A self-organizing map approach for constrained multi-objective optimization problems

Chao He 1 · Ming Li 1,2 · Congxuan Zhang 3 · Hao Chen 2 · Peilong Zhong 2 · Zhengxiu Li 2 · Junhua Li 2

Received: 24 September 2021 / Accepted: 15 April 2022 / Published online: 19 May 2022
© The Author(s) 2022

Abstract
There exist many multi-objective optimization problems (MOPs) containing several inequality and equality constraints in practical applications, which are known as CMOPs. CMOPs pose great challenges for existing multi-objective evolutionary algorithms (MOEAs) since the difficulty in balancing the objective minimization and constraint satisfaction. Without loss of generality, the distribution of the Pareto set for a continuous m-objective CMOP can be regarded as a piecewise continuous manifold of dimension \((m - 1)\). According to this property, a self-organizing map (SOM) approach for constrained multi-objective optimization problems is proposed in this article. In the proposed approach, we adopt the strategy of two population evolution, in which one population is evolved by considering all the constraints and the other population is used to assist in exploring the areas. In the evolutionary stage, each population is assigned a self-organizing map for discovering the population distribution structure in the decision space. After the topological mapping, we utilize the extracted neighborhood relationship information to generate promising offspring solutions. Afterwards, the neuron weight vectors of SOM are updated by the objective vectors of the surviving offsprings. Through the proposed approach, we can make the population efficiently converge to the feasible region with suitable levels of diversity. In the experiments, we compare the proposed method with several state-of-the-art approaches by using 48 benchmark problems. The evaluation results indicate that the overwhelmingly superior performance of the proposed method over the other peer algorithms on most of the tested problems. The source code is available at https://github.com/hccccc92918/CMOSMA.

Keywords Constrained multi-objective optimization · Self-organizing map · Evolutionary algorithms

Introduction
In many real-world applications, we often face complex decision-making problems that need to optimize a number of conflicting objectives simultaneously with a set of equality and/or inequality constraints [1–3]. These problems are called constrained multi-objective optimization problems (CMOPs). Generally speaking, a continuous CMOP can be defined as follows:

\[
\begin{align*}
\text{minimize} & \quad F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\
\text{subject to} & \quad x \in \Omega \\
g_i(x) \leq 0, & \quad i = 1, \ldots, p \\
h_j(x) = 0, & \quad j = 1, \ldots, q
\end{align*}
\]

(1)

where \(\Omega\) denotes the decision space, the notation \(x = (x_1, x_2, \ldots, x_n)\) is a candidate solution, and \(f\) indicates the objective vector. \(F : \Omega \rightarrow R^m\) represents the continuous objective function, and \(m\) is the number of objectives. \(g_i(x) \leq 0\) are \(p\) objectives inequality constraints. \(h_j(x) = 0\) is an equality constraint, and \(q\) is the number of equality constraints. In the CMOP, the set of all Pareto optimal solutions is called the Pareto set (PS) and the objective vectors corresponding to PS are named the Pareto front (PF). In addition, the existing methods usually adopt the overall constraint violation degree to reflect the feasibility of each solution \(x\), the constraint violation value of \(x\) is calculated as
\[ CV(x) = \sum_{i=1}^{p} \max\{0, g_i(x)\} + \sum_{j=1}^{q} |h_j(x)|. \] (2)

If \( CV(x) = 0 \), it means that \( x \) is a feasible solution. Otherwise, \( x \) is infeasible. Different from the unconstrained multi-objective optimization problems (MOPs), we need to consider equality and inequality constraints when solving CMOPs. Therefore, CMOPs are much more difficult to be solved than MOPs with an unconstrained Pareto front [4]. Constraint functions greatly increase the difficulty of optimizing problems. Having been developed for more than two decades, multi-objective evolutionary algorithms (MOEAs) have shown excellent results in different types of benchmarking MOPs and practical applications [3, 5–7], which give us some new patterns to solve CMOPs. Recently, some constrained MOEAs (CMOEs) have been proposed. The literature [8] presented a categorization scheme for existing constraint handling techniques in multi-objective optimization, and classified different CMOEAs into the following three categories.

The first category is mainly driven by the feasibility information, where feasible solutions are always guaranteed a higher priority to survive in the next generation. C-MOEA/D and constrained NSGA-II are two representative algorithms of this type [9, 10]. In C-MOEA/D and constrained NSGA-II, a constrained dominance principle is adopted to handle CMOPs. Specifically, a solution \( x \) is said to constraint dominate (CD) \( y \) if the following conditions hold:

\[
\begin{align*}
x \prec_{\text{CD}} y & \iff \begin{cases} 
x < y, & \text{if } CV(x) = CV(y) = 0 \\
CV(x) < CV(y), & \text{otherwise.}
\end{cases}
\end{align*}
\] (3)

where \( \prec \) represents the Pareto dominate. According to Eq. (3), we can find that the MOEA/D and NSGA-II can be further extended for handling CMOPs by simply replacing the Pareto dominance relation with this constrained dominance relation. Borrowing a similar idea, several MOEA/D variants [11, 12] introducing the constraint dominance principle into the decomposition strategy to improve the performance of solving CMOPs. In addition, some new constraint dominance principles proposed recently also provide research ideas for this type of CMOEA [13, 14].

The second category aims to properly balance convergence and feasibility during the evolutionary process. Inspired by the success of BGE [15] algorithm in solving MOPs [4], proposed a Tri-Goal evolution (TiGE) framework for solving CMOPs. In TiGE, the proposed framework carefully designs two indicators for convergence and diversity, respectively, and converts the constraints into the third indicator for feasibility. Analogously, Wang et al. [8] presented an upgraded version of the Two_Arch algorithm [16], which has shown effectiveness in solving CMOPs. By adding feasibility information to the convergence-oriented archive, the Two_Arch algorithm can be readily used to tackle CMOPs. To balance objective minimization and constraint satisfaction, a biphasic CMOEA is proposed [17], namely PPS. PPS is an interesting CMOEA, where the push and pull search strategy is proposed. This push and pull search strategy is embedded into a decomposition-based multi-objective evolutionary framework MOEA/D, which has shown good performance in solving CMOPs. Furthermore, [18] proposed a two-stage CMOEA, which can adaptively adjust the fitness evaluation strategies during the whole search process. As a result, this method has shown promising performance in handling CMOPs with different feasible regions.

Algorithms in the third category try to repair the infeasible solutions to drive them toward the feasible region. The main idea is to convert the infeasible solution into a feasible solution through some repair strategy. For instance, [19] proposed a Pareto descent repair operator, in which ideas derived from multi-objective local search and gradient projection method are incorporated. However, this method has shown some limitations in practical problems where corresponding gradient information is not easily accessible. The C-PSA proposed in [20] is a repair-based CMOEA, in which a simulated annealing method is adopted to accelerate the progress of movements from infeasible solutions toward feasible ones. This acceleration strategy is helpful when improving the search efficiency of the population. Another representative work is proposed by Jiao [21], where the infeasible solutions are guided toward the feasible region by leveraging the information provided by the feasible direction.

Although existing CMOEAs have exhibited excellent performance on some constrained multi-objective benchmark problems and practical applications [22, 23], there are still several noticeable challenges such as the problems have several disconnected feasible regions and only several narrow feasible regions. Recently, some structured MOEAs have been suggested, including the IRM-MEDA [24], cMOGAs [25], MOEA/D [26], and SMEA [27]. These structured MOEAs use the regularity property of MOP to guide their search. For instance, cMOGA exploits a cell topological structure to explore the decision space, which can improve search efficiency. MOEA/D makes use of the regularity property, the neighborhood relationship is established by a set of predefined reference/weight vectors. Then, the MOP is decomposed into a number of scalar objective optimization subproblems and optimized simultaneously. Without loss of generality, the distribution of the Pareto set for a continuous \( m \)-objective MOP is a piecewise continuous manifold, which has a dimension of \( m-1 \). For the CMOP, it is can be regarded as a special case of MOP. The constrained PF of CMOP is a part of the unconstrained PF. Therefore, the PF of CMOP also has a regularity property. It provides a new idea
for CMOEAs to deal with CMOPs, especially complex constraints optimization problems. Although many structured MOEAs are specifically designed for solving MOPs, little work regarding using regularity property to handle CMOPs.

Inspired by the success of structured MOEAs in the field of MOPs, we present a self-organizing map (SOM) approach for solving CMOPs. Our main contributions to this article are summarized as follows.

(1) In this article, a new two-population evolutionary framework named CMOSMA is proposed. The two populations have different behaviors and complementary effects: one is denoted as the feasibility-oriented population (FP); the other is denoted as the auxiliary-oriented population (AP). Different from the existing two-population/stage CMOEAs, the unique feature of CMOSMA is that SOM is embedded into two collaborative and complementary populations during the evolutionary process.

