A ε -indicator-based shuffled frog leaping algorithm for many-objective optimization problems

WANG Na^{1,2,*}, SU Yuchao¹, CHEN Xiaohong¹, LI Xia^{1,2}, and LIU Dui¹

1. College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China;

2. Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518060, China

Abstract: Many-objective optimization problems take challenges to multi-objective evolutionary algorithms. A number of non-dominated solutions in population cause a difficult selection towards the Pareto front. To tackle this issue, a series of indicator-based multi-objective evolutionary algorithms (MOEAs) have been proposed to guide the evolution progress and shown promising performance. This paper proposes an indicator-based many-objective evolutionary algorithm called ε -indicator-based shuffled frog leaping algorithm (ε -MaOSFLA), which adopts the shuffled frog leaping algorithm as an evolutionary strategy and a simple and effective ε -indicator as a fitness assignment scheme to press the population towards the Pareto front. Compared with four state-of-the-art MOEAs on several standard test problems with up to 50 objectives, the experimental results show that ε -MaOSFLA outperforms the competitors.

Keywords: evolutionary algorithm, many-objective optimization, shuffled frog leaping algorithm (SFLA), ε -indicator.

DOI: 10.21629/JSEE.2020.01.15

1. Introduction

In past decades, multi-objective evolutionary algorithms (MOEAs) have been regarded as one of the most successful approaches for solving multi-objective optimization problems (MOPs). However, with an increase in the number of objectives, the efficiency of Pareto-based MOEAs has been invalid to press population towards Pareto front [1,2], such as non-dominated sorting genetic algorithm II (NSGA-II) [3] and strength Pareto evolutionary algorithm 2 (SPEA2) [4]. The reason is mainly the fast increasing proportion of non-dominated solutions in population. In the extreme case, all the solutions in population are non-dominated, Pareto-based MOEAs cannot well distinguish the solutions and press them towards the Pareto front (PF).

Recently, a number of many-objective evolutionary al-

gorithms (MaOEAs) have been designed to address many-objective optimization problems (MaOPs) [5]. The environmental selection is of great importance in MOEAs. According to the selection, MOEAs can be classified into the domination-based [6], the decomposition-based [7], and the indicator-based [8] MOEAs. To Pareto-based MOEAs, some improved strategies have been proposed for MaOPs, the strategies can be categorized into two types. The first one is to modify the definition of the Pareto domination, such as corner sort [9], fuzzy-dominance [10], and reference point-based dominance [11]. The other is to improve the diversity management. For example, shift-based density estimation (SDE) [12] used the shift-based density to strength the diversity, NSGA-III [13] used the weight vectors to decide the diversity. For decomposition-based MOEAs, one difficulty is to set uniformly distributed weight vectors in many-objective space. An MaOEA based on decomposition and Pareto selection [14] designed a two-layer generation method to decrease the number of the weight vectors, as the method in [15] generated a large number of the weight vectors. Some approaches have been applied to enhance the diversity [16,17]. Regarding the indicator MOEAs, the optimization goal is to determine a Pareto approximation set by a quality indicator, which qualifies a solution as a fitness value to decide the next population. Hypervolume (HV) is a popular indicator. However, HV has been impeded by the efficiency in MaOPs. A number of studies have been done to reduce its computational complexity [18–20]. The other indicators, such as the ε -indicator [21] and R_2 indicator [22,23], have been proposed to capture different preferences in the optimization process.

The shuffled frog leaping algorithm (SFLA) [24] was originally proposed as a memetic evolutionary algorithm. SFLA mimics the behavior of a group of frogs to hunt. Each frog improves its position by itself and other members during the hunt search. SFLA combines the ideas of

Manuscript received August 13, 2018.

*Corresponding author.

This work was supported by the Shenzhen Innovation Technology Program (JCYJ20160422112909302).

particle swarm optimization as a local search and the shuffled complex evolution algorithm for a global information exchange. SFLA has been successfully used for a wide range of real-world applications, such as the water distribution network design [24], the capacitated vehicle routing problems [25], and the image segmentation [26].

The SFLA for MOPs called MOSFLA was firstly proposed [27]. In MOSFLA, the crowding distance and Pareto dominance relation are regarded as the criteria to replace an old frog by a new frog. An archive is used to maintain the non-dominated solutions. The most crowded solution is deleted when the size of the archive is excessive. MOSFLA was used to a bi-criteria permutation flow-shop problem and showed better performance than the competitors. A variable neighborhood search was also incorporated into MOSFLA to enhance the convergence and diversity for the large-sized problems [28]. Hybrid MOSFLA (HMOSFLA) with bacterial optimization [29] was proposed to solve a mixed-model assembly line sequencing problem with three objectives. The experimental results further confirmed that SFLA was more efficient for solving MOPs than the rivals.

