Research Article

Multiobjective Parallel Chaos Optimization Algorithm with Crossover and Merging Operation

Qingxian Li, Liangjiang Liu, and Xiaofang Yuan

1Hunan Institute of Metrology and Test, Changsha, Hunan 410014, China
2College of Electrical and Information Engineering, Hunan University, Changsha, Hunan 410082, China

Correspondence should be addressed to Liangjiang Liu; 41326337@qq.com

Received 11 November 2019; Accepted 27 April 2020; Published 8 June 2020

Academic Editor: Andras Szekrenyes

Copyright © 2020 Qingxian Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chaos optimization algorithm (COA) usually utilizes chaotic maps to generate the pseudorandom numbers mapped as the decision variables for global optimization problems. Recently, COA has been applied to many single objective optimization problems and simulations results have demonstrated its effectiveness. In this paper, a novel parallel chaos optimization algorithm (PCOA) will be proposed for multiobjective optimization problems (MOOPs). As an improvement to COA, the PCOA is a kind of population-based optimization algorithm which not only detracts the sensitivity of initial values but also adjusts itself suitable for MOOPs. In the proposed PCOA, crossover and merging operation will be applied to exchange information between parallel solutions and produce new potential solutions, which can enhance the global and fast search ability of the proposed algorithm. To test the performance of the PCOA, it is simulated with several benchmark functions for MOOPs and mixed $H_2/H_\infty$ controller design. The simulation results show that PCOA is an alternative approach for MOOPs.

1. Introduction

Optimization problems that at least two objectives need to be optimized simultaneously are called multiobjective, and multiobjective optimization problems (MOOPs) are very common in real world and many engineering areas. Solving MOOPs has become a crucial part of the optimization field. The solution of MOOPs is usually not easy because the multiple objectives tend to be in conflict with each other. Generally, the optimal solution of a MOOPs is a set of optimal solutions, largely known as Pareto-optimal solutions [1]. Each solution represents a particular performance trade-off between multiple objectives and can be considered optimal.

Generating the Pareto-optimal set can be computationally expensive and is often infeasible, because the complexity of the underlying application prevents exact methods from being applicable. For this reason, a number of multiobjective search strategies have been developed over the past decades. One of the most important aspects is the development of evolutionary algorithms and a number of multiobjective evolutionary algorithms have been suggested [2–9]. Evolutionary algorithms are used for MOOPs because they provide a set of solutions in a single run and do not require objectives to be aggregated. Additionally, the performance of evolutionary algorithms does not get affected by the shape of the Pareto front [6]. Other recent effective multiobjective optimization algorithms are multiobjective particle swarm optimization [10, 11], multiobjective artificial immune algorithm [12, 13], ant colony optimization [14], or artificial bee colony (ABC) [15] for MOOPs.

Facing single objective optimization problems, a lot of existing algorithms, such as genetic algorithm (GA), simulated annealing (SA), particle swarm optimization (PSO), differential evolution (DE), harmony search algorithm (HSA), and compact pigeon-inspired optimization [16–18], have demonstrated excellent performance, but trapping in local optimum still remains a challenge. Chaos optimization algorithm (COA) is a recently developed global optimization technique based on chaos theory and particularly is the specification of the use of numerical chaotic sequences. Recently, literatures have demonstrated that the COA can
Mathematical Problems in Engineering

carry out overall global searches at higher speed than stochastic ergodic searches that depend on the probabilities [16–22]. In addition to the development of COA, chaos has also been integrated with optimization algorithms for MOOPs, such as chaotic nondominated sorting genetic algorithm (CNSGA) [23, 24], chaotic sequences based multiobjective differential evolution (CS-MODE) [25], chaos multiobjective immune algorithm [26], multiobjective chaotic particle swarm optimization [27, 28], multiobjective chaotic ant swarm optimization [29], and multiobjective chaotic artificial bee colony algorithm [30]. Because of the ergodicity and the pseudorandomness of chaos, applying chaos in multiobjective optimization algorithms becomes an interesting alternative for enriching population diversity and escaping from local optimum [24, 29].

The parallel strategy is widely used for its great capability of overcoming the sensitivity to the initial, by producing a number of the initial solutions which are uniformly distributed in solution space. Some new optimization algorithms, developed by applying the parallel strategy to conventional intelligent algorithm framework, such as parallel particle swarm optimization algorithm and parallel multiverse optimizer, show excellent performance in robustness and convergence speed.

Although COA has been successfully applied to single objective optimization problems, as far as we know, there are no literatures concerning COA for MOOPs until now. The reasons for this may be as follows: (1) COA is an individual-based algorithm which is not suitable for MOOPs with Pareto-optimal solutions; (2) the chaotic sequences are pseudorandom and sensitive to the initial conditions; therefore, the success of COA crucially depends on appropriate starting values.

In this paper, a novel population-based parallel chaos optimization algorithm (PCOA) with crossover and merging operation will be proposed for MOOPs. In the PCOA, multiple chaos variables (like population) are simultaneously mapped onto one decision variable, so PCOA searches from diverse initial points and detracts the sensitivity of initial values. In addition, crossover and merging operation will be used to exchange information within population and produce new potential solutions. Actually, the proposed algorithm provides a combination of the ability of global search of PCOA and local search accuracy of crossover and merging operation. To preserve the diversity of the Pareto-optimality, an external elitist archive and accurate crowding measure method is applied in the PCOA for MOOPs.

The rest of this paper is organized as follows. Section 2 briefly describes MOOPs. The PCOA approach is introduced in Section 3. Section 4 gives presentation of PCOA for MOOPs. Test problems simulation results show the effectiveness of the PCOA approach in Section 5. In Section 6, PCOA approach is applied through mixed $H_2/H_{\infty}$ controller design. Conclusions are presented in Section 7.