(2) Two-SOM collaborative framework is developed to extract neighborhood relationship information of solutions in FP and AP at each generation, respectively. Then, each of the two SOMs is used to guide recombination within the neighborhood to generate promising offspring. In such a way, the generated offspring of FP have good feasibility or good convergence, and the offspring of AP can jump over the infeasible band to enable FP to evolve towards the true Pareto Front.

(3) In the training procedure, the neuron weight vectors of SOM are updated by the objective vectors of the surviving offspring, which ensures that the algorithm has good computational efficiency. In addition, the unsupervised learning approach does not require a lot of data for pre-training. Therefore, our method is very suitable for handling different types of benchmark CMOPs and practical applications.

The rest of this paper is organized as follows. “Related background and motivations” briefly describes the challenges of existing constraint handling techniques. Details of the self-organizing map approach are given in “Proposed framework”. The empirical results and discussions are presented in “Experimental results”. Finally, “Conclusions” concludes this paper and provides research directions for future work.

Related background and motivations

Self-organizing map

SOM is a self-organizing (competitive) neural network [28], which maps high-dimensional data input to low-dimensional space. Generally, SOM is composed of the input layer and latent layer. The input layer is n-dimensional and uses the solutions as the training points. As for the latent layer, it is a two-dimensional planar array composed of neurons. Let us consider an example, Fig. 1 depicts the topology of a 2-D SOM, which is composed of an input layer and a latent layer. As shown in Fig. 1, we can find that SOM consists of 25 neurons and each input solution is connected to all output neurons. SOM network adopts unsupervised and competitive learning methods. It has the advantage of preserving the topological properties of data, and can map nearby data in high-dimensional input space to adjacent units in SOM. Therefore, SOM is often used for cluster analysis of high-dimensional data. Based on this property, some studies have been carried out using SOM to modify MOEAs for solving MOPs. For instance, [29] proposed a high-dimensional object space visualization method based on SOM. By this method, the high-dimensional nondominated solution can be mapped to two-dimensional SOM. Reference [30] proposed a novel weight design method by using a SOM to estimate the shape of the PF, effectively improving population diversity. However, there are few works that have utilized SOM in solving CMOPs. The successful integration of SOM with MOEA in solving unconstrained problems can provide a reference for solving CMOPs. For this reason, we desire to devise a novel SOM-based framework to deal with CMOPs.

Related concepts

In the Karush–Kuhn–Tucker (KKT) condition, the distribution of the Pareto set for a continuous m-objective CMOP is an \((m - 1)\)-dimensional piecewise continuous manifold, which is called the regularity property [31–33]. According to this property, PS is a piecewise continuous curve.
when we deal with two-objective problems. For a continuous three-objective problem, PS can be regarded as a piecewise continuous curved surface as well. The definition of regularity property is as follows.

**Theorem 1** Assume that the objective functions of a multi-objective optimization problem $f_i(x^*)$, $i = 1, 2, \ldots, m$ are continuously differentiable at $x^* = (x_1, \ldots, x_n) \in \Omega$ ($\Omega$ denotes the decision space and $n$ is the number of decision variables). The necessary condition for $x^*$ to be a (local) Pareto optimal solution is that there exists a vector $w \geq 0$ satisfying the following conditions

$$
\begin{align*}
\sum_{i=1}^{m} w_i \nabla f_i(x^*) &= 0 \\
\sum_{i=1}^{m} w_i &= 1.
\end{align*}
$$

(4)

The points that satisfy the above condition are called the KKT points. It is obvious that Eq. 4 has $n + 1$ equality constraints and $n + m$ variables. Therefore, without loss of generality, the PS of a continuous $m$-objective MOP is a piecewise continuous manifold, which has a dimension of $m - 1$. Generally speaking, we can treat the CMOP as a special case of MOP, so continuous CMOPs meet the regularity property as well. Due to this fact, PS has a regular topological structure. Following this property, we utilize the SOM for discovering the population distribution structure in the decision space of CMOPs. On the one hand, we use the evolutionary operation to approach PS. On the other hand, the SOM is applied to explore the manifold structure of PS. By doing all of these, the search efficiency will be significantly improved.

**Challenges to existing MOEAs with constraint handling techniques**

Recently, despite the progress made regarding the performances of CMOEAs in handling CMOPs, there exist some limitations in the state-of-the-art CMOEAs. For instance, the problem that has a highly multimodal landscape and a band of the infeasible region. Taking the 2-objective C1-DTLZ3 [9] in Fig. 2 as an illustrative example, we find that the feasible region of this test problem is intersected by an infeasible ribbon. As can be further observed from Fig. 2a, the size of the infeasible region is large. Therefore, the C1-DTLZ3 has brought challenges regarding the convergence of existing CMOEAs. To better illustrate this fact, Fig. 2b shows the nondominated solution set obtained by CMOEA-MS [18] on 2-objective C1-DTLZ3. As can be seen from Fig. 2b, the final solutions found by CMOEA-MS are stuck in the outermost feasible boundary after 50,000 function evaluations. This means that CMOEA-MS is trapped in local optima when solving C1-DTLZ3.

Another test function worth considering is MW13 [34], it is an irregular instance, which the PF is disconnected and the size of the feasible region is very small. Therefore, MW13 poses a great challenge for the CMOEAs to push the population toward the global optimum. To validate the performances of existing CMOEAs on MW13, we employ the C-MOEA/D [9] and CMOEA-MS [18] as the benchmark algorithms to evaluate. Figure 3 depicts the final populations obtained by C-MOEA/D and CMOEA-MS on MW13 with two objectives. It can be clearly observed that neither C-MOEA/D and CMOEA-MS cannot find all feasible regions. This means that existing CMOEAs may lead to invalid searching when handling the CMOPs with small feasible regions.

There are also some other noticeable problems, where the constrained landscape is significantly complex, such as the discrete feasible region with a huge infeasible barrier and only several narrow feasible regions. As a result, these widespread difficulties may lead CMOEAs to obtain a bad convergence or diversity of the population. To solve the abovementioned issues, the proposed CMOSMA uses a novel self-organizing map approach for solving CMOPs, which has higher search efficiency than the state-of-the-art CMOEAs. In the following sections, the main components of the proposed CMOSMA will be described in detail.

**Proposed framework**

**Main framework of CMOSMA**

To describe the procedure of CMOSMA in detail, Fig. 4 depicts the flow chart of its complete framework. In CMOSMA, we simultaneously maintain two collaborative and complementary populations. Specifically, FP is mainly responsible for driving the population toward the feasible region. Therefore, FP is evolved by considering all the constraints and objectives. Different from FP, AP is a complement, which is used to assist in exploring the areas that have not been exploited by the FP. As shown in Fig. 4, the proposed CMOSMA begins with the random initialization of two populations FP and AP with size $N$. Then, two SOMs are initialized. During the initialization of SOM1, we assign each neuron weight vector a randomly chosen training point from FP. As for SOM2, the same initialization strategy is adopted. In the reproduction phase, two mating pools are selected from FP and AP by the mating selection strategy, respectively. Afterwards, each of the mating pools is used to generate an offspring population by the reproduction operators. Then, both FP and AP are combined with the offspring populations, and further truncated by the environmental selection strategy. In this way, useful information is shared between the two populations. In other words, CMOSMA shares the offspring generated by all the populations to assist in solving
Fig. 2  a Feasible region formed by C1-DTLZ3, b final populations obtained by CMOEA-MS

Fig. 3  a Feasible region formed by MW13, b final populations obtained by C-MOEA/D, c final populations obtained by CMOEA-MS
Next, the SOM update strategy is used to iteratively update weight vectors. The above steps are repeated until a termination condition is met. Finally, the FP is reported as the final output.

In the next section, the main components of the proposed CMOSMA method are described in detail.

**Initialization**

The initialization of CMOSMA is essential, which plays an important role in the process of optimization. The initialization of the population is relatively simple, and two populations FP and AP with the size of N are randomly generated. We mainly focus on the initialization strategy of SOM. In CMOSMA, SOM is applied to explore the manifold structure of PS. Therefore, we need to initialize the neuron weight vector of SOM. The initialization of SOM is based on the current population. In the initialization stage, every neuron of SOM network is assigned with a weight vector randomly. For example, when initializing SOM1, the weight value of each neuron is assigned by randomly selecting a solution from FP. The initialization of SOM2 also uses the same strategy. In addition, we also need to select reasonable numerical values for the free parameters during the initialization phase.