For MaOPs, a number of non-dominated solutions in population cause a difficult selection towards the PF. The main motivation of the present paper is to develop a new hybrid MaOEA for MaOPs, and the proposed MaOEA called ε -MaOSFLA combines SFLA with the ε -indicator. Different from MOEAs and MaOEAs, SFLA was proposed for single-objective optimization, in which the qualities of the individuals can be distinguished by the objective values directly. However, for MOPs and MaOPs, the comparison cannot be executed. As a result, we select the ε -indicator to discriminate the individuals. Thus, SFLA with the ε -indicator can tackle MaOPs. With the increasing of the number of objective, objective space is expanded exponentially [30]. Thus, the diversity of population is important to cover the whole PF due to the use of a limited number of individuals to approximate the PF. Actually, the subpopulations of SFLA, in which they do the local search for single-objective optimization, can be used to maintain the diversity on MaOPs.

The remainder of this paper is organized as follows. Section 2 introduces the relative background. In Section 3, we describe the proposed ε -MaOSFLA in detail. The experimental analysis is reported to compare ε -MaOSFLA with the four state-of-the-art MOEAs for tackling MaOPs with up to 50 objectives and the parameter analysis is executed in Section 4. The conclusions are given in Section 5.

2. Background

In this section, we describe the definitions of multi- and

many-objective optimization, the frog leaping functions in SFLA and the ε -indicator, respectively.

2.1 Definition of multi- and many-objective optimization problem

An MOP can be defined as

$$\begin{aligned} \min \quad & F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})) \\ \text{s.t.} \quad & \mathbf{x} \in \Omega \text{ and } 1 < M \leq 3 \end{aligned} \quad (1)$$

where M is the number of objectives and $\mathbf{x} = (x_1, x_2, \dots, x_v)$ is a V -dimensional decision vector in the decision space Ω . $F : \Omega \rightarrow R^M$ defines m conflicting objective functions and R^M indicates the objective space. The purpose of a multi-objective optimizer is to find the Pareto optimal set (PS) and the mapping of PS in R^M is called the PF [31]. When $M > 3$, (1) is called an MaOP.

2.2 SFLA

The SFLA was originally designed for single-objective optimization problems. Basically, the evolutionary process of the SFLA is composed of three actions: partitioning, memetic evolution and shuffling. In the SFLA, the population is partitioned into m subpopulations (also known as memeplexes), each of which contains n frogs. Each memeplex aims to move the worst frog towards a better region inspired by the local best frog or the global best frog. A new frog is generated by

$$\mathbf{D} = r(\mathbf{x}_b - \mathbf{x}_w) \quad (2)$$

$$\mathbf{x}' = \mathbf{x}_w + \mathbf{D} \quad (3)$$

where $\mathbf{x}_b$ and $\mathbf{x}_w$ denote the best and the worst frog in a memeplex, respectively, r is a random number from 0 to 1. If a new frog $\mathbf{x}'$ is not better than $\mathbf{x}_w$, and then the new frog is generated by (4) instead of (2).

$$\mathbf{D} = r(\mathbf{x}_g - \mathbf{x}_w) \quad (4)$$

where $\mathbf{x}_g$ is the global best frog. $\mathbf{x}'$ replaces $\mathbf{x}_w$ if $\mathbf{x}'$ is better than $\mathbf{x}_w$. Otherwise, a random solution is generated to replace the worst frog.

2.3 ε -indicator

The ε -indicator [32,33] was reported to be comparable to the HV indicator [34], and the advantage is the lower computational complexity. It is defined by

$$I_\varepsilon(A, B) = \inf_{\varepsilon > 0} \{ \forall \mathbf{b} \in B, \exists \mathbf{a} \in A : \mathbf{a} \succ_\varepsilon \mathbf{b} \} \quad (5)$$

$$\forall 1 \leq i \leq M : f_i(\mathbf{a}) - \varepsilon \leq f_i(\mathbf{b}). \quad (6)$$

When (5) holds, $\mathbf{a}$ is said to ε -dominate $\mathbf{b}$ (denotes as $\mathbf{a} \succ_\varepsilon \mathbf{b}$). $I_\varepsilon(A, B)$ represents the minimum distance by which any solution in B is not worse than at least one solu-

tion in A in each objective value. Then, the fitness is computed by

$$F_{\varepsilon}(\mathbf{x}_1) = \sum_{\mathbf{x}_2 \in P \setminus \{\mathbf{x}_1\}} -e^{-I_{\varepsilon}(\mathbf{x}_2, \mathbf{x}_1)/(c \cdot k)} \quad (7)$$

$$c = \max_{\mathbf{x}_i, \mathbf{x}_j \in P} |I_{\varepsilon}(\mathbf{x}_i, \mathbf{x}_j)|, \quad i, j = 1, 2, \dots, |P| \quad (8)$$

where P is a population, k is a parameter that is greater than 0, and $k = 0.01$ is set in [32].

3. The proposed ε -MaOSFLA

When modeling the SFLA for MaOPs, three issues need to be tackled:

(i) Assigning the fitness. For MOPs and MaOPs, the fitness cannot be directly calculated based on the objective values due to their conflicts. The SFLA requires a single-valued fitness assignment to directly compare two solutions when partitioning and selecting solutions. Moreover, the discrimination of solutions should not deteriorate with the increasing of the objective number.

(ii) Selecting the global optimal solution. The global optimal solution is used to update the worst solution if it cannot leap to a better region where the local best frog locates. For MOPs and MaOPs, the issue is how to identify the global optimum because of a number of non-dominated solutions.