2. Multiobjective Optimization Problems (MOOPs)

MOOPs with conflicting objectives do not have a single solution. Therefore, multiobjective algorithms (MOAs) aim to obtain a diverse set of nondominated solutions, i.e., solutions that balance the trade-off between the various objectives, referred to as the Pareto-optimal front (POF). Another goal of MOAs is to find a POF that is as close as possible to the true POF of MOOPs.

The objectives of MOOPs are normally in conflict with one another; i.e., improvement in one objective leads to a worse solution for at least one other objective. Therefore, when solving MOOPs the definition of optimality used for single objective optimization problems has to be adjusted. For MOOPs, when one decision vector dominates another, the dominating decision vector is considered as a better decision vector.

The MOOPs can be mathematically described as

\[
\begin{align*}
\min f(x) &= [f_1(x), f_2(x), \ldots, f_D(x)]^T, \\
\text{s.t. } g_i(x) &\leq 0, \quad i = 1, 2, \ldots, m, \\
\quad h_j(x) &= 0, \\
\quad j &= 1, 2, \ldots, p, \\
R &= \{x \mid g_i(x) \leq 0, h_j(x) = 0\}, \quad x \in R,
\end{align*}
\]

where $f(x)$ is the objective vector to be optimized and $D$ is the number of objective functions. $g_i(x)$ is the set of inequality constraints, $h_j(x)$ is the set of equality constraints, and $m$ and $p$ are the number of inequality constraints and equality constraints, respectively. We call $x = [x_1, x_2, \ldots, x_n]^T$ the vector of decision variables, and $R$ is the feasible region. The MOOPs determine the particular set of values $x_1^*, x_2^*, \ldots, x_n^*$, which yield the optimum values of all the objective functions, from the set $F$ of all vectors which satisfy (2) and (3).

Several definitions for MOOPs are given by the following [1].

Definition 1 (dominance): a vector $u = (u_1, \ldots, u_p)$ is said to dominate $v = (v_1, \ldots, v_p)$ (denoted by $u \prec v$) if and only if $v_i \leq u_i \quad \forall i \in \{1, \ldots, D\}$.

Definition 2 (Pareto-optimal solution): a point $x^* \in \Omega$ is Pareto-optimal if there is not another $x \in \Omega$ that satisfies with $f(x) < f(x^*)$.

Definition 3 (Pareto-optimal set): for a given MOP $f(x)$, the Pareto-optimal set ($P^*$) is defined as $P^* = \{x \in \Omega \mid \exists x^* \in \Omega: f(x^*) < f(x)\}$.

Definition 4 (Pareto front): for a given MOP $f(x)$ and Pareto-optimal set $P^*$, the Pareto front $PF^*$ is defined as $PF^* = \{f(x) \mid x \in P^*\}$.

In the general case, it is impossible to find an analytical expression of the line or surface that contains these points. The normal procedure to generate the Pareto front is to compute the feasible points $\Omega$ and their corresponding $f(\Omega)$. When there are a sufficient number of these, it is then possible to determine the non-dominated points and to produce the Pareto front.
3. PCOA Approach

A novel population-based parallel chaos optimization algorithm (PCOA) is proposed for MOOPs. The salient feature of PCOA lies in its pseudoparallel mechanism. In addition, crossover and merging operation will be applied to utilize the fitness and diversity information of the population. Actually, the proposed algorithm provides a combination of the ability of global search of PCOA and local search accuracy of crossover and merging operation.

3.1. Twice Carrier Wave Mechanism-Based PCOA. Consider a single objective optimization problem for nonlinear multimodal function with boundary constraints as

\[
\min f(x) = f(x_1, x_2, \ldots, x_n), \quad x_i \in [a_i, b_i].
\]  

(5)

In the PCOA, multiple stochastic parallel chaos variables (like population) are simultaneously mapped onto one decision variable, and the search result is the best solution of parallel candidate individuals.

The process of PCOA is based on the twice carrier wave mechanism, which is described in Table 1. The first part of PCOA is the raw search in different chaotic traces, and the second part of PCOA is refined search to enhance the search precision.

3.2. Crossover and Merging Operation within Population.

In this paper, crossover and merging operation within population are employed in the PCOA. Both crossover and merging operation will exchange information within population and produce new potential parallel variables, which usually are different from chaotic sequences.

3.2.1. Crossover Operation. The motion step of chaotic maps between two successive iterations is usually big, which results in the big jump of the decision variable in search space. This kind of randomness of chaotic maps is benefit to jump out local optimum; however, it is not efficient for local exploitation. In this paper, the crossover is used for information interaction between parallel solutions. The crossover operation within population is illustrated in Figure 1. In the crossover, decision variable from one parallel solution \(x_j^*\) is randomly chosen to be crossed with the corresponding one from the other parallel solution. The new candidate individual by crossover operation is denoted by

\[
x_{ij}^{(C)} = \begin{cases} x_{im}^*, & \text{if } m\text{th individual is crossed}, \\ x_{ij}^*, & \text{otherwise}. \end{cases}
\]  

(6)

3.2.2. Merging Operation. Even if PCOA has reached the neighborhood of the global optimum, it needs to spend much computational effort to reach the optimum eventually by search numerous points [22]. In order to improve PCOA’s precise exploitation capability, merging operation within population is employed here. The merging operation within population is illustrated in Figure 2.