**Mating selection and offspring reproduction**

Recent reports show that individuals with similar structural information can generate higher-quality offspring [35, 36]. This is mainly because the PS of a CMOP is an \((m - 1)\)-dimensional piecewise continuous manifold. Therefore, the generated offspring by individuals with similar structure can explore decision space more efficiently. To improve the quality of new solutions, the proposed CMOSMA uses the SOM to explore the population distribution structure in the decision space and utilizes the extracted neighborhood relationship information to guide the generation of offspring population at each generation. Specifically, we first partition the FP into several clusters according to the trained SOM1 network, which makes the individuals in each cluster have similar structural information. Such neighborhood relationship is established by the unsupervised learning method, so it has good reliability and can be applied to different types of CMOPs. Then, the offspring of FP is generated by restricting the recombination within neighboring solutions based on the discovered neighborhood relationship information. The offspring generation process of AP is similar to that of the FP. To better illustrate this fact, Fig. 5 shows the procedure of mating selection and offspring reproduction strategy in CMOSMA. As shown in Fig. 5, we first perform a clustering operation on the population. Each of the individual associates a neuron (find the closest neuron), and similar individuals are assigned to adjacent neurons. Then, we create a mating pool based on
the clustering results. Individuals have a higher probability
to generate offspring with individuals with similar structure
information. As a result, the topological properties of the pop-
ulation are introduced into the offspring generation process
improve the quality of new solutions.

The details of the above procedure are given in Algorithm
1. In Algorithm 1, we use the simulated binary crossover
(SBX) and polynomial mutation (PM) \cite{10} to generate new
solutions. It should be noted that, in the offspring gen-
eration, any commonly adopted recombination operators such
as differential evolution (DE) \cite{17} and other novel genetic
operators \cite{37} can be applied to reproduce promising off-
spring solutions.

**Algorithm 1 Mating Selection and Offspring Reproduction**

\begin{algorithm}[h]
\caption{Mating Selection and Offspring Reproduction \((P, SOM)\)}
\begin{algorithmic}
\Statex \textbf{Input:} \(P = (x^1, x^2, \ldots, x^N)\) (finite set of feasible solutions
under consideration), \(SOM\) (self-organizing map), \(N\) (size
of population), \(H\) (Size of neighborhood mating pools) The
neuron weight vector \(w = (w^1, w^2, \ldots, w^N)\). Let \(u^k\) be the
index of the \(k^{th}\) nearest neuron to neuron \(u\) in the latent space.
\Statex \textbf{Output:} \(P'\) (Offspring)
\end{algorithmic}
\end{algorithm}

1: Set \(S = P, P' = \emptyset \) and \(U = (1, \ldots, N)\).
2: \While{\(S \neq \emptyset\)}
3: \hspace{1em} Randomly choose \(x \in S\) and set \(S = S\setminus x\).
4: \hspace{1em} Set \(u^k = x\) where \(u = \arg\min_{u' \in U} \|x - w^{u'}\|_2\).
5: \hspace{1em} Set \(U = U\setminus u\).
6: \EndWhile
7: \For{\(u = 1\) to \(N\)}
8: \hspace{1em} Set the mating pool \(Q\) for \(x^u\) as
9: \hspace{1em} \If{\(rand() < 0.9\)}
10: \hspace{1em} \hspace{1em} \(Q = \cup_{k=1}^H x^{u^k}\).
11: \hspace{1em} \Else
12: \hspace{1em} \hspace{1em} \(Q = P\).
13: \hspace{1em} \EndIf
14: \hspace{1em} Generate new solutions \(y = \text{Generate}(x^u, Q)\).
15: \hspace{1em} \(P' = P' \cup y\).
16: \EndFor
17: \Return \(P'\).

**Environmental selection**

Similar to most existing CMOEAs, the CMOSMA also
adopts the elite strategy by performing the environmental
selection to select the solutions for passing to the popula-
tion of the next generation from the combined population.
In CMOSMA, we adopt the environmental selection strategy of
SPEA2 \cite{38, 39} to truncate the combined population and maintain population diversity. It is worth noting that FP
and AP implement different environment selection strategies
according to the properties of the constraint problem. In
the process of FP environmental selection, the constrained
dominance and crowding distance are introduced to calculate
fitness. Then, we select reserved solutions for the next
 generation by the fitness values and the truncation method.
This means that FP locates in the feasible regions and feasible solutions are guaranteed a higher priority to pass to the next generation FP. In contrast, without considering any constraint, the individuals in AP are closer to the ideal point than those in FP. The main reason for this is that the AP only considers objectives, leading to the individuals with better convergence and diversity are easier to survive. This strategy of cooperative evolution of two populations can significantly improve the search efficiency, thus making more effective the evolutionary search toward the true Pareto Front. The detailed procedure of the environmental selection in the proposed CMOSMA is given in Algorithm 3.

Algorithm 3 Environmental Selection ($P$, $N$)

Input: $P = (x^1, x^2, ..., x^N)$ (finite set of feasible solutions under consideration), $N$ (size of population).

Output: $Q$ (next population)
1: $Q = \emptyset$.
2: $FV \leftarrow$ Fitness Assignment ($P$).
3: for $i = 1$ to $|P|$ do
4: if $FV(i) < 1$ then
5: $Q \leftarrow Q \cup x_i$.
6: end if
7: end for
8: if $|Q| > N$ then
9: while $|Q| > N$ do
10: Calculate the crowding distances of remaining individuals.
11: $x' \leftarrow$ Find the solution with the worst crowding distances from $P$.
12: $Q = Q \setminus x'$.
13: end while
14: end if
15: return $Q$.

SOM training

The purpose of SOM training is to find the representation weight vectors of the neurons that can effectively estimate the manifold structure of PS. In CMOSMA, the training step of SOM adopts the unsupervised learning approach does not require a lot of data for pre-training. After implementing the environmental selection, we use the surviving offsprings to update the neuron weight vectors in SOM. Algorithm 4 details the procedure of SOM training strategy in CMOSMA. In Algorithm 4, $\delta_0$ and $\tau_0$ are two related parameters used in the SOM training strategy. Among them, $\delta_0$ represents the width of the initial neighborhood radius, and $\tau_0$ is the initial learning rate. Before the weight vectors are updated, we first need to create a training data set. During the new data set creating process, the SOM procedure is not restarted. Because each retraining of the SOM network requires a mass of data, which will increase the computational complexity and time consumption. In line 1, we only use the offsprings who survive to the next generation to build new training data set, which means that the previously learned distribution information of the data is reused. After that, the width of the neighborhood radius $\delta$ and learning rate $\tau$ is updated. Then, in lines 2–11, we find the nearest neuron for each training point. The weight of the nearest neuron and its neighboring neurons are updated by the training point. At last, the trained SOM network will be returned by Algorithm 4. This unsupervised learning method can significantly reduce the complexity of the SOM network for manifold structure estimation and is very suitable for handling different types of CMOPs. A noteworthy phenomenon is that there is a special case in the SOM training process, that is, the training data is empty. In this case, no training will happen.

Algorithm 4 SOM Training ($P_t, P_{t+1}, SOM$)

Input: $P_t$ (The population at the $t$-th iteration),$P_{t+1}$ (the population for the $(t + 1)$-th iteration), SOM (self-organizing map), $N$ (size of population), $T$ (The maximum number of iterations), $w = (u^1, u^2, ..., u^N)$ (The location of neurons in latent space). Let $u^k$ be the index of the $k^{th}$ nearest neuron to neuron $u$ in the latent space.

Output: $SOM$
1: Create the training data: $S = P_{t+1} \setminus P_t$.
2: for $x^* \in S, s = 1, ..., |S|$ do
3: Update the training parameters:
4: $\sigma = \sigma_0 \ast (1 - \frac{t - 1}{T - 1})$.
5: $\tau = \tau_0 \ast (1 - \frac{t - 1}{T - 1})$.
6: Find the closest neuron to $x^*$:
7: $u = \text{arg min}_{1 \leq u \leq N} ||x^* - u^u||_2$.
8: Locate the neighboring neurons:
9: $U = \{u|1 \leq u \leq N \land ||x^* - u^u||_2 < \delta \}$.
10: Update all neighboring neurons ($u \in U$) as
11: $u^u = u^u + \tau * exp(-||x^* - u^u||_2)(x - u^u)$.
12: end for
13: return $SOM$.

Computational complexity

Although CMOSMA adopts the two populations strategy, it has a quite simple framework. Since CMOSMA evolves two populations with the same strategies, the worst-case time complexity of the mating selection, offspring generation, environmental selection, and SOM training of CMOSMA is $2 \times O(N), 2 \times O(ND), 2 \times O(MN^2)$, and $2 \times O(N^3)m$ denotes the number of objectives, $D$ is the number of decision
variables and $N$ represents the size of population). Therefore, the overall computational complexity of the proposed CMOSMA is $O(MN^2)$. Since we perform each search strategy twice a generation, the proposed CMOSMA method requires more time consumption than the CMOEA that has the same worst-case time complexity.

**Analysis of CMOSMA**

In the genetic operator, the offsprings are usually generated via the crossover technique as follows.