(iii) Retaining non-dominated solutions. With the evolution, the size of non-dominated solutions may be more than N (population size), which causes a high computational cost and a difficulty to choose solutions for the next population. Thus, the issue is how to store a fix-sized set of non-dominated solutions and maintain the diversity of population.

To tackle the above issues, we propose an indicator-based many-objective optimization algorithm, called ε -MaOSFLA. This algorithm adopts the SFLA as the local and global optimizer and the ε -indicator to evaluate the fitness values of the solutions for the environmental selection.

3.1 Memetic evolution within each memplex

As described in Section 2, the core of the SFLA is the memetic evolution within each memplex, where the worst frog is improved to proceed the search towards the PF. For MaOPs, some modifications on the SFLA are required.

The first modification is related to the criterion of replacement. To single-objective optimization, an offspring is either better or worse than the current one, which will be replaced or not. However, in terms of MaOPs, the comparison is infeasible. In ε -MaOSFLA, the Pareto dominance

and ε -indicator are used to distinguish solutions. $\mathbf{x}_w$ and $\mathbf{x}'$ are defined in (3) in a memplex E . When $\mathbf{x}'$ can update $\mathbf{x}_w$, there are the three following cases:

- (i) If $\mathbf{x}' \succ \mathbf{x}_w$, $\mathbf{x}_w$ is replaced by $\mathbf{x}'$.
- (ii) When $\mathbf{x}'$ and $\mathbf{x}_w$ are not dominated by each other, $\mathbf{x}'$ dominates any individual in E .
- (iii) $F_{\varepsilon}(\mathbf{x}') > F_{\varepsilon}(\mathbf{x}_w)$.

This criterion includes the Pareto dominance and ε -indicator. To tackle MaOPs, the Pareto dominance is invalid in most cases but it is more accurate to distinguish two solutions if one dominates other one. In most cases, the ε -indicator discriminates solutions to confirm the convergence. The assignment of fitness for SFLA can be designed by this strategy.

For selecting the global optimal solution, all the non-dominated solutions in the current population are equally optimal. We identify the nearest frog in the archive (described in Section 3.2) to the current worst frog as its global optimal solution. Thus, each worst frog may be assigned to different global optimal solutions, and this procedure is performed to move the population towards different regions in the decision space, which can avoid falling into local optima. The pseudo code of the evolution is shown in Algorithm 1. In Algorithm 1, the input includes a memplex S , the number of the maximal iteration t in a memplex, and the non-dominated solution set A . In lines 1–14, the iteration is executed to evolve the frogs. In lines 2–4, a frog $\mathbf{x}'$ is generated by (2) and (3), and it can update $\mathbf{x}_w$ according to the three cases. In lines 6–9, $\mathbf{x}'$ is generated by (3) and (4), and $\mathbf{x}_w$ is updated by $\mathbf{x}'$ if one of the three cases holds. Otherwise, randomly regenerate $\mathbf{x}_w$. In the line 15, S is outputted.

Algorithm 1 ε -SFLA

Input a memplex S , the maximal iteration t for a memplex and the non-dominated solution set A

```

1  for  $i = 1$  to  $t$ 
2    generate  $\mathbf{x}'$  by (2) and (3)
3    if  $\mathbf{x}'$  is better than  $\mathbf{x}_w$ 
4      set  $\mathbf{x}'$  as  $\mathbf{x}_w$ 
5    else
6      set the nearest frog in  $A$  to  $\mathbf{x}_w$  as  $\mathbf{x}_g$ 
7      generate  $\mathbf{x}'$  by (3) and (4)
8      if  $\mathbf{x}'$  is better than  $\mathbf{x}_w$ 
9        set  $\mathbf{x}'$  as  $\mathbf{x}_w$ 
10     else
11        $\mathbf{x}_w$  is regenerated randomly
12     end if
13   end if
14 end for
15 return  $S$ 
```

3.2 Update non-dominated archive

In ε -MaOSFLA, an archive A is employed for collecting the non-dominated solutions found in the evolution. Assuming the pre-size is N , and when the size is larger than N , the solution with the smallest fitness value in A is deleted. Suppose P' is the non-dominated set of the current population P . The procedure of updating A is introduced as follows.

Step 1 Delete dominated solutions from $P' \cup A$ to form a set A' . If the size of A' does not exceed N , A' is served as A . Otherwise, go to Step 2.

Step 2 For all x_i in A' , compute the fitness values by (9) and (10).

$$F_\varepsilon(x_i) = \sum_{x_j \in A' \setminus \{x_i\}} -e^{-I_\varepsilon(x_j, x_i)/(c \cdot k)} \quad (9)$$

$$c = \max_{x_i, x_j \in A'} |I_\varepsilon(x_i, x_j)| \quad (10)$$

Step 3 Select an individual x^w in A' with the smallest fitness value and remove it from A' . If the size of A' exceeds N , go to Step 4. Otherwise, stop.

