Applied Intelligence (2022) 52:1157–1187
https://doi.org/10.1007/s10489-021-02313-6

An improved multiobjective cultural algorithm with a multistrategy knowledge base

Zhengyan Mao[1] · **Mandan Liu**[1]

Accepted: 3 March 2021 / Published online: 18 May 2021

Abstract

Based on the dual-inheritance framework of cultural evolution, an improved multiobjective cultural algorithm (IMOCA) with a multistrategy knowledge base is presented in this paper. Inspired by the original versions of the cultural algorithm (CA), four basic types of knowledge sources, i.e., normative, situational, topographical and historical knowledge, are effectively utilized in the proposed IMOCA. Several modifications with the knowledge base of the IMOCA are made to tackle the characteristics of the multiobjective problem. Situational knowledge is used as an external repository for storing elite individuals, and the redesigned topographical knowledge functions as a search engine to broaden the expansion of the obtained solution set. The historical knowledge used in the IMOCA aims to select a productive knowledge source to generate new individuals. Furthermore, a simple mutation scheme is introduced into the knowledge base as an influence function for the purpose of fine tuning in the late stage of search. After configuring the parameters used in IMOCA, two classic benchmark suites, i.e., WFG and MaF, are used to assess the performance of the IMOCA in approaching the Pareto fronts (PFs) with accuracy and diversity. Nondominated solution sets obtained by the IMOCA are compared with 8 state-of-the-art multiobjective algorithms available in the literature. A statistical analysis is conducted, which reveals that, by modifying the basic knowledge structure of the CA, the proposed multiobjective cultural algorithm is competent enough to handle multiobjective problems with competitive performance.

Keywords Multiobjective optimization · Evolutionary algorithm · Cultural algorithm · Dual-inheritance framework · Knowledge-based evolutionary computation · Nondominated sorting

1 Introduction

After John Holland presented the genetic algorithm (GA) as an abstraction of biological evolution and gave a theoretical framework for adaptation under the GA in his book [23], for the last four decades, a large amount of research focusing on nature-inspired optimization algorithms has been conducted, leading to the emergence of a series of classic metaheuristic optimization algorithms, such as simulated annealing, tabu search, ant colony optimization, and particle swarm optimization.

In 1994, a conceptual model of the cultural algorithm (CA) was first proposed by [34], which was further developed by Reynolds and his students in the following years [9, 25, 36, 37]. As a novel evolutionary framework with the principle of dual-inheritance between culture and individuals, the CA soon attracted much attention and was proved to be of practical success in a variety of problem domains [20, 44, 45].

When considering the CA, it is necessary to mention the memetic algorithm (MA), which was put forward by [31]. In essence, the key difference between the CA and MA lies in the way in which the individuals search and evolve. Specifically, the MA can be seen as a combination of a global search of the whole population and a local search of individuals, which is based on imitation from a certain subset of the population [32], while in the CA the learning process of each individual is based on the belief space, and the knowledge source stored in the belief space

✉ Mandan Liu
liumandan@ecust.edu.cn

Zhengyan Mao
020130072@mail.ecust.edu.cn

[1] Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China

during the evolution can be any information that benefits progress.

Nevertheless, returning to optimization problems, decision makers usually need to consider more than one objective (which are often conflicting) when addressing practical optimization problems. For instance, minimization of the operational costs of a certain plant is one objective, while the quality of products is another aspect that should be taken into account. Though an intuitive approach to handling multiobjective problems is to add all the objectives together under a set of weighting coefficients according to the importance of each objective, the configuration of the preference vector is difficult, as it is inevitably problem-specific. Moreover, the outcome of using a classical optimization method is a single optimized solution. Benefiting from the ability of intelligent optimization algorithms to find multiple optimal solutions in one single simulation run, however, multiobjective optimization problems (MOOPs) can be solved without converting the task of finding multiple trade-off solutions into finding one single solution; thus, the task of decision makers is to simply pick out one solution among the resulting set of optimal trade-offs according to their preference.

Both evolutionary and swarm-based optimization algorithms have been successfully extended to multiobjective fields and quite a few of them have shown satisfactory performance on test and/or practical applications. Deb et al. [17] introduced a fast nondominated sorting approach called the NSGA-II based on the previous nondominated sorting genetic algorithm (NSGA) by [40] to reduce the computational complexity. Additionally, an elitism and the notion of crowding distance were incorporated for the purpose of preserving elites and omitting solutions located in overcrowded regions, respectively. As a successful improvement to the NSGA, the NSGA-II was one of the most popular multiobjective optimization algorithms in the two decades following its inception. In 2001, on the basis of their Pareto envelope-based selection algorithm (PESA, presented by [14]), [15] used a new selection technique in their improved algorithm PESA-II, in which selective fitness is assigned to the hyperboxes in the objective space instead of to individuals. Compared to the individual-based selection used in the PESA, the hyperboxes-based selection was shown to be more able to ensure a good spread of development along the Pareto front. In the same year, the strength Pareto evolutionary algorithm 2 (SPEA2), which integrates a fine-grained fitness assignment strategy, a density estimation technique, and an enhanced archive truncation method into its predecessor the SPEA in [49], was proposed by [51] and performed competitively on both combinatorial and continuous test problems compared with the SPEA, NSGA-II and PESA. One of the most famous swarm-based optimization algorithms, i.e., the particle swarm optimization (PSO) algorithm, was also extended to address MOOPs in [13]. In the multiobjective particle swarm optimization (MOPSO), the Pareto dominance is adopted, and the previously found nondominated vectors are maintained in an external repository to guide the flight of other particles. In 2007, [46] proposed a multiobjective evolutionary algorithm based on decomposition (MOEA/D), which decomposes an MOOP into a number of scalar optimization subproblems and optimizes them simultaneously. The application of the MOEA/D with three different decomposition methods on both multiobjective 0/1 knapsack problems and multiobjective continuous test instances was compared with the MOGLS and NSGA-II, respectively.

Over the past five years, several newer multiobjective evolutionary algorithms (MOEAs) have been proposed, with competitive performance compared to the representative ones mentioned above. A new multiobjective optimization framework, comprising nondominated sorting, local search and the farthest-candidate approach, named the nondominated sorting and local search (NSLS) algorithm was introduced by [5]. Tian et al. [42] proposed an MOEA based on an enhanced inverted generational distance metric (termed MOEA/IGD-NS) that can omit noncontributing solutions during the evolutionary search. In the same year, [27] proposed a framework containing a bicriterion evolution (BCE) with an indicator-based evolutionary algorithm (IBEA) embedded into its non-Pareto criterion (NPC) evolution part (termed BCE-IBEA). The effectiveness of this framework was shown by experiments on seven groups of test problems with various characteristics.

Moreover, the potential of the CA, which is an expandable framework that allows communication between the population and knowledge base, in solving MOOPs has attracted researchers' attention [11]. However, there is still a lack of clear demonstrations of the merits of multiobjective cultural algorithms (MOCAs) compared to other classic multiobjective optimization algorithms. Furthermore, few comprehensive statistical analyses of the performance of MOCAs in solving a sufficient number of benchmark problems are available in the existing literature. In addition, there is still a large space in which to explore the potential of MOCAs in terms of the usage of various types of knowledge sources.

In this work, an improved multiobjective cultural algorithm (IMOCA), as a variant version of the previously proposed MOCAs, is presented. Following Reynolds' basic framework of the CA in [34], the IMOCA incorporates several strategies into the basic knowledge sources in the belief space, which aim to balance convergence and coverage in finding a proper set of Pareto optimal solutions with efficiency. The situational knowledge is designed as an

elite repository to avoid losing promising solutions during the search process, whereas the topographical knowledge is utilized to expand the spread of solutions. The historical knowledge functions as a selector that determines which influence function will be used to generate offspring. A mutation scheme is also included as one of the influence functions to improve the diversity of the solution set.

The remainder of this paper is organized as follows. Some related studies on MOCAs are reviewed in Section 2. Section 3 briefly introduces a basic description of an MOOP. Section 4 gives a concise description of the basic cultural algorithm. Section 5 presents the proposed IMOCA. Section 6 presents configurations of the experimentation on a series of standard test problems. The experimental results and a statistical analysis are given in Section 7. Section 8 concludes this paper and suggests some directions for future studies.

2 Related work

In this section, some of the recent studies on expanding CAs from solving single-objective to multiobjective problems are reviewed.

In 2003, the first extension of the CA to solving MOOPs was presented by [12]. In their approach, named the multiobjective cultural algorithm with evolutionary programming ((MO)CAEP), evolutionary programming (EP) is used as the search engine, and the belief space comprises a phenotypic normative part and a segmentation scheme that aim to improve the distribution of the solutions along the PF. In the belief space of the (MO)CAEP, the phenotypic normative part is not updated at every iteration, as doing so involves the rebuilding of the grids once the upper and lower bounds stored in the normative part are changed. On the other hand, the counters of the nondominated individuals contained within each cell are updated in every generation.

Instead of utilizing only one type of knowledge source, [3] introduced another multiobjective version of the CA, in which situational, domain, normative, historical and topographical knowledge are incorporated in the belief space. The five knowledge sources were defined as follows: situational knowledge was recognized as a set of exemplary individuals that were nondominated; domain knowledge was designed to perform an incremental search of a certain region in the search space; normative knowledge provided a promising region based on the interval for each dimension over a set of high-performing individuals; historical knowledge recorded the progress of the elites for each objective separately; topographical knowledge was designed to divide the search space and identified regions containing well-performing solutions. In each generation, only one of the five is chosen to influence the solutions in the population by the corresponding influence function. The test function DTLZ1 was used to investigate the capability of this version of the MOCA, and a detailed analysis of the productivity of the five knowledge sources was carried out. It appeared that the topographical knowledge performed far from satisfactorily in producing nondominated solutions, which indicated that certain adjustments should be made to the influence function of topographical knowledge.

Then, in the following year, [38] provided another extension of the CA called cultural algorithms for multiobjective optimization (MOCAT), in which all of the available categories of knowledge sources are fully utilized. In MOCAT, the normative, situational, domain and historical knowledge are generally inherited from the MOCA of [3], while the implementation of topographical knowledge is reversed from top-down to bottom-up to avoid high computational consumption in higher-dimensional objective space. However, the application of MOCAT on the test function ZDT1 did not reflect a good distribution on the Pareto front. Unfortunately, neither an experimental comparison with other multiobjective algorithms nor a thorough analysis was presented in this article.

Several other contemporary variations of the cultural algorithm for solving multiobjective problems were proposed in the past decade. Qin et al. [33] presented a multiobjective cultured differential evolution (MOCDE) algorithm based on the cultural algorithm framework and used it to deal with reservoir flood control operation (RFCO) problems. In MOCDE, situational, normative and historical knowledge are used to influence the evolution of a population, where the historical knowledge stores the coverage metrics of the past h generations and the adaptive Cauchy mutation operation is carried out on a certain dimension when the change in the coverage record is smaller than a predefined threshold. Before being applied to a case study of RFCO, the MOCDE algorithm was tested on benchmark problems ZDT1/3/4/6, SRN and TNK, and competitive results were obtained compared with six well-known multiobjective optimization algorithms. A hybrid multiobjective cultural algorithm (HMOCA) was introduced by [29], and a performance comparison with the NSGA-II was made in an application to two case studies of a short-term environmental/economic hydrothermal scheduling problem. In their method, a self-adaptive mutation operator taken from DE is used in the population space, and the preservation of the elite Pareto optimal solutions found along with the evolution process, which is similar to the archive strategy in the SPEA2, is incorporated in the belief space. In addition, the historical knowledge in the HMOCA was redefined as a local search operator based on the dominance concepts and crowding distance to carry out a guided local

fine tuning process on the Pareto optimal solutions in the archive set to enhance the convergence properties. Zhang et al. [47] proposed an enhanced multiobjective cultural algorithm (EMOCA) that integrates PSO into the evolutionary process in the population space of the cultural algorithm. Situational knowledge was used as an external repository to preserve elite particles, whereas the historical knowledge in the EMOCA was redesigned as a local search operation to search the area around the particles in situational knowledge. The EMOCA was implemented on an economic environment dispatch optimization problem and the simulation results were compared with those of the NSGA-II. Liu et al. [28] introduced the cultural evolution mechanism into the multiobjective quantum-behaved particle swarm optimization framework and proposed the cultural MOQPSO algorithm. In the belief space of the cultural MOQPSO, a local-search-based strategy and a combination-based update operator were utilized to guide the search in the population space. In [21] and [22], the cultural evolution framework was combined with particle swarm optimization and quantum-inspired evolution for solving uncertain multiobjective optimization problems and interval multiobjective optimization problems respectively. Stanley et al. [41] proposed a parallelized multiobjective cultural algorithm particle swarm optimizer, CAPSO, which is a hybrid system composed of a particle swarm and the vector-evaluated genetic algorithm population component operating under the control of the cultural algorithm framework. In their work, the relative contribution of different CA knowledge sources was analyzed under the deployment of PSO swarms in solving constrained multiobjective optimization problems. In [1], an enhanced cultural algorithm using only normative and situational knowledge was applied to solve the multi-objective attribute reduction problem, which was a discrete problem. In the work of [30], the evolution scheme of a multiobjective five-element cycle optimization algorithm (MOFECO) was successfully incorporated into the population space of the CA, and this novel combination, named MOCAFECO, was proved to be feasible and efficient in solving MOOPs based on statistical results.

In the research of [3, 12] and [38], a set of basic knowledge sources was utilized to solve multiobjective problems; however, the results did not reflect satisfactory performance of the knowledge sources. The lack of thorough experimentation and analysis is another key problem. In the rest of the aforementioned research, some novel search strategies were adopted in the framework of the CA to tackle multiobjective problems. However, the productivity of each knowledge source in the belief space of the CA was rarely analyzed or discussed, and the performance of the proposed MOCAs was seldom evaluated or compared with that of other multiobjective algorithms using Pareto-compliant metrics. In this paper, we aim to introduce a simple but effective multiobjective cultural algorithm framework using the most basic knowledge source without adopting other novel evolutionary notions or strategies, say, to keep the algorithm as vanilla as possible, and to validate the effectiveness of each knowledge source, and analyze its performance in comparison with that of other algorithms under several Pareto-compliant metrics.

3 Description of the multiobjective optimization problem

The definition of an MOOP addressing more than one objective function is given. Consider, without loss of generality, a multiobjective minimization problem with n decision variables, M objectives, J inequality constraints and K equality constraints:

$$\begin{aligned} \text{Minimize} \quad & \mathbf{y} = f(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_M(\mathbf{x})) \\ \text{subject to} \quad & g_j(\mathbf{x}) \leq 0, \ j = 1, 2, \cdots, J \\ & h_k(\mathbf{x}) = 0, \ k = 1, 2, \cdots, K \\ \text{where} \quad & \mathbf{x} = (x_1, \cdots, x_n) \in \mathbf{X} \\ & \mathbf{y} = (y_1, \cdots, y_M) \in \mathbf{Y}. \end{aligned} \tag{1}$$

Here, the n-dimensional vector $\mathbf{x}$ is called the decision vector with n decision variables in it, $\mathbf{X}$ is called the decision space, $\mathbf{y}$ is the objective vector, and $\mathbf{Y}$ is the objective space.

4 Basic notions of the cultural algorithms

The cultural algorithm was proposed and developed by [34] as a dual-inheritance process that occurs simultaneously at the micro-evolutionary level (population space) and at the macro-evolutionary level (belief space), which metaphorically models the cultural evolution of human society. The basic framework of the CA is shown in Fig. 1, from which we can see the two main components of the CA: a population space where the regeneration of individuals takes place and a belief space where individual experiences are preserved and passed from generation to generation. Communication between the two components is realized through protocols that enable not only knowledge extraction from the population space to the belief space but also knowledge to exert influence on the population in turn. We call the two protocol functions the acceptance function $Accept()$ and the influence function $Influence()$, respectively. Further explanations of the CA framework are given in the rest of this section.