Parent1 : $a = (x_1, x_2, \ldots, x_n)$ crossover
Parent2 : $b = (y_1, y_2, \ldots, y_n)$

offspring1 : $c = (\delta x_1 + (1 - \delta)y_1, \ldots, \delta x_n + (1 - \delta)y_n)$
offspring2 : $d = (\delta y_1 + (1 - \delta)x_1, \ldots, \delta y_n + (1 - \delta)x_n)$

where $\delta \in (0, 1)$ is a learning factor. Existing CMOEAs mainly use random matching methods to construct a mating pool, which means that parents are randomly selected from the current population. This random crossover technique is very effective in solving single-objective optimization problems (SOPs). Because when the population is close to the optimal solution, the offspring generated by random parent recombination still have a large probability distribution around the optimal solution. Different from the SOP, the MOP/CMOP usually obtains a set of tradeoff solutions due to the conflicting nature between the multiple objectives. Without loss of generality, the distribution of the Pareto set for a continuous $m$-objective CMOP is a piecewise continuous manifold, which has a dimension of $m - 1$ (Theorem 1). In this case, the offspring solutions generated via two randomly selected parents may be far away from the PS. Let us consider a simple example as shown in Fig. 6a, where $a$ and bare two solutions randomly selected from the population. In this example, PS is a continuous curve. Since Eq. 5 is essentially a linear combination, its offsprings are located between the two parents. As shown in the figure, we can find that the offsprings of $a$ and bare far away from PS. This situation will increase the invalid search and weaken the search efficiency. It is obvious that the traditional offspring reproduction strategy has randomness and lacks sufficient pressure toward the PS. Therefore, CMOEAs need a more efficient reproduction strategy to improve the search efficiency for solving CMOPs.

In the proposed CMOSMA, we use SOM to discover the population distribution structure and create the mating pool to generate new solutions. An illustrative example of our reproduction method is given in Fig. 6b. It can be seen from the figure that our generated offspring solutions are better than those generated by the traditional reproduction strategy. On the one hand, the parent solutions with similar structural information can reproduce high-quality offspring solutions. The generated offspring are close to PS, and thus the local search capability of CMOSMA is promising. On the other hand, these generated solutions are distributed around PS, which ensures the diversity of the population to some degree. It is worth mentioning that Euclidean distance cannot be used as a criterion for judging individual similarity, because the distance measure will be invalid in high-dimensional space. By contrast, SOM has the advantage of preserving the topological properties of the population, and can well explore the manifold structure of PS by unsupervised learning.

To examine the benefits of SOM strategy in our method for improving the search efficiency in solving CMOPs, we first utilize four 2-objective test problems to conduct an ablation experiment. Figure 7 shows the comparison results of CMOSMA method with different modeling choices, where the CMOSMA and CMOSMA-I respectively represent the CMOSMA method with or without the SOM component. As shown in the figure, the CMOSMA performs faster convergence speed than the CMOSMA-I in almost all test problems. This indicates that the SOM component can improve search efficiency and increase the speed of convergence. Furthermore, an interesting phenomenon is observed that the IGD values of CMOSMA are slightly higher than that of CMOSMA-I in the early stage of some test problems. This finding is unsurprising and is mainly because the SOM training needs to consume more evaluation functions in the early stage.

Generally, complex constraint problems usually contain diverse constraints, and these different constraints collaboratively form a complex constraint landscape. C2-DTLZ2, DC3-DTLZ1 and MW14 are three representative complex constraint problems. Among them, C2-DTLZ2 and MW14 have irregular feasible regions, and DC3-DTLZ1 has a narrow feasible region on the top. Furthermore, their feasible regions are discrete. Therefore, these instances can be challenging for existing CMOEAs. In contrast to existing two-population/stage CMOEAs, our method adopts a two-SOM collaboration framework based on two-population. Figure 8 displays the feasible and non-dominated solutions obtained by CMOSMA on the above problems. As shown in Fig. 8, the proposed CMOSMA can converge toward the global PFs and maintain the diversity of the population. The non-dominated solutions of CMOSMA cover all feasible regions. The good performance of CMOSMA is benefiting from the excellent ability of the proposed framework in searching feasible regions and maintain the diversity of the population.

To further demonstrate the benefits of the proposed CMOSMA for handling CMOPs with complex constraints, we run our method and three state-of-the-art methods on MW13. As for MW13 instance, whose feasible regions are continuous but disjoint from the unconstrained PF.
Offspring solutions generated by traditional offspring reproduction strategy and our method. a Traditional offspring reproduction strategy, b our method.

Fig. 6

![Offspring solutions](image)

Convergence profiles of IGD values obtained by CMOSMA and CMOSMA-I on four 2-objective test problems. a MW2, b MW6, c LIR-CMOP3, d LIR-CMOP6

Fig. 7

![Convergence profiles](image)

Feasible and non-dominated solutions obtained by CMOSMA on C2-DTLZ2, DC3-DTLZ1 and MW14

Fig. 8

![Feasible and non-dominated solutions](image)

Figure 9a plots the convergence profiles of IGD values obtained by CMOSMA, C-MOEA/D [9], CMOEA-MS [18] and MSCMO [23] on MW13, averaged over 30 runs. As shown in Fig. 9a, we first find that the IGD curve of MSCMO is relatively smooth, which indicates that MSCMO has not found the true Pareto front (PF). For CMOEA-MS and C-MOEA/D, they converge slowly and thus have failed to achieve an acceptable accuracy level by the end of the evolution. In contrast, the proposed CMOSMA is significantly faster than the other compared algorithms, especially in the early stage. To visually show the optimization process of CMOSMA for CMOPs with complex constraints, Fig. 10 plots the populations in the early, middle, and last generations of the compared CMOEAs on MW13. As can be observed, the population of MSCMO cannot find the whole feasible region in the search process and only cover a small part of the PF. This is because MSCMO considers the constraints one by one and divides the process of solving CMOPs into multiple stages. This will cause MSCMO to prefer feasible solutions that satisfy some constraints during the evolution process, and these solutions have easy trouble in the local optimal feasible region. For the populations of CMOEA-MS and C-MOEA/D, they have similar performance to CMOSMA in the early generations. All solutions in the population are located in the feasible regions. However, CMOEA-MS and C-MOEA/D do not find the global optimum in the middle.
and last generations. The reason that may make C-MOEA/D ineffective for the CMOP with complex constraints is that the constrained dominance principle is overemphasized in the evolution process. In C-MOEA/D, the feasible solutions always dominate the infeasible solutions, which makes the search very difficult to jump out of local optima. For CMOEA-MS, we need to set a user-defined parameter to decide when to switch from stage A to stage B. Unfortunately, we are difficult to find an appropriate value for different types of benchmarks CMOPs. In Fig. 10, some generated solutions of CMOEA-MS even move away from the PF, which will reduce the exploration ability in decision space. Different from CMOEA-MS and C-MOEA/D, the proposed CMOSMA have successfully found all the feasible regions of MW13. On the one hand, CMOSMA considers the regularity property of continuous CMOP. The Two-SOM collaborative framework is developed to extract neighborhood relationship information of the population and use such information to generate promising offspring solutions. On the other hand, CMOSMA adopts the strategy of two population evolution. In the early generations, FP only searches some feasible regions and these regions are far away from the PF. On the contrary, AP have a good spread since it is evolved without considering the constraints. Although part of the FP solution has trouble in the top local optimal, AP can help FP quickly jump over the local optimal and infeasible band. AP is used to assist in exploring the areas that have not been exploited by the FP. Therefore, once the AP finds more promising feasible solutions, these solutions can be entered into the next generation of FP through an environment selection strategy. As shown in the second column of Fig. 10, we find that AP and FP converge to the global PF through cooperative search. As a consequence, it is obvious from the figure that the solutions of CMOSMA can find all the current feasible regions with good diversity and converge to the global optimum in the last generations.

To further analyze the performance of the proposed CMOSMA, Fig. 9b plots the trajectories of IGD with generations for CMOSMA, C-MOEA/D, CMOEA-MS and MSCMO on DC1-DTLZ3 that have the disconnected feasible regions. As illustrated in Fig. 9b, we can see that CMOSMA is significantly faster than the other compared algorithms during the whole evolutionary process. CMOSMA has already converged to a promising accuracy level at a very early stage of its evolution. The comparison results in Fig. 9b indicate that the proposed framework has a fast convergence speed and low IGD value for solving CMOPs. For a visual comparison, Fig. 11 shows the solution sets obtained by CMOSMA and three compared CMOEAs on DC1-DTLZ3. As shown in Fig. 11, the feasible and nondominated solutions obtained by our method are a significant improvement compared with those of other evaluation approaches, as evidenced by our solution set, which is closer to the shape of the global optimum. Combined with the analysis of the previous paragraph, it can be summarized that the proposed approach is effective in dealing with complex constraint problems.