Step 4 Update the fitness values of the remaining individuals in A' by (11), and then go to Step 3.

$$F_\varepsilon(x) = F_\varepsilon(x^w) + e^{-I_\varepsilon(x^w, x)/(c \cdot k)}, \quad x \in A' \quad (11)$$

Repeat Step 3 and Step 4 until the size of A' is equal to N and return A' as A . When a solution is removed, the fitness values of the remaining solutions in A' are updated by (11) that is very efficient, because the item $I_\varepsilon(x^w, x)$ has been computed by (9). Thus, retaining non-dominated solutions can be completed by this method with a high efficiency.

3.3 ε -MaOSFLA

Algorithm 2 shows the pseudo code of ε -MaOSFLA. In lines 1 and 2, the population P is initialized, and add non-dominated solutions in P into A . The main iteration is executed from the line 3 to the line 11. Sort frogs in P to a descending order and divide them into n memplexes in lines 4 and 5. Execute Algorithm 1 for n memplexes to update them in lines 6–8. Shuffle the n memplexes as P and use P to update A in lines 9 and 10. A is served as the output set in the line 12.

Algorithm 2 ε -MaOSFLA

Input population size N , the number of memplexes n , the number of individuals m in each memplex and the maximal iteration t for a memplex

1 randomly generate the population P with N individuals

2 set non-dominated solutions in P as A

3 **while** do not satisfy the stop criterion do
 4 sort frogs in P to a descending order by (9) and (10)
 5 divide P into n memplexes $S_1, S_2, \dots, S_n$
 6 **for** $i = 1$ **to** n
 7 run Algorithm 1 to update S_i
 8 **end for**
 9 shuffle the n memplexes as P
 10 update A by non-dominated solutions in P
 11 **end while**
 12 **return** A

3.4 Complexity analysis

The time complexity of ε -MaOSFLA is identified by the while loop in Algorithm 2. Assume there are M objectives to be optimized and the population size is N . In each iteration, dividing the population into n memplexes requires $O(n)$ times. The memetic evolution runs in $O(m)$ times for each memplex, so $O(mn)$ is necessary for n memplexes. The shuffle operator spends about $O(N^2)$ times to sort the memplexes at least and the non-dominated sorting runs $O(N(\log_2 N)^{M-2})$ times. The complexity of updating A is $O(N^2)$. Therefore, the overall time complexity is $O(N(\log_2 N)^{M-2})$ or $O(N^2)$.

4. Computational experiment

4.1 Experimental setup

The DTLZ1-7 [35] and the WFG2-4 [36] benchmark, which have 3, 5, 10, 20, 30 and 50 objectives, are used to test the performance of ε -MaOSFLA. In Table 1, we summarize the parameters settings and the main features of all the problems [37].

Table 1 Features and parameters of the test problems

Problem	Number of variables	Geometry of PF
DTLZ1	$M+4$	Linear
DTLZ2	$M+9$	Concave
DTLZ3	$M+9$	Concave
DTLZ4	$M+9$	Concave
DTLZ7	$M+9$	Disconnected, mixed
WFG2	$M+19$	Convex, disconnected
WFG3	$M+19$	Linear, degenerate
WFG4	$M+19$	Concave

To assess the performance of ε -MaOSFLA, we compare it with respect to the four state-of-the-art algorithms, including NSGA-II [1], hypervolume-based search algorithm (HypE) [38], indicator-based evolutionary algorithm (IBEA) [32] and many objective metaheuristic based on the R2 indicator (MOMBI) [39].

HypE is a representative indicator-based MOEA that uses the HV indicator to guide in the evolution, and its advantages for MaOPs have been reported in [40]. IBEA is based on the ε -indicator and the performance is shown

in [41]. MOMBI uses the $R2$ indicator to rank solutions, and NSGA-II works well for the optimizations that involve two or three objectives. Note that HypE, IBEA and NSGA-II are implemented within [42] and the source code of MOMBI is available at <http://compuacion.cs.cinvesta.mx/~rhernandez/mombi>. The population size for each algorithm is set to 100. The maximum of fitness evaluations is 20 000 for each test problem with no more than 10 objectives and 50 000 for the others. The simulated binary crossover [43], in which its index and the crossover probability are set to 20 and 0.9, respectively, is employed as the crossover operator of all tested algorithms. The polynomial mutation, in which its index and the mutation probability are set to 20 and $1/V$, respectively, is used as the mutation operator. The sample number adopted in HypE is set to 10^5 as suggested in [38]. For MOMBI, the number of weight vectors is set to the population size. About the parameters of ε -MaOSFLA, the number of memplexes n , the number of solutions in a memplex m and the maximum iterations t allowed for the memetic evolution are set to 4, 25 and 4, respectively.

4.2 Performance assessment

To compare the performance of all the algorithms, we perform 30 independent runs for each algorithm on each benchmark instance. Three performance assessments are adopted to measure the approximation sets obtained by the algorithms.

Hypervolume metric **H** was proposed [39] to measure the convergence and diversity of obtained population simultaneously, and it measures the volume in the objective space, which is calculated by obtained population and a reference point. In this paper, the reference points are set to be larger by an additive term of 1 than the maximal objective values in the PF as in [44]. The setting of reference points for each test problems is shown in Table 2. A larger **H** value indicates that the obtained population is closer to the PF and has a more even convergence on the PF.