The merging operation between two parallel solutions can be denoted by

\[
x_{ij}^{(M)} = y \cdot x_{1j} + (1 - y) \cdot x_{2j},
\]  

(7)

where \(y\) is chaotic map and frequently used Logistic map is defined by the following equation [21, 29]:

\[
y_{n+1} = 4y_n(1 - y_n),
\]  

\[
y_n \in (0, 1).
\]  

(8)

This kind of merging operation within population randomly chooses two parallel solutions to merge, and it may produce new potential solutions for optimization problem. In essence, the merging operation within population is a kind of local exploiting search as shown in (7).

The crossover and merging operation within population are also used as the supplement to the twice carrier wave search during each iteration. This means that if the new parallel variables by crossover and merging operation have reached better objective function value than the original two parallel variables, the new parallel variables will replace one original parallel variable. In another situation, if the new parallel variables bring a worse objective function value than the original two parallel variables, the new parallel variable will be given up.

Both crossover and merging operation within population are conducted during each iteration in the PCOA search procedure. Another problem is to choose how many parallel variables for crossover or merging operation. The more the crossover or the merging operation, the more the diversity of parallel variables and the more the computing cost. In this paper, the crossover rate and the merging operation rate are set as \(P_{cross} = 0.1 \sim 0.5\) and \(P_{merging} = 0.1 \sim 0.5\); that is, about 10% to 50% of parallel variables have been applied through the crossover or merging operation. These parameters values are usually chosen by trial, taking into account both optimal search and computing cost.

4. PCOA for MOOPs

As far as we know, there are no literatures concerning COA for MOOPs until now, and this motivates us to extend PCOA for MOOPs. In the following discussion, the Pareto dominance concept and elitist archive mechanism are used to extend the PCOA to tackle MOOPs.

4.1. Dominance Selection Operator. To extend PCOA to MOOPs, the most important work is the selection mechanism, where the selection operation is based on the concept of Pareto dominance. According to the dominance relation between potential solutions \(x_i(k)\) and \(x_j(k)\), there may be at most three situations: (i) \(x_i(k)\) dominates \(x_j^*(k)\), (ii) \(x_j^*(k)\) dominates \(x_i(k)\), and (iii) \(x_i(k)\) and \(x_j^*(k)\) are nondominated with each other.

Thus, dominance selection operator is defined as follows:
Table 1: Algorithmic description of the PCOA.

Initialization: specify the maximum iterations \(K_1\) and \(K_2\) for twice carrier wave; set population size \(N\); random initial value of chaos variable \(y_{ij}\); parallel optimum \(P_j^* = \infty\); global optimum \(P^* = \infty\).

Begin the first carrier wave
for \(k = 1\) to \(K_1\)
  Map chaos variable \(y_{ij}(k)\) into the variance range of decision variable \(x_{ij}(k)\) by: \(x_{ij}(k) = L_x + y_{ij}(k)(U_x - L_x)\).
  Compute the objective function value of \(x_{ij}(k)\).
  Update the search result of \(P_j^*\) and \(P^*\) with \(f(x_{ij}(k))\).
  Generate next value of chaos variable by one-dimensional map function \((M): y_{ij}(k + 1) = M(y_{ij}(k))\).
end for

End the first carrier wave

Begin the second carrier wave
for \(k' = 1\) to \(K_2\)
  Compute the second carrier wave around parallel solution \(x_j^*\) by: \(x_{ij}(k') = x_j^* + \lambda_i(y_{ij}(k') - 0.5)\).
  Compute the objective function value of \(x_{ij}(k')\).
  Update the search result of \(P_j^*\) and \(P^*\) with \(f(x_{ij}(k'))\).
  Generate next value of chaos variable by one-dimensional map function \((M): y_{ij}(k' + 1) = M(y_{ij}(k'))\).
  Adjust small ergodic range by: \(\lambda_i \leftarrow t\lambda_i\).
end for

End the second carrier wave

![Crossover operation within population.](image1)

![Merging operation between parallel solutions.](image2)
$$x_i(k+1) = \{ x_i(k), \quad x_i(k) < x'_i(k), x'_i(k) < x_i(k), LC(x_i(k), x'_i(k)), x_i(k) = x'_i(k) \wedge x'_i(k) = x_i(k),$$

where $LC(x'_i(k), x_i(k))$ denotes the less crowded one between $x_i(k)$ and $x'_i(k)$. The crowding degree estimation is introduced in [7].

### 4.2. Handling Constraints

For optimization problems with constraints, the penalty function approach is frequently used to handle constraints where the constrained-domination is used to handle constraints, which is a penalty-parameterless constraint handling approach [2]. A solution $A$ is said to dominate a solution $B$, if any of the following conditions is true [9]: (i) solution $A$ is feasible and solution $B$ is not; (ii) solutions $A$ and $B$ are both infeasible, but solution $A$ has a smaller overall constraint violation; (iii) solutions $A$ and $B$ are feasible and solution $A$ dominates solution $B$. The effect of using this constrained-domination principle is that any feasible solution has a better nondomination rank than any infeasible solution. All feasible solutions are ranked according to their nondomination level based on the objective function values. However, among two infeasible solutions, the solution with a smaller constraint violation has a better rank [6].

### 4.3. External Elitist Archive

Since Zitzler and Thiele formally introduced the strength Pareto evolutionary algorithm (SPEA) [31] with the elitist reservation mechanism in 1999, many researchers have adopted similar elitist reservation concepts in practice [1, 7–9].

The main motivation for this mechanism is the fact that a solution that is nondominated with respect to its current population is not necessarily nondominated with respect to all the populations that are produced by an optimization algorithm. Thus, what we need is a way of guaranteeing that the solutions that we will report to the user are nondominated with respect to every other solution that our algorithm has produced [6]. Therefore, the most intuitive way of doing this is by storing in an external memory (or archive) all the nondominated solutions found. If a solution that wishes to enter the archive is dominated by its contents, then it is not allowed to enter. Conversely, if a solution dominates anyone stored in the archive, the dominated solution must be deleted.

