2014

The Effects of Using Chaotic Map on Improving the Performance of Multiobjective Evolutionary Algorithms

Hui Lu
Beihang University, China

Xiaoteng Wang
Beihang Univeristy, China

Zongming Fei
University of Kentucky, zongming.fei@uky.edu

Meikang Qiu
San Jose State University

Digital Object Identifier (DOI)
http://dx.doi.org/10.1155/2014/924652

Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 924652, 16 pages
http://dx.doi.org/10.1155/2014/924652

Research Article

The Effects of Using Chaotic Map on Improving the Performance of Multiobjective Evolutionary Algorithms

Hui Lu,[1] Xiaoteng Wang,[1] Zongming Fei,[2] and Meikang Qiu[3]

[1] *School of Electronic and Information Engineering, Beihang University, Beijing 100191, China*
[2] *Department of Computer Science, University of Kentucky, Lexington, KY 40506-0046, USA*
[3] *Department of Computer Engineering, San Jose State University, San Jose, CA 95192, USA*

Correspondence should be addressed to Hui Lu; mluhui@buaa.edu.cn

Received 3 December 2013; Accepted 10 January 2014; Published 27 February 2014

Academic Editor: Rongni Yang

Chaotic maps play an important role in improving evolutionary algorithms (EAs) for avoiding the local optima and speeding up the convergence. However, different chaotic maps in different phases have different effects on EAs. This paper focuses on exploring the effects of chaotic maps and giving comprehensive guidance for improving multiobjective evolutionary algorithms (MOEAs) by series of experiments. NSGA-II algorithm, a representative of MOEAs using the nondominated sorting and elitist strategy, is taken as the framework to study the effect of chaotic maps. Ten chaotic maps are applied in MOEAs in three phases, that is, initial population, crossover, and mutation operator. Multiobjective problems (MOPs) adopted are ZDT series problems to show the generality. Since the scale of some sequences generated by chaotic maps is changed to fit for MOPs, the correctness of scaling transformation of chaotic sequences is proved by measuring the largest Lyapunov exponent. The convergence metric γ and diversity metric Δ are chosen to evaluate the performance of new algorithms with chaos. The results of experiments demonstrate that chaotic maps can improve the performance of MOEAs, especially in solving problems with convex and piecewise Pareto front. In addition, cat map has the best performance in solving problems with local optima.

1. Introduction

Multiobjective evolutionary algorithms have attracted widespread attention and have been applied successfully in many areas, such as test task scheduling problem (TTSP) [1], reservoir operation [2], proportional integral and derivative (PID) controller [3], and distribution feeder reconfiguration (DFR) [4]. One key challenge in multiobjective evolutionary algorithms is the problem of resolving local optima and the speed of convergence. There are different solutions for improving evolutionary algorithms. Some approaches have been devoted to propose new algorithms, such as MOEA/D [5], SPEA-2 [6], and NSGA-II [7]. Other researchers have proposed a variety of hybrid algorithms, which combined the advantages of two different methods. For example, a new hybrid evolutionary algorithm (EA) based on the combination of the honey bee mating optimization (HBMO) and the discrete particle swarm optimization (DPSO), called DPSO-HBMO, is applied to solve the multiobjective distribution feeder reconfiguration (DFR) problem [4]. Another approach has focused on modifying original algorithms. For example, new particle swarm optimization (PSO) methods were proposed by using chaotic maps for parameter adaptation [8]. The results showed that chaos embedded PSO can improve the quality of results in some optimization problems. Chaos variables are loaded into the variable colony of the immune algorithm in the immune evolutionary algorithm, and the experimental results indicate that the new immune evolutionary algorithm improves the convergence performance and search efficiency [9]. Due to the characteristics such as randomness, regularity, ergodicity, and initial value sensitiveness, chaos has been widely applied in the original evolutionary algorithms to improve the performance.

Recently researches have been done to the chaos embed in evolutionary algorithms. For example, Alatas et al. [8] applied seven chaotic maps to generate seven new chaotic artificial bee colony algorithms. Three phases were adopted in generating these algorithms to solve three different benchmark single objective problems. Results showed that these methods have somewhat improved the solution quality. Tavazoei and Haeri [10] introduced ten chaotic maps in weighted gradient direction to solve two test functions. Results showed that none of these maps transcends other maps for all of the problems and desired criteria. Those researches demonstrated that chaotic sequences replacing the random parameters in three phases, including initial population, crossover operator, and mutation operator, can improve the performance of evolutionary algorithms. However, questions remain that for a given MOP, which chaotic map should be chosen in order to achieve the best performance. It is also not clean what kinds of combination of chaotic maps used in a particular phase have the best property. Therefore, it is difficult to give comprehensive guidance to improve the performance of evolutionary algorithms.

In addition, from the problems solved by COA, it can be seen that single objective optimization problems are the focus. Comparisons of different chaotic maps in improving the effects of COAs for solving single objective problems are common, but it is rare in solving multiobjective optimization problems (MOPs). Yu et al. [11] revealed that COA is not effective for solving MOPs, whereas the experiments in Alatas and Akin [12] showed the opposite. The results on these foregoing researches demonstrate that COAs are successful and competitive for solving single objective optimization problem, but effects of COAs on solving MOPs are not consistent.

In summary, although there have been many researches about the chaos and its application in COAs, the effects of different chaotic maps used in different phases on the performance of evolutionary algorithms have not yet been fully evaluated, especially for the multiobjective evolutionary algorithms.

In this paper, we explore the relationships of chaotic maps and phases on improving multiobjective evolutionary algorithms by a series of experiments. We will answer the question whether chaotic maps are suitable to improve the evolutionary algorithms in solving MOPs. We also investigate which phase should be chosen when one chaotic map is used to improve a multiobjective evolutionary algorithm.

In this research, NSGA-II is chosen as the main optimization algorithm, because it captures the core ideas and characteristics of MOEAs with the properties of a fast nondominated sorting procedure, an elitist strategy, a parameterless approach, and a simple yet efficient constraint-handling method [7]. Despite these good aspects of NSGA-II for solving MOPs, it may be entrapped into local optimal solutions. Thus, the properties of chaos can help to improve the performance of NSGA-II.

In order to reflect the diversity of chaotic maps, ten chaotic maps that have been widely used in pioneering researches are studied in this paper. They are circle map, cubic map, Gauss map, ICMIC map, logistic map, sinusoidal map, tent map, Baker's map, cat map, and Zaslavskii map. Each chaotic map has its own property and has its own effect on improving the performance of evolutionary algorithms. For example, logistic map has Chebyshev-type distribution but not uniform distribution. As a result, it is necessary for optimal solution to go through multiple iterations.

Similar to past researches, chaotic maps are used in three common phases in evolutionary algorithms in experiments, that is, chaotic sequences for initial population, chaotic sequences for crossover operator, and chaotic sequences for mutation operator.

Five benchmark MOPs including ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 [7] are chosen as test problems. These MOPs have different characteristics and can reflect the property of evolutionary algorithms from different aspects. For example, we can use problem ZDT4 to evaluate the performance of evolutionary algorithms for resolving local optimal, because ZDT4 has different local Pareto-optimal solutions in the search space.

In addition, ranges of chaotic maps are not always fit for test problems. Scaling transformation is needed to apply chaotic sequences. For example, Coelho and Mariani [13] adopted Zaslavskii's map by changing its range to $(0, 1)$ and Alatas [12, 14, 15] took a similar approach. The problem is whether the chaotic sequences through scaling transformation still maintain the properties of chaos. In this paper, the correctness of scaling transformation of chaotic sequences is proved by measuring the largest Lyapunov exponent.

Finally, the criteria of convergence and distribution proposed by Deb et al. [7] are adopted in this paper to evaluate the effects of the combinations of phases and chaotic maps on improving the performance of multiobjective evolutionary algorithms. One is metric γ, which measures the extent of convergence to a known set of Pareto-optimal solutions. The other is metric Δ, which measures the extent of spread achieved among the obtained solutions.

From the results of experiments, it can be seen that NSGA-II embedded with chaotic maps in most cases get better results with regard to the metrics γ and Δ. The effects of using chaotic maps depend on which chaotic map is selected and in which phase it is used. In particular, chaos can improve the ability of NSGA-II in solving ZDT3 and ZDT6, which are difficult for the original NSGA-II algorithm. Besides, cat map is good at solving problems with local optima, such as ZDT4.

The rest of paper is organized as follows. Section 2 gives a summary of related work on applying chaos to improve evolutionary algorithms. Section 3 shows the phases in which chaos can be embedded in evolutionary algorithms. Section 4 defines ten chaotic maps which are embedded in NSGA-II in the experiments. Section 5 proves that the chaotic sequences through scaling transformation still hold the properties of chaos. Section 6 describes the test problems and metrics used in the experiments. Section 7 presents the performance results of the experiments. Section 8 concludes the paper.

2. Related Work

Applying chaotic maps to improve evolutionary algorithms has been studied for a while. There are two different strategies to apply the chaotic maps in the evolutionary algorithms.

One is to use chaotic sequences generated by chaotic maps to replace the random parameters needed by evolutionary algorithms. Coelho [16] proposed a quantum-behaved particle swarm optimization (QPSO). Random sequences of mutation operator in QPSO were replaced with chaotic sequences based on Zaslavskii map. The results demonstrated that it is a powerful strategy to diversify the population and improve the performance in preventing premature convergence to local minima. Dos Coelho and Alotto considered the chaotic crossover operator using the Zaslavskii map to solve multiobjective optimization problems [17]. Zhang et al. [18] proposed three chaotic sequences based multiobjective differential evolution (CS-MODE) to solve short-term hydrothermal optimal scheduling with economic emission (SHOSEE). In the modified mutation operator, chaotic theory is used to increase the population diversity, and some adaptive tuning parameters are produced by chaotic mappings to control the evolution.

The other strategy is to use the chaos optimization as an operator. For example, Alatas [14] applied chaotic search in case that a solution does not obtain improvements in artificial bee colony (ABC) algorithm. The results showed that the strategy has better performance than that of ABC algorithm. Wang and Zhang [19] employed chaos analogously. When the value of objective function had no improvement in continuous iterations, one chaotic system was applied to reinitialize half of the population. It replaced the worst half part of the population in order to jump out of the local optimum, whereas the best half part is kept unchanged.

Since evolutionary algorithms have sensitive dependence on their initial condition and parameters, the improvements on these parameters can have a good effect. That may be one of the reasons that the first strategy is widely adopted. In the first strategy, it is necessary to consider the phases of replacing random sequences with chaotic sequences and the different chaotic maps adopted.

For the phases of the evolutionary algorithms, Caponetto et al. [20] introduced chaotic sequences instead of random ones during all the phases of the evolution process. Results showed that the behaviors of all operators were influenced by chaotic sequences. Alatas [15], Ahmadi and Mojallali [21], and Ma [22] focused on random parameters in initial population. Coelho [16] and Zhang et al. [18] did their research on mutation operator. However, which phase is the best choice was not discussed.

To study the performance of different chaotic maps, some researchers give the comparisons of different chaotic maps solving both single objective optimization problems and MOPs. Talatahari et al. [23] proposed a novel chaotic improved imperialist competitive algorithm (CICA) for global optimization. Seven chaotic maps were utilized to improve the movement step of the algorithm, and the logistic and sinusoidal maps were found as the best choices. Caponetto et al. [20] proposed an experimental analysis on the convergence of evolutionary algorithms. Six chaotic maps, four phases, and single-objective statistical tests showed an improvement of evolutionary algorithms when chaotic sequences were used instead of random processes. Lu et al. [1] proposed a chaotic nondominated sorting genetic algorithm (CNSGA) to solve the automatic test task scheduling problem (TTSP). According to the different capabilities of the logistic and the cat chaotic operators, the CNSGA approach using the cat population initialization, the cat or logistic crossover operator, and the logistic mutation operator performs well and is very suitable for solving the TTSP. The comparisons of the performance of chaotic maps in these researches are based on solving one specific problem, so the results cannot be generalized to offer guidance on how to choose a chaotic map for solving other problems. Furthermore, most researches focus on single objective problems.

In contrast, this paper performs extensive experiments on genetic multiobjective evolutionary algorithms embedded with chaotic sequences. It focuses on exploring the relationships of phases and chaotic maps on improving multiobjective evolutionary algorithms. As mentioned above ten chaotic maps and three phases of evolutionary algorithms are considered. Five general benchmark problems are used to demonstrate that the conclusions can be generalized. Finally the guidance is presented to help researchers choose the suitable chaotic map and phases in multiobjective evolutionary algorithms for different MOPs.

3. Phases in Chaos Embedded Evolutionary Algorithms

With the ergodic property, chaos is adopted to enrich the searching behavior and to avoid solutions being trapped into local optimum in optimization problems. In this section three key phases in evolutionary algorithms, initialization, crossover, and mutation, are chosen to be embedded with chaos. Those three phases are described as follows.

3.1. Initialization. Initial population is the starting point of iterations. Ergodicity and diversity of initial population are very important for making sure that the individuals in the population spread in the search spaces uniformly as far as possible. In this case, initial population is generated by chaotic maps which can form a feasible solution space with good distribution by the properties of randomicity and ergodicity of chaos. Chaotic sequences can guarantee the diversity of the initial population, speed up its convergence, and enhance global search capability.

More specifically, a chaotic map, such as logistic map or cat map, is adopted instead of random population initialization of evolutionary algorithms. In the experiments of multiobjective evolutionary algorithms with chaos, the initial population is generated by chaos maps. For example, one of the individuals can be denoted by $x_s = \{x_s^1, x_s^2, \dots x_s^i, \dots x_s^n\}$, $s = 1, 2, \dots N$, $i = 1, 2, \dots N$. For the logistic map initialization, $x_s^{i+1} = 4x_s^i(1 - x_s^i)$.

3.2. Crossover Operator. Crossover operator is most important for evolutionary algorithms. Most of the offsprings are generated through the crossover operator. It has a great influence on the convergence speed. A good crossover operator may prevent premature convergence. Ergodicity of chaos helps search all the solutions, avoid solutions from falling into local optimum, and gain the global optimum.

There are many different crossover operators, such as simulated binary crossover operator [7] in NSGA-II algorithm and multiparent arithmetic crossover operator. Chaotic sequences substitute random parameters in the crossover operators. Chaotic sequences do not change the randomness of the parameter but display better randomness and therefore enhance the global performance of evolutionary algorithms.

In this paper, simulated binary crossover (SBX) operator is adopted in the experiment. According to SBX, two child individuals $x_{c1} = \{x_{c1}^1, \ldots, x_{c1}^i, \ldots, x_{c1}^n\}$ and $x_{c2} = \{x_{c2}^1, \ldots, x_{c2}^i, \ldots, x_{c2}^n\}$ are generated by a pair of parents $x_{p1} = \{x_{p1}^1, \ldots, x_{p1}^i, \ldots, x_{p1}^n\}$ and $x_{p2} = \{x_{p2}^1, \ldots, x_{p2}^i, \ldots, x_{p2}^n\}$ as follows:

$$\begin{aligned} x_{c1}^i &= \frac{1}{2}\left[(1-\beta)\,x_{p1}^i + (1+\beta)\,x_{p2}^i\right], \\ x_{c2}^i &= \frac{1}{2}\left[(1+\beta)\,x_{p1}^i + (1-\beta)\,x_{p2}^i\right], \end{aligned} \tag{1}$$

and β is generated in the following manner:

$$\beta = \begin{cases} (2u)^{1/(\eta_c+1)}, & \text{if } u \le 0.5, \\ \left(\dfrac{1}{2(1-u)}\right)^{1/(\eta_c+1)}, & \text{others}, \end{cases} \tag{2}$$

where u is a random number in the range $[0, 1]$. η_c is the distribution index for the crossover operator.

Since u is a random number, u can be generated by chaotic maps. For instance, if the chaotic map is a logistic map and in the ith iteration $u = u_i$, then in the $(i+1)$th iteration, $u_s = u_{i+1} = 4 \times u_i(1-u_i)$.

3.3. Mutation Operator. Mutation operator is indispensable in the process of evolutionary algorithms. This mechanism avoids solutions from falling into local optimum and guarantees more possibilities of obtaining global optimum. The properties of chaos, like randomness and sensitivity to initial conditions, contribute to preventing solutions from being trapped into local optimum.

Random parameters in mutation operators, for instance, polynomial variation, are replaced by chaotic sequences. For a solution x_s, the polynomial mutation is described as

$$x_s^* = x_s + \left(x_s^u - x_s^l\right) \times \delta_s, \tag{3}$$

where x_s^u and x_s^l are the upper and lower bounds of x_s, and

$$\delta_s = \begin{cases} (2u_s)^{1/(\eta_m+1)} - 1, & \text{if } u_s < 0.5, \\ 1 - (2 \times (1-u_s))^{1/(\eta_m+1)}, & \text{others}, \end{cases} \tag{4}$$

where u_s is a random number ranging from 0 to 1. η_m is the distribution index for the mutation operator.

The phase for mutation is that u_s is calculated by chaotic maps in iterations. For example, if the chaotic map is logistic map, and in the ith iteration $u_s = u_i$, then in the $(i+1)$th iteration, $u_s = u_{i+1} = 4 \times u_i(1-u_i)$.

As a representative of MOEAs, the framework of NSGA-II algorithm is adopted in the experiments. In order to eliminate the effect of NSGA-II algorithm, other two different mutation operators, that is, Gauss mutation and Cauchy mutation, are chosen to replace polynomial variation.

3.3.1. Gauss Mutation. If random variable X has the probability density function:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-((x-\mu)^2/2\sigma^2)}, \quad -\infty < x < +\infty, \tag{5}$$

then X obeys Gauss normal distribution with the parameters μ, σ; that is, $X \sim N(\mu, \sigma^2)$.

Gauss mutation means that the random numbers obeying gauss distribution substitute δ_s in polynomial mutation; that is, $\delta_s \sim N(\mu, \sigma^2)$.

3.3.2. Cauchy Mutation. The probability density function of Cauchy distribution concentrated near the origin. It is defined as

$$f(x) = \frac{1}{\pi} \frac{t}{t^2 + x^2}, \quad -\infty < x < +\infty, \; t > 0. \tag{6}$$

It is similar to Gauss probability density function. The difference is that the value of Cauchy distribution is lower than the value of Gauss distribution in the vertical direction, and Cauchy distribution is closer to the horizontal axis in the horizontal direction. Cauchy mutation means that the random numbers obeying Cauchy distribution substitute δ_s in polynomial mutation.

4. Chaotic Maps

Chaotic maps generate chaotic sequences in the process of evolutionary algorithms. Ten chaotic maps including both one-dimensional maps and two-dimensional maps are introduced in this section. They will be used to improve the performance of MOP algorithms.

4.1. One-Dimensional Maps

(1) Circle Map. Circle map is a member of a family of dynamical systems on the circle first defined by Andrey Kolmogorov. He proposed this family as a simplified model for driven mechanical rotors specifically, a free-spinning wheel weakly coupled by a spring to a motor. The circle map equations also describe a simplified model of the phase-locked loop in electronics. The circle map [24] is given by iterating the map:

$$x_{k+1} = \left\{x_k + b - \left(\frac{a}{2\pi}\right)\sin(2\pi x_k)\right\} \bmod (1), \tag{7}$$

with $a = 0.5$ and $b = 0.2$; it generates chaotic sequence in $(0, 1)$.

(2) Cubic Map. Cubic map is one of the most commonly used maps in generating chaotic sequences in various applications. This map is formally defined by the following equation [25]:

$$x_{k+1} = \rho x_k \left(1 - x_k^2\right), \quad x_k \in (0, 1). \tag{8}$$

Cubic map generates chaotic sequences in $(0, 1)$ with $\rho = 2.59$.

(3) Gauss Map. Gauss map is also one of the well-known and commonly employed maps in generating chaotic sequences [26] as follows:

$$x_{k+1} = \begin{cases} 0, & x_k = 0, \\ \dfrac{1}{x_k} \bmod (1), & \text{otherwise.} \end{cases} \tag{9}$$

This map also generates chaotic sequences in $(0, 1)$.

(4) ICMIC Map. The iterative chaotic map with infinite collapses (ICMIC) [27] is defined by the following equation:

$$x_{k+1} = \sin\left(\frac{a}{x_k}\right), \quad a \in (0, \infty), \ x_k \in (-1, 1). \tag{10}$$

The parameter "a" is an adjustable parameter. This paper chooses $a = 2$. Because the range of x_k is not $(0, 1)$, the chaotic sequences need to be transformed to change the range.

(5) Logistic Map. As a well-known chaotic map, logistic map is one of the simplest maps and was introduced by May in 2004 [28]. It is often cited as an example of how complex behavior can arise from a very simple nonlinear dynamical equation. Logistic map generates chaotic sequences in $(0, 1)$. This map is formally defined by the following equation:

$$x_{k+1} = a x_k (1 - x_k). \tag{11}$$

Parameter a is set to 4 in the simulation.

(6) Sinusoidal Iterator. The sinusoidal iterator [29] is formally defined by the following equation:

$$x_{k+1} = a x_k^2 \sin(\pi x_k), \quad x_k \in (0, 1). \tag{12}$$

In this paper the simplified equation is used in the following iteration:

$$x_{k+1} = \sin(\pi x_k), \quad x_k \in (0, 1). \tag{13}$$

(7) Tent Map. Tent chaotic map is very similar to the logistic map, which displays specific chaotic effects [30]. This map is defined by the following equation:

$$x_{k+1} = \begin{cases} 2x_k, & x_k < 0.5, \\ 2(1 - x_k), & x_k \geq 0.5, \end{cases} \tag{14}$$

where x_k is ranging from 0 to 1.

Tent map generates chaotic sequences in $(0, 1)$.

4.2. Two-Dimensional Maps

(1) Baker's Map. The Baker map [31] is described by the following formulas:

$$B(x, y) = \begin{cases} (2x, 2y), & \text{for } 0 \leq x < 0.5, \\ \left(2 - 2x, 1 - \dfrac{y}{2}\right), & \text{for } 0.5 \leq x < 1. \end{cases} \tag{15}$$

In the following simulations, one dimension of Baker's map, which is similar to tent map, is adopted. The equation is defined by

$$x_{k+1} = \begin{cases} 2x_k, & \text{for } 0 \leq x_k < 0.5, \\ 2 - 2x_k, & \text{for } 0.5 \leq x_k < 1. \end{cases} \tag{16}$$

This map generates chaotic sequences in $(0, 1)$.

(2) Arnold's Cat Map. Arnold's cat map is named after Vladimir Arnold, who demonstrated its effects in the 1960s using an image of a cat. It is represented by [32]

$$\begin{aligned} x_{k+1} &= x_k + y_k \bmod (1), \\ y_{k+1} &= x_k + 2y_k \bmod (1). \end{aligned} \tag{17}$$

It is obvious that the sequences $x_k \in (0, 1)$ and $y_k \in (0, 1)$.

(3) Zaslavskii Map. Zaslavskii map [33] is an interesting dynamic system with chaotic behavior. The discretized equation is given by

$$\begin{aligned} x_{k+1} &= (x_k + v + a y_{k+1}) \bmod (1), \\ y_{k+1} &= \cos(2\pi x_k) + e^{-r} y_k. \end{aligned} \tag{18}$$

The Zaslavskii map shows a strange attractor with the largest Lyapunov exponent for $v = 400$, $r = 3$, and $a = 12.6695$. In this case, it can be calculated that $y_{k+1} \in [-1.0512, 1.0512]$. Only one dimension is chosen in the following simulation. Since the scale of y_{k+1} is not $[0, 1]$, the chaotic sequences generated need scale transformation.

5. Chaotic Properties of Sequences Generated by Scale Transformation

As mentioned in the previous sections, the scale of sequences generated by chaotic maps is not always fit for the problems to be solved. Some sequences have to change their scale, and some sequences are generated by one dimension of a two-dimension chaotic map. Hence, it is necessary to demonstrate the chaotic properties of sequences after these changes.

Detecting the presence of chaos in a dynamical system is usually solved by measuring the largest Lyapunov exponent which describes quantitatively the speed of index divergence or convergence between the adjacent phase space orbits. A positive largest Lyapunov exponent indicates chaos. Since the chaotic sequences adopted in this paper are discrete, the Lyapunov exponent of discrete series can be calculated by *small data sets arithmetic* [34]. This method makes full use of all the data, obtains higher accuracy, and has stronger robustness for the amount of data, the embedding dimension, and the time delay.

5.1. Small Data Sets Arithmetic. The reconstructed trajectory, X, can be expressed as a matrix where each row is a phase-space vector; that is,

$$X = (X_1, X_2, \ldots, X_M)^T, \tag{19}$$

where X_i is the state of the system at discrete time i. For an N-point time series, $\{x_1, x_2, \ldots, x_N\}$, each X_i is given by

$$X_i = \left(x_i, x_{i+J}, \ldots, x_{i+(m-1)J}\right), \quad (20)$$

where J is the lag or reconstruction delay, and m is the embedding dimension. Thus, X is an $M \times m$ matrix, and the constants m, M, J, and N are related as

$$M = N - (m-1)J. \quad (21)$$

After reconstructing the dynamics, the algorithm locates the nearest neighbor of each point on the trajectory. The nearest neighbor $X_{\hat{j}}$, where $\hat{j} \in \{1, 2, \ldots M\}$, is found by searching for the point that minimizes the distance to the particular reference point X_j. This is expressed as

$$d_j(0) = \min_{X_{\hat{j}}} \left\| X_j - X_{\hat{j}} \right\|, \quad (22)$$

where $d_j(0)$ is the initial distance from the jth point to its nearest neighbor, and $\|\|$ denotes the Euclidean norm. We impose an additional constraint that the nearest neighbors have a temporal separation greater than the mean period of the time series:

$$\left| j - \hat{j} \right| > p, \quad (23)$$

where p is the mean period of time series. p can be estimated by the reciprocal of the mean frequency of the power spectrum. This allows us to consider each pair of neighbors as nearby initial conditions for different trajectories. The largest Lyapunov exponent is estimated as the mean rate of separation of the nearest neighbors.

For each reference point X_j, $d_j(i)$ is the distance between the jth pair of nearest neighbors after i discrete time:

$$d_j(i) = \left\| X_{j+i} - X_{\hat{j}+i} \right\|, \qquad i = 1, 2, \ldots, \min\left(M - j, M - \hat{j}\right). \quad (24)$$

Assume that reference point X_j and its nearest neighbor $X_{\hat{j}}$ have index divergence rate λ_1; then

$$d_j(i) = C_j e^{\lambda_1 (i \cdot \Delta t)}, \qquad C_j = d_j(0), \quad (25)$$

where C_j is the initial separation. By taking the logarithm of both sides of (25) we get

$$\ln d_j(i) \approx \ln C_j + \lambda_1 (i \cdot \Delta t). \quad (26)$$

Equation (26) represents a set of approximately parallel lines (for $j = 1, 2, \ldots, M$), each with a slope s roughly proportional to λ_1. The largest Lyapunov exponent is easily and accurately calculated using a least square fit to the "average" line defined by

$$y(i) = \frac{1}{\Delta t} \left\langle \ln d_j(i) \right\rangle, \quad (27)$$

where $\langle\ \rangle$ denotes the average over all values of j. So

$$y(i) = \frac{1}{q\Delta t} \sum_{j=1}^{q} \ln d_j(i), \quad (28)$$

where q is the number of $d_j(i)$ with $d_j(i) \neq 0$.

Choose a linear area of the curve $y(i) \sim i$, and apply the least square method to get the regression straight line. Then the slope of the regression straight line is the largest Lyapunov exponent λ_1.

5.2. The Lyapunov Exponent of Sequences. In the calculation process, the embedding dimension m is calculated through the method of false nearest neighbors (FNN). For the time delay J, a good approximation of J is equal to the number lagging where the autocorrelation function drops to $1 - 1/e$ of its initial value.

Since different test problems have different ranges, chaotic sequences need to be changed to different scales. Two kinds of sequences used in experiments need to be investigated: sequences with scales changed and sequences generated by one dimension of a two-dimension chaotic map.

5.2.1. Sequences with Scales Changed. Since the sequence x_1 to x_{100} generated by ICMIC is not in (0, 1), the new sequence y_1 to y_{100} has to be generated by the following function:

$$y_i = \frac{1}{2}\left(x_i + 1\right), \quad i \in [1, 100]. \quad (29)$$

The sequence y_1 to y_{100} is in the range of (0, 1). The Lyapunov exponent of the new sequence is calculated through *small data sets arithmetic*. The average Lyapunov exponent of 10 runs is 0.0744. Since it is a positive number, the new sequence y_1 to y_{100} conforms to the chaotic nature.

5.2.2. Sequences Generated by One Dimension of a Two-Dimension Chaotic Map. For the Zaslavskii map, one dimension y_k is chosen in the following simulation. The sequence y_1 to y_{100} is generated by 100 iterations through Zaslavskii map. The new sequence z_1 to z_{100} is generated by the following function:

$$z_i = \frac{\left(y_i + 1.0513\right)}{2.1026}, \qquad i \in [1, 100]. \quad (30)$$

Then the sequence z_1 to z_{100} is in (0, 1). By a similar processing with ICMIC, the average Lyapunov exponent is 0.00194. Then the new sequence z_1 to z_{100} conforms to the chaotic nature.

6. Test Problem and Performance Measures

6.1. Test Problems. Two-objective optimization problems are chosen to test and measure the performance improvement of the evolutionary algorithms using chaotic maps in three phases. We use well-defined benchmark functions as objective functions. Their properties are shown in Table 1.

6.2. Performance Measures. Two criteria are used to evaluate the performance of multiobjective optimization: (1) convergence to the Pareto-optimal set and (2) maintenance of diversity in solutions of the Pareto-optimal set [7]. Two metrics are adopted to evaluate the effects of the combinations of phases and chaotic maps.

The first metric γ measures the extent of convergence to a known set of Pareto-optimal solutions. It is defined as

$$\gamma = \frac{1}{N} \sum_{i=1}^{N} d_i, \quad (31)$$

Table 1: Test problems.

Problem	n	Variable bounds	Objective functions	Optimal solutions
ZDT1	30	$[0,1]$	$f_1(x)=x_1$ $f_2(x)=g(x)[1-\sqrt{x_1/g(x)}]$ $g(x)=1+\left(9\left(\sum_{i=2}^{n}x_i\right)/(n-1)\right)$	$x_1\in[0,1]$ $x_i=0,$ $i=2,\ldots,n$
ZDT2	30	$[0,1]$	$f_1(x)=x_1$ $f_2(x)=g(x)[1-(x_1/g(x))^2]$ $g(x)=1+\left(9\left(\sum_{i=2}^{n}x_i\right)/(n-1)\right)$	$x_1\in[0,1]$ $x_i=0,$ $i=2,\ldots,n$
ZDT3	30	$[0,1]$	$f_1(x)=x_1$ $f_2(x)=g(x)[1-\sqrt{x_1/g(x)}-(x_1/g(x))\sin(10\pi x_1)]$ $g(x)=1+\left(9\left(\sum_{i=2}^{n}x_i\right)/(n-1)\right)$	$x_1\in[0,1]$ $x_i=0,$ $i=2,\ldots,n$
ZDT4	10	$x_1\in[0,1]$ $x_i\in[-5,5],$ $i=2,\ldots,n$	$f_1(x)=x_1$ $f_2(x)=g(x)[1-\sqrt{x_1/g(x)}]$ $g(x)=1+\left(10(n-1)+\sum_{i=2}^{n}[x_i^2-10\cos(4\pi x_i)]\right)$	$x_1\in[0,1]$ $x_i=0,$ $i=2,\ldots,n$
ZDT6	10	$[0,1]$	$f_1(x)=1-\exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x)=g(x)[1-(f_1(x)/g(x))^2]$ $g(x)=1+\left(9[(\sum_{i=2}^{n}x_i)/(n-1)]^{0.25}\right)$	$x_1\in[0,1]$ $x_i=0,$ $i=2,\ldots,n$

where d_i is the minimum Euclidean distance of every obtained solution to the Pareto-optimal front. The smaller the value of this metric is, the nearer the convergence toward Pareto-front is.

The other metric Δ measures the extent of spread achieved among the obtained solutions. The metric Δ is defined by

$$\Delta=\frac{d_f+d_l+\sum_{i=1}^{N-1}\left|d_i-\overline{d}\right|}{d_f+d_l+(N-1)\overline{d}}. \quad (32)$$

The parameter d_i is the Euclidean distance between consecutive solutions in the obtained nondominated set of solutions. The parameters d_l and d_f are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained nondominated set. The parameter $\overline{d}$ is the average of all distances d_i, $i=1,2,\ldots,N-1$, assuming that there are N solutions on the best nondominated front.

7. Experiments and Results

To explore the relationship of phases and chaotic maps to solve MOPs, NSGA-II algorithm is chosen as the main framework. The ten chaotic maps mentioned in Section 4 are embedded in three different phases in the original NSGA-II algorithm. Each time only one parameter is modified. For example, if initial population is generated by chaotic map, the crossover and mutation operator are not changed. Similarly, if crossover operator is modified by a chaotic map, the initial population and mutation operator are not changed. The solutions, generated by the chaos embedded NSGA-II algorithm, are evaluated by two metrics: γ and Δ. For reader's convenience, the new algorithms with different combinations of chaotic maps and phases are named as "cns_[chaotic map]_[phase]," and the results of different algorithms on test problems are named as "cns_[chaotic map]_[phase]_[test problem]." In addition, "i" represents the phase for initial population, "c" represents the phase for crossover operator, and "m" represents the phase for mutation operator. For example, the results through modified initial population by logistic map solving ZDT1 problem are named as "cns_logistic_i_zdt1."

Each combination of one chaotic map and one phase needs one experiment. In this research, 10 chaotic maps with 3 different phases based on 2 metrics solving 5 test problems need 150 basic experiments and obtain 300 results. Each experiment obtains a Pareto front. The values of convergence metric γ and the diversity metric Δ are also calculated.

In order to compare with the results of original NSGA-II algorithm, we focused on the difference of the γ and Δ values of the original NSGA-II algorithm and the new algorithm. For example, the γ of results of "cns_sinusoidal_i_zdt1" is named as "cns_sinusoidal_i_zdt1_gama," and the γ of results of NSGA-II solving ZDT1 problem is named as "ns_zdt1_gama." Then the difference is named as "ns_zdt1_gama—cns_sinusoidal_i_zdt1_gama." When the processes of algorithms get to convergence, the difference is very small. The properties of convergence and diversity in the process of iterations need to be taken into account, so the γ values of each generation in the iterations are recorded and the differences of γ of each generation are obtained. This process also applies to Δ.

Some main parameters in the process of NSGA-II algorithm are introduced in the following paragraphs. Then the results of experiments are shown and analyzed.

7.1. The Main Parameters.

The main parameters in the process of NSGA-II algorithm are presented in this section. Choosing an appropriate representation of a chromosome is very important for solving problems. Real numbers are chosen to represent the genes. One chromosome represents one individual. The initial population has 100 individuals, and each chromosome has a certain number of genes which are represented by a real number. Each individual of the initial population is generated randomly with the range

Table 2: Parameters in the process of algorithms.

	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
n_{iter}			250		
n_{pop}			100		
n_{var}	30	30	30	10	10
p_c			0.9		
p_m	1/30	1/30	1/30	1/10	1/10

based on the test problems. The iteration will not terminate until the number of iterations gets to 250. For the process of NSGA-II algorithm, a parent population is selected by tournament selection depending on the nondominated rank and the crowed-comparison operator. Then the new population is generated by crossover and mutation operators. The crossover operation is executed with the probability of p_c = 0.9. The probability of mutation p_m is equal to the reciprocal of n_{var}, which is the dimension number of a chromosome; that is, $p_m = 1/n_{\text{var}}$.

Those parameters are summarized in Table 2. In the table, n_{iter} is the number of iterations, n_{pop} is the scale of the population, n_{var} is the number of dimensions of a chromosome, and p_c and p_m are the probabilities of crossover and mutation operations.

7.2. Convergence Performance. It is known that the γ difference is used to evaluate the performance of the chaotic maps in different phases in multiobjective evolutionary algorithms. An example is chosen for further explanation in detail. As in Figure 1, the graph shows the results of solving ZDT1 problems with Baker's map in crossover operator in NSGA-II. The differences of γ between the experiment "cns_baker_c_zdt1" and the experiment "ns_zdt1" in the 250 iterations are given. As seen from the figure, the black line is above the red line which represents 0, so the new algorithm "cns_bakers_c" is better than NSGA-II algorithm in solving ZDT1 problem with regard to the convergence metric.

The γ results of all the experiments are given similar to Figure 1. Since it is difficult to show so many graphs in this paper, the results of three typical problems are chosen, that is, ZDT1, which is a simple convex problem, ZDT3, whose Pareto front is piecewise, and ZDT4, which has local optima. The graphs in Figures 2, 3, and 4 provide a comparison of the performance of solving different MOPs with chaotic maps in initial population. ZDT4 is chosen to show the performance of chaotic maps in different phases on solving the same MOP, as shown in Figures 4, 5, and 6. Each subgraph is labeled with the name of the chaotic map used.

In order to quantify the effect of chaotic maps and phases with regard to the metric γ, the average of γ difference in 250 generations is calculated to represent the effect of the new algorithms.

Since the order of magnitude of γ is not the same, the comparison of these γ values is not convenient. The normalized values are obtained by dividing the γ values by the maximum of the absolute values of the γ based on one test problem. The results of normalization are shown in Table 3.

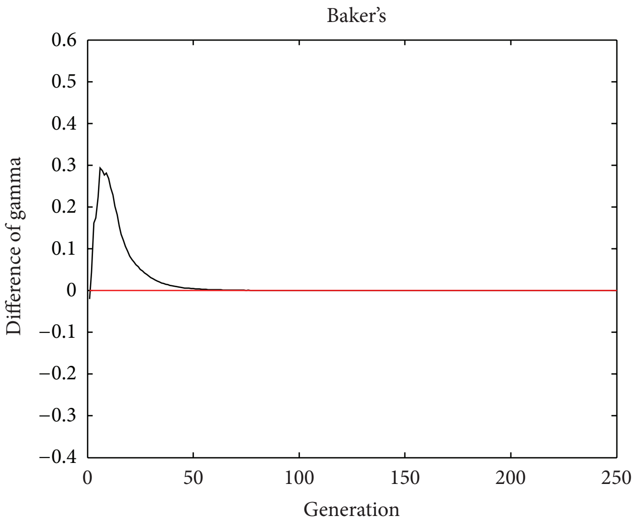

Figure 1: Performance of Baker's maps in crossover operator in solving ZDT1.

Table 3 can be presented in a more intuitive way. If $\gamma \geq 0.3$, the numerical value of γ is replaced by "++." Similarly, "+" represents $0.1 \leq \gamma < 0.3$, "0" represents $-0.1 \leq \gamma < 0.1$, "−" represents $-0.3 \leq \gamma < -0.1$, and "− −" represents $\gamma < -0.3$. Therefore "++" means that the effect of the new algorithm with chaotic maps is much better, whereas "− −" is much worse. Table 4 shows the results.

As shown in Table 4, most of the combinations of chaotic maps and phases have a positive effect on improving the performance of NSGA-II algorithm. The effect of some chaotic maps is very good, especially in some particular phases. For example, Baker's map in crossover operator, Gauss map in crossover operator and initial population, ICMIC map in initial population, sinusoidal map in initial population, tent map in crossover operation, and Zaslavskii map in initial population have very good effect.

Since ZDT4 problem has 21^9 or 7.94×10^{11} different local Pareto-optimal fronts in the search space, the solutions easily get entrapped into local optimum. As seen from Table 4, chaotic maps used for crossover and mutation operator have significant improvement on evolutionary algorithms solving ZDT4 problem; especially cat map has the best performance in ten maps. Circle map and cubic map have less contribution in solving those MOPs. The distribution of cat map is relatively uniform. It is probably the reason for the good performance in solving problems with local optima.

The original NSGA-II algorithm is not good at solving ZDT3 and ZDT6 problems, because Pareto-optimal front of ZDT3 is disconnected and solutions of ZDT6 are nonuniformly spaced. However, it can be seen in Table 4 that chaotic maps can improve NSGA-II especially in crossover operation and initial population in solving ZDT3 and ZDT6 problem.

In order to eliminate the special effect of the NSGA-II algorithm, the polynomial mutation operator in NSGA-II is changed by the Gauss mutation and Cauchy mutation operators. Four typical chaotic maps, which include two chaotic maps with best performance and two chaotic maps