Table 2 Setting of reference points for computing **H** values

Test problem	Reference point
DTLZ1	(1.5, 1.5, ...)
DTLZ2-4	(2, 2, ...)
DTLZ7	(2, 2, ..., $2M + 1$)
WFG2-4	(3, 5, ..., $2M + 1$)

Epsilon metric **E** was proposed in [33]. It is a unary quality indicator based on (5) and (6), where A in (5) is the output population by a tested algorithm, and B in (5) represents a reference set. The epsilon metric only measures the convergence of a non-dominated set to the reference set. In this paper, we adopt B as non-dominated solutions from all output solutions by all the compared algorithms. A smaller **E** value means that A is more similar to B .

Schott's spacing metric **SS** was proposed by Schott [45] to measure the diversity of solutions in objective space, and it is calculated as follows:

$$SS(A) = \sqrt{\frac{\sum_{i=1}^{|A|} (d_i^* - d_i)^2}{(|A| - 1)}} \quad (12)$$

where A is a non-dominated set, d_i is the distance between the i th solution x in A and a solution of A (do not include x) nearest to it, and d^* is the mean of all d_i . A smaller **SS** value indicates solutions of A are more evenly distributed in objective space. The Kruskal-Wallis statistical test at a significance level of $\alpha = 0.01$ [40] is used to test whether the results of the algorithms on each problem are different. The P -values are given when the first algorithm outperforms the second algorithm. The performance score of each algorithm is calculated by the number of algorithms who outperform it and the definition is shown as follows:

$$S_i = \sum_{j=1, j \neq i}^l \sigma_{ij}$$

where σ_{ij} is equal to 1 if the j th algorithm outperforms the i th algorithm. Otherwise, it is equal to 0. $S_i = 0$ implies that A_i is the best algorithm and outperforms all the competitors.

4.3 Experimental results

4.3.1 Performance comparison

In this subsection, an assessment on convergence is executed to compare ε -MaOSFLA with the other algorithms. Figs. 1–3 show the **H**, **E** and **SS** metrics, respectively, where N, H, I, M and E respectively indicate NSGA-II, HypE, IBEA, MOMBI and ε -MaOSFLA. Note MOMBI requires a considerably long computational time to obtain a series of weight vectors with uniform distribution on the high-dimensional problems, we only perform MOMBI on the problems with no more than ten objectives. Regardless MOMBI, in terms of **H**, The results of ε -MaOSFLA are better than HypE and IBEA, in which it wins on 19 out of 48 instances, followed by HypE (9 instances) and IBEA (5 instances). In terms of **E**, ε -MaOSFLA obtains the best results on 23 instances, followed by IBEA (5 instances), HypE (4 instances) and NSGA-II (4 instances). In terms of **SS**, NSGA-II outperforms the other algorithms on 15 instances, followed by HypE (13 instances), ε -MaOSFLA (5 instances) and IBEA (5 instances). For some instances, the values of **H** obtained by some algorithms are equal to zero. This means that all outputted solutions of the algorithms are dominated by the corresponding reference points in Table 2.

Fig. 1 H metric results on test problems with 3, 5, 10, 20, 30 and 50 objectives

Fig. 3 SS metric results on the test problems with 3, 5, 10, 20, 30 and 50 objectives

Figs. 4–9 show the objective values of the obtained solutions (corresponding to the median value of **H**) on DTLZ3 with all the objective numbers. From Figs. 4–9, it is clear that the approximation sets of ε -MaOSFLA are the closest to the PF than the others on the test problems with all the objective numbers except three objectives. On all the problems, the maximal objective values obtained by ε -MaOSFLA are less than 8. When the number of objectives is greater than 10, the maximal objective values of NSGA-II, HypE and IBEA are greater than 100, i.e., they are difficult to converge on the problems. Although the ε -indicator is employed in ε -MaOSFLA and IBEA, ε -MaOSFLA is clearly better than IBEA on dealing with DTLZ3. On each number of objectives, most solutions obtained by NSGA-II are far away from the PF, resulting in a zero value of **H** seen from Fig. 1. The similar phenomena occur on HypE and IBEA. Although the convergence performances of MOMBI and ε -MaOSFLA are competitive on the problem with no less than 20 objectives, the diversity of the former is worse than that of the latter.

Fig. 4 Objective values of the output populations on DTLZ3 with three objectives

Fig. 5 Objective values of the output populations on DTLZ3 with five objectives

Fig. 6 Objective values of the output populations on DTLZ3 with 10 objectives

Fig. 7 Objective values of the output populations on DTLZ3 with 20 objectives

Fig. 8 Objective values of the output populations on DTLZ3 with 30 objectives

Fig. 9 Objective values of the output populations on DTLZ3 with 50 objectives

To sum up, ε -MaOSFLA is better than the competitors about the convergence on most instances, but worse than some algorithms about the diversity on some instances. For an overall consideration of the convergence and diversity, ε -MaOSFLA still outperforms its competitors.