In this paper, the elitist reservation strategy is also adopted. Initially, this archive is empty. As the evolution progresses, the trial vectors that are not dominated by the corresponding target vectors obtained at each generation are compared one by one with the current archive, and good solutions enter the archive. There are three cases when the nondominated trial vectors compare with the current archive [7, 9]: (a) if the trial vector is dominated by member(s) of the external archive, the trial vector is rejected; (b) if the trial vector dominates some member(s) of the archive, then the dominated members are deleted from the archive and the trial vector enters the archive; and (c) the trial vector does not dominate any archive members and none of the archive members dominates the trial vector, which implies that the trial vector belongs to the Pareto front and it enters the archive.

When the external archived population reaches its maximum capacity, the crowding entropy measure is used to keep the external archive at its maximum size as [1, 7]

$$CE_i = \sum_{j=1}^{O} \frac{c_{ij}E_{ij}}{f_{j_{\text{max}}} - f_{j_{\text{min}}}} = - \sum_{j=1}^{O} \frac{d_{ij}\log_2 (p_{ij}) + du_{ij}\log_2 (pu_{ij})}{f_{j_{\text{max}}} - f_{j_{\text{min}}}}$$

where the parameters $f_{j_{\text{max}}}$ and $f_{j_{\text{min}}}$ are the maximum and minimum values of the $j$th objective function and $D$ is the number of objective functions. Other variables have been described as in [1, 7].

### 4.4. Algorithm Procedure of PCOA for MOOPs

Consider a multiobjective optimization problem as in (1); $x$ is the decision solution vector consisting of $n$ variables $x_i \in \mathbb{R}^n$ bounded by lower ($L_i$) and upper ($U_i$) limits. $i = 1, 2, \ldots, n$, which represents each decision variable; $j = 1, 2, \ldots, N$, which represents each decision variable simultaneously mapped by multiple $N$ chaos variables. $N$ can also be considered as population size as other evolutionary algorithms. The process of the PCOA approach for MOOPs is described as follows, and it is illustrated in Figure 3.

Step 1: set the iterations number $k = 1$, specify the maximum number of iterations $k_{\text{max}}$, switch point from the first carrier wave to the second carrier wave $S_1$, initialize chaotic maps $0 < \gamma_{ij}(k) < 1$ randomly, set population size $N$, set crossover probability and merging probability, and initialize the external elitist archive $A^{(0)} = \emptyset$ and its maximum size $N_{\text{max}}$.

Step 2: map chaotic maps $\gamma_{ij}(k)$ onto the variance range of decision variables $x_{ij}(k)$ by the following two ways.

If $k < S_1$, PCOA search using the first carrier wave is

$$x_{ij}(k) = L_i + \gamma_{ij}(k)(U_i - L_i).$$

If $k \geq S_1$, PCOA search using the second carrier wave is

$$x_{ij}(k) = x_{ij}^* + \lambda(\gamma_{ij}(k) - 0.5),$$

where $\lambda$ is a very important local search parameter and adjusts small ergodic ranges around so far global solutions $x_{ij}^*$.

Step 3: evaluate the fitness value of each target vector $x_{ij}(k)$.

Step 4: in crossover operation within population, produce $x_{ij}^{(C)}(k)$ and evaluate the trial vector $x_{ij}^{(C)}(k)$.

Step 5: in merging operation within population, produce $x_{ij}^{(M)}(k)$ and evaluate the trial vector $x_{ij}^{(M)}(k)$.

Step 6: perform selection operation between $x_{ij}(k)$, $x_{ij}^{(C)}(k)$, and $x_{ij}^{(M)}(k)$.
If \( x_{ij}(k) \) dominates \( x_{ij}^{(C)}(k) \), update \( A(k) \) with the elitist archive update rules. If \( x_{ij}(k) \), \( x_{ij}^{(C)}(k) \), and \( x_{ij}^{(M)}(k) \) are nondominated with respect to each other, update \( A(l) \) with elitist archive update rules. Each vector in \( A(k) \) has stored its respective global solution \( x^*_ij \).

Step 7: size external elitist archive \( A(k) \). When \( A(k) \) exceeds the maximum size, the less crowded vectors based on the distance in (10) keep the archive size at \( N_{\text{max}} \).

Step 8: generate next values of chaotic maps by a chaotic mapping function \( (M) \) as in (8):

\[
y_{ij}(k + 1) = M\left( y_{ij}(k) \right).
\] (13)

Step 9: if \( k \geq k_{\text{max}} \), stop the search process; otherwise \( k \leftarrow k + 1 \) and go to Step 2.

5. MOOPs Test Simulation

This section is devoted to presenting the experiments performed in this work. First, the set of MOOPs used as a benchmark and the quality indicators applied for measuring the performance of the resulting Pareto fronts are introduced. Next, our preliminary experiments of PCOA are described and analyzed. Then, PCOA is evaluated and compared to other multiobjective optimization algorithms.

5.1. MOOPs Test Problems. Different sets of classical test problems suggested in the MOOPs literature are used to estimate the performance of the PCOA.

Among these classical test problems, the first set is the following biobjective unconstrained problems: Schaffer, Fonseca, and Kursawe, as well as the problems ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 [1, 2]. The second set includes the constrained biobjective problems ConstrEx, Srinivas, and Tanaka [1, 2]. In these test problems, ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 can be considered as high-dimensional tests (with \( n = 30 \) for ZDT1, ZDT2, and ZDT3; \( n = 10 \) for ZDT4 and ZDT6).