For the purpose of illustration, in the rest of this paper, we use an $N \times n$ matrix $\mathbf{p}$ to denote the population space

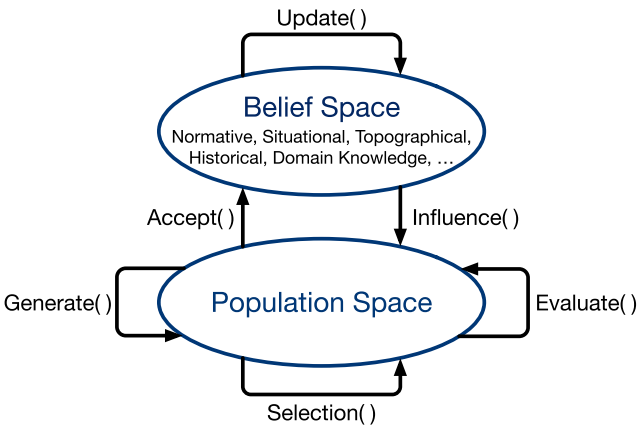


Fig. 1 Framework of the cultural algorithm

with capacity N, in which element x_{ij} is the j-th decision variable of the i-th individual in the current population:

$$\mathbf{p} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{11}x_{12} & \cdots x_{1n} \\ x_{21}x_{22} & \cdots x_{2n} \\ \vdots\vdots & \ddots\vdots \\ x_{N1}x_{N2} & \cdots x_{Nn} \end{bmatrix}, \tag{2}$$

where $\mathbf{x}_i = (x_{i1}, x_{i2}, \cdots, x_{in})$, with $i \in \{1, 2, \cdots, N\}$, denotes one of the N solutions to the MOOP at hand.

4.1 Population space

At the micro-evolutionary level of cultural algorithms, there is a population of individuals, each of which represents a solution to the problem to be optimized. The number of individuals that the population accommodates, namely, the size of the population, is defined by the user as a predefined parameter of the algorithm. In the population space, the individuals evolve following a certain evolutionary mechanism over the course of an iteration.

As Reynolds stated in [34], as a class of computational models of cultural evolution that support the dual-inheritance perspective, cultural algorithms provide a framework in which to describe all of the current models of cultural evolution from a computational point of view. More specifically, as illustrated by [12], any evolutionary computational technique in which a population is adopted could be used in the population space of cultural algorithms. For example, [8] introduced a cultural-algorithm-based testbed with the genetic algorithm and evolutionary programming as the evolutionary technique in the population space to achieve constrained numerical optimization. It was also addressed in their paper that evolution strategies are another alternative to be embedded in the population space. In addition, [2] incorporated differential evolution into the population space and applied the cultural algorithm to constrained optimization problems.

4.2 Belief space

One of the most distinguishing characteristics of the CA compared with other nature-inspired metaheuristic evolutionary algorithms is its information-sharing scheme among individuals through five major knowledge types, namely, normative knowledge, situational knowledge, domain knowledge, historical knowledge, and topographical knowledge. Normative, topographical, and situational knowledge have been used to solve real-valued function optimization problems under static environments [39]. The other two knowledge sources, i.e., historical and domain knowledge, were added because of their particular use in solving dynamic problems [4]. It should be noted that either the way the data structures are employed or even how the knowledge sources are defined changes along with the development of the CA.

Among the two basic knowledge sources, situational knowledge provides a set of exemplars that represent specific individual experience, and normative knowledge provides standards for individual behavior and sets guidelines within which individual adjustments can be made [7]. When it was first proposed in [26], topographical knowledge was used to handle constraints in solving single-objective optimization problems (SOOPs). Within the region given by the intervals saved in normative knowledge, the topographical knowledge segments the search space into $(nGrid)^M$ hypercubes; the promising areas are then re-split into sub-hypercubes, while poorly performing areas are fused back into larger ones. Historical knowledge was originally proposed for dynamic objective functions and was used to record the location of the best individual ever found before each environmental change to find patterns in the environmental changes [39]. Domain knowledge was proposed to cope with dynamic optimization problems.

4.3 Acceptance function

As one of the two protocol functions between the population space and the belief space, the acceptance function is used to glean the experience of the selected individuals, just as in human society, where knowledge is generalized by certain experts or elites [26]. Simply speaking, the acceptance function determines which individuals in the current population can be used to impact the update of the belief space knowledge. It is usually defined as a percentage so that a certain proportion of the population is selected to exert the impact. There is also a dynamic acceptance function with an acceptance percentage that can be adjusted to manipulate the pace of the belief space update [35].

4.4 Influence function

The influence function is a vehicle for reproducing individuals in the population under the guidance of the knowledge in the belief space. In general, each knowledge source has its influence function, which can be modified according to the nature of the problem at hand. The influence functions can be based on more than just one knowledge source. To save space, the influence functions for the different knowledge structures in the CA are presented, along with an illustration of the IMOCA, in the next section, and some of the classic configurations of influence functions of the basic knowledge sources can be found in the literature [4].

5 The improved multiobjective cultural algorithm with a multistrategy knowledge base (IMOCA)

As an improved multiobjective version of the basic CA (which was originally designed for SOOPs), the proposed IMOCA integrates necessary modifications in the knowledge base to cope with MOOPs, taking advantage of various knowledge sources. First, the structure of historical and topographical knowledge is redesigned to improve the efficiency of the knowledge selection scheme and to gain a better spread of the obtained PF, respectively. Thus, there is a corresponding adjustment in the influence functions of the two sources of belief space knowledge. Second, in addition to the influence functions of the four knowledge sources, a straightforward mutation method is used as an "influence function" in the influence step instead of in the evolution step of the population. In other words, the mutation operation affects the population only when it is selected in the influence phase. Third, due to the fact that multiobjective problems involve more than one single objective, the Pareto dominance relationship should be introduced in the comparison between solutions. Furthermore, the nondominated sorting method, which firstly proposed for the NSGA-II [17], is adopted in the IMOCA for ranking individuals efficiently. Finally, another modification, i.e., an elite maintenance mechanism, is included in the IMOCA to avoid losing promising solutions during the search process. Apart from the adjustments mentioned above, the evolutionary scheme adopted in the IMOCA population space is inherited from the prototype of [8]. The details of the proposed IMOCA algorithm are given in the rest of this section.

5.1 Evolution scheme in the population space of the IMOCA

Based on the simplified EP algorithm suggested by [8], a framework analogous to the EP algorithm is adopted as the evolution mechanism of the population space in the IMOCA, except that the standard mutation operator is replaced by one of the five influence functions driven by knowledge sources and a simple mutation scheme.

5.2 Knowledge sources in the belief space of the IMOCA

5.2.1 Situational knowledge

When dealing with SOOPs, situational knowledge contains a set of exemplars that are used to guide the search. Accordingly, in the IMOCA, the situational knowledge serves as an external archive /repository preserving elite solutions, i.e., nondominated solutions found during the search process. This tactic is analogous to the use of an external repository in quite a few classic multiobjective algorithms such as the SPEA2, MOPSO, etc.

Similar to the basic configuration of situational knowledge in the CA for SOOPs, the situational knowledge $\mathbf{S}$ adopted in the IMOCA is represented in the following form:

$$S = \{\mathbf{E}_1,\ \mathbf{E}_2,\ \cdots,\ \mathbf{E}_s\}, \tag{3}$$

where s is the predefined capacity of situational knowledge, which here we make equal to the size of the population N, and $\mathbf{E}_k$ is the k-th nondominated individual, with $k \in \{1, 2, \cdots, s\}$. The nondominated exemplars in situational knowledge are arranged in ascending order from the most preferable one to the least preferable, where the ordering operation is performed according to their crowding distances, as demonstrated in Section 5.4.

Since the situational knowledge functions as a leader for the other individuals in the population to follow, when it is selected to influence the population, its influence function is to push each individual towards a certain exemplar that is randomly selected from the situational repository. For the j-th decision variable x_{ij} of an arbitrary individual $\mathbf{x}_i$ in the current population, the increment/decrement Δx_{ij} is computed as follows:

$$\begin{cases} \sigma_{\mathrm{s}} = 0.1 \times (x_j^{\max} - x_j^{\min}) \\ \Delta x_{ij} = \sigma_{\mathrm{s}} \times r \end{cases}, \tag{4}$$

where $x_j^{\max}$ and $x_j^{\min}$ are the j-th elements of $\mathbf{x}^{\max}$ and $\mathbf{x}^{\min}$ (the upper and lower bounds of the decision variables given by the optimization problem), respectively, and r is a random scalar drawn from the standard normal distribution $N(0, 1)$. Then, the new individual influenced by the situational knowledge, $x_{ij}^{\text{new\_s}}$, can be produced by (5):

$$x_{ij}^{\text{new\_s}} = \begin{cases} x_{ij} + |\Delta x_{ij}|, & \text{if } x_{ij} < E_{cj} \\ x_{ij} - |\Delta x_{ij}|, & \text{if } x_{ij} > E_{cj} \\ x_{ij} + \Delta x_{ij}, & \text{if } x_{ij} = E_{cj} \end{cases}, \tag{5}$$

where E_{cj} is the j-th variable of a randomly chosen elite $\mathbf{E}_c$ from repository $\mathbf{S}$.

The influence function driven by both normative and situational knowledge is illustrated in the next subsection on normative knowledge.

When the situational knowledge is updated, all nondominated individuals in the current population space are inserted into the repository. Then, the fast nondominated sorting method (see Section 5.4) is implemented on the new archive, and the dominated ones and repeated ones are discarded. Finally, if the number of the current exemplars exceeds the maximum size N, the individuals with the smallest crowding distances (which is also explained in Section 5.4) will be omitted.

5.2.2 Normative knowledge

Normative knowledge preserves the intervals where nondominated solutions located. Specifically, it saves the upper and lower bounds of both the decision and the corresponding objective space of the nondominated individuals in the current population. In the IMOCA, the data structure and update function of the normative knowledge are inherited from the dissertation of [7]. The structure of normative knowledge in the IMOCA can be described as follows:

$$N = \{\mathbf{l},\ \mathbf{u},\ \mathbf{L},\ \mathbf{U}\}, \tag{6}$$

where $\mathbf{l} = (l_1, \cdots, l_n)$ and $\mathbf{u} = (u_1, \cdots, u_n)$ are the minimum and maximum values of each decision variable that are found in all the individuals sorted out by the acceptance function, while $\mathbf{L} = (L_1, \cdots, L_M)$ and $\mathbf{U} = (U_1, \cdots, U_M)$ are the minimum and maximum values of each objective found in all the accepted individuals.

When only normative knowledge itself is chosen to influence the population, increment/ decrement can simply be applied to individuals that are smaller/larger than the lower/ upper bounds $\mathbf{l}$ and $\mathbf{u}$ saved in the normative knowledge. Here, the increment/decrement is computed based on the interval $[\mathbf{l}, \mathbf{u}]$ using a coefficient α and random scalar r drawn from the standard normal distribution $N(0, 1)$:

$$\begin{cases} \sigma_{\mathrm{n}} = \alpha \times (u_j - l_j) \\ \Delta x_{ij} = \sigma_{\mathrm{n}} \times r \end{cases}. \tag{7}$$

Hence, the adjustment produced by the normative knowledge can be expressed by (8):

$$x_{ij}^{\text{new\_n}} = \begin{cases} x_{ij} + |\Delta x_{ij}|, & \text{if } x_{ij} < l_j \\ x_{ij} - |\Delta x_{ij}|, & \text{if } x_{ij} > u_j \\ x_{ij} + \beta \times \Delta x_{ij}, & \text{otherwise} \end{cases}. \tag{8}$$

Under the other circumstance, when the influence function of the combination of the normative knowledge and the situational knowledge is selected, we apply a normative-knowledge-based increment/decrement to the current individual if it is smaller/larger than the exemplar, which is randomly selected from the situational knowledge. For the j-th decision variable x_{ij} of an arbitrary individual $\mathbf{x}_i$ to be improved in the population, as in the case where normative knowledge is used alone, the increment/decrement is computed as follows:

$$\begin{cases} \sigma_{\mathrm{ns}} = \alpha \times (u_j - l_j) \\ \Delta x_{ij} = \sigma_{\mathrm{ns}} \times r \end{cases}, \tag{9}$$

where u_j and l_j are the j-th elements of $\mathbf{l}$ and $\mathbf{u}$, the first two vectors stored in normative knowledge, defined as the lower and upper bounds of the closed interval that contains all the accepted exemplar individuals, respectively, and r is a random scalar drawn from the standard normal distribution $N(0, 1)$. Then, the new individual generated by normative and situational knowledge can be computed as follows:

$$x_{ij}^{\text{new\_ns}} = \begin{cases} x_{ij} + |\Delta x_{ij}|, & \text{if } x_{ij} < E_{cj} \\ x_{ij} - |\Delta x_{ij}|, & \text{if } x_{ij} > E_{cj} \\ x_{ij} + \Delta x_{ij}, & \text{if } x_{ij} = E_{cj} \end{cases}, \tag{10}$$

where E_{cj} is the j-th variable of $\mathbf{E}_c$, a randomly chosen elite from the situational repository $\mathbf{S}$.

The update operation of the normative knowledge is performed in every iteration by recovering the lower and upper bounds of the decision and objective values according to the accepted individuals selected by the acceptance function.

5.2.3 Topographical knowledge

To obtain a nondominated solution set with better coverage and diversity, rather than promoting the promising regions, in the IMOCA we utilize the topographical knowledge to expand the spread of solutions in the objective space.

Taking a simple bi-objective problem as an example, topographical knowledge locates all the individuals into $nGrid \cdot nGrid$ cells and chooses the ones that accommodate on the edge of the objective space. As Fig. 2 presents below, suppose that we have partially reached the true PF (denoted by the orange dashed line) of the optimization problem of minimizing f_1 and f_2; thus, the next step is to expand the spread of the obtained individuals to cover the unsearched part of the orange dashed line. Both the blue and red triangles denote the solutions in the first rank obtained in the current iteration, among which the red ones are selected by topographical knowledge as they are located in the edge areas. Thus, in the IMOCA we sort out all the individuals located in the edge cells (cells in light blue in Fig. 2) to which adjustment is performed.

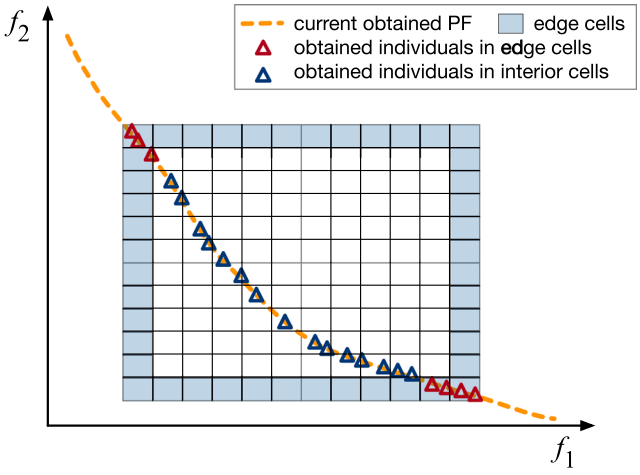


Fig. 2 Implementation of topographical knowledge in the IMOCA

The influence function of topographical knowledge is implemented to exert a certain mutation operation on the individuals "in red" to achieve better diversity among edge solutions. For each individual located in the edge cells, we randomly select $\mu_t \times n$ of its decision variables to perform fine tuning, where μ_t is a percentage parameter. Then, the adjustments to the $\mu_t \times n$ decision variables of the edge individual $\mathbf{x}_i$ are made by

$$\begin{cases} \sigma_{\mathrm{t}} = 0.1 \times (x_j^{\max} - x_j^{\min}) \\ \Delta x_{ij} = \sigma_{\mathrm{t}} \times r \end{cases}, \tag{11}$$

where $j \in \{1, 2, \cdots, n\}$. Thus, the j-th decision variable of an arbitrary individual $\mathbf{x}_i$ located in the edge cells is updated by

$$x_{ij}^{\text{new\_t}} = x_{ij} + \Delta x_{ij}. \tag{12}$$

It should be pointed out that, topographical knowledge is not updated in every generation, but only when it is chosen to influence the population to conserve computation resources. When topographical knowledge is selected, the search region spanned by the current elites in the situational knowledge is re-split into $(nGrid)^M$ grids, and each individual in the population is relocated to its own residence grid.

5.2.4 Historical knowledge

Here, we redesigned the historical knowledge as a meta-knowledge source that records the performance of the four influence functions generated by the other three knowledge sources and the mutation strategy. In other words, the historical knowledge saves the total number of surviving individuals in the situational knowledge over the past h generations produced by each influence function or the mutation strategy to manipulate the selection operation of the knowledge source to be used in generating offspring.

In the previous h generations, i.e., when the current iteration number has not reached a predefined threshold h, one of the five influence functions randomly selected to generate new individuals. Under this condition, we have

$$p_{\mathrm{n}} = p_{\mathrm{s}} = p_{\mathrm{ns}} = p_{\mathrm{t}} = p_{\mathrm{m}} = 0.2, \tag{13}$$

where p_{n}, p_{s}, p_{ns}, p_{t}, and p_{m} are the probabilities of each knowledge to be chosen.