The running time is a crucial performance indicator of MOEAs. Fig. 10 shows the average running time of the algorithms on the different problems with the different objective numbers, where D1, D2, D3, D4, D7, W2, W3 and W4 respectively indicate DTLZ1-4, DTLZ7 and WFG2-4. Obviously, NSGA-II, IBEA, MOMBI and ε -MaOSFLA receive similar results. Moreover, the running times of these four algorithms are significantly less than HypE. For the problems with 20, 30 and 50 objectives, ε -MaOSFLA spends the shortest time.

To obtain better insight on how fast ε -MaOSFLA converges to PF, we run each algorithm on DTLZ4 with five objectives, and then calculate the $\mathbf{H}$ values in each generation. The results are shown in Fig. 11.

Fig. 10 Average times over the problems with 3, 5, 10, 20, 30 and 50 objectives

Fig. 11 Changing of the H values on DTLZ4 with five objectives

In Fig. 11 the horizontal axis denotes the number of function evaluations and the vertical axis indicates the $\mathbf{H}$ values. It is obvious that ε -MaOSFLA is superior to the other four algorithms due to the faster increasing of $\mathbf{H}$ values. These results indicate the SFLA is more efficient than IBEA (which uses the same indicator as ε -MaOSFLA but a different evolution strategy). As a result, ε -MaOSFLA is more efficient for solving MaOPs.

4.3.2 Parameter analysis

Three parameters in ε -MaOSFLA, i.e., the number of memplexes n , the number of solutions in a memplex m and the maximum iterations allowed for the memetic evolution t , are critical factors that affect the performance of the algorithm.

The parameters n and m are investigated. The parameters n and m are set to 2×50 , 4×25 , 5×25 , and 10×10 , respectively. The population size is set to 100, and t is equal to m . DTLZ4 is served as the test problem.

Table 3 shows the average values of $\mathbf{H}$ for the configurations, where the best results are presented in boldface. Clearly, for DTLZ4 with the different numbers of objectives M , 4×25 is better than the others. To the parameter set of 2×50 , a too small number of memplexes cause a number of individuals exploiting in the same area of the decision space. In this case, the population is easy to fall into the local optimal. On the other hand, when the population is partitioned into too many memplexes, e.g., 5×20 and 10×10 , the evolution in each memplex is ineffective since there are not enough individuals exploiting the local area.

Table 3 Average $\mathbf{H}$ values of different $n \times m$ sets on DTLZ4

M	2×50	4×25	5×20	10×10
3	7.292 8	7.388 8	7.374 8	7.364 8
5	31.566 9	31.573 6	31.512 1	31.419 6
10	1 020	1 022	1.020	1 016

Three different t values are set to test the influence of the parameter on DTLZ4 with the different numbers of objectives M , i.e., $t = 0.8m$, $t = m$ and $t = 1.2m$. The parameters n and m are set to 4 and 25, respectively. Table 4 shows the average $\mathbf{H}$ values of t . The best results about three and five objectives are given when $t = m$, and when the parameter t is equal to $1.2m$, the performance on 10 objectives is better than the rivals. A small t value may slower the convergence speed because a few frogs are improved. On the other hand, a large t value may cause the waste of the evolutionary resource.

Table 4 Average $\mathbf{H}$ values of the different t values on DTLZ4

M	$t = 0.8m$	$t = m$	$t = 1.2m$
3	7.374 1	7.381 5	7.379 0
5	31.476 0	31.559 9	31.529 1
10	1 021	1 019	1 021

5. Conclusions

The indicator-based evolutionary many-objective optimization algorithm are known as an effective method for solving MaOPs. Nevertheless, they are commonly criticized on their high computational complexities. In this study, we propose a new indicator-based MaOEA, called

ε -MaOSFLA, which is composed of the SFLA as the evolutionary strategy and the ε -indicator as the fitness assignment scheme. Taking the fast convergence of the SFLA and the ε -indicator's low computational complexity, the experimental results compare ε -MaOSFLA with three indicator-based MOEA (i.e., HypE, IBEA and MOMBI) and one Pareto-based MOEA (i.e., NSGA-II) and show the efficiency of ε -MaOSFLA for solving several standard test problems, which have 3, 5, 10, 20, 30 and 50 objectives. ε -MaOSFLA can find a better Pareto approximation set than the other algorithms under the same stop criterion. On the other hand, a smaller computation is required by ε -MaOSFLA.

This paper presents the preliminary studies on exploring the specific aspects of SFLA for addressing many-objective optimization and ε -MaOSFLA needs to be further applied to other problems, especially to some real-work optimization problems.