5.2. Performance Measures. In order to determine whether an algorithm can solve MOOPs efficiently, the algorithm’s performance should be quantified with functions referred to

Figure 3: Flowchart of the PCOA approach for MOOPs.
as performance measures. A comprehensive overview of performance measures currently used in the multiobjective optimization problems literatures has been provided in [32]. There are three goals in a multiobjective optimization: (i) convergence to the Pareto-optimal set, (ii) maintenance of diversity in solutions of the Pareto-optimal set, and (iii) maximal distribution bound of the Pareto-optimal set. In this article, three quality indicators evaluating each type of the above goals are introduced as follows.

(a) Generational distance (GD) [1]: the concept of generational distance was used to measure how far the elements are in the set of nondominated vectors found so far from those in the Pareto-optimal set. This indicator is defined as

\[
GD = \sqrt{\frac{\sum_{i=1}^{n} d_i^2}{n}},
\]  

(14)

where \( n \) is the number of vectors in the set of nondominated solutions found so far and \( d_i \) is the Euclidean distance (measured in objective space) between each of these solutions and the nearest member of the Pareto-optimal set. A smaller value of GD demonstrates a better convergence to the Pareto front.

(b) Spread (\( \Delta \)) [2]: this indicator is to measure the extent of spread archived among the obtained nondominated solutions. The metric is defined as

\[
\Delta = \frac{d_j + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_j + d_l + (N - 1)\bar{d}},
\]  

(15)

where \( N \) is the number of nondominated solutions found so far, \( d_j \) is the Euclidean distance between neighboring solutions in the obtained nondominated solutions set, \( \bar{d} \) is the mean of all \( d_j \) and \( d_f \) and \( d_l \) are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained nondominated set. A smaller value of \( \Delta \) indicates a better distribution and diversity of the nondominated solutions.

(c) Hypervolume (HV) [1]: the reference point can be found simply by constructing a vector of the worst objective function values. Thereafter, a union of all hypercubes is found and its HV is calculated:

\[
HV = \bigcup_{i=1}^{\Omega} v_i,
\]  

(16)

where \( \Omega \) is the members of nondominated set of solutions. Algorithms with larger HV values are desirable. Since the calculation of the HV is related to the reference point, in our experiment, the HV value of a set of solutions is normalized by a reference set of Pareto-optimal solutions with the same reference point. After normalization, the HV values are confined in [0, 1].

In order to know how competitive the proposed PCOA approach is, it is compared with several popular multiobjective algorithms (MOAs) as follows: nondominated sorting genetic algorithm-II (NSGA-II), strength Pareto evolutionary algorithm 2 (SPEA2), and multiobjective particle swarm optimization (MOPSO), and these MOAs are representative of the state of the art [1]. In the simulations, the parameter values of NSGA-II, SPEA2, and MOPSO are the same as in [1]. The parameters of PCOA used in this simulation are chosen by trial as follows: parallel number \( N = 100 \), Logistic map in (8) used as the chaotic map, crossover probability \( P_{cross} = 0.5 \), merging probability \( P_{merging} = 0.5 \), archive size \( N_{max} = 100 \), \( k_{max} = 300 \) for low-dimensional tests (Schaffer, Fonseca, Kursawe, ConstrEx, Srinivas, and Tanaka) and \( k_{max} = 600 \) for high-dimensional tests (ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6), switch point \( S_1 = 0.5 \times k_{max} \), and local search parameter in (15) \( \lambda = 1.5 \). To avoid randomness, all these MOAs independently run 30 times on each test problem.

5.3. Simulation Results. The Pareto fronts obtained with the proposed PCOA on different MOOPs have been illustrated in Figures 4 and 5. It can be seen from Figures 4 and 5 that the proposed PCOA can usually obtain Pareto fronts on different MOOPs, and Pareto fronts obtained with the PCOA are more perfect and uniformly distributed for low-dimensional tests (Figure 4).

Tables 1–3 and Figures 6–8 have reported the simulation results of the previously described quality indicators (generational distance (GD), spread (\( \Delta \)), and hypervolume (HV)) using the four MOAs: PCOA, NSGA-II, SPEA2, and MOPSO. In the simulation results, the "mean" is the average values of 30 runs, and "SD" is the standard deviation (SD) of 30 runs of each MOAs.

Table 1 shows the generational distance (GD) indicator in these tests, and Figure 6 shows the mean of GD for these MOAs. In the group of low-dimensional tests (Schaffer, Fonseca, Kursawe, ConstrEx, Srinivas, and Tanaka), the resulting Pareto fronts from PCOA are as close as those computed by NSGA-II, SPEA2, and MOPSO. In the group of ZDT problems (high-dimensional tests), PCOA obtains better results than MOPSO. This can also be seen from Figures 4 and 5 that the Pareto fronts in Figure 5 (high-dimensional tests) are not as perfect as those in Figure 4 (low-dimensional tests). From Table 1 and Figure 6, it can be seen that PCOA can reach Pareto fronts for all these tests, while its performance in low-dimensional tests is more competitive.

Table 2 shows the spread indicator (\( \Delta \)) in these tests, while Figure 7 shows the mean of \( \Delta \) for these MOAs. The results obtained from \( \Delta \) in Table 2 and Figure 7 indicate that PCOA outperforms the other three MOAs concerning the diversity of the obtained Pareto fronts. PCOA usually yields
Figure 4: Pareto fronts obtained with the PCOA on different MOOPs. (a) Schaffer. (b) Kursawe. (c) Fonseca. (d) ConstrEx. (e) Srinivas. (f) Tanaka.