Otherwise, the probability for each influence function to be chosen in the selection phase is then calculated according to the number of corresponding survivals recorded. For this purpose, for each of the N elites stored in the situational knowledge, through which knowledge source it was generated is also saved in the situational knowledge. Then, the probability of each knowledge to be chosen, which is based on the productivity of each knowledge in the previous iterations, can be simply calculated by

$$\begin{cases} p_{\mathrm{n}} = \dfrac{c_{\mathrm{n}}}{N} \\ p_{\mathrm{s}} = \dfrac{c_{\mathrm{s}}}{N} \\ p_{\mathrm{ns}} = \dfrac{c_{\mathrm{ns}}}{N} \\ p_{\mathrm{t}} = \dfrac{c_{\mathrm{t}}}{N} \\ p_{\mathrm{m}} = \dfrac{c_{\mathrm{m}}}{N} \end{cases}, \tag{14}$$

where c_{n}, c_{s}, c_{ns}, c_{t}, and c_{m} are the numbers of individuals generated from normative knowledge, situational knowledge, a combination of normative and situational knowledge, topographical knowledge, and the mutation strategy, respectively. Furthermore, to prevent starvation of any knowledge source, here, we introduce a parameter $\epsilon = 0.03$, and thereby, an adjustment of each probability p is made as follows:

$$p' = \frac{p + \epsilon}{1 + 5\epsilon}, \tag{15}$$

where $p \in \{p_{\mathrm{n}}, p_{\mathrm{s}}, p_{\mathrm{ns}}, p_{\mathrm{t}}, p_{\mathrm{m}}\}$, which guarantees that every knowledge source has the opportunity to produce new individuals while keeping $p'_{\mathrm{n}} + p'_{\mathrm{s}} + p'_{\mathrm{ns}} + p'_{\mathrm{t}} + p'_{\mathrm{m}} = 1$. By doing so, from the $(h+1)$-th generation, when selecting influence functions, the selector has a reference based on the performance of the influence functions in the past h generations. At the same time, even the most poorly performing influence function has chance to impact the current population.

As illustrated in (16), in the influence phase, among normative knowledge, situational knowledge, the combination of normative and situational knowledge, and the mutation scheme, one influence strategy is chosen to accomplish the evolution of the current population in each generation.

The selection is based on the performance of the influence strategies stored in the historical knowledge.

$$x_{ij}^{\text{new}} = \begin{cases} x_{ij}^{\text{new\_s}}, & \text{if situational knowledge is selected} \\ x_{ij}^{\text{new\_n}}, & \text{if normative knowledge is selected} \\ x_{ij}^{\text{new\_ns}}, & \text{if normative and situational knowledge is selected} \\ x_{ij}^{\text{new\_t}}, & \text{if topographical knowledge is selected} \\ x_{ij}^{\text{new\_m}}, & \text{if mutation scheme is selected} \end{cases}, \tag{16}$$

where $x_{ij}^{\text{new\_m}}$ denotes the new value for the j-th decision variable of the individual x_i, as illustrated in Section 5.3.

To update the historical knowledge, simply add the number of the current survivals in the situational knowledge generated by each influence function to its corresponding counter.

5.3 A mutation scheme

As illustrated previously, the mutation scheme is also among the influence strategies used to alter the individuals living in a population. The mutation operation used here is analogous to the one in the GA and is described as follows. First, for each individual $\mathbf{x}_i$ in the current population ($i \in \{1, 2, \cdots, N\}$), randomly select n_μ decision variables from the total n ones to apply the mutation operation to, where n_μ is calculated by

$$n_\mu = n \times \mu_m, \tag{17}$$

where n is the number of decision variables and μ_m is the mutation percentage. Then, use the equation given below to obtain σ_m, i.e., the mutation step size on the j-th decision variable:

$$\sigma_m = 0.1 \times (x_j^{\max} - x_j^{\min}). \tag{18}$$

Thereafter, execute the mutation operation on the j-th variable of the individual $\mathbf{x}_i$:

$$x_{ij}^{\text{new\_m}} = x_{ij} + \sigma_m \times r, \tag{19}$$

where r is a random scalar drawn from the standard normal distribution $N(0, 1)$. Finally, if any dimension of the mutated individual exceeds the corresponding constraints, set it to the boundary value.

Here, it should be noted that the topographical knowledge and the mutation scheme use the same form of equations to produce offspring, but the key point here is the selection of individuals to be operated on. In other words, the difference between the two strategies lies in which kind of individual is selected for variation. The topographical knowledge is used to find promising solutions and broaden the obtained front, the influence function of which is therefore applied to the individuals located on the edge of the objective space, while the mutation scheme aims to carry out fine tuning in the late stage of search, which is applied to every individual in the current population. The results prove that these two influence functions are superior in in producing nondominated solutions at different stages of evolution (see Fig. 4).

In the IMOCA, the mutation operation is not applied to the population in every generation. Instead, it plays the same role as that of the other influence functions of knowledge sources. That is, the use of mutation depends on its performance in terms of productivity in the preceding h generations stored in the historical knowledge. As the simulation results (shown in Section 7) indicate, the evolution process makes full use of the mutation operation in the last stage of search.

5.4 Pareto dominance and the fast nondominated sorting approach

For MOOPs, since there is more than one decision value to be considered, it is not as simple as for SOOPs to compare the performances of different solutions with less computational effort. Hence, adopting an efficient sorting strategy among solutions is one of the significant aspects of an MOOP optimization algorithm. To address this problem, we utilize the fast nondominated sorting approach together with the notion of crowding distance inspired from the NSGA-II. For a detailed explanation of the two techniques, refer to [17].

5.5 Elite preservation

In the competition of individuals in a population space, because newborn individuals are always merged into the parent population once generated, no excellent parent individuals are omitted. On the other hand, during the update procedure of the situational knowledge, the nondominated solutions sorted out from the current population are in competition with all the elites previously saved in the repository to prevent the best solutions in the first PF from dying out. Furthermore, when the algorithm is terminated, the repository of the situational knowledge is saved as the final solutions to the problem at hand. In general, the IMOCA avoids any loss of the well-performing individuals that have emerged in the population space.

5.6 Communication channel

5.6.1 Acceptance function

The acceptance function is used to accept a proportion of high-performing individuals to update the knowledge in the belief space. Here, we use a very simple acceptance function: 35% of the individuals in the current population with the best nondominated ranks are accepted.

5.6.2 Influence function

The influence functions of situational, normative and topographical knowledge have already been described in detail in Section 5.2. The influence function generated from the mutation scheme is demonstrated in Section 5.3.

5.7 Implementation framework of the IMOCA

Within the EP framework mentioned above, the IMOCA works as follows:

INPUT:

- a multiobjective problem with n decision variables and M objectives;
- a stopping criterion;
- the population size N.

OUTPUT: individuals stored in the current situational knowledge.

Step 1) Initialization:

Step 1.1) Set generation counter $t = 0$.

Step 1.2) Generate N individuals $\mathbf{x}_1, \cdots, \mathbf{x}_N$ randomly within the range of decision variables as the initial population, where N is the size of the population. Here, each individual $\mathbf{x}_i = (x_{i1}, \cdots, x_{in})^T$, with $i \in \{1, 2, \cdots, N\}$, where n is the number of decision variables.

Step 1.3) Evaluate the N individuals in the initial population by the objective functions and obtain N corresponding objective vectors $\mathbf{y}_1, \cdots, \mathbf{y}_N$, where $\mathbf{y}_i = (y_{i1}, \cdots, y_{iM})^T$, with $i \in \{1, 2, \cdots, N\}$, and M is the number of objectives.

Step 1.4) Initialize the knowledge sources in the belief space according to the knowledge structure of each source, which is described in Section 5.2.

Step 2) Population Space Evolution:

Step 2.1) If the generation counter t has not exceeded h, randomly choose one influence function (among normative knowledge, situational knowledge, normative and situational knowledge, topographical knowledge, and the mutation scheme) to influence the evolution of the current population. Otherwise, carry out the selection according to the former performance of the five influence functions stored in the historical knowledge. If the topographical knowledge is chosen, update it before utilizing.

Step 2.2) For the N individuals $\mathbf{x}_i$ with $i \in \{1, 2, \cdots, N\}$, generate N offspring $\mathbf{x}_{N+1}, \cdots, \mathbf{x}_{2N}$ using the chosen influence function. Then, evaluate the new individuals and merge the old and the new ones; thus, we have $2 \times N$ individuals in total.

Step 2.3) Conduct the nondominated sorting method on the entire $2 \times N$ population and discard the last N individuals with larger rank values. Accordingly, the best N ones survive as the new population of the next generation.

Step 2.4) Set $t = t + 1$.

Step 3) Belief Space Update: Update the 3 knowledge sources (except topographical knowledge) in the belief space using the accepted individuals in the new population.

Step 3.1) Update situational knowledge.

Step 3.2) Update normative knowledge.

Step 3.3) Update historical knowledge according to the productivities of the other four influence functions in the last generation.

Step 4) Stopping Criterion: If the stopping criterion, i.e., the generation counter t exceeds the maximum amount, is satisfied, stop and output the current individuals stored in the situational knowledge as the results of the optimization. Otherwise, go to **Step 2)**.

The implementation flowchart of the algorithm is shown in Fig. 3.

6 Experimentation and related configurations

6.1 Multiobjective continuous test suites

For simulation, two classic benchmark suites WFG [24] and MaF [6] are used in the comparison between the IMOCA and 5 classic multiobjective algorithms and 3 recently proposed MOEAs. It should be noted that since the problems of WFG1, WFG2 and WFG9 are used in the MaF suites as MaF10, MaF11 and MaF12 without any change, in the experiment, we omit the redundant ones in the WFG suites. All of the 21 test instances are tri-objective problems and involve minimization of the objectives, including problems with a variety of characteristics.

6.2 Parameter configuration of the IMOCA for simulation

The parameters used in each knowledge source of the IMOCA are configured as shown in Table 1.

First, we tested a set of values for the parameters α and β, which were used in the influence function of normative knowledge and the influence function of normative and situational knowledge in (7), (8) and (9). As shown in Tables 2, 3 and 4, the performance of the IMOCA when adopting six different combinations of the parameters α and β was evaluated by the MaF functions. In the three tables, the best indicator values are marked in bold. The results reveal that when $\alpha = 0.3$ and $\beta = 0.5$, the best performance of the IMOCA could be achieved compared to the other combinations.

Then, for the influence function of topographical knowledge, the segmentation number on each dimension $nGrid$ is specified as 10 in accordance with the literature [38]. For

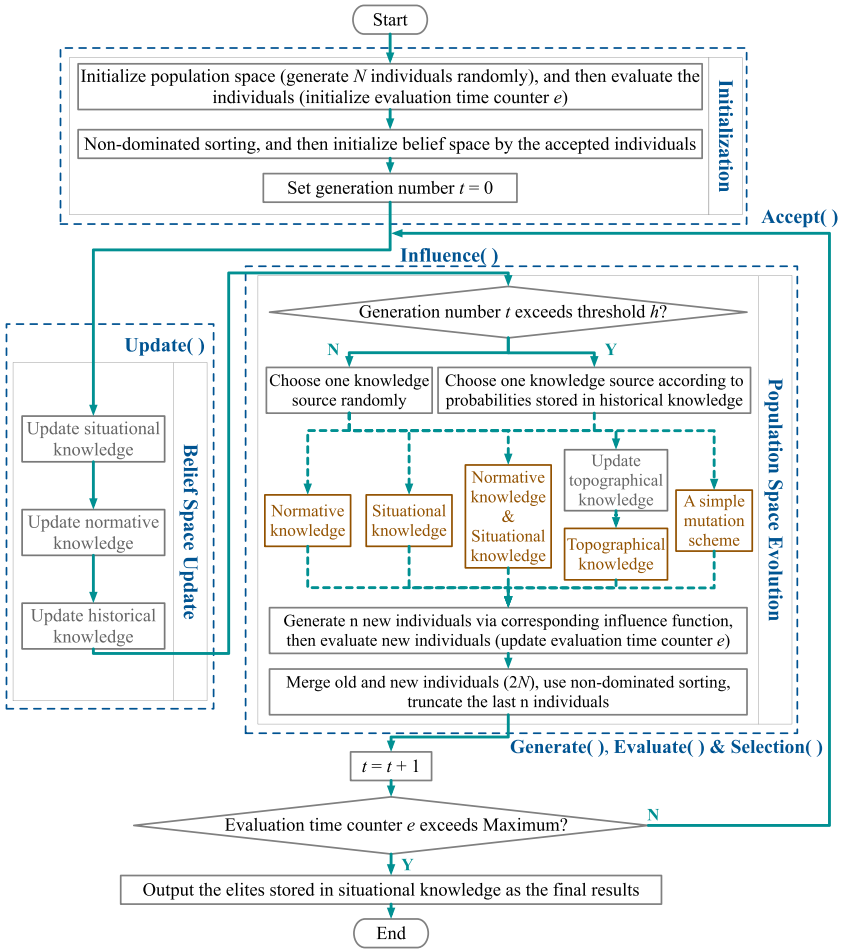


Fig. 3 Framework of the IMOCA

the parameter μ_t, which decides the number of decision variables to be adjusted, it is specified as 0.02 so that only one decision variable is tuned for problems with less than 50 decision variables; for problems with (50, 100] decision variables, two decision variables need to be changed. For the same reason, the value 0.02 is also assigned to the parameter μ_m in the mutation scheme. Last, for historical knowledge, we assign h as 50 so that at every iteration, the productivity of each influence function in the last 50 iterations is a reference for selecting the influence function to be used in the current iteration. The parameter ϵ is set to avoid starvation of any knowledge source.

6.3 Algorithms for comparison

In this paper, we choose eight multiobjective algorithms, including five classic multiobjective algorithms (NSGA-II,

Table 1 Parameter configuration in the IMOCA

The influence function the parameters are used in	Parameter names	Values used in the IMOCA
Influence function of normative knowledge	α (see (7))	0.3
	β (see (8))	0.5
Influence function of situational and normative knowledge	α (see (9))	0.3
Influence function of topographical knowledge	$nGrid$	10
	μ_t	0.02
Influence function of historical knowledge	h	50
	ϵ (see (15))	0.03
Mutation scheme	μ_m (see (17))	0.02

Table 2 Medians of the inverted generational distance (IGD) of 20 runs under different settings of the parameters α and β

	$\alpha = 0.3$ $\beta = 0.3$	$\alpha = 0.3$ $\beta = 0.5$	$\alpha = 0.3$ $\beta = 0.7$	$\alpha = 0.5$ $\beta = 0.3$	$\alpha = 0.5$ $\beta = 0.5$	$\alpha = 0.7$ $\beta = 0.3$
MaF1	0.073218	0.069127	0.068027	0.069014	**0.067049**	0.069049
MaF2	0.596158	0.596457	0.596310	0.595035	**0.594968**	0.595377
MaF3	0.347454	0.325043	**0.318657**	0.400711	0.459670	0.428111
MaF4	3.736833	3.810884	3.762974	3.730497	**3.717573**	3.748048
MaF5	4.014352	3.999706	4.024609	3.994386	3.995444	**3.994142**
MaF6	0.179881	0.179307	**0.179071**	0.179427	0.179267	0.179529
MaF7	**2.957703**	2.964635	2.963736	2.964219	2.965030	2.961734
MaF8	**0.620189**	0.628236	0.624343	0.623818	0.625612	0.639203
MaF9	5.435639	**0.754918**	0.912336	1.254550	2.852567	1.381800
MaF10	1.262508	1.260272	1.267940	**1.259995**	1.262095	1.268234
MaF11	**2.254036**	2.256029	2.259387	2.257765	2.259533	2.258941
MaF12	3.147283	3.15109	3.147852	3.147987	3.147441	**3.144435**
MaF13	0.094895	0.094385	0.095658	0.095362	**0.093508**	0.093514
MaF14	0.441295	**0.430414**	0.434587	0.441788	0.433716	0.446452
MaF15	0.128081	0.127588	**0.127100**	0.128951	0.127435	0.129540
$\overline{IGD}$	1.685968	**1.376539**	1.385506	1.409567	1.518727	1.422541
Rank of $\overline{IGD}$	6	**1**	2	4	5	3

PESA-II, SPEA2, MOPSO and MOEA/D) and three well-performing algorithms (NSLS, MOEA/IGD-NS and BCE-IBEA) selected from more recent research, to compare with the IMOCA in terms of the performance.