**Experimental design**

This section is devoted to the experimental design for verifying the performance of the proposed CMOSMA. First, we briefly describe the benchmark problems used in the experiments. Then, some representative CMOEAs used for evaluating the performance of CMOSMA are briefly introduced. Finally, both the parameter settings of each algorithm and other experimental settings are presented.

**Benchmark problems**

A total of 47 benchmark problems, including the C-DTLZ test suite [9], the MW test suite [34], DC-DTLZ test suite [8]
Fig. 10 Populations in the early, middle, and last generations of CMOSMA, C-MOEA/D, CMOEA-MS and MSCMO on 2-objective MW13

Fig. 11 Feasible and nondominated solutions with the median IGD values among 30 runs on DC1-DTLZ1, shown by parallel coordinates. a CMOSMA, b C-MOEA/D, c CMIEA-MS, d MSCMO, e True PF

and the LIR-CMOP test suite [40], which are widely used to evaluate the performance of the algorithm. These test suites include the problems with various complex PFs (e.g., the feasible region is discrete or the size of the feasible region is very small), which pose a significant challenge to existing CMOEAs in finding feasible nondominated solutions. As suggested in [23], the number of decision variables $D$ is set to 10 for LIR-CMOP problems. For C1-DTLZ1, DC1-DTLZ1, DC2-DTLZ1, and DC3-DTLZ1, $D$ is set to 12, and $D = 15$ for the remaining constrained C-DTLZ and DC-DTLZ problems. As recommended in [41], we set $D = 10$ for LIR-CMOP test suite. For each test problem, we consider the number of objectives $m \in \{2, 3\}$.

**Algorithms for comparison**

To investigate the performance of the proposed CMOSMA in handling CMOPs, we carried out a series of simulation
experiments and compared it with six well-known CMOEAs:

(1) MSCMO [23]: MSCMO is a multi-stage evolutionary algorithm, where constraints are added one after the other and handled in different stages of evolution.

(2) C-TAEA [8]: C-TAEA is a two-archive algorithm following Two_Arch [16] to tackle the CMOPs.

(3) CMOEA-MS [18]: CMOEA-MS is a novel algorithm, which adaptively balance objective optimization and constraint satisfaction during the evolutionary process.

(4) ToP [22]: ToP suggests a brand-new two-phase framework, which has an excellent performance in finding the feasible region.

(5) TiGE-2 [4]: TiGE-2 is a popular CMOEA which uses a tri-goal evolution framework to balance the convergence, diversity and feasibility of the evolutionary process.

(6) C-MOEA/D [9]: C-MOEA/D can be regarded as a variation of MOEA/D to deal with CMOPs, by using a constraint dominance principle.

(7) PPS [17]: PPS is a two-stage evolutionary algorithm, where a push and pull search strategy is proposed to adaptively handle different CMOPs.

The above algorithms are shown to be effective in solving CMOPs, making the comparisons more comprehensive.

**Experimental settings**

(1) Performance metrics:

In this report, the inverted generational distance (IGD) [42] is employed to evaluate the performance of various approaches, using the formula in the following:

\[
\text{IGD}(P, P^*) = \frac{\sum_{z \in P^*} \text{dist}(z, P)}{|P^*|},
\]

where \(P^*\) denotes the set of points uniformly distributed along with the PF, \(P\) indicates the nondominated solution set obtained by an algorithm, \(\text{dist}(z, P)\) is the Euclidean distance between \(z\) and its nearest neighbor in \(P\), and \(\text{dist}\) represents the \(L_1\)-norm. IGD can measure the convergence and diversity between the solution sets and PF. Regarding the IGD metric, the smaller the value is, the better performance of the algorithm. Furthermore, to demonstrate the capability of our method in solving CMOPs, the Wilcoxon signed-rank test [43] at a 5% significance level is performed to analyze the significant performance differences between CMOSMA and other seven popular CMOEAs. To make a fair comparison, all the test instances are run for 30 times independently, and the mean and standard deviation of each metric value are collected for comparison purposes.

(2) Population size and termination criterion:

According to [41], for the test problems of 2, and 3 objectives, we set the population size for all the compared algorithms to 100, 105, respectively. For a quantitative comparison, the total number of function evaluations is adopted as the termination criterion. Following the suggestion in [41], the maximal number of evaluations is set to 100,000 for the C-DTLZ and MW problems. For DC-DTLZ and LIR-CMOP test instances, we set the maximal number of function evaluations to 300,000.

(3) Parameters in the compared CMOEAs:

In this paper, the parameters of all compared algorithms are set according to their original papers. C-TAEA, MSCMO, TOP, TiGE-2, CMOSMA and CMOEA-MS adopt the simulated binary crossover operator and polynomial mutation to generate offspring, so they have several common parameters. Following the suggestion in [10], the distribution indexes of both SBX and polynomial mutation are set to 20. In addition, the crossover and mutation probabilities are set to be 1 and 1/D, respectively. According to the suggestion of [17], the differential evolution operator with \(CR = 1\) and \(F = 0.5\) are adopted to generate offspring in C-MOEA/D and PPS. Furthermore, the neighborhood size \(T\) is set to 20 in C-MOEA/D and PPS. Regarding CMOEA-MS, the parameter \(\lambda\) for determining the current stage is set to 0.5. For the TiGE-2, the scaling factor \(\rho\) is set to 0.05 and the penalty factor \(\varepsilon\) is set to 1.01. In the ToP framework, the feasibility proportion \(P_f\) is larger than 1/3 or the difference \(\delta\) is less than 0.2. For the proposed CMOSMA approach, the initial SOM learning rate is set to 0.7, the initial neighborhood radius is set to \(\sqrt{\frac{\sum_{i=1}^{m} N_i^2}{(m-1)}}\) \(m\) denotes the number of objectives and \(N\) represents the size of neighborhood mating pools is set 5, and probability of mating restriction within the neighborhood is set 0.9. Moreover, all the parameters of CMOSMA are specified following suggestions in [10, 27]. Therefore, we do not discuss the influences of the free parameters on the performance of the proposed approach in this paper.

(4) Other parameters:

The source codes of evaluated algorithms are provided in the PlatEMO [44]. All the compared MOEAs in this paper are implemented by a DELL computer equipped with an Intel Core I9-9880H CPU and an NVIDIA RTX 5000 GPU.
Experimental results

In this section, a series of experiments are conducted to verify the performance of CMOSMA in handling CMOPs.

C-DTLZ benchmark suite

To show the superiority of our approach in dealing CMOPs, we first compare our method with several state-of-the-art methods on C-DTLZ benchmark suite. Table 1 presents the IGD values obtained by CMOSMA and seven popular CMOEAs on C1-DTLZ1, C1-DTLZ3, C2-DTLZ2 and C3-DTLZ4. According to Table 1, we can find that CMOSMA significantly outperforms the other CMOEAs in terms of the IGD values. Specifically, for C1-DTLZ1 instance, it is a relatively simple constraint problem, in which the feasible region is simple and contains only one constraint. Therefore, all compared CMOEAs can yield good results on C1-DTLZ1. For C1-DTLZ3 with many local optima of a constraint violation, CMOSMA and PPS outperformed the rest of the algorithms with respect to their IGD values. This is mainly because PPS has a two-stage framework, its bi-phase search process can help the population jumps out of local optima. Different from PPS, CMOSMA adopts a two-population strategy, where each population cooperates with the other to escape from the local optimum. As for the C2-DTLZ2, CMOSMA also exhibits excellent performance. For C3-DTLZ4 instance, the median IGD value by CMOSMA is higher than C-MOEA/D on C3-DTLZ4. This finding is unsurprising and is mainly because the decomposition-based method is very suitable for dealing with regular PF such as that of C3-DTLZ4. In addition, according to the Wilcoxon rank-sum test in Table 1, it is obvious that CMOSMA achieves the best performance among all of the evaluated methods.

Performance comparisons on MW test suite

Recently, MW test suite has become increasingly popular in evaluating the performance of various CMOEAs. MW test suite includes not only the problems with simple feasible regions (such as MW1 and MW12) but also problems with complex feasible regions that are narrow, discrete and so on (i.e., MW5, MW11 and the MW13). These problems thus pose a great challenge for CMOEAs to obtain a distribution of feasible solutions with suitable levels of diversity and convergence. To further assess the performance of the proposed CMOSMA, we run our method on the MW test suite to conduct a qualitative comparison with the state-of-the-art.