Figure 5: Pareto fronts obtained with the PCOA on different MOOPs. (a) ZDT1. (b) ZDT2. (c) ZDT3. (d) ZDT4. (e) ZDT6.
the lowest values in almost all MOOPs as PCOA is a kind of global search. From Table 2 and Figure 7, we can see that PCOA obtains better results for Δ in all the test problems than NSGA-II, while PCOA obtains better results in most test problems than SPEA2 and MOPSO. This means that the PCOA approach has shown good diversity, and this may attribute its success to PCOA’s parallel search pattern and escaping from local optima.

Table 3 shows the hypervolume (HV) indicator in these tests, while Figure 8 shows the mean of HV for these MOAs. From Table 3 and Figure 8, we can see that PCOA approach obtains very large HV values in low-dimensional tests while its HV in high-dimensional tests is also good. It can be seen from these results that PCOA approach outperforms MOPSO based on the HV indicator Table 4.

From the above results, it can be seen that the proposed PCOA approach has shown good performance based on generational distance, spread indicator, and HV indicator. In all these simulation results, the PCOA outperforms MOPSO and it is as good as NSGA-II and SPEA2. This means that the PCOA can be used as an alternative approach for MOOPs.

5.4. Comparing Algorithm Parameters. In order to test the performance of crossover probability and merging probability on the proposed PCOA approach, here the PCOA with different crossover probability and merging probability is also compared (other PCOA parameters values are the same as those in the former simulation). Case 1, crossover probability and merging probability are all 0.5 (denoted by PCOA1); Case 2, crossover probability and merging probability are all 0.3 (denoted by PCOA2); Case 3, crossover probability and merging probability are all 0.2 (denoted by PCOA3). The simulation results of the PCOA with different crossover probability and merging probability have been shown in Figure 9.

It can be seen from Figure 9 that, with higher crossover probability and merging probability, we may have more potential trial solutions for MOOPs; therefore, its performance with quality indicators (generational distance (GD), spread (Δ), and hypervolume (HV)) will be better.

6. Mixed $H_2/H_\infty$ Controller Design

The mixed $H_2/H_\infty$ control synthesis problem is an important multiobjective controller design problem in the field of control theory, and it has received a great deal of attention in recent years. The most popular approach for solving this problem is the linear matrix inequalities (LMIs) approach [33, 34]. Recently, the mixed $H_2/H_\infty$ control synthesis problem is stated as a multiobjective optimization problem, with objectives of minimizing $H_2$ and $H_\infty$ norms simultaneously. In this section, we will apply the proposed PCOA approach for the mixed $H_2/H_\infty$ control multiobjective optimal design. Consider the following system equations for $H_2/H_\infty$ control synthesis:
Table 2: Comparison results of PCOA and other MOAs based on the generational distance.

| Function  | PCOA  | NSGA-II | SPEA2 | MOPSO |
|-----------|-------|---------|-------|-------|
| Schaffer  | 2.3407e-3 | 2.9498e-4 | 2.1572e-3 | 2.0999e-4 |
| Kursawe   | 3.1164e-3 | 3.7829e-4 | 2.8974e-3 | 2.3637e-4 |
| Fonseca   | 2.1940e-3 | 3.8326e-4 | 2.5656e-3 | 2.0082e-4 |
| ConstrEx  | 4.9281e-3 | 7.3116e-3 | 5.1349e-3 | 2.4753e-4 |
| Srinivas  | 2.5572e-3 | 4.9703e-4 | 3.7069e-3 | 5.1034e-4 |
| Tanaka    | 3.7950e-3 | 1.0075e-3 | 4.0488e-3 | 3.4365e-4 |
| ZDT1      | 1.3367e-1 | 4.2378e-2 | 1.3437e-3 | 1.4078e-4 |
| ZDT2      | 2.0960e-1 | 5.4732e-2 | 9.8112e-4 | 6.4133e-4 |
| ZDT3      | 2.8144e-1 | 3.5513e-2 | 2.4783e-3 | 1.2746e-4 |
| ZDT4      | 6.5237e-2 | 4.2511e-2 | 5.1635e-3 | 1.3281e-3 |
| ZDT6      | 7.3128e-2 | 6.5390e-3 | 7.5818e-2 | 6.0797e-3 |

Table 3: Comparison results of PCOA and other MOAs based on the spread indicator $\Delta$.

| Function  | PCOA  | NSGA-II | SPEA2 | MOPSO |
|-----------|-------|---------|-------|-------|
| Schaffer  | 0.19945 | 2.1953e-2 | 0.29228 | 2.1393e-2 |
| Kursawe   | 0.21514 | 2.8631e-2 | 0.43404 | 1.9817e-2 |
| Fonseca   | 0.19583 | 1.7106e-2 | 0.37672 | 2.5222e-2 |
| ConstrEx  | 0.40773 | 6.1873e-2 | 0.54863 | 2.7171e-2 |
| Srinivas  | 0.16950 | 3.7642e-2 | 0.3869  | 2.5115e-2 |
| Tanaka    | 0.69843 | 6.9733e-2 | 0.82286 | 2.8678e-2 |
| ZDT1      | 0.37906 | 4.1615e-2 | 0.50429 | 3.9251e-2 |
| ZDT2      | 0.45027 | 3.7390e-2 | 0.48775 | 2.7686e-2 |
| ZDT3      | 0.42798 | 6.3524e-2 | 0.59025 | 3.0439e-2 |
| ZDT4      | 0.36412 | 5.0661e-2 | 0.37524 | 2.4448e-2 |
| ZDT6      | 0.20572 | 2.9758e-2 | 0.48611 | 3.6054e-2 |

Table 4: Comparison results of PCOA and other MOAs based on the HV indicator.