Referring to the recommendations regarding the experimental setup in the literature [6], for all nine algorithms, the population size was set to 25 times the number of objectives of the problems, and for the IMOCA, PESA-II, SPEA2 and MOEA/D, which are algorithms with external repositories, the size of the repositories was set equal to the population size. The other parameter settings of the 8 algorithms are as follows.

Table 3 Medians of the hypervolume (HV) of 20 runs under different settings of the parameters α and β

	$\alpha = 0.3$ $\beta = 0.3$	$\alpha = 0.3$ $\beta = 0.5$	$\alpha = 0.3$ $\beta = 0.7$	$\alpha = 0.5$ $\beta = 0.3$	$\alpha = 0.5$ $\beta = 0.5$	$\alpha = 0.7$ $\beta = 0.3$
MaF1	1.099999	1.099999	1.099999	1.099999	1.099999	1.099999
MaF2	0.712305	**0.787706**	0.724117	0.723971	0.713761	0.744828
MaF3	1.099998	1.099998	**1.099998**	1.099998	1.099998	1.099998
MaF4	2.199998	2.199998	2.199998	2.199998	2.199998	2.199998
MaF5	8.800000	8.800000	8.800000	8.800000	8.800000	8.800000
MaF6	0.777817	0.777817	0.777817	0.777817	0.777817	0.777817
MaF7	0.945341	0.945341	0.945341	0.945341	0.945341	0.945341
MaF8	1.665953	**1.715514**	1.662875	1.707283	1.700960	1.702080
MaF9	1.572484	**1.590801**	1.569413	1.575329	1.550658	1.568517
MaF10	1.376749	1.363852	**1.398516**	1.362191	1.370329	1.397717
MaF11	2.190963	**2.191980**	2.190273	2.187861	2.188651	2.190273
MaF12	**2.181592**	2.178299	2.165982	2.169219	2.170965	2.162307
MaF13	1.100000	1.100000	1.100000	1.100000	1.100000	1.100000
MaF14	1.100000	1.100000	1.100000	1.100000	1.100000	1.100000
MaF15	1.099999	1.099999	1.099999	1.099999	1.099999	1.099999
$\overline{HV}$	1.861546	**1.870087**	1.862289	1.863267	1.861232	1.865925
Rank of $\overline{HV}$	5	**1**	4	3	6	2

Table 4 Medians of the unary ϵ-indicator ($I_{\epsilon 1}$) of 20 runs under different settings of the parameters α and β

	$\alpha = 0.3$ $\beta = 0.3$	$\alpha = 0.3$ $\beta = 0.5$	$\alpha = 0.3$ $\beta = 0.7$	$\alpha = 0.5$ $\beta = 0.3$	$\alpha = 0.5$ $\beta = 0.5$	$\alpha = 0.7$ $\beta = 0.3$
MaF1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
MaF2	0.066895	**0.008506**	0.055083	0.055229	0.065439	0.034372
MaF3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
MaF4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
MaF5	0.000002	0.000002	0.000002	0.000002	0.000002	0.000002
MaF6	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
MaF7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
MaF8	0.222136	**0.172575**	0.225214	0.180806	0.187129	0.186009
MaF9	0.055197	**0.036880**	0.058267	0.052352	0.077022	0.059163
MaF10	0.817039	0.829937	**0.795272**	0.831597	0.823459	0.796071
MaF11	0.009037	**0.008020**	0.009727	0.012139	0.011349	0.009727
MaF12	**0.018406**	0.021699	0.034016	0.030779	0.029033	0.037691
MaF13	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
MaF14	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
MaF15	0.732228	**0.000000**	0.540390	0.309080	0.628766	0.561862
$\overline{I_{\epsilon 1}}$	0.079248	**0.071841**	0.078506	0.077527	0.079562	0.074869
Rank of $\overline{I_{\epsilon 1}}$	5	**1**	4	3	6	2

- NSGA-II: crossover rate $P_c = 0.7$, mutation rate $P_m = 0.4$, mutation step size $\sigma = 0.1$;
- PESA-II: number of grids per dimension $nGrid = 7$, grid inflation factor $IF = 0.1$, crossover rate $P_c = 0.5$, crossover parameter $\gamma = 0.15$, mutation rate $P_m = 0.5$, mutation step size $\sigma = 0.3$;
- SPEA2: crossover rate $P_c = 0.7$, crossover parameter $\gamma = 0.1$, mutation rate $P_m = 0.3$, mutation step size $\sigma = 0.2$;
- MOPSO: inertia weight $w = 0.4$, inertia weight damping rate $wdamp = 0.99$, personal learning coefficient $c_1 = 1$, global learning coefficient $c_2 = 2$, number of grids per dimension $nGrid = 30$, inflation rate $\alpha = 0.1$, leader selection pressure $\beta = 2$, deletion selection pressure $\gamma = 2$, mutation rate $P_m = 0.5$;
- MOEA/D: crossover rate $P_c = 0.5$, number of neighbors: 15% of the population but no larger than 15;
- NSLS: mean value of the Gaussian distribution $\mu = 0.5$, mutation strength $\sigma = 0.1$;
- MOEA/IGD-NS: the archive size NA is set to 5 times the population size;
- BCE-IBEA: the scaling factor $\kappa = 0.05$.

6.4 Experimental configurations

The IMOCA and the other 8 algorithms are written in MATLAB scripts and the testing experiments were implemented in MATLAB R2018a on a 3.1 GHz Intel Core i7 processor under macOS Mojave 10.14.4. The source codes of the NSGA-II, PESA2, SPEA-II, MOPSO and MOEA/D are available from the EMOO repository [10], and the implementations of the NSLS, MOEA/IGD-NS, and BCE-IBEA are carried out on the PlatEMO [43].

All the algorithms were run 20 times independently for each benchmark problem. Additionally, to conduct a fair comparison among the nine methods, instead of adopting a maximum number of iterations as the termination condition, we terminate each algorithm once the calculation times of the objective functions reach $\max(100000,\ 10000 \times n)$.

6.5 Performance metrics

Six performance metrics are adopted in this paper to evaluate the performances of all 9 algorithms in terms of both diversity and convergence.

1. Set Coverage ($C(A,\ B)$) [16]. It calculates the proportion of solutions in B, which are weakly dominated by the solutions in A:

$$C(A, B) = \frac{|\{b \in B \mid \exists\, a \in A :\ a \preceq b\}|}{|B|}.$$

The metric value $C(A, B) = 1$ means that all members of B are weakly dominated by A. On the other hand, $C(A, B) = 0$ means that no member of B is weakly dominated by A. It can be noted that $C(A, B)$ is not necessarily equal to $1 - C(B, A)$ because the domination operator is not a symmetric operator. It is thus necessary to calculate both $C(A, B)$ and $C(B, A)$ to investigate how many solutions of A are covered by B and vice versa.

2. Inverted Generational Distance (IGD). This measure was proposed by [52]. It is similar to the generational distance (GD) and formulated as follows:

$$IGD = \frac{\sum_{i=1}^{n} d_i^2}{n},$$

where n is the number of true Pareto optimal solutions and d_i indicates the Euclidean distance between the i-th true Pareto optimal solution and the closest obtained Pareto optimal solutions in the reference set. The Euclidean distance between obtained solutions and the reference set is different here. For the IGD, the Euclidean distance is calculated for every true solution with respect to its nearest obtained Pareto optimal solutions in the objective space.

3. Hypervolume (HV) [48]. This metric calculates the volume (in the objective space) covered by members of the solution set A. For each solution $\mathbf{x} \in A$, a hypercube v_i is constructed with a reference point W and the solution itself as two diagonal corners of the hypercube. Thereafter, the hypervolume of all the hypercubes is calculated as

$$HV = volume\left(\cup_{i=1}^{|A|} v_i\right).$$

Obviously, an algorithm with a larger HV is desirable.

4. Unary ϵ-indicator ($I_{\epsilon 1}$) [52]. This indicator is based on the binary ϵ-indicator, which is defined as

$$I_\epsilon(A,\ B) = \inf_{\epsilon \in \mathbb{R}} \{\forall\, \mathbf{z}^2 \in B,\ \exists\, \mathbf{z}^1 \succeq_\epsilon \mathbf{z}^2\}$$

for any two approximation sets $A,\ B \in \Omega$. For the unary ϵ-indicator, substitute the set B with a reference set of points:

$$I_{\epsilon 1}(A) = I_\epsilon(A,\ P),$$

where P is the true PF set. If P is unknown, any reference set R can be used instead.

5. Spacing (S) [16]. This is a relative distance measure between consecutive solutions in the obtained nondominated set:

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

where $d_i = \min_{k \in Q \wedge k \neq i} \sum_{m=1}^{M} |f_m^i - f_m^k|$ and $\bar{d} = \sum_{i=1}^{|Q|} d_i / |Q|$. Obviously the metric S calculates the standard deviations of different d_i values. Therefore, an algorithm finding a set of a nondominated solutions having smaller spacing S is better.

6. Spread (Δ) [16]. This metric reflects the extent of spread:

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{m=1}^{|Q|} |d_i - \bar{d}|}{\sum_{m=1}^{M} d_m^e + |Q|\bar{d}},$$

where d_i can be any distance measure between neighboring solutions, $\bar{d}$ is the mean value of these distance measures, and d_m^e is the distance between the extreme solutions of P^* and Q in the m-th objective function. For an ideal distribution of obtained solutions, $\Delta = 0$.

The set coverage metric is a binary measure, while the inverted generational distance, hypervolume and unary ϵ-indicator are unary Pareto-compliant indicators [19]. The other two indicators, spacing and spread, are unary but non-Pareto-compliant metrics evaluating the distribution and diversity, respectively.

7 Results and discussions

7.1 Performance of knowledge sources in the IMOCA

To examine the performances of the knowledge sources used in the IMOCA, we examine the elite solutions saved in the situational knowledge in different search phases in a random run when solving a basic test problem ZDT1 from the classic multiobjective test suite ZDT [50].

We use a population size of 50 and take a number of objective functions $M = 2$ and number of decision variables $n = 30$ for ZDT1. Figure 4 plots the minimum values of the sum of all the objective functions among all the solutions saved in the current situational knowledge. The abscissa is based on the logarithm of the iteration so that the performance of each knowledge source in the early stage of the iteration can be seen clearly. It should be stated here that in this experiment, we set the terminal condition as a maximum generation of 300 because for ZDT1, convergence can be achieved by the IMOCA before the 300th iteration; in the competition experiments between the IMOCA and the other chosen algorithms, to guarantee that most of the algorithms have enough chances to search for solutions, we terminate the algorithms according to the calculation times of the objective functions.

To better analyze how each knowledge source guides the search process, in Fig. 4, the knowledge sources that produce the minimum sum of objective functions are indicated in different colors and shapes. It is obvious from the figure that all five influence functions made efforts to find new promising individuals. At the 5th iteration, topographical knowledge first found promising solutions different from the initial ones, which were generated randomly in the first iteration. As the optimization process progressed, the five influence functions took turns generating individuals with small values of $\sum_{m=1}^{M} f_m(\mathbf{x})$. From the 30th iteration, the mutation scheme and the influence functions of topographical knowledge began to take up a larger portion of the

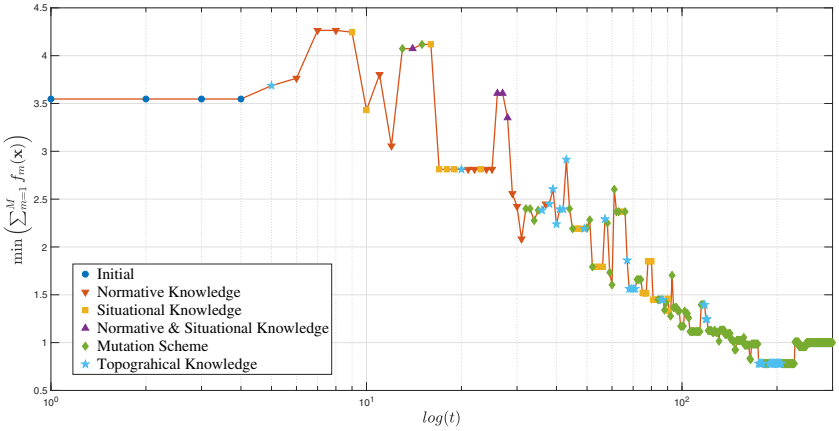


Fig. 4 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing ZDT1 using the IMOCA (iteration $t \in [1, 300]$)

producers of promising individuals. From the 43th to the 90th iterations, situational knowledge also played a significant role in generating new elite solutions. Similar performances of the five influence functions were also observed when solving the 15 MaF instances (see Appendix).

In [3], the productivity of different knowledge sources was also observed by plotting the minimum value of $\sum_{m=1}^{M} f_m(\mathbf{x})$ for all the individuals $\mathbf{x}$ in the population in each iteration. By comparing the figures given in both [3] and our paper, the implementation of topographical knowledge in [3] produced very few promising individuals, while that in the IMOCA contributed to finding better solutions, especially in the middle and late stages of evolution.

In a nutshell, it can be concluded that the effectiveness of all the adopted influence functions is verified. In the late stage of evolution, the mutation scheme seems to be more productive in promoting the quality of a population. It can be also noted that the productivity of the adapted topographical knowledge in the proposed IMOCA is much better than that of the topographical knowledge used in the MOCA in [3].

7.2 Results on the test functions

7.2.1 Obtained nondominated solutions

According to the simulation experimentation, in which the obtained Pareto optimal solutions of the 9 algorithms in 20 runs are compared, the IMOCA is able to converge to the true PFs of most of the 21 test problems. Here, three typical simulation results of the test problems MaF4, MaF14 and WFG6 are given in Fig. 5a, b and c, respectively. In the three figures, the true PFs of these three functions are plotted as gray surfaces, and the nondominated solutions obtained in the 20 runs are represented by small circles in different colors (the solutions of each run correspond to each of the different colors).

MaF4 is a concave, multimodal, and badly scaled optimization problem with no single optimal solution in any subset of the objectives. Its PF surface is a part of a sphere that intersects at the following coordinates: (0, 4, 8), (2, 4, 0), and (2, 0, 8). From Fig. 5a, it can be observed that hardly any of the solutions found by the algorithms occurred on or near the true Pareto front in the 20 runs, but the axis tick values in each subfigure in Fig. 5a indicate that the solution set obtained by the IMOCA is the one nearest the true Pareto front of MaF4. This reveals that the IMOCA is more capable of dealing with badly scaled PFs.

The test problem of MaF14 is linear but with a complicated fitness landscape. Its decision variables are nonuniformly correlated with different objectives, and the decision variables are partially separable. Similarly, from the axis labels in Fig. 5b, it can be inferred that only the proposed algorithm and NSLS approached the Pareto front of MaF14, which is a triangle plane spanned by the points [0, 1, 0], [1, 0, 0] and [0, 0, 1]. Compared to those of the NSLS, the solutions obtained by the IMOCA seemed to cover a larger portion of the Pareto front and were relatively evenly distributed, as there were at least four sets of solutions that reached the upper part of the Pareto plane instead of falling into a single straight line in the xy-plane.

WFG6 is a concave problem with nonseparable reduction. The obtained solutions of the 9 algorithms on WFG6 are depicted in Fig. 5c, where almost all 9 algorithms found solutions near the Pareto sphere surface in the 20 runs. Under this circumstance, the distribution of the solutions becomes notable when comparing the performances of the multiobjective algorithms. According to the figure, the solutions obtained by the IMOCA covered most of the Pareto front without any omission, while there was always more or less space without coverage in the cases of the other 8 algorithms.