![Table 1: Quantitative comparison between the median IGD values for 30 independent runs of various methods on C1-DTLZ1, C1-DTLZ3, C2-DTLZ2 and C3-DTLZ4](image-url)
Table 2 presents the IGD values obtained by the CMOSMA and other compared CMOEAs on the MW test suite for 30 independent runs. As shown in Table 2, we find that the proposed CMOSMA produced the competitive results compared with the other state-of-the-art approaches, especially gained the overwhelmingly superior performance in instances with complex constraints. For example, for CMOPs with narrow, discrete, or irregular feasible regions, such as the MW5, MW11 and MW13 problems, our method can deal with the above problems quite well. Moreover, it can observe that the C-TAEA obtains the best result on MW6. The main reason is that the selection mechanisms of the convergence-oriented archive and the diversity-oriented archive can help C-TAEA obtains good IGD values on problems with a simple feasible region. However, for problems with complex feasible regions C-TAEA cannot find a sufficient number of feasible and well-converged solutions.

For a visual comparison, Fig. 12 depicts the populations with median IGD value achieved by each algorithm on MW5. As illustrated in Fig. 12, we can observe that MW5 instance includes the discontinuous feasible regions and the optimal solution is only a few points on the unconstrained PF. Therefore, MW5 poses a great challenge for the existing CMOEAs to obtain all the feasible regions. For ToP, it exhibits the worse performance of convergence in MW5. This means that the ToP framework has low search efficiency when solving constraint problems with discontinuous feasible regions. MSCMO has successfully converged to the true PF, but its solution set contains too many infeasible nondominated solutions. TiGE-2 and C-TAEA obtain similar experimental results on MW5. It can be seen from the figure that neither TiGE-2 nor C-TAEA converges to the optimal feasible region, and the proportion of nondominated solutions in the solution set is relatively low. As for C-MOEA/D, it cannot find feasible solutions with the optimal objective value, and part of the optimal feasible solutions are concentrated at the edge of the feasible region. Although both CMOEA-MS and PPS can find the entire feasible region, their solution set only contains part of the optimal feasible solutions. Being different from the above-compared algorithms, CMOSMA performs well in terms of searching efficiency and population diversity. It is obvious from the figure that the feasible solutions obtained by CMOSMA are much closer to the True PF than the other peer algorithms. For further observations, Fig. 13 depicts the final solutions obtained by compared methods on MW10 problem. As can be seen from Fig. 13, we find that all evaluated algorithms except for ToP and PPS...
| Problem | Obj | Publication year | MSCMO | CMOEA-MS | TiGE-2 | C-MOEA/D | ToP | PPS | C-TAEA | CMOSMA |
|---------|-----|----------------|-------|----------|--------|----------|-----|-----|--------|--------|
| MW1     | 2   | 2021           | 3.5944e-3 | 1.8481e-3 | 7.4487e-2 | 4.1569e-3 | NaN (NaN) | 5.6703e-3 | 2.0087e-3 | 1.6539e-3 |
|         |     | (1.23e-4) - 4) | (9.23e-2) - (1.88e-3) | (2.198e-2) - (1.4235e-1) | (7.38e-2) - (5.08e-2) | (1.26e-2) - (1.95e-2) |
| MW2     | 2   | 2021           | 2.2059e-2 | 2.2133e-2 | 6.3623e-2 | 2.1232e-2 | 1.2305e-1 | 1.8055e-1 | 1.5891e-2 | 9.5036e-3 |
|         |     | (8.06e-3) - 3) | (1.30e-1) - (6.69e-3) | (9.01e-2) - (1.17e-1) | (1.03e-1) - (7.04e-2) | (1.26e-2) - (1.95e-2) |
| MW3     | 2   | 2020           | 5.1517e-2 | 5.3577e-3 | 2.9223e-2 | 5.4438e-3 | 5.1377e-1 | 6.3263e-3 | 5.0606e-3 | 4.9293e-3 |
|         |     | (7.68e-4) - 4) | (2.29e-2) - (4.56e-4) | (3.91e-1) - (5.34e-4) | (3.55e-1) - (3.32e-4) | (2.06e-4) |
| MW4     | 3   | 2014           | 5.0641e-2 | 4.0741e-2 | 1.0755e-1 | 3.8207e-2 | 5.6568e-1 | 6.0966e-2 | 4.3029e-2 | 3.7272e-2 |
|         |     | (2.27e-3) - 4) | (4.24e-2) - (5.19e-4) | (2.02e-3) - (7.04e-2) | (7.04e-2) - (3.15e-3) | (1.48e-3) |
| MW5     | 2   | 2019           | 5.3060e-1 | 2.9956e-2 | 4.7383e-2 | 1.0304e-2 | 8.7586e-1 | 2.8332e-1 | 1.2558e-2 | 5.2333e-4 |
|         |     | (1.72e-3) - 2) | (1.05e-2) - (4.60e-2) | (0.00e+0) - (3.54e-1) | (2.75e-3) - (3.15e-4) | (4.15e-4) |
| MW6     | 2   | 2017           | 4.2022e-2 | 2.1521e-2 | 2.4129e-1 | 3.2787e-2 | 6.4756e-1 | 4.4913e-1 | 8.3680e-3 | 8.6580e-3 |
|         |     | (9.02e-2) - 2) | (3.28e-1) - (8.80e-2) | (2.89e-1) - (3.35e-1) | (7.81e-2) - (4.75e-4) | (4.75e-4) |
| MW7     | 2   | 2019           | 2.0637e-1 | 2.4660e-2 | 3.5054e-2 | 4.6962e-3 | 4.8890e-2 | 5.6463e-3 | 7.0006e-3 | 4.8012e-3 |
|         |     | (2.13e-3) - 2) | (1.65e-2) - (1.29e-4) | (7.21e-2) - (5.22e-4) | (7.81e-2) - (4.75e-4) | (5.26e-4) |
| MW8     | 3   | 2014           | 6.4496e-2 | 4.9104e-2 | 5.1375e-1 | 5.3426e-2 | 5.3519e-1 | 1.4399e-1 | 5.3935e-2 | 4.3792e-2 |
|         |     | (8.74e-3) - 2) | (2.62e-1) - (5.61e-3) | (3.42e-1) - (5.93e-2) | (1.73e-1) - (1.38e-3) | (4.43e-3) |
| MW9     | 2   | NaN (NaN)      | 1.4417e-1 | 9.2191e-2 | 8.8945e-3 | 5.3542e-1 | 3.5165e-2 | 8.4277e-3 | 4.4345e-3 | 3.108e-3 |
|         |     | (1.29e-2) - 1) | (1.93e-1) - (3.03e-3) | (3.65e-1) - (6.33e-2) | (6.22e-2) - (4.30e-4) | (9.99e-3) |
| MW10    | 2   | 6.3998e-2      | 7.0414e-2 | 9.1467e-2 | 7.1105e-2 | 9.1467e-2 | 3.9449e-1 | 1.8242e-2 | 1.3685e-2 | 6.0472e-3 |
|         |     | (3.89e-2) - 1) | (1.09e-1) - (6.65e-2) | (1.35e-1) - (1.35e-1) | (1.35e-1) - (1.35e-1) | (3.95e-3) |
| MW11    | 2   | NaN (NaN)      | 4.0332e-2 | 3.3181e-2 | 2.7262e-1 | 6.3187e-1 | 7.3488e-3 | 1.3888e-2 | 6.0472e-3 | 2.55e-4 |
|         |     | (1.38e-2) - 1) | (5.23e-3) - (3.33e-1) | (1.90e-1) - (3.76e-4) | (1.58e-3) - (1.58e-3) | (2.55e-4) |
| MW12    | 2   | NaN (NaN)      | 5.2521e-3 | 7.2366e-2 | 5.2948e-3 | 7.8081e-1 | 2.0496e-2 | 7.9085e-3 | 4.8183e-3 | 9.51e-5 |
|         |     | (2.62e-2) - 4) | (1.73e-1) - (2.85e-3) | (6.64e-2) - (4.84e-2) | (5.26e-4) - (5.26e-4) | (9.51e-5) |
| MW13    | 2   | 2.0395e-1      | 1.1695e-1 | 5.6194e-1 | 7.4165e-2 | 6.5971e-1 | 3.3232e-1 | 3.0902e-2 | 2.3573e-2 | 1.76e-2 |
|         |     | (1.12e-1) - 2) | (1.84e-1) - (2.76e-2) | (5.08e-1) - (2.58e-1) | (1.95e-2) - (1.95e-2) | (1.76e-2) |
| MW14    | 3   | 1.1776e-1      | 1.2521e-1 | 1.6045e-1 | 2.1232e-1 | 3.8620e-1 | 1.3800e-1 | 1.1151e-1 | 1.0022e-1 | 1.79e-3 |
|         |     | (1.87e-3) - 2) | (1.52e-2) - (4.04e-1) | (1.26e-2) - (3.05e-3) | (1.79e-3) | (1.79e-3) |

The best result in each row is highlighted in bold.
can converge to feasible regions. Unfortunately, the populations obtained by CMOEA-MS, TiGE-2, C-MOEA/D and C-TAEA cannot cover all feasible regions. For MSCMO with a multi-stage framework, it only finds a nondominated solution set. The main reason may be that the strategy of iterative optimization constraints requires more evaluation functions. Although both C-TAEA and CMOSMA adopt the two-population evolution strategy, C-TAEA cannot find the top of the feasible solutions with the optimal objective value. In contrast, CMOSMA shows significantly outstanding performance and finds the global optimal solutions. This shows that the method of generating solutions through the SOM framework has stronger global search capabilities.