| Function  | PCOA  | NSGA-II | SPEA2 | MOPSO |
|-----------|-------|---------|-------|-------|
| Schaffer  | 0.996461 | 2.1130e-4 | 0.996845 | 1.0374e-4 |
| Kursawe   | 0.981255 | 4.1259e-3 | 0.786379 | 3.6251e-4 |
| Fonseca   | 0.984921 | 2.5437e-3 | 0.979604 | 1.0286e-3 |
| ConstrEx  | 0.994527 | 5.9124e-4 | 0.991616 | 5.8481e-4 |
| Srinivas  | 0.993786 | 5.1601e-4 | 0.99316  | 2.2948e-4 |
| Tanaka    | 0.995682 | 1.0024e-3 | 0.99759  | 9.2767e-4 |
| ZDT1      | 0.853766 | 6.1083e-2 | 0.994704 | 1.8910e-4 |
| ZDT2      | 0.783503 | 1.1973e-1 | 0.995543 | 2.1983e-3 |
| ZDT3      | 0.771385 | 1.3570e-1 | 0.992548 | 9.8126e-3 |
| ZDT4      | 0.916307 | 8.254e-3  | 0.989682 | 1.7974e-3 |
| ZDT6      | 0.901684 | 8.863e-3  | 0.872076 | 1.0198e-2 |

\[
\begin{bmatrix}
-0.3868 & 0.0751 \\
0 & -0.0352 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
x \\
x
\end{bmatrix} +
\begin{bmatrix}
-0.6965 \\
1.6961 \\
0
\end{bmatrix}u +
\begin{bmatrix}
0.0591 \\
0 \\
1.7971
\end{bmatrix}w,
\]

\[
\begin{bmatrix}
-0.0346 & 0.0535 \\
0 & 0 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
x \\
x
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
1.5297
\end{bmatrix}u.
\]

where \( z = z_2 = z_3 \).

For the multiobjective control synthesis, the solutions obtained by the PCOA will be compared with the solutions calculated with the function "msfyn" provided by the MATLAB LMI Control Toolbox. As the LMI-based approach can only find a single solution in each run, the set of solutions of the multiobjective problem is obtained by varying \( H_\infty \), upper bound as \( \gamma = 0.1, 0.2, \ldots, 0.9 \).

Figure 10 illustrates the Pareto estimates (\( H_1 \) and \( H_\infty \) closed-loop norms) obtained by the two approaches. It can be seen that PCOA is able to find a set of estimates of the Pareto front that are more equally distributed and are with a better extension over the conflicting objectives. Figure 11 shows the obtained solutions in the parameters space \((k_1 \times k_3)\). Considering the analyzed example, the proposed synthesis procedures present better results than LMI-based approaches.
Figure 9: PCOA approach with different parameter values. Comparison results of mean of (a) GD, (b) spread, and (c) HV.

Figure 10: Pareto solutions obtained with PCOA and LMI.

Figure 11: Pareto solutions in the parameters space.
7. Conclusion
As far as we know, there are no literatures concerning COA for MOOPs until now, and this motivates us to propose PCOA for MOOPs. In this paper, a PCOA with crossover and merging operation is proposed for MOOPs. Both crossover and merging operation can exchange information between parallel variables and produce new potential solutions, which can enhance the global and fast search ability of the proposed algorithm. To test the effectiveness of the PCOA, it is simulated with several benchmark functions for MOOPs and mixed $H_2/H_{\infty}$ controller design. The simulation results have shown that PCOA can be an alternative approach for MOOPs.

Data Availability
The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
This work was supported in part by the National Key R&D Program of China (no. 2018YFF0212900).