7.2.2 Statistical results on the pareto-compliant metrics

To quantitatively observe the quality of the solutions obtained by the proposed IMOCA, the median values of the three

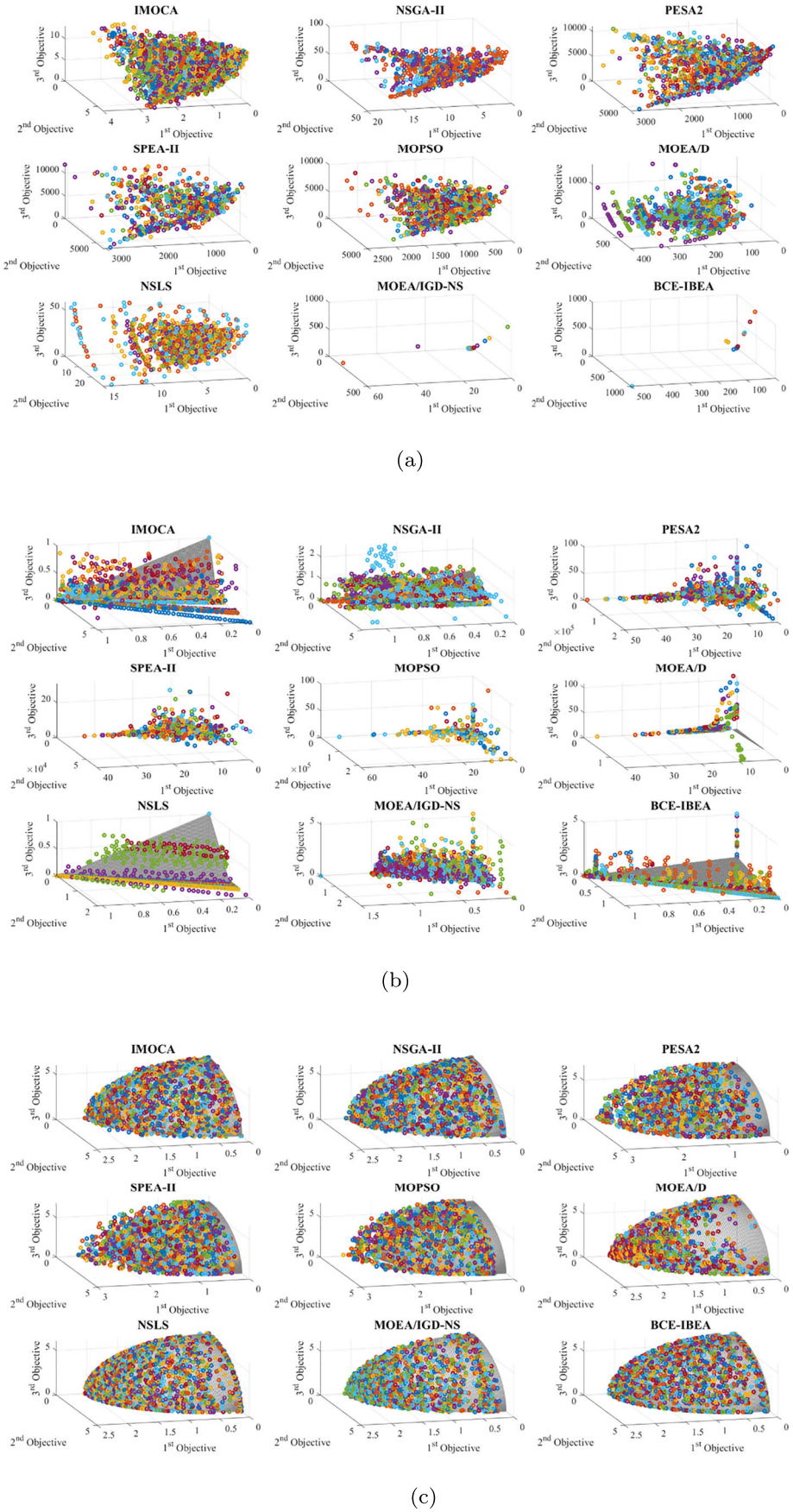


Fig. 5 Non-dominated solutions obtained by 9 algorithms in 20 independent runs in the objective space of MaF4, MaF14 and WFG6. **a** Non-dominated solutions obtained by 9 algorithms in 20 independent runs on MaF4. **b** Non-dominated solutions obtained by 9 algorithms in 20 independent runs on MaF14. **c** Non-dominated solutions obtained by 9 algorithms in 20 independent runs on WFG6

Pareto-compliant indicators (IGD, HV and unary ϵ-indicator) for the 21 test problems over 20 runs were computed, as shown in Tables 5, 6 and 7, respectively. The Friedman test was also performed on the results, as recommended in [18].

In Table 5, the result of the Friedman test reflected the IGD values of the 9 algorithms on the 21 test problems from an overall perspective, and the proposed IMOCA seemed to have an average performance on this metric. However, the p-value on the Friedman test was larger than 0.01 (but still smaller than 0.05), which indicated that there was not an explicit significance in terms of the difference between the IGD values of the 9 algorithms. Moreover, if we consider the Friedman mean rank, there was not a large difference between the rank values of the first five algorithms, while the rank differences in Tables 6 and 7 were considerably large. This occurred because the algorithms performed disparately in terms of the IGD for the different test problems. For example, the IMOCA also had a better IGD performance on several problems, such as MaF3, MaF5, and MaF10, than that of the first-ranked algorithm NSLS.

All of three metrics could measure both the diversity and convergence of an obtained nondominated set in a sense. Specifically, the IGD indicator calculates the average distance from the uniformly distributed points along the PF to the obtained approximation set; HV calculates the area covered by the approximation set with respect to a set of predefined reference points; and the unary ϵ-indicator calculates how far the obtained set should move to cover every part of the true PF. If an algorithm has a superior performance in terms of the HV and unary ϵ-indicator but an average performance on the IGD indicator, the reason may be the uneven distribution of the obtained nondominated set.

Therefore, combined with the results in Tables 6 and 7, which show that the IMOCA outperformed its competitors in terms of the HV and unary ϵ-indicator with statistical significance, it could be concluded that the solutions obtained by the IMOCA have a wider distribution along the whole PF with better diversity but are distributed somewhat unevenly on some of the test problems. The superiority of the proposed algorithm over the other ones may be generally attributed to its full use of different knowledge

Table 5 Medians of the inverted generational distance (IGD) and the ranks achieved by the Friedman test (A: Algorithms, P: Problems)

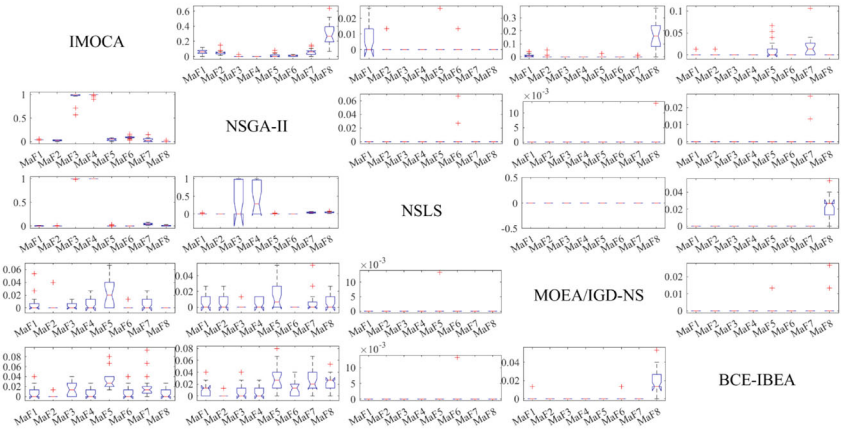
IGD A / P	IMOCA	NSGA-II	PESA2	SPEA-II	MOPSO	MOEA/D	NSLS	MOEA/ IGD-NS	BCE- IBEA
MaF1	0.069127	0.059033	0.117887	0.095604	0.162825	0.076905	0.048571	0.044996	0.048476
MaF2	0.596457	0.589407	0.631322	0.618410	0.647875	0.594880	0.586274	0.591409	0.586845
MaF3	0.325043	15.22894	149661.8	46156.26	182644.9	52661.12	24.77425	0.050707	0.045855
MaF4	3.810884	1.966237	152.3839	43.95356	195.7980	65.30562	2.045685	4.122527	4.087418
MaF5	3.999706	4.056822	3.137925	3.423809	3.673384	4.070455	4.000472	3.999068	3.995224
MaF6	0.179307	0.179086	0.313292	0.186830	0.190383	0.182843	0.178456	0.178496	0.178222
MaF7	2.964635	2.964138	1.343458	1.978762	2.810360	1.923563	2.962115	2.975142	2.963811
MaF8	0.628236	0.713789	20.19838	0.777988	4.114733	153.9574	0.674722	0.710802	0.684543
MaF9	0.754918	0.417066	0.621625	0.381577	0.685884	0.329862	0.283492	0.312261	0.299580
MaF10	1.260272	1.322007	1.234122	1.234730	1.235308	1.422659	1.504999	1.505942	1.504176
MaF11	2.256029	2.241249	2.049511	2.106867	2.100148	2.454968	2.266790	2.295643	2.285038
MaF12	3.151090	3.158409	2.894917	2.948502	2.957855	3.180359	3.139051	3.164418	3.169480
MaF13	0.094385	0.144658	0.503394	0.243494	0.303238	0.115139	0.081957	0.167277	0.135415
MaF14	0.430414	0.436481	3.735738	1.884243	1.088949	5.728214	0.328652	0.274863	0.362551
MaF15	0.127588	0.197353	1.924609	0.742012	3.659655	0.399609	0.092445	0.105716	0.193129
WFG3	1.496902	1.506820	1.294940	1.370189	1.300587	1.506385	1.491000	1.498191	1.504487
WFG4	3.156928	3.152629	3.023305	3.084248	3.037741	3.175135	3.174921	3.168637	3.171670
WFG5	3.144812	3.160057	3.044761	3.073862	3.078044	3.170920	3.158484	3.158342	3.156399
WFG6	3.162833	3.147130	2.922245	2.981786	3.009032	3.169542	3.167762	3.149022	3.144736
WFG7	3.128845	3.169453	2.985589	3.047090	2.987953	3.188021	3.120353	3.166467	3.172101
WFG8	3.124054	3.120201	2.949690	3.004771	2.960251	3.153578	3.128565	3.127279	3.123302
Friedman mean rank	4.8095	5.1905	4.3810	4.4286	5.2857	7.1429	4.2381	5.0476	4.4762
final rank	5	7	2	3	8	9	1	6	4
p-value	0.0238								

Table 6 Medians of the hypervolume (HV) and the ranks achieved by the Friedman test (A: Algorithms, P: Problems)

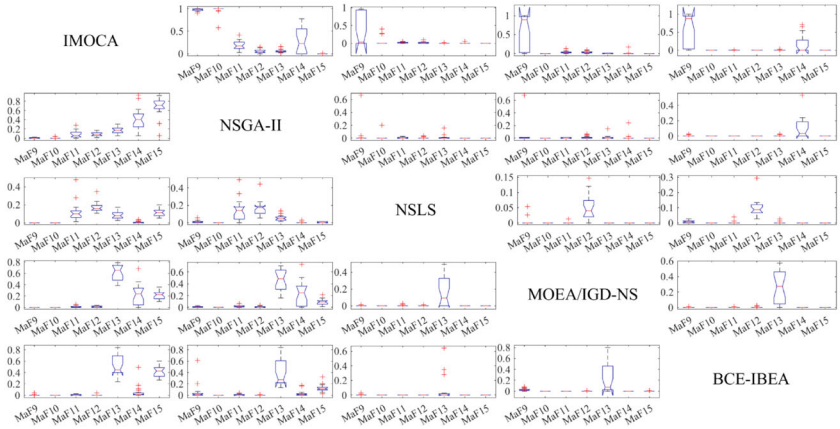
HV \ A / P	IMOCA	NSGA-II	PESA2	SPEA-II	MOPSO	MOEA/D	NSLS	MOEA/ IGD-NS	BCE-IBEA
MaF1	1.099999	1.099999	0.826206	0.689619	0.649821	0.699533	0.833800	0.762811	0.620982
MaF2	0.787706	0.787705	0.571813	0.580768	0.551649	0.551469	0.679991	0.565094	0.551822
MaF3	1.099998	1.099998	0.000000	0.000000	0.000000	0.900368	1.099998	1.070959	1.099011
MaF4	2.199998	2.199998	0.000000	0.000000	0.000000	0.000000	0.936951	1.537858	1.686619
MaF5	8.800000	8.800000	8.800000	8.799999	8.800000	8.699051	8.800000	8.436382	8.246617
MaF6	0.777817	0.777817	0.762108	0.441606	0.505800	0.322903	0.452516	0.605385	0.462596
MaF7	0.945341	0.945341	0.696127	0.648013	0.936370	0.086044	0.945341	0.721735	0.811703
MaF8	1.715514	1.814192	0.000000	1.138757	0.000000	0.000000	1.892087	1.324838	1.330587
MaF9	1.590801	1.627746	1.358045	1.454144	0.000000	1.411091	1.627742	1.555787	1.578290
MaF10	1.363852	1.642272	1.174315	1.172509	1.204704	1.640279	2.193788	2.014984	2.101533
MaF11	2.191980	2.190534	1.969394	2.026187	1.861817	1.877204	1.948194	2.053898	2.085841
MaF12	2.178299	2.170024	1.353768	1.300132	1.291971	1.215577	2.095685	1.937484	1.405935
MaF13	1.100000	1.100000	0.997004	0.818549	0.878926	0.961643	0.945635	0.859009	1.084175
MaF14	1.100000	1.100000	1.100000	1.050915	1.100000	1.100000	1.100000	1.099846	1.100000
MaF15	1.099999	1.099999	0.920223	0.685776	0.701804	0.833055	0.951467	0.987424	1.081947
WFG3	2.174671	2.185114	1.393022	1.496392	1.118911	1.345899	1.312526	1.866991	1.404859
WFG4	2.199350	2.196763	1.530188	1.569514	1.824897	1.207731	1.774998	1.891940	2.097426
WFG5	2.149639	2.149900	1.277216	1.347096	1.844558	1.724509	2.137688	1.551251	1.706306
WFG6	2.189382	2.165826	1.149088	1.371485	1.617050	1.561608	1.731286	1.919685	1.649564
WFG7	2.198887	2.197992	1.294846	1.289364	1.426695	1.428825	2.102357	2.023960	1.768640
WFG8	2.197502	2.188655	1.137852	1.290701	1.171363	1.316274	1.496490	1.902946	1.363954
Friedman mean rank	8.0714	8.0238	3.4048	2.7857	3.2381	3.0238	6.1190	5.0952	5.2381
final rank	1	2	6	9	7	8	3	5	4
p-value	4.2758E-18								

Table 7 Medians of the unary ϵ-indicator ($I_{\epsilon 1}$) and the ranks achieved by the Friedman test (A: Algorithms, P: Problems)

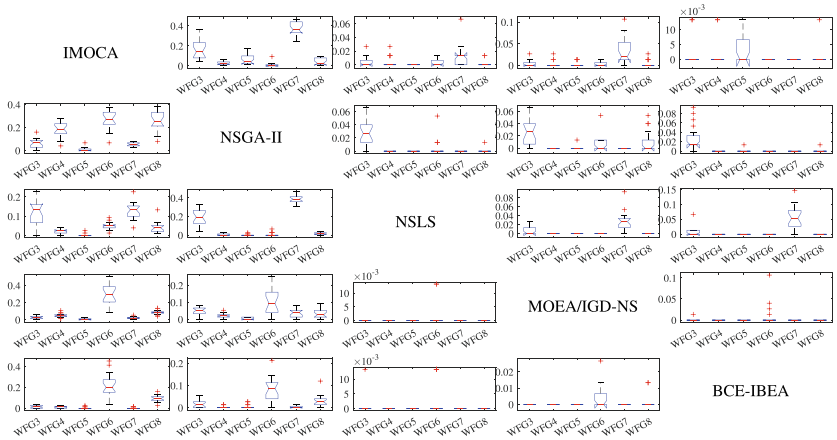
$I_{\epsilon 1}$ A / P	IMOCA	NSGA-II	PESA2	SPEA-II	MOPSO	MOEA/D	NSLS	MOEA/IGD-NS	BCE-IBEA
MaF1	0.000000	0.000000	0.273793	0.410380	0.450178	0.400466	0.266198	0.337188	0.479017
MaF2	0.008506	0.008506	0.207387	0.198432	0.227551	0.227731	0.099209	0.214106	0.227378
MaF3	0.000000	0.000000	11441.74	5089.625	334.9264	0.199630	0.000000	0.029038	0.000987
MaF4	0.000000	0.000000	128.2923	48.35021	237.1097	95.28383	1.263046	0.662139	0.513379
MaF5	0.000002	0.000002	0.000002	0.000001	0.000002	0.100947	0.000002	0.363616	0.553381
MaF6	0.000001	0.000001	0.015708	0.336210	0.272017	0.454914	0.325300	0.172432	0.315221
MaF7	0.000000	0.000000	0.249214	0.297328	0.008971	0.859297	0.000000	0.223606	0.133638
MaF8	0.172575	0.073897	21.07622	0.749332	28.71752	153.0767	0.003998	0.563251	0.557502
MaF9	0.036880	0.000065	0.269636	0.173536	5.317898	0.216590	0.000061	0.071893	0.049391
MaF10	0.829937	0.551516	1.019473	1.021279	0.989085	0.553509	0.000000	0.178804	0.092255
MaF11	0.008020	0.009466	0.230606	0.173813	0.338183	0.322796	0.251806	0.146102	0.114159
MaF12	0.021699	0.029974	0.846230	0.899866	0.908027	0.984421	0.104313	0.262514	0.794063
MaF13	0.000001	0.000001	0.102995	0.281450	0.221073	0.138356	0.154364	0.240990	0.015824
MaF14	0.000001	0.000001	0.000001	0.049084	0.000001	0.000001	0.000001	0.000153	0.000001
MaF15	0.000000	0.000000	0.179776	0.414223	0.398195	0.266944	0.148532	0.112575	0.018052
WFG3	0.019117	0.008674	0.800766	0.697397	1.074877	0.847889	0.881262	0.326798	0.788929
WFG4	0.000648	0.003235	0.669810	0.630484	0.375101	0.992267	0.425000	0.308058	0.102572
WFG5	0.050359	0.050098	0.922782	0.852902	0.355440	0.475489	0.062310	0.648747	0.493692
WFG6	0.010616	0.034172	1.050910	0.828513	0.582948	0.638390	0.468712	0.280313	0.550434
WFG7	0.001111	0.002006	0.905152	0.910634	0.773303	0.771173	0.097641	0.176038	0.431358
WFG8	0.002496	0.011343	1.062146	0.909297	1.028635	0.883724	0.703508	0.297052	0.836044
Friedman mean rank	1.9524	1.9524	6.619	7.1429	6.7857	7.0000	3.8810	4.9048	4.7619
final rank	1	1	6	9	7	8	3	5	4
p-value	6.6585E-18								



(a)

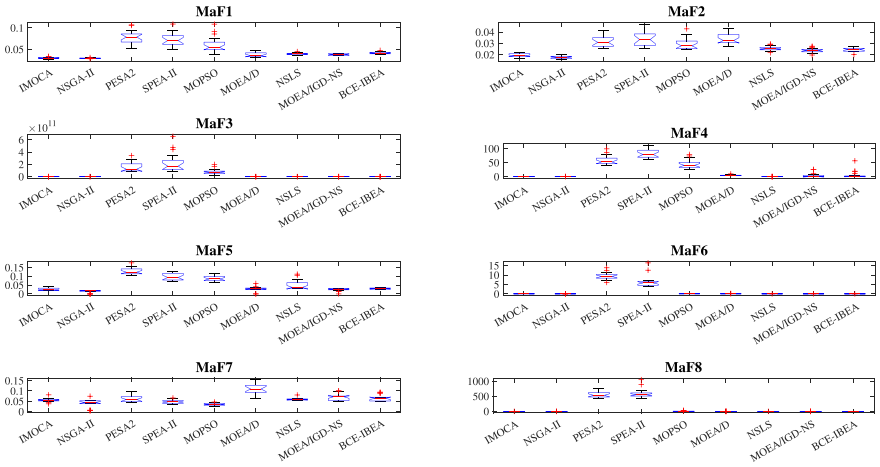


(b)

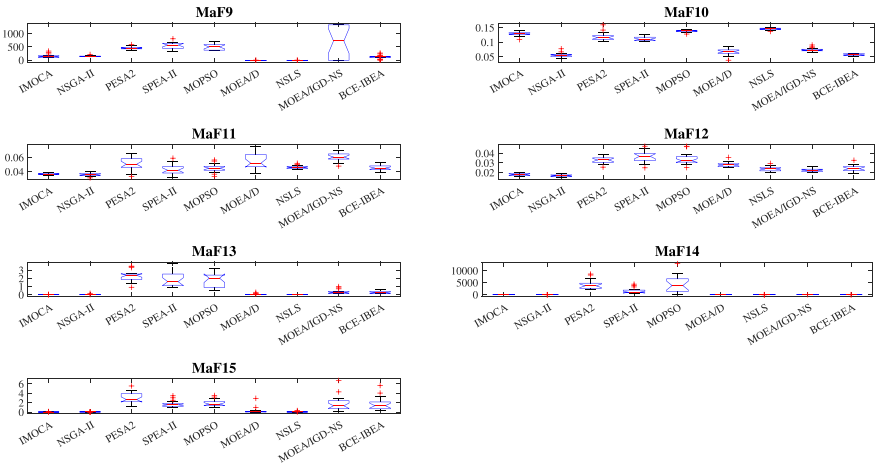


(c)

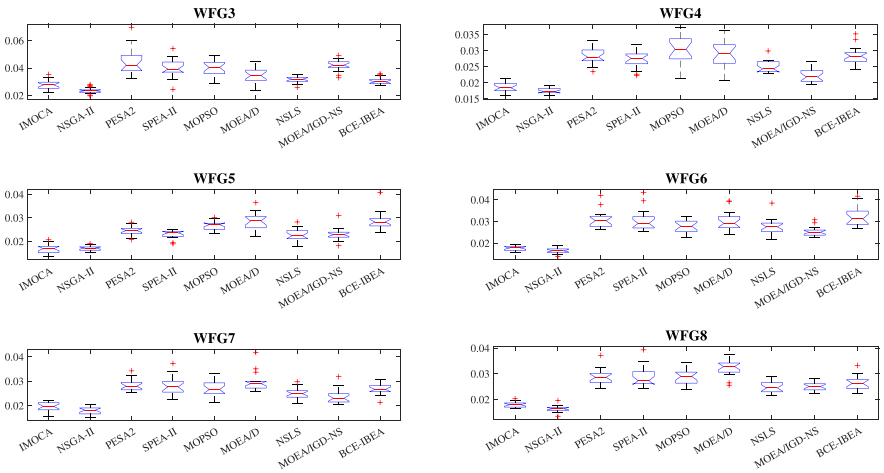
Fig. 6 Box plots of pairwise comparison of set coverage C(A, B) based on 20 independent runs obtained by the 9 algorithms. **a** Box plots of pairwise comparison of set coverage C(A;B) on MaF1-MaF8. **b** Box plots of pairwise comparison of set coverage C(A;B) on MaF9-MaF15. **c** Box plots of pairwise comparison of set coverage C(A;B) on WFG3-8



(a)

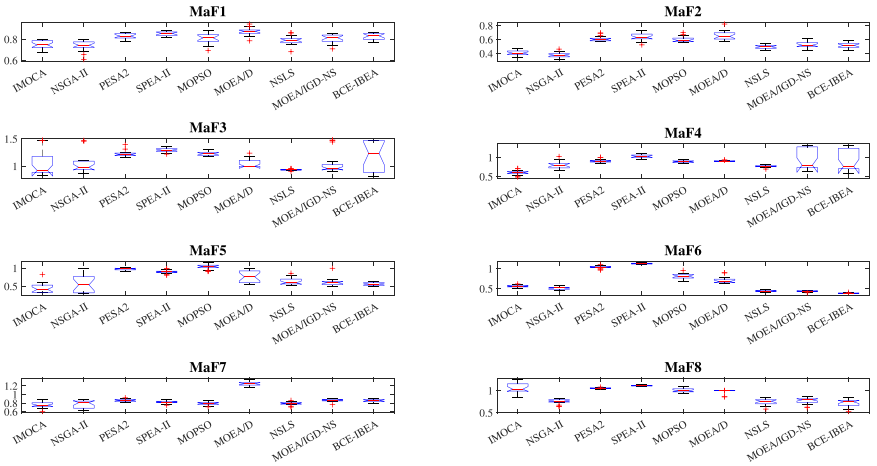


(b)

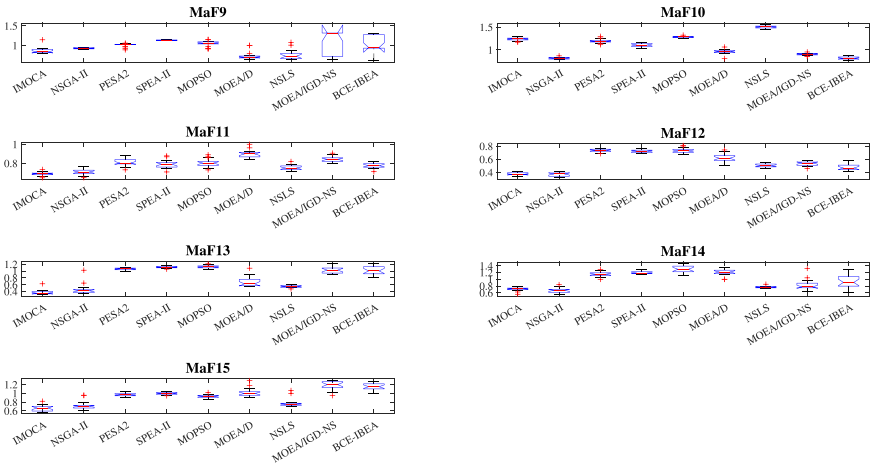


(c)

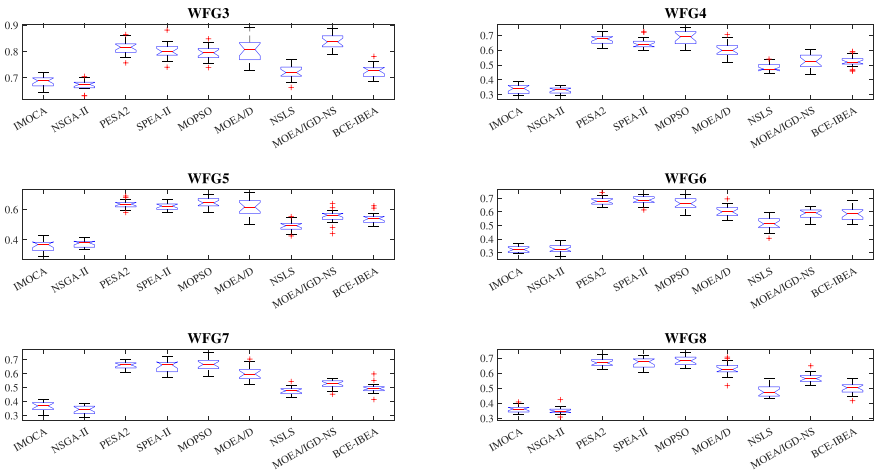
Fig. 7 Box plots of metric Spacing S based on 20 independent runs obtained by the 9 algorithms in solving MaF and WFG problems. **a** Box plots of metric Spacing S on MaF1-8. **b** Box plots of metric Spacing S on MaF9-15. **c** Box plots of metric Spacing S on WFG3-8



(a)



(b)



(c)

Fig. 8 Box plots of metric Spread Δ based on 20 independent runs obtained by the 9 algorithms in solving MaF and WFG problems. **a** Box plots of metric Spread (insert symbol here) on MaF1-8. **b** Box plots of metric Spread ((insert symbol here) on MaF9-15. **c** Box plots of metric Spread ((insert symbol here) on WFG3-8

categories at different stages of the evolution according to their productivity.

7.2.3 Set coverage

As illustrated in Section 6.5, when evaluating the solution sets obtained by algorithms A and B, it is necessary to compute both $C(A, B)$ and $C(B, A)$. To compare the quality of the solution sets obtained by the different algorithms, or more specifically, to compare the dominance relationship between the two final populations resulting from two different algorithms, a pairwise comparison for the metric $C(A, B)$ is made based on the solutions obtained by the IMOCA and four representative algorithms, i.e., NSGA-II, NSLS, MOEA/IGD-NS and BCE-IBEA (see Fig. 6a, b and c). In this comparison, we selected the first four algorithms among the remaining 8 algorithms based on the statistical results of the hypervolume and unary ϵ-indicator.

In Fig. 6a, the subfigures in the first row starting with "IMOCA" are the box plots of the set coverage of the IMOCA on test problems MaF1-MaF8. For example, the subfigure in row 1 and column 2 (the box on the right side of "IMOCA" in Fig. 6a) denotes the proportion of solutions obtained by the IMOCA dominated by the ones obtained by the NSGA-II on the 8 test problems. Similarly, the subfigure in row 2 and column 2 (the box on the left side of "NSGA-II") denotes the proportion of solutions obtained by the NSGA-II dominated by the ones obtained by the IMOCA. According to these two boxes, it can be inferred that, except for problem MaF8, the IMOCA found better solutions than those found by the NSGA-II, especially for problems MaF3 and MaF4. The results on the other 13 test problems are given in Fig. 6b and c.

From a comparison of the first rows and the first columns in Fig. 6a, b and c, it can be concluded that the solution sets of the IMOCA outperform those of the other four competitive algorithms in producing nondominated solutions when solving most of the 21 test problems, whereas for MaF8/9/10 and WFG7, the solution sets of the IMOCA are worse.

7.2.4 Spacing and spread

Last, we examine the performance of the proposed IMOCA in terms of the distribution and diversity by the spacing and spread metrics, respectively. The box plots of the spacing S are given in Fig. 7a, b and c, while the box plots of the spread Δ are given in Fig. 8a, b and c.

These figures show that the IMOCA has lower or nearly equal values of both the spacing and spread indicators

Table 8 Medians of time consumed by each algorithm (A: Algorithms, P: Problems; time unit: second(s))

Time (s) \ A / P	IMOCA	NSGA-II	PESA2	SPEA-II	MOPSO	MOEA/D
MaF1	194.57	184.08	230.62	144.16	55.25	257.18
MaF2	193.03	184.41	295.45	145.03	93.28	257.91
MaF3	207.48	223.32	163.50	143.94	36.80	142.26
MaF4	227.53	224.07	181.83	144.94	52.44	257.15
MaF5	189.04	193.80	247.37	146.88	45.53	243.55
MaF6	214.83	198.40	174.48	143.94	38.73	254.39
MaF7	349.10	343.59	391.82	265.15	89.61	301.17
MaF8	121.92	194.16	98.19	128.18	33.64	163.04
MaF9	154.24	171.14	188.64	122.02	22.93	215.73
MaF10	198.93	191.03	263.13	151.34	57.03	258.44
MaF11	200.87	191.50	215.97	148.35	57.06	261.35
MaF12	195.69	190.37	254.91	150.76	65.83	262.87
MaF13	169.94	168.35	152.03	122.52	40.06	217.11
MaF14	1080.16	1111.38	895.89	792.17	195.42	1056.53
MaF15	1077.76	1033.77	1082.49	821.96	311.65	1387.32
WFG3	197.52	189.83	251.22	149.45	78.94	262.84
WFG4	197.29	187.87	271.21	149.07	77.86	260.24
WFG5	196.36	186.94	283.32	149.13	88.52	261.83
WFG6	197.03	186.85	245.65	148.71	81.41	260.16
WFG7	198.58	189.14	263.49	151.57	73.41	264.33
WFG8	198.43	189.34	264.75	151.49	79.96	261.02

compared with those of the other algorithms on most of the problems. Considering the excellence that the IMCOA reveals in terms of the hypervolume, unary ϵ-indicator and set coverage metric $C(A, B)$, the performance of the IMOCA in terms of the spacing and spread indicates that the solution sets obtained by the IMOCA approach the PFs with both accuracy and diversity in most cases.

7.2.5 Time consumption

The time consumed by the IMOCA and other five algorithms in the comparison is given in Table 8. Since the rest three algorithms are carried out on another platform (the PlatEMO platform), the time consumptions are not commensurable and thus are not listed here. It can be seen from Table 8 that there is no big difference between the time consumptions of the IMOCA and the NSGA-II, which due to the fact that the computational complexity of the influence functions and the sorting part in the IMOCA is the same as the corresponding operations in the NSGA-II.

Overall, the experimental results confirm that, with several simple but necessary modifications, the IMOCA is competitive in capturing well-distributed, near-optimal nondominated solutions under acceptable execution time when solving MOOPs.

8 Conclusions and future work

In this paper, we proposed a modified version of the CA with a multistrategy knowledge base for solving multiobjective optimization problems. Four basic knowledge sources, i.e., normative, situational, topographical and historical knowledge, are adopted in the knowledge base of the proposed algorithm, and necessary modifications were made to the situational, historical and topographical knowledge. In addition, a simple mutation scheme was integrated into the algorithm to perform fine tuning in the late stage of evolution. The experimental results showed that, on most of the selected test problems, the proposed IMOCA was able to converge better to the true PFs and provided a wide and well-distributed spread of solutions for most of the problems compared to the other 8 multiobjective algorithms.

There is still abundant room to refine the proposed algorithm in future work. Inspired by the performance metric computation used in the elimination of noncontributing solutions in the MOEA/IGD-NS, exploitation of the performance evaluation in the selection operation in the IMOCA may further improve the quality of the obtained solution set. Moreover, given the fact that the framework of the CA can integrate any form of evolutionary technique, exploring other effective and efficient knowledge categories for solving MOOPs will also be addressed in future work. Furthermore, attempts at applying the IMOCA to practical MOOPs could be made in the near future. In addition, since the proposed IMOCA is an unconstrained search approach, developing an efficient mechanism to deal with the constraints in MOOPs could be a future direction of research.

Appendix : Performance of different knowledge source as IMOCA solving MaF test suite

Here in Appendix, Figs. 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22 and 23 show the minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when the IMOCA optimizing the MaF test suite in a random run.

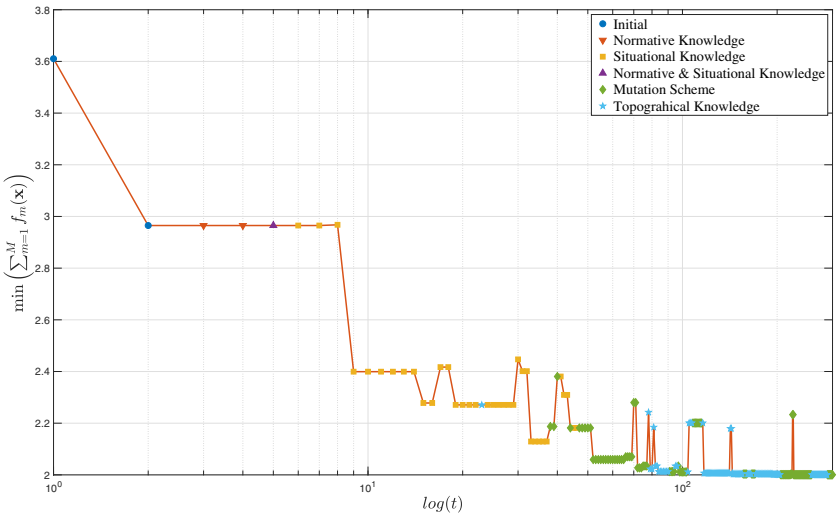


Fig. 9 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF1 using the IMOCA (iteration $t \in [1, 300]$)

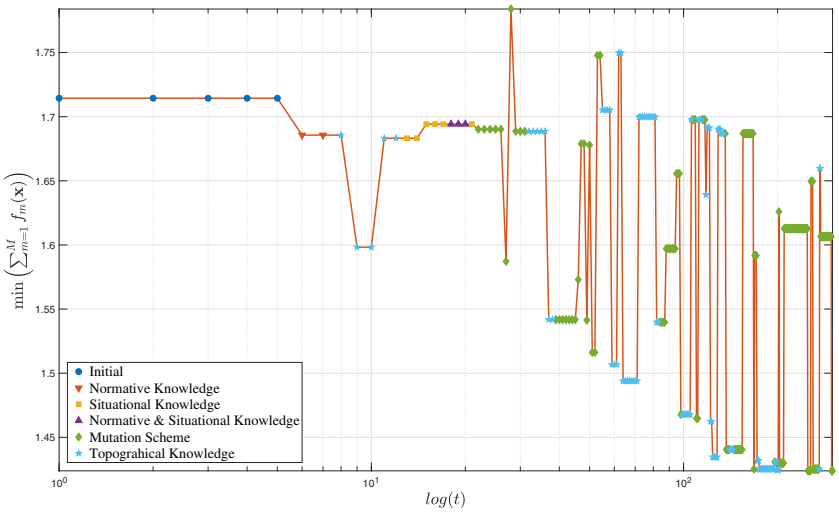


Fig. 10 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF2 using the IMOCA (iteration $t \in [1, 300]$)

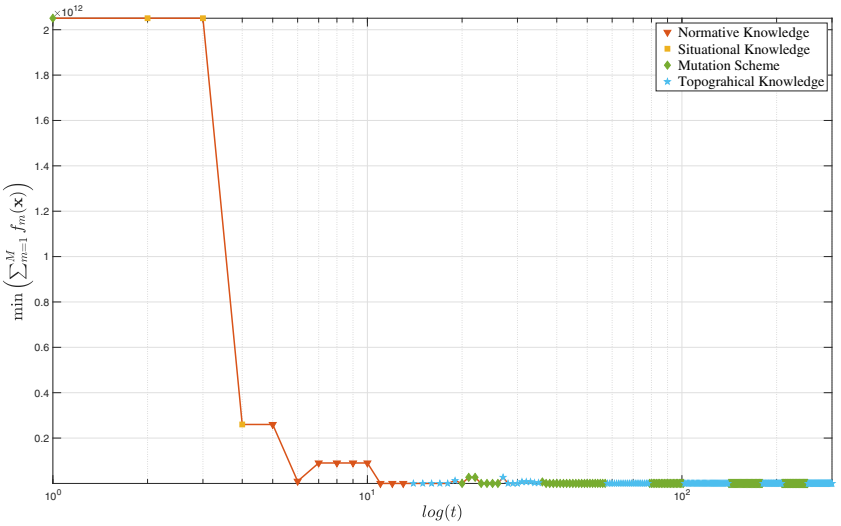


Fig. 11 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF3 using the IMOCA (iteration $t \in [1, 300]$)

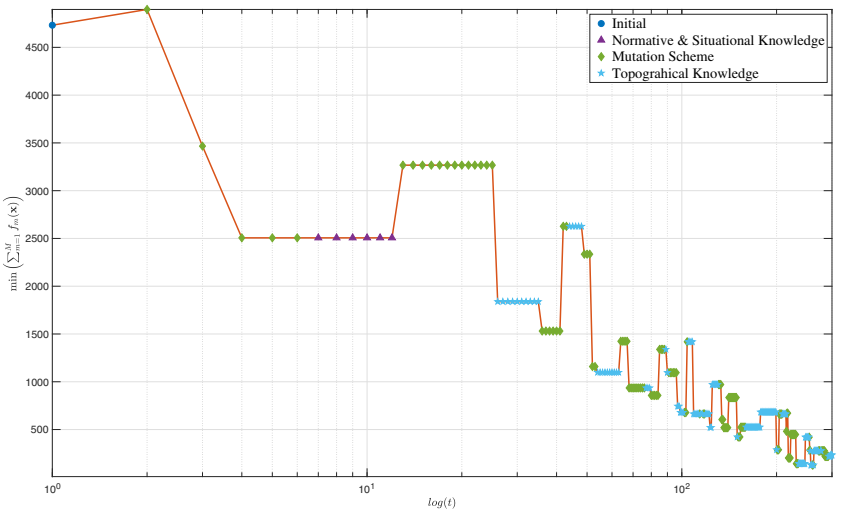


Fig. 12 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF4 using the IMOCA (iteration $t \in [1, 300]$)

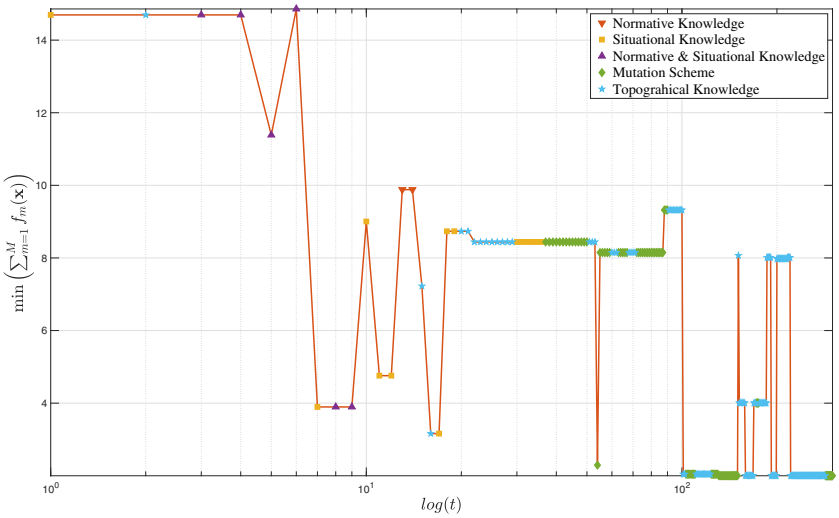


Fig. 13 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF5 using the IMOCA (iteration $t \in [1, 300]$)

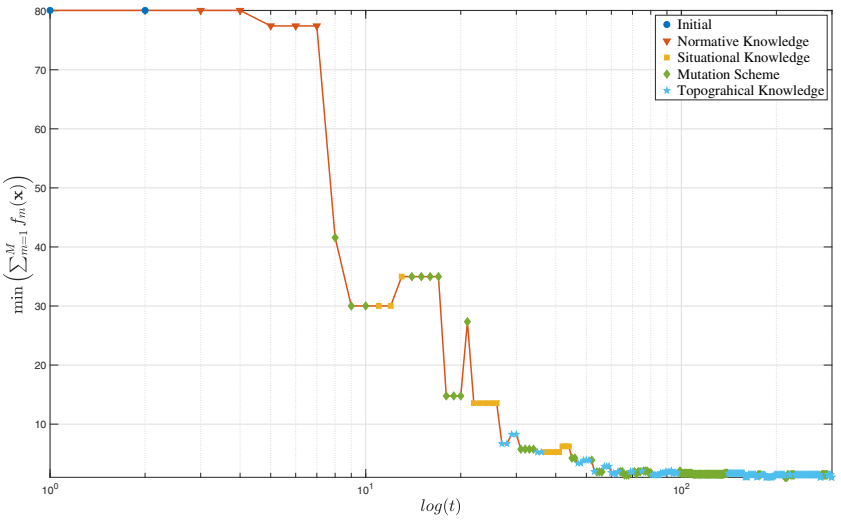


Fig. 14 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF6 using the IMOCA (iteration $t \in [1, 300]$)

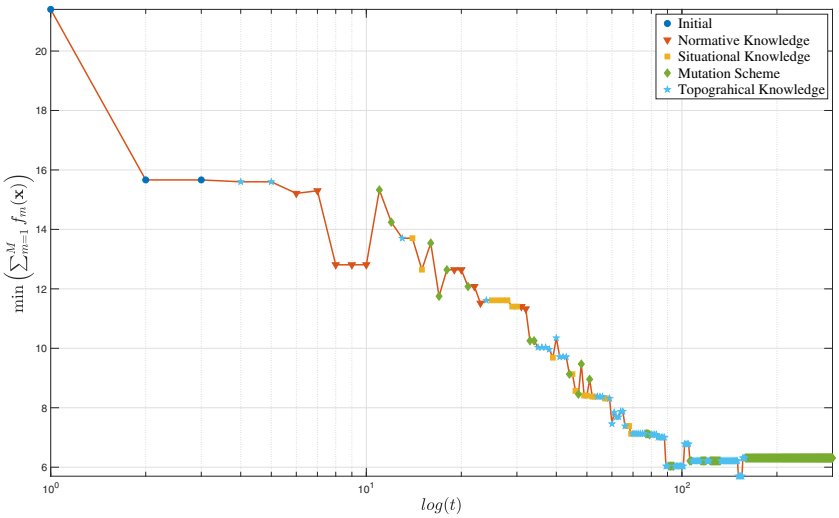


Fig. 15 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF7 using the IMOCA (iteration $t \in [1, 300]$)

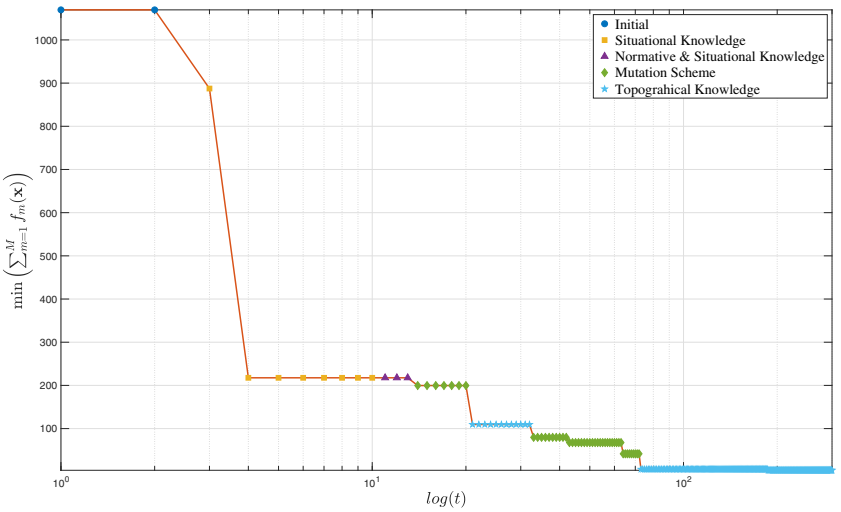


Fig. 16 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ as when optimizing MaF8 using the IMOCA (iteration $t \in [1, 300]$)

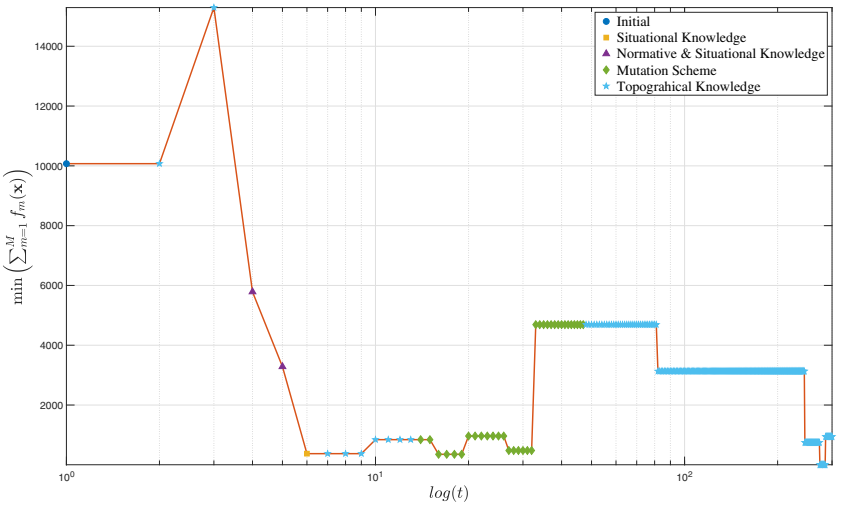


Fig. 17 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF9 using the IMOCA (iteration $t \in [1, 300]$)

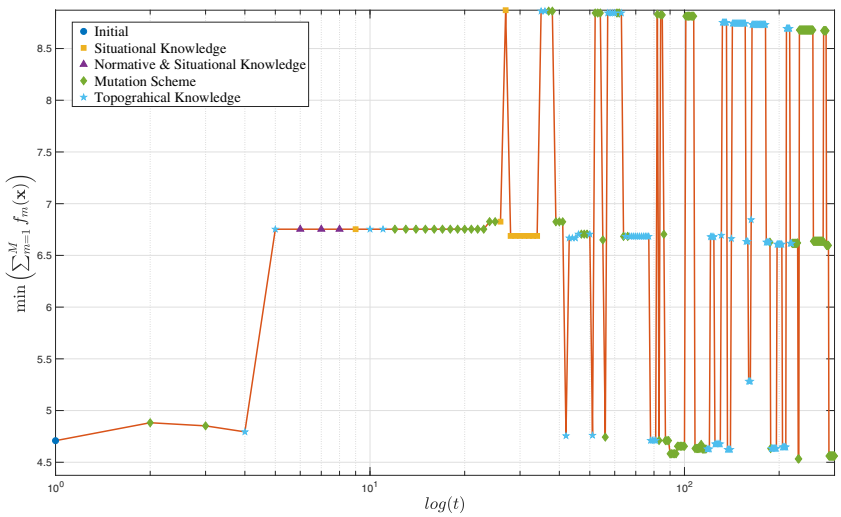


Fig. 18 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF10 using the IMOCA (iteration $t \in [1, 300]$)

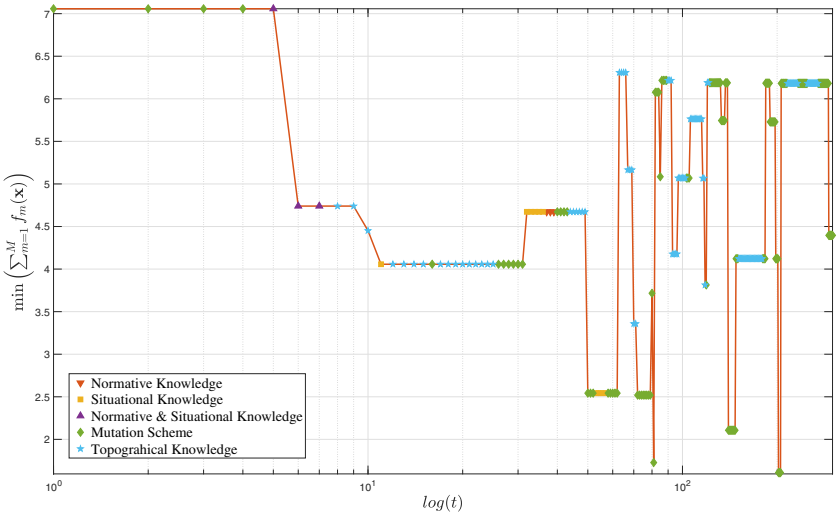


Fig. 19 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF11 using the IMOCA (iteration $t \in [1, 300]$)

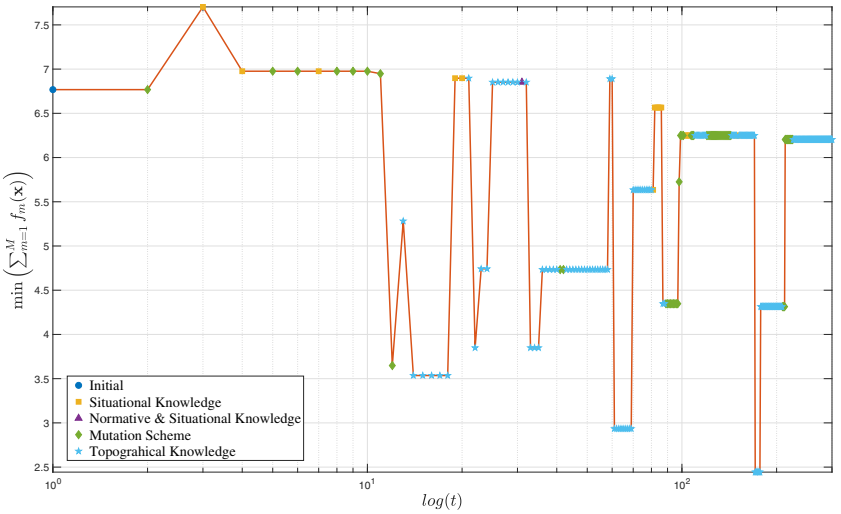


Fig. 20 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF12 using the IMOCA (iteration $t \in [1, 300]$)

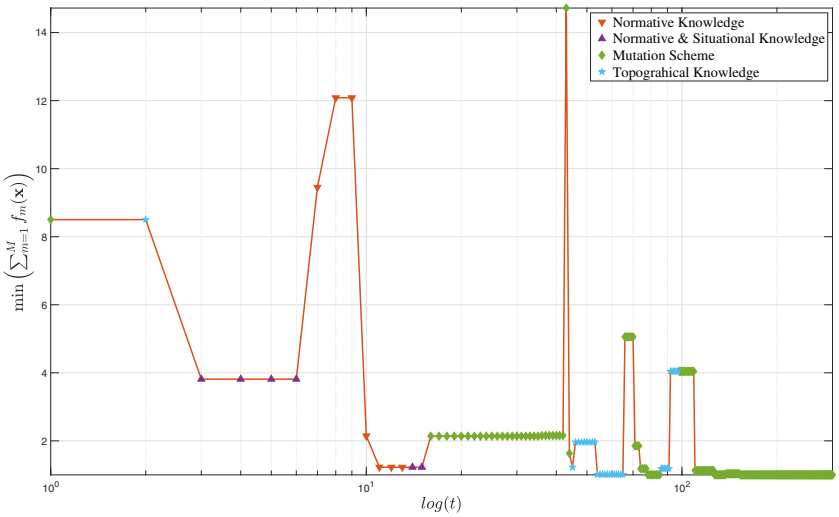


Fig. 21 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF13 using the IMOCA (iteration $t \in [1, 300]$)

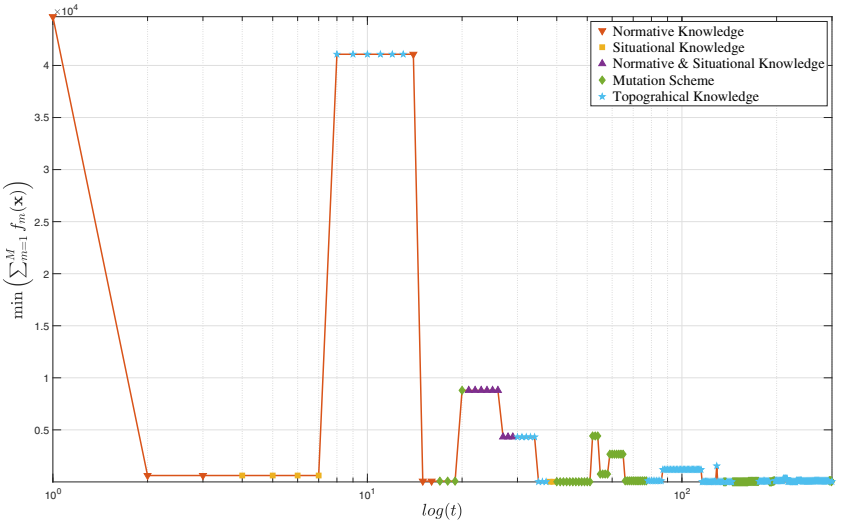


Fig. 22 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF14 using the IMOCA (iteration $t \in [1, 300]$)

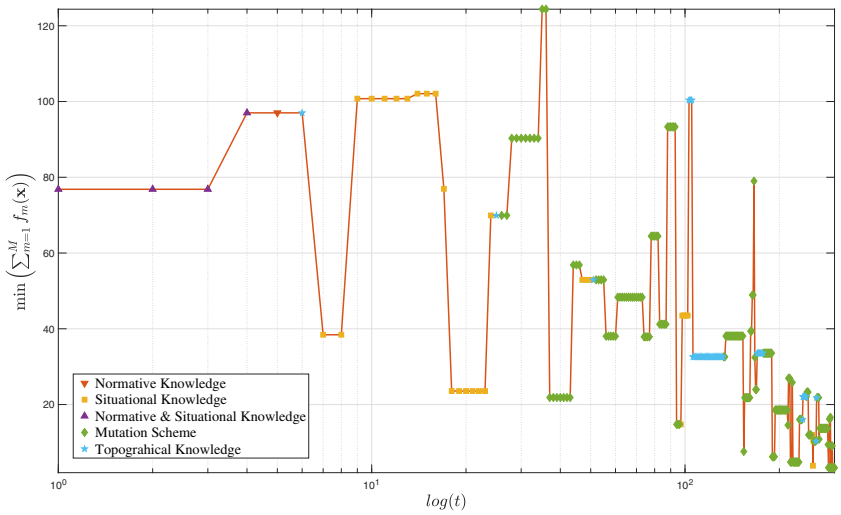


Fig. 23 The minimum values for $\sum_{m=1}^{M} f_m(\mathbf{x})$ when optimizing MaF15 using the IMOCA (iteration $t \in [1, 300]$)

References

1. Abdolrazzagh-Nezhad M, Radgohar H, Salimian SN (2020) Enhanced cultural algorithm to solve multi-objective attribute reduction based on rough set theory. Math Comput Simul 170:332–350
2. Becerra RL, Coello CAC (2004) Culturizing differential evolution for constrained optimization. In: Proceedings of the fifth mexican international conference in computer science, 2004. ENC 2004. IEEE, pp 304–311
3. Best C, Che X, Reynolds RG, Liu D (2010) Multi-objective cultural algorithms. In: IEEE congress on evolutionary computation. IEEE, pp 1–9
4. Booker L, Forrest S, Mitchell M, Riolo R (2005) Perspectives on adaptation in natural and artificial systems. Oxford University Press
5. Chen B, Zeng W, Lin Y, Zhang D (2015) A new local search-based multiobjective optimization algorithm. IEEE Trans Evol Comput 19(1):50–73
6. Cheng R, Li M, Tian Y, Zhang X, Yang S, Jin Y, Yao X (2017) A benchmark test suite for evolutionary many-objective optimization. Complex & Intelligent Systems 3(1):67–81
7. Chung C (1997) Knowledge-based approaches to self-adaptation in cultural algorithms. PhD thesis, Wayne State University, Detroit, Michigan
8. Chung CJ, Reynolds RG (1996) A testbed for solving optimization problems using cultural algorithms. In: Evolutionary programming, pp 225–236
9. Chung CJ, Reynolds RG (1998) CAEP - an evolution-based tool for real-valued function optimization using cultural algorithms. International Journal on Artificial Intelligence Tools 07(03):239–291
10. Coello CAC (2015) EMOO repository. http://delta.cs.cinvestav.mx/ccoello/EMOO/
11. Coello CAC (2015) Multi-objective evolutionary algorithms in real-world applications: some recent results and current challenges. In: Advances in evolutionary and deterministic methods for design, optimization and control in engineering and sciences. Springer, pp 3–18
12. Coello CAC, Becerra RL (2003) Evolutionary multiobjective optimization using a cultural algorithm. In: Proceedings of the 2003 IEEE swarm intelligence symposium. SIS'03 (Cat. No.03EX706). IEEE, pp 6–13
13. Coello CAC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 8(3):256–279
14. Corne DW, Knowles JD, Oates MJ (2000) The Pareto envelope-based selection algorithm for multiobjective optimization. In:

International conference on parallel problem solving from nature. Springer, Berlin, pp 839–848
15. Corne DW, Jerram NR, Knowles JD, Oates MJ (2001) PESA-II: region-based selection in evolutionary multiobjective optimization. In: Proceedings of the genetic and evolutionary computation conference, pp 283–290
16. Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, New York
17. Deb K, Agrawal S, Pratap A, Meyarivan T (2000) A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: International conference on parallel problem solving from nature. Springer, Paris, France, pp 849–858
18. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1(1):3–18
19. Fonseca C, Knowles J, Thiele L, Zitzler E (2005) A tutorial on the performance assessment of stochastic multiobjective optimizers. Tech. rep., Evolutionary Multi-Criterion Optimization Conference (EMO 2005), Guanajuato, Mexico
20. Gao H, Diao M (2011) Cultural firework algorithm and its application for digital filters design. International Journal of Modelling Identification & Control 14(4):324
21. Guo YN, Yang Z, Wang C, Gong D (2017) Cultural particle swarm optimization algorithms for uncertain multi-objective problems with interval parameters. Nat Comput 16(4):527–548
22. Guo YN, Zhang P, Cheng J, Wang C, Gong D (2018) Interval multi-objective quantum-inspired cultural algorithms. Neural Comput & Applic 30(3):709–722
23. Holland J (1975) Adaptation in natural and artificial Systems. Second edition (1992). (First edition, University of Michigan Press, 1975). MIT Press, Cambridge
24. Huband S, Hingston P, Barone L, While L (2006) A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans Evol Comput 10(5):477–506
25. Jin X, Reynolds RG (1999) Using knowledge-based evolutionary computation to solve nonlinear constraint optimization problems: a cultural algorithm approach. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406), vol 3. IEEE, pp 1672–1678
26. Jin X, Reynolds RG (1999) Using knowledge-based system with hierarchical architecture to guide the search of evolutionary computation. In: Proceedings 11th international conference on tools with artificial intelligence. IEEE, Chicago, pp 29–36. https://doi.org/10.1109/tai.1999.809762
27. Li M, Yang S, Liu X (2016) Pareto or non-pareto: bi-criterion evolution in multiobjective optimization. IEEE Trans Evol Comput 20(5):645–665
28. Liu T, Jiao L, Ma W, Ma J, Shang R (2016) A new quantum-behaved particle swarm optimization based on cultural evolution mechanism for multiobjective problems. Knowl-Based Syst 101:90–99
29. Lu Y, Zhou J, Qin H, Wang Y, Zhang Y (2011) A hybrid multi-objective cultural algorithm for short-term environmental/economic hydrothermal scheduling. Energy Convers Manag 52(5):2121–2134
30. Mao Z, Xiang Y, Zhang Y, Liu M (2020) A novel multi-objective cultural algorithm embedding five-element cycle optimization. In: 2020 IEEE congress on evolutionary computation (CEC). IEEE, pp 1–10
31. Moscato P (1989) On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms. Caltech Concurrent Computation Program
32. Moscato P, Cotta C (2003) A gentle introduction to memetic algorithms. In: Handbook of metaheuristics. Kluwer Academic Publishers, Boston, pp 105–144
33. Qin H, Zhou J, Lu Y, Li Y, Zhang Y (2010) Multi-objective cultured differential evolution for generating optimal trade-offs in reservoir flood control operation. Water Resour Manag 24(11):2611–2632
34. Reynolds RG (1994) An introduction to cultural algorithms. In: Proceedings of the third annual conference on evolutionary programming. World Scientific, Singapore, pp 131–139
35. Reynolds RG (2018) Culture on the edge of chaos: cultural algorithms and the foundations of social intelligence. SpringerBriefs in Computer Science, Springer International Publishing
36. Reynolds RG, Chung C (1997) Knowledge-based self-adaptation in evolutionary programming using cultural algorithms. In: IEEE international conference on evolutionary computation, pp 71–76
37. Reynolds RG, Chung CJ (1996) A self-adaptive approach to representation shifts in cultural algorithms. In: IEEE international conference on evolutionary computation. IEEE, pp 94–99
38. Reynolds RG, Liu D (2011) Multi-objective cultural algorithms. In: 2011 IEEE congress of evolutionary computation (CEC). IEEE, pp 1233–1241
39. Saleem SM (2001) Knowledge-based solution to dynamic optimization problems using cultural algorithms. PhD thesis, Wayne State University, Detroit, Michigan
40. Srinivas N, Deb K (1994) Muiltiobjective optimization using nondominated sorting in genetic algorithms. MIT Press 2(3):221–248
41. Stanley SD, Kattan K, Reynolds RG (2020) CAPSO. Cultural Algorithms 9:169–194. John Wiley & Sons, Ltd https://doi.org/10.1002/9781119403111.ch9, https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119403111.ch9, eprint https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119403111.ch9
42. Tian Y, Zhang X, Cheng R, Jin Y (2016) A multi-objective evolutionary algorithm based on an enhanced inverted generational distance metric. In: 2016 IEEE congress on evolutionary computation (CEC), pp 5222–5229
43. 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
44. Yan X, Song T, Wu Q (2017) An improved cultural algorithm and its application in image matching. Multimedia Tools & Applications 76(13):14951–14968
45. Yuan X, Yuan Y (2006) Application of cultural algorithm to generation scheduling of hydrothermal systems. Energy Conversion & Management 47(15/16):2192–2201
46. Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation 11(6):712–731
47. Zhang R, Zhou J, Mo L, Ouyang S, Liao X (2013) Economic environmental dispatch using an enhanced multi-objective cultural algorithm. Electr Power Syst Res 99:18–29
48. Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and applications, vol 63. Citeseer
49. Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evolutionary Computation 3(4):257–271
50. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms - empirical results. Evol Comput 8(2):173–195
51. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm. TIK-report 103
52. Zitzler E, Thiele L, Laumanns M, Fonseca CM, Fonseca VGD (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput 7(2):117–132

Zhengyan Mao received the B.Eng. degree from East China University of Science and Technology, Shanghai, China, in 2013. Currently, she is a Ph.D. candidate within a master-doctor combined program in the School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China. Her current research interests include evolutionary algorithms, multiobjective optimization, and their application.

Mandan Liu received the B.S. degree in process automation and the Ph.D. degree in control theory and control engineering from Zhejiang University, Hangzhou, China, in 1995 and 2000, respectively. She was a Research Assistant and an Associate Researcher with the Institute of Automation, East China University of Science and Technology, Shanghai, China, from 2000 to 2008. Since 2008, she has been a Professor with the Department of Automation, School of Information Science and Engineering, East China University of Science and Technology. Her current research interests include process modeling and optimization, intelligent control theory and applications, and intelligent optimization algorithms and applications. Dr. Liu received several awards, including the Venus Fellowship (Shanghai Science and Technology Commission), the Yucai Education Award (Shanghai Education Commission), and the Outstanding Teacher Award (Baosteel Education Foundation).