sci China Inf Sci 62(007):1–17
42. Duan HB, Huo MZ, Shi YH (2020) Limit-cycle-based mutant multiobjective pigeon-inspired optimization. IEEE Trans Evol Comput 24(5):948–959
43. Shang J, Li Y, Sun Y, Li F, Zhang Y, Liu JX (2021) MOPIO: a multiobjective pigeon-inspired optimization algorithm for community detection. Symmetry 13:497–506
44. Zou J, Li Q, Yang S, Bai H, Zheng J (2017) A prediction strategy based on center points and knee points for evolutionary dynamic multi-objective optimization. Appl Soft Comput 61:806–818
45. Branke J, Deb K, Dierolf H, Osswald M (2004) Finding knees in multi-objective optimization. Proc Int Conf Parallel Problem Solving Nat: 722–731
46. Ikeda K, Kita H, Kobayashi S (2001) Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal?. Proc Congr Evol Comput Seoul South Korea 2: 957–962
47. Yu G, Jin Y, Olhofer M (2019) An a priori knee identification multi-objective evolutionary algorithm based on α-dominance. Proc Genet Evol Comput Conf Compan: 241–242
48. Yu G, Jin Y, Olhofer M (2021) A multiobjective evolutionary algorithm for finding knee regions using two localized dominance relationships. IEEE Trans Evol Comput 25(1):145–158
49. Lin Q, Li J, Du Z et al (2015) A novel multi-objective particle swarm optimization with multiple search strategies. Eur J Oper Res 247(3):732–744

50. Dulebenets MA (2020) Archived elitism in evolutionary computation: towards improving solution quality and population diversity. Int J Bio-Inspired Comput 15(3):135–146
51. Zhang YW (2021) Subjectivity and nominal property concepts in Mandarin Chinese. [Doctoral dissertation, Indiana University]. ProQuest Dissertation Publishing
52. Zhang YW (2020) Nominal property concepts and substance possession in mandarin Chinese. J East Asian Linguis 29:393–434
53. Grano T, Zhang YW (2019) Getting aspectual -guo under control in Mandarin Chinese: An experimental investigation. In: Proc. Congress on the 30th North American Conference on Chinese Linguistics (NACCL-30) 1: 208–215
54. Tian Y, Cheng R, Zhang X, Jin Y (2017) PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput Intell Mag 12(4):73–87
55. Yu G, Jin Y, Olhofer M (2020) Benchmark problems and performance indicators for search of knee points in multiobjective optimization. IEEE Trans Cybern 50(8):3531–3544
56. Deb K, Thiele L, Laumanns M et al (2002) Scalable multi-objective optimization test problems Evolutionary Computation. In: Proc. Congress on IEEE (CEC'02) 825–830
57. Huband S, Hingston P, Barone L et al (2006) A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans Evol Comput 10(5):47
58. Yang S, Li M, Liu X, Zheng J (2013) A grid-based evolutionary algorithm for many-objective optimization. IEEE Trans Evol Comput 17(5):721–736
59. Cheng R, Jin YC, Olhofer M, Sendhoff B (2016) A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Trans Evol Comput 20(5):773–791
60. Xiang Y, Zhou Y, Li M, Chen Z (2017) A vector angle-based evolutionary algorithm for unconstrained many-objective optimization. IEEE Trans Evol Comput 21(1):131–152
61. Pan LQ, He C, Tian Y, Wang HD, Zhang XY, Jin YC (2018) A classification based surrogate-assisted evolutionary algorithm for expensive many-objective optimization. IEEE Trans Evol Comput 23(1):74–88
62. Chen HK, Tian Y, Pedrycz W, Wu GH, Wang R, Wang L (2020) Hyperplane assisted evolutionary algorithm for many-objective optimization problems. IEEE Trans Cybern 7:3367–3380
63. Hu ZM, Lan Y, Zhang ZX, Cai XJ (2021) A many-objective particle swarm optimization algorithm based on multiple criteria for hybrid recommendation system. KSII Trans Internet Inf Syst 15(2):442–460