### Performance comparisons on DC-DTLZ and LIR-CMOP test suite

To further examine the capability of CMOSMA in dealing with CMOPs, we evaluate our method on DC-DTLZ and LIR-CMOP test suite because the DC-DTLZ and LIR-CMOP test suite include discrete, narrow, irregular and complex feasible regions. Table 3 summarizes the IGD results for DC-DTLZ and LIR-CMOP problems, indicating that the proposed CMOSMA achieves the competitive performance among the compared algorithms. Although PPS produces the best results on some problems (such as LIR-CMOP1-7), it suffers from considerable performance deterioration on problems with complex constraints. For example, the mean IGD values of PPS are worse than CMOSMA on LIR-CMOP9 with only several narrow feasible regions and LIR-CMOP10...
| Publication year | 2021 | 2021 | 2020 | 2014 | 2019 | 2017 | 2019 |
|-----------------|------|------|------|------|------|------|------|
| **Problem**     |      |      |      |      |      |      |      |
| **Obj**         |      |      |      |      |      |      |      |
| **MSCM**        | 2.0959 e-2 | 1.6737 e-2 | 4.1882 e-1 | 2.1520 e-2 | 1.8208 e-2 | 1.8351 e-2 | 1.3635 e-2 | 1.0325 e-2 | 7.93 e-5 |
| **CMOEA-MS**    | (7.71 e-4) | (3.9 e-4) | (6.94 e-3) | (4.18 e-3) | (9.07 e-4) | (3.53 e-4) | (5.53 e-4) | (7.93 e-5) |      |
| **TIGE-2**      |      |      |      |      |      |      |      |      |      |
| **C-MOEA/D**    | 1.4830 e-1 | 1.1359 e-1 | 1.0099 e+0 | 1.1486 e-1 | 1.7345 e-1 | 1.3367 e-1 | 1.2231 e-1 | 1.1294 e-1 | 8.53 e-4 |
| **ToP**         | (7.11 e-3) | (4.0 e-4) | (1.74 e-1) | (5.0 e-1) | (8.52 e-2) | (1.36 e-3) | (1.23 e-3) | (1.36 e-3) |      |
| **PPS**         |      |      |      |      |      |      |      |      |      |
| **C-TAEA**      | 1.9629 e-2 | 2.0201 e-2 | 3.9152 e-1 | 1.2013 e-1 | 2.6319 e-2 | 2.1539 e-2 | 1.5836 e-2 | 1.34 e-4 |      |
| **CMOSMA**      | (3.9 e-4) | (4.0 e-4) | (3.60 e-2) | (9.0 e-4) | (8.4 e-4) | (1.90 e-4) | (1.34 e-3) | (4.34 e-3) |      |
| **LIR-CMOP1**   | 5.2173 e-2 | 1.5314 e-1 | 7.9553 e-1 | 6.3261 e-1 | 2.2400 e-2 | 5.7873 e-2 | 4.8261 e-2 | 4.34 e-3 |      |
| **Publication year** | 2021 | 2020 | 2014 | 2019 | 2017 | 2019 |      |
| **Obj**         | NaN | NaN | 3.014 e-1 | 3.014 e-1 | 3.014 e-1 | 3.014 e-1 |      |
| **LIR-CMOP2**   | 2.7718 e-2 | 3.084 e-2 | 7.6410 e-1 | 1.6800 e-1 | 2.1977 e-2 | 9.1576 e-3 | 6.1560 e-3 | 5.77 e-4 |      |
| **LIR-CMOP3**   | 2.9217 e-1 | 1.838 e-1 | 2.4764 e+0 | 1.0626 e+0 | 4.750 e-1 | 1.6412 e-1 | 1.5866 e-1 | 1.45 e-3 |      |
| **LIR-CMOP4**   | 3.1168 e-1 | 2.2294 e-1 | 2.7222 e-1 | 3.3861 e-1 | 5.21 e-1 | 2.73 e-1 |      |      |      |
| **LIR-CMOP5**   | 2.8739 e-2 | 2.1079 e-1 | 2.6548 e-1 | 3.0981 e-1 | 4.99 e-1 | 2.2486 e-1 | 2.1843 e-1 | 2.22 e-2 |      |
| **LIR-CMOP6**   | 2.6433 e-1 | 3.021 e-1 | 7.8108 e-1 | 1.2169 e+0 | 1.129 e+0 | 1.0655 e+0 | 1.2770 e-1 | 1.88 e-2 |      |
| **LIR-CMOP7**   | 2.4652 e-1 | 3.3833 e-1 | 1.0522 e+0 | 1.3453 e+0 | 1.130 e+0 | 7.5645 e+0 | 1.2511 e+0 | 2.99 e-1 |      |
| **LIR-CMOP8**   | 2.1216 e-1 | 3.2585 e-1 | 8.5391 e-1 | 8.7238 e-1 | 1.0605 e-2 | 1.9605 e-1 | 1.0935 e-1 | 1.10 e-1 |      |
| **LIR-CMOP9**   | 9.4696 e-1 | 1.6998 e-1 | 4.2171 e-1 | 1.6336 e+0 | 1.2006 e+0 | 8.3868 e-1 | 1.4727 e-1 | 5.72 e-2 |      |
| **LIR-CMOP10**  | 8.3893 e-1 | 8.4063 e-1 | 8.2511 e-1 | 4.6686 e-1 | 4.2436 e-1 | 4.3952 e-1 | 4.1261 e-1 | 1.34 e-1 |      |
| **LIR-CMOP11**  | 9.6013 e-2 | 3.4208 e-1 | 1.0638 e+0 | 5.8341 e-1 | 1.7647 e-1 | 2.4277 e-1 | 1.5348 e-1 |      |      |
| **LIR-CMOP12**  | 7.3932 e-1 | 2.2272 e-1 | 1.1564 e+0 | 8.0655 e-1 | 8.6470 e-2 | 1.8548 e-1 |      |      |      |
| **LIR-CMOP13**  | 9.7579 e-1 | 3.0500 e-1 | 4.2689 e-1 | 5.9069 e-1 | 2.1835 e-1 | 1.5685 e-1 | 2.0284 e-1 | 9.42 e-2 |      |
| **LIR-CMOP14**  | 9.1772 e-2 | 9.024 e-2 | 1.2277 e+0 | 1.3010 e+0 | 1.2251 e-1 | 9.9880 e-2 | 8.6007 e-2 | 9.04 e-4 |      |
| **LIR-CMOP15**  | 6.5737 e-1 | 9.2313 e-2 | 1.2795 e+0 | 1.1016 e-1 | 1.0139 e-1 | 8.7737 e-2 | 7.73 e-4 |      |      |

The best result in each row is highlighted in bold.
with discrete feasible regions. On the contrary, the proposed CMOEA-MS and ToP perform well on most test instances. We win the best or is comparable with the best on 10 out of the 20 test instances. Since the IGD is a direct indicator for diversity and convergence of population, the comparison results of IGD value demonstrate that the proposed CMOSMA has good robustness for handling CMOPs, and it especially exhibits excellent performance in terms of convergence and diversity.

To verify the computational efficiency of the proposed CMOSMA, Fig. 14 shows the average runtimes of the various evaluated methods on the 2-objective LIR-CMOP test suite. Clearly, C-TAEA is the slowest algorithm. On the contrary, the TiGE-2 performs significantly better than the other algorithms on real-time computation due to it has a quite simple framework. Although CMOEA-MS and ToP have higher efficiency than the proposed CMOSMA, their IGD values are higher than that of CMOSMA. Besides, it is worth mentioning that C-TAEA requires more time consuming than the proposed CMOSMA though they all adopt the two-population evolution strategy since CMOSMA employs the truncation strategy instead of the decomposition-based technique, where the truncation strategy is less time-consuming than association operation. Combined with the analysis of the previous paragraph, it can be confirmed that the proposed CMOSMA presents a competitive performance on computational efficiency compared with the other evaluated approaches.

**Conclusions**

In this work, a novel self-organizing map approach has been presented to solve constrained multi-objective optimization problems with complex feasible regions. Without loss of generality, the distribution of the Pareto set for a continuous m-objective MOP is a piecewise continuous manifold, which has a dimension of m-1. Following this property, we designed a Two-SOM collaborative framework to estimate the topological structure information of FP and AP at each generation, respectively. A set of offspring solutions were generated from their mating pools constructed based on the neighborhood relationship information. Furthermore, a two-population evolution strategy is adopted, which can effectively improve the search efficiency and avoid trapping in local optima.