References
[1] Y.-N. Wang, L.-H. Wu, and X.-F. Yuan, "Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure," Soft Computing, vol. 14, no. 3, pp. 193–209, 2010.
[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.
[3] H. M. Lu and G. G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," IEEE Transactions on Evolutionary Computation, vol. 7, no. 4, pp. 325–343, 2003.
[4] L. Ke, Q. Zhang, and R. Battiti, "MOEA/D-ACO: a multi-objective evolutionary algorithm using decomposition and AntColony," IEEE Transactions on Cybernetics, vol. 43, no. 6, pp. 1845–1859, 2013.
[5] M. Li, S. Yang, K. Li, and X. Liu, "Evolutionary algorithms with segment-based search for multiobjective optimization problems," IEEE Transactions on Cybernetics, vol. 44, no. 8, pp. 1295–1313, 2014.
[6] C. Coello and A. Carlos, "Evolutionary multi-objective optimization: a historical view of the field," IEEE Computational Intelligence Magazine, vol. 1, no. 1, pp. 28–36, 2006.
[7] L. H. Wu, Y. N. Wang, X. F. Yuan, and S. W. Zhou, "Environmental/economic power dispatch problem using multi-objective differential evolution algorithm," Electric Power Systems Research, vol. 80, no. 9, pp. 1171–1181, 2010.
[8] H. Xia, J. Zhuang, and D. Yu, "Combining crowding estimation in objective and decision space with multiple selection and search strategies for multi-objective evolutionary optimization," IEEE Transactions on Cybernetics, vol. 44, no. 3, pp. 378–393, 2014.
[9] L. Wu, Y. Wang, X. Yuan, and Z. Chen, "Multiobjective optimization of HEV fuel economy and emissions using the self-adaptive differential evolution algorithm," IEEE Transactions on Vehicular Technology, vol. 60, no. 6, pp. 2458–2470, 2011.
[10] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: a multi-objective approach," IEEE Transactions on Cybernetics, vol. 43, no. 6, pp. 1656–1671, 2013.
[11] Z. H. Zhan, J. Li, J. Cao, J. Zhang, H. S Chung, and Y. H Shi, "Multiple populations for multiple objectives: a coevolutionary technique for solving multiobjective optimization problems," IEEE Transactions on Cybernetics, vol. 43, no. 2, pp. 445–463, 2013.
[12] F. Freschi and M. Repetto, "VIS: an artificial immune network for multi-objective optimization," Engineering Optimization, vol. 38, no. 8, pp. 975–996, 2006.
[13] I. Aydin, M. Karakose, and E. Akin, "A multi-objective artificial immune algorithm for parameter optimization in support vector machine," Applied Soft Computing, vol. 11, no. 1, pp. 120–129, 2011.
[14] B. Yagmahan and M. M. Yenisey, "Ant colony optimization for multi-objective flow shop scheduling problem," Computers & Industrial Engineering, vol. 54, no. 3, pp. 411–420, 2008.
[15] S. N. Omkar, J. Senthilnath, R. Khandelwal, G. Narayana Naik, and S. Gopalakrishnan, "Artificial Bee Colony (ABC) for multi-objective design optimization of composite structures," Applied Soft Computing, vol. 11, no. 1, pp. 489–499, 2011.
[16] Q. Zhu, X. Yuan, and H. Wang, "An improved chaos optimization algorithm-based parameter identification of synchronous generator," Electrical Engineering, vol. 94, no. 3, pp. 147–153, 2012.
[17] D. Yang, Z. Liu, and J. Zhou, "Chaos optimization algorithms based on chaotic maps with different probability distribution and search speed for global optimization," Communications in Nonlinear Science and Numerical Simulation, vol. 19, no. 4, pp. 1229–1246, 2014.
[18] T. Hamaizia, R. Lozi, and N.-E. Hamri, "Fast chaotic optimization algorithm based on locally averaged strategy and multifold chaotic attractor," Applied Mathematics and Computation, vol. 219, no. 1, pp. 188–196, 2012.
[19] B. Li and W. S. Jiang, "Optimizing complex function by chaos search," Cybernetics and Systems, vol. 29, no. 4, pp. 409–419, 1998.
[20] D. Yang, G. Li, and G. Cheng, "On the efficiency of chaos optimization algorithms for global optimization," Chaos, Solitons & Fractals, vol. 34, no. 4, pp. 1366–1375, 2007.
[21] M. S. Tavazoei and M. Haeri, "Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms," Applied Mathematics and Computation, vol. 187, no. 2, pp. 1076–1085, 2007.
[22] X. F. Yuan and Y. N. Wang, "Parameter selection of support vector machine for function approximation based on chaos optimization," Journal of Systems Engineering and Electronics, vol. 19, no. 1, pp. 191–197, 2008.
[23] I. Pan and S. Das, "Frequency domain design of fractional order PID controller for AVR system using chaotic multi-objective optimization," International Journal of Electrical Power & Energy Systems, vol. 51, pp. 106–118, 2013.
[24] H. Lu, R. Niu, J. Liu, and Z. Zhu, "A chaotic non-dominated sorting genetic algorithm for the multi-objective automatic
test task scheduling problem,” Applied Soft Computing, vol. 13, no. 5, pp. 2790–2802, 2013.

[25] H. Zhang, J. Zhou, Y. Zhang, N. Fang, and R. Zhang, “Short term hydrothermal scheduling using multi-objective differential evolution with three chaotic sequences,” International Journal of Electrical Power & Energy Systems, vol. 47, pp. 85–99, 2013.

[26] J. Chen, Q. Lin, and Z. Ji, “Chaos-based multi-objective immune algorithm with a fine-grained selection mechanism,” Soft Computing, vol. 15, no. 7, pp. 1273–1288, 2011.

[27] B. Alatas and E. Akin, “Multi-objective rule mining using a chaotic particle swarm optimization algorithm,” Knowledge-Based Systems, vol. 22, no. 6, pp. 455–460, 2009.

[28] J. Cai, X. Ma, Q. Li, L. Li, and H. Peng, “A multi-objective chaotic particle swarm optimization for environmental/economic dispatch,” Energy Conversion and Management, vol. 50, no. 5, pp. 1318–1325, 2009.

[29] J. Cai, X. Ma, Q. Li, L. Li, and H. Peng, “A multi-objective chaotic ant swarm optimization for environmental/economic dispatch,” International Journal of Electrical Power & Energy Systems, vol. 32, no. 5, pp. 337–344, 2010.

[30] K. Ayan and U. Kilic, “Solution of multi-objective optimal power flow with chaotic artificial bee colony algorithm,” International Review of Electrical Engineering, vol. 6, no. 3, pp. 1365–1371, 2011.

[31] C. Scherer, P. Gahinet, and M. Chilali, “Multiobjective output-feedback control via LMI optimization,” IEEE Transactions on Automatic Control, vol. 42, no. 7, pp. 896–911, 1997.

[32] B. S. Chen, C. S. Tseng, and H. J. Uang, “Mixed H₂/H-infinity fuzzy output feedback control design for nonlinear dynamic systems: an LMI approach,” IEEE Transactions on Fuzzy Systems, vol. 8, no. 3, pp. 249–265, 2000.

[33] L. Wu, Y. Wang, S. Zhou, and X. Yuan, “Design of mixed H-two/H-infinity optimal control systems using multi-objective differential evolution algorithm,” Journal of Control Theory and Applications, vol. 11, no. 3, pp. 521–528, 2013.

[34] F. G. Guimarães, R. M. Palhares, F. Campelo, and H. Igarashi, “Design of mixed H2/H∞ control systems using algorithms inspired by the immune system,” Information Sciences, vol. 177, no. 20, pp. 4368–4386, 2007.