References

- [1] CHAND S, WAGNER M. Evolutionary many-objective optimization: a quick-start guide. *Surveys in Operations Research and Management Science*, 2015, 20(2): 35–42.
- [2] BECHIKH S, ELARBI M, SAID L B. Many-objective optimization using evolutionary algorithms: a survey. *Recent Advances in Evolutionary Multi-objective Optimization*, 2017, 20: 105–137.
- [3] DEB K, AGRAWAL S, PRATAP A, et al. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. *Proc. of the 6th International Conference on Parallel Problem Solving from Nature*, 2000: 849–858.
- [4] ZITZLER E, LAUMANN S, THIELE L. SPEA2: improving the strength Pareto evolutionary algorithm. Technical Report 103. Zurich: Swiss Federal Institute of Technology, 2001.
- [5] FARINA M, AMATO P. On the optimal solution definition for many-criteria optimization problems. *Annual Meeting of the North American Fuzzy Information Processing Society Proceedings*, 2002: 233–238.
- [6] LIN Q Z, CHEN J Y, ZHAN Z H, et al. A hybrid evolutionary immune algorithm for multi-objective optimization problems. *IEEE Trans. on Evolutionary Computation*, 2016, 20(5): 711–729.
- [7] SU Y C, LIN Q Z, WANG J, et al. A constrained solution update strategy for multiobjective evolutionary algorithm based on decomposition. *Complexity*, 2019: 3251349.
- [8] LI B D, TANG K, LI J L, et al. Stochastic ranking algorithm for many-objective optimization based on multiple indicators. *IEEE Trans. on Evolutionary Computation*, 2016, 20(6): 924–938.
- [9] WANG H D, YAO Y. Corner sort for Pareto-based many-objective optimization. *IEEE Trans. on Cybernetics*, 2014, 44(1): 92–102.
- [10] HE Z N, YEN G G, ZHANG J. Fuzzy-based Pareto optimality for many-objective evolutionary algorithms. *IEEE Trans. on Evolutionary Computation*, 2014, 18(2): 269–285.
- [11] ELARBI M, BECHIKH S, GUPTA A, et al. A new decomposition-based NSGA-II for many-objective optimization. *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, 2018, 48(7): 1191–1210.
- [12] LI M Q, YANG S X, LIU X H. Shift-based density estimation for Pareto-based algorithms in many-objective optimization. *IEEE Trans. on Evolutionary Computation*, 2014, 18(3): 348–365.
- [13] DEB K, JAIN H. An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: solving problems with box constraints. *IEEE Trans. on Evolutionary Computation*, 2014, 18(4): 577–601.
- [14] LI K, DEB K, ZHANG Q F, et al. An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Trans. on Evolutionary Computation*, 2015, 19(5): 694–716.
- [15] ZHANG Q F, LI H. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. on Evolutionary Computation*, 2017, 11(6): 712–731.
- [16] WANG L P, ZHANG Q F, ZHOU A M, et al. Constrained subproblems in a decomposition-based multiobjective evolutionary algorithm. *IEEE Trans. on Evolutionary Computation*, 2016, 20(3): 475–480.
- [17] CAI X Y, YANG Z X, FAN Z, et al. Decomposition-based sorting and angle-based-selection for evolutionary multiobjective and many-objective optimization. *IEEE Trans. on Cybernetics*, 2017, 47(9): 2824–2837.
- [18] ROSTAMI S, NERI F. A fast hypervolume driven selection mechanism for many-objective optimisation problems. *Swarm and Evolutionary Computation*, 2017, 34: 50–67.
- [19] VOß T, FRIEDRICH T, BRINGMANN K, et al. Scaling up indicator-based MOEAs by approximating the least hypervolume contributor: a preliminary study. *Proc. of the 12th Annual Conference on Genetic and Evolutionary Computation*, 2010: 1975–1978.
- [20] ISHIBUCHI H, TSUKAMOTO N, SAKANE Y, et al. Indicator-based evolutionary algorithm with hypervolume approximation by achievement scalarizing functions. *Proc. of the 12th Annual Conference on Genetic and Evolutionary Computation*, 2010: 527–534.
- [21] ZITZLER E, KNIZLI S. Indicator-based selection in multiobjective search. *Proc. of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, 2004: 832–842.
- [22] MA X L, ZHANG Q F, TIAN G D, et al. On Tchebycheff decomposition approaches for multi-objective evolutionary optimization. *IEEE Trans. on Evolutionary Computation*, 2018, 22(2): 226–244.
- [23] ALAN D M, GREGORIO T P, JOSE H B Z, et al. R2-based multi/many-objective particle swarm optimization. *Computational Intelligence and Neuroscience*, 2016: 1898527.
- [24] EUSUFF M M, LANSEY K E. Optimization of water distribution network design using the shuffled frog leaping algorithm. *Journal of Water Resources Planning and Management*, 2003, 129(3): 210–225.
- [25] LUO J P, LI X, CHEN M. Improved shuffled frog leaping algorithm for solving CVRP. *Journal of Electronics and Information Technology*, 2011, 33(2): 429–434.
- [26] WANG N, LI X, CHEN X H. Fast three-dimensional Otsu thresholding with shuffled frog leaping algorithm. *Pattern Recognition Letters*, 2010, 31(13): 1809–1815.
- [27] RAHIMI-VAHED A, MIRZAEI A H. Solving a bi-criteria permutation flow-shop problem using shuffled frog leaping algorithm. *Soft Computing*, 2008, 12(5): 435–452.
- [28] RAHIMI-VAHED A, DANGCHI M, RAFIEI H, et al. A novel hybrid multi-objective shuffled frog leaping algorithm for a bi-criteria permutation flow shop scheduling problem. *International Journal of Advanced Manufacturing Technology*, 2009, 41(11): 1227–1239.