In the experimental studies, we ran our CMOSMA framework on 48 well-known constrained multi-objective benchmark problems to conduct a comprehensive comparison with several state-of-the-art algorithms. The satisfactory results obtained demonstrate that the proposed CMOSMA has high search efficiency and good robustness for solving CMOPs, especially for CMOPs with complex constraints.

In the future, we will further enhance the performance of CMOSMA and extend current work to other scenarios, such as constrained many-objective optimization problems [45] and dynamic problems [46].

**Acknowledgements** The authors would like to thank the anonymous reviewers for their valuable suggestions on improving this paper.

**Funding** This work was supported in part by the National Key Research and Development Program of China (2020YFC2003800), the National Natural Science Foundation of China (61866026, 61772255, 61866025 and 62066031), the Natural Science Foundation of Jiangxi Province (2018BAB202025, 20202BAB120101, 20202ACB214007), the Technology Innovation Guidance Program of Jiangxi Province (No.20212AEB10005), the Aeronautical Science Foundation of China (2018ZC56008), the Outstanding Young Scientist Project of Jiangxi Province (20192BCB23011).

**Declarations**

**Conflict of interest** The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.
References

1. Chen Y, Wang R, Ming M, Cheng S, Bao Y, Zhang W, Zhang C (2021) Constraint multi-objective optimal design of hybrid renewable energy system considering load characteristics. Complex Intell Syst. https://doi.org/10.1007/s40747-021-00363-4

2. Zhu L, Lin J, Li Y, Wang ZI (2021) A decomposition-based multi-objective genetic programming hyper-heuristic approach for the multi-skills resource constrained project scheduling problem. Knowl-Based Syst 225:1–18

3. Gu Q, Wang Q, Xiong NN, Jiang S, Chen L (2021) Surrogate-assisted evolutionary algorithm for expensive constrained multi-objective discrete optimization problems. Complex Intell Syst. https://doi.org/10.1007/s40747-020-00249-x

4. Zhou Y, Zhu M, Wang J, Zhang Z, Xiang Y, Zhang J (2020) Tri-goal evolution framework for constrained many-objective optimization. IEEE Trans Evol Comput 50(8):3086–3099

5. Ewees AA, Elaziz MA, Oliva D (2021) A new multi-objective optimization algorithm combined with opposition-based learning. Expert Syst Appl 165(3):1–30

6. Du K, Li J, Wang H, Zhang J (2022) Multi-objective multi-criteria evolutionary algorithm for multi-objective multi-task optimization. Complex Intell Syst. https://doi.org/10.1007/s40747-022-00650-8

7. Wang YK, Wang SL, Lo D, Shen CF, Yang B (2021) An improved multi-objective whale optimization algorithm for the hybrid flow shop scheduling problem considering device dynamic reconfiguration processes. Expert Syst Appl 174:1–14

8. Li K, Chen R, Fu G, Yao X (2018) Two-archive evolutionary algorithm for constrained multi-objective optimization. IEEE Trans Evol Comput 23(2):303–315

9. Jain H, Deb K (2014) An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. IEEE Trans Evol Comput 18(4):602–622

10. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

11. Peng C, Liu H, Goodman E, Tan KC (2022) A two-phase framework of locating the reference point for decomposition-based constrained multi-objective evolutionary algorithms. Knowl-Based Syst 239:1–12

12. Fan Z, Fang Y, Li W, Cai X, Wei C, Goodman E (2019) MOEA/D with angle-based constrained dominance principle for constrained multi-objective optimization problems. Appl Soft Comput 74:621–633

13. Yang Y, Liu J, Tan S (2020) A constrained multi-objective evolutionary algorithm based on decomposition and dynamic constraint-handling mechanism. Appl Soft Comput 89:1–16

14. Jiao R, Zeng S, Li C, Ong Y (2021) Two-type weight adjustments in MOEA/D for highly constrained many-objective optimization. Inf Sci 578:592–614

15. Li M, Yang S, Liu X (2015) Bi-goal evolution for manyobjective optimization problems. Artif Intell 228:45–65

16. Wang H, Jiao L, Yao X (2015) Two Arch2: an improved two-archive algorithm for many-objective optimization. IEEE Trans Evol Comput 19(4):524–541

17. Fan Z, Li W, Cai X, Hui L, Goodman ED (2019) Push and pull search for solving constrained multi-objective optimization problems. Swarm Evol Comput 44:665–679

18. Tian Y, Zhang Y, Su Y, Zhang X, Tan KC, Jin Y (2021) Balancing objective optimization and constraint satisfaction in constrained evolutionary multi-objective optimization. IEEE Trans Cybernet. https://doi.org/10.1109/TCYB.2020.3021138

19. Harada K, Sakuma J, Ono I and Kobayashi S (2006) Constraint-handling method for multi-objective function optimization: pareto descent repair operator. In: Proceedings of 4th International Conference Evolution Multi Criterion Optim. (EMO), pp 156–170

20. Singh HK, Ray T, Smith W (2010) C-PSA: constrained Pareto simulated annealing for constrained multi-objective optimization. Inf Sci 180(13):2499–2513. https://doi.org/10.1016/j.ins.2010.03.021

21. Jiao L, Luo J, Shang R, Liu F (2014) A modified objective function method with feasible-guiding strategy to solve constrained multi-objective optimization problems. Appl Soft Comput 14:363–380

22. Liu Z, Wang Y (2019) Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces. IEEE Trans Evol Comput 23(13):870–884

23. Ma H, Wei H, Tian Y, Cheng R, Zhang X (2021) A multi-stage evolutionary algorithm for multi-objective optimization with complex constraints. Inf Sci 560:68–91

24. Li Y, He H, Wang Y, Xu X, Jiao L (2015) An improved multi-objective estimation of distribution algorithm for environmental economic dispatch of hydrothermal power systems. Appl Soft Comput 28:559–568

25. Zhang H, Song S, Zhou A, Gao XZ (2015) A multiobjective cellular genetic algorithm based on 3D structure and cosine crowding measurement. Int J Mach Learn Cybern 6(3):487–500

26. Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731

27. Zhang H, Zhou A, Song S, Zhang Q (2016) A self-organizing multiobjective evolutionary algorithm. IEEE Trans Evol Comput 20(5):792–806

28. Kohonen T (1998) The self-organizing map. Neurocomputing 21(1–3):1–6

29. Obayashi S and Sasaki D (2003) Visualization and data mining of Pareto solutions using self-organizing map. In: Proceedings of evolution and multi-criterion optimization, Faro, Portugal, pp 796–809

30. Gu F, Cheung YM (2018) Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm. IEEE Trans Evol Comput 22(2):211–225. https://doi.org/10.1109/tevc.2017.2695579

31. Ma X et al (2016) A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with largescale variables. IEEE Trans Evol Comput 20(2):275–298

32. Miettinen K (1999) Nonlinear multiobjective optimization. Kluwer Academic, Boston

33. Hillermeier C (2001) Nonlinear multiobjective optimization—a generalized homotopy approach. Birkhauser, Boston

34. Ma Z, Wang Y (2019) Evolutionary constrained multiobjective optimization: test suite construction and performance comparisons. IEEE Trans Evol Comput 23(6):972–986

35. Zhang Y, Li Z, Zhang H, Yu Z, Lu T (2018) Fuzzy c-means clustering-based mating restriction for multiobjective optimization. Int J Mach Learn Cybern 9:1609–1621

36. Wang S, Zhang H, Zhang Y, Zhou A, Wu P (2019) A spectral clustering-based multi-source mating selection strategy in evolutionary multi-objective optimization. IEEE Access 7:131851–131864

37. Zillke H, Ishibuchi H, Mostaghim S and Nojima Y (2016) Mutation operators based on variable grouping for multi-objective large-scale optimization. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI)

38. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. IEEE Trans Evol Comput 6(2):182–197

39. Li M, Yang S, Liu X (2014) Shift-based density estimation for Pareto-based algorithms in many-objective optimization. IEEE
40. Fan Z, Li W, Cai X, Huang H, Fang Y, You Y, Mo J, Wei C, Goodman E (2019) An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions. Soft Comput 23:12491–12510
41. Tian Y, Zhang T, Xiao J, Zhang X, Jin Y (2021) A coevolutionary framework for constrained multi-objective optimization problems. IEEE Trans Evol Comput 25(1):102–116
42. Bosman PAN, Thierens D (2003) The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Trans Evol Comput 7:174–188
43. Wilcoxon F (1945) Individual comparisons by ranking methods. Biometrics Bull 1(6):80–83
44. Tian Y, Cheng R, Zhang X, Jin Y (2017) PlatEMO: a MATLAB platform for evolutionary multi-objective optimization. IEEE Comput Intell Mag Mag 12(4):73–87