- [29] RAHIMI-VAHED A, MIRZAEI A H. A hybrid multi-objective shuffled frog leaping algorithm for a mixed-model assembly line sequencing problem. *Computers and Industrial Engineering*, 2007, 53(4): 642–666.
- [30] ZHANG Y H, GONG Y J, GU T L, et al. DECAL: decomposition-based coevolutionary algorithm for many-objective optimization. *IEEE Trans. on Cybernetics*, 2019, 49(1): 27–41.
- [31] MIETTINEN K. *Nonlinear multi-objective optimization*. USA: Kluwer Academic, 2001.
- [32] ZITZLER E, KNZLI S. Indicator-based selection in multiobjective search. *Proc. of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, 2004: 832–842.
- [33] ZITZLER E, THIELE L, LAUMANN S M, et al. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. on Evolutionary Computation*, 2003, 7(2): 117–132.
- [34] BASSEUR M, LIEFOOGHE A, LE K, et al. The efficiency of indicator-based local search for multi-objective combinatorial optimization problems. *Journal of Heuristics*, 2012, 18(2): 263–296.
- [35] DEB K, THIELE L, LAUMANN S M, et al. Scalable test problems for evolutionary multiobjective optimization. *Advanced Information and Knowledge Processing Series*. Berlin, Germany: Springer, 2005: 105–145.
- [36] HUBAND S, BARONE L, WHILE L, et al. A scalable multi-objective test problem toolkit. *Lecture Notes in Computer Science*, 2005, 3410: 280–295.
- [37] HUBAND S, HINGSTON P, BARONE L, et al. A review of multi-objective test problems and a scalable test problem toolkit. *IEEE Trans. on Evolutionary Computation*, 2006, 10(5): 477–506.
- [38] BADER J, ZITZLER E. HypE: an algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 2011, 19(1): 45–76.
- [39] RAQUEL H G, COELLO COELLO C A. Improved meta-heuristic based on the R2 indicator for many-objective optimization. *Proc. of the Annual Conference on Genetic and Evolutionary Computation*, 2015: 679–686.
- [40] KNOWLES J, THIELE L, ZITZLER E. A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical report TIK-Report No. 214. Zurich: Computer Engineering and Networks Laboratory, ETH, 2005.
- [41] LI M Q, YANG S X, LIU X H, et al. A comparative study on evolutionary algorithms for many-objective optimization. *Proc. of the 7th International Conference on Evolutionary Multi-Criterion Optimization*, 2013: 261–275.
- [42] BLEULER S, LAUMANN S M, THIELE L, et al. PISA—a platform and programming language independent interface for search algorithms. *Proc. of the International Conference on Evolutionary Multi-Criterion Optimization*, 2003: 494–508.
- [43] SU Y C, WANG J, MA L J, et al. A hybridized angle-encouragement-based decomposition approach for many-objective optimization problems. *Applied Soft Computing*, 2019, 78: 355–372.
- [44] BEUME N, NAUJOKS B, EMMERICH M. SMS-EMOA: multiobjective selection based on dominated hypervolume. *European Journal on Operational Research*, 2007, 181(3): 1653–1669.

- [45] SCHOTT J R. *Fault tolerant design using single and multi-criteria genetic algorithm optimization*. Boston, USA: Massachusetts Institute of Technology, 1995.

Biographies

WANG Na was born in 1977. She received her B.S. degree in electronic engineering from Dalian Maritime University in 1998. She later took a successive postgraduate and doctoral programs of study and was conferred a Ph.D. degree on signal and information processing by Dalian Maritime University in 2003. Now she is a professor of College of Electronics and Information Engineering at Shenzhen University. Her main research interests include intelligent computing, machine learning and pattern recognition.
E-mail: wangna@szu.edu.cn

SU Yuchao was born in 1992. He received his B.S. degree from Minnan Normal University, Zhangzhou, China in 2016. He is pursuing his M.S. degree in Shenzhen University, China. His research interests include single-, multi-, and many-objective optimization, and artificial neural network.
E-mail: yuchaosu@126.com

LI Xia was born in 1968. She received her B.S. and M.S. degrees in electronic engineering and signal and information processing from Xidian University in 1989 and 1992 respectively. She was later conferred a Ph.D. degree in Department of Information Engineering by the Chinese University of Hong Kong in 1997. She is a director of Shenzhen Key Laboratory of Advanced Communication and Information Processing. Her research interests include intelligent computing and its applications, image processing and pattern recognition.
E-mail: lixia@szu.edu.cn

CHEN Xiaohong was born in 1977. She received her B.S. and M.S. degrees in electronic engineering and signal and information processing from Liaoning University in 2003 and 2006 respectively. She later was conferred a Ph.D. degree on signal and information processing by Shenzhen University in 2015. Her main research interests include intelligent computing, machine learning and pattern recognition.
E-mail: chenxh@szu.edu.cn

LIU Dui was born in 1995. He received his B.S. degree from the College of Electronics and Information Engineering of Shenzhen University in 2017, where he is currently pursuing his M.S. degree. His main research interests include intelligent computing and machine learning.
E-mail: liud@szu.edu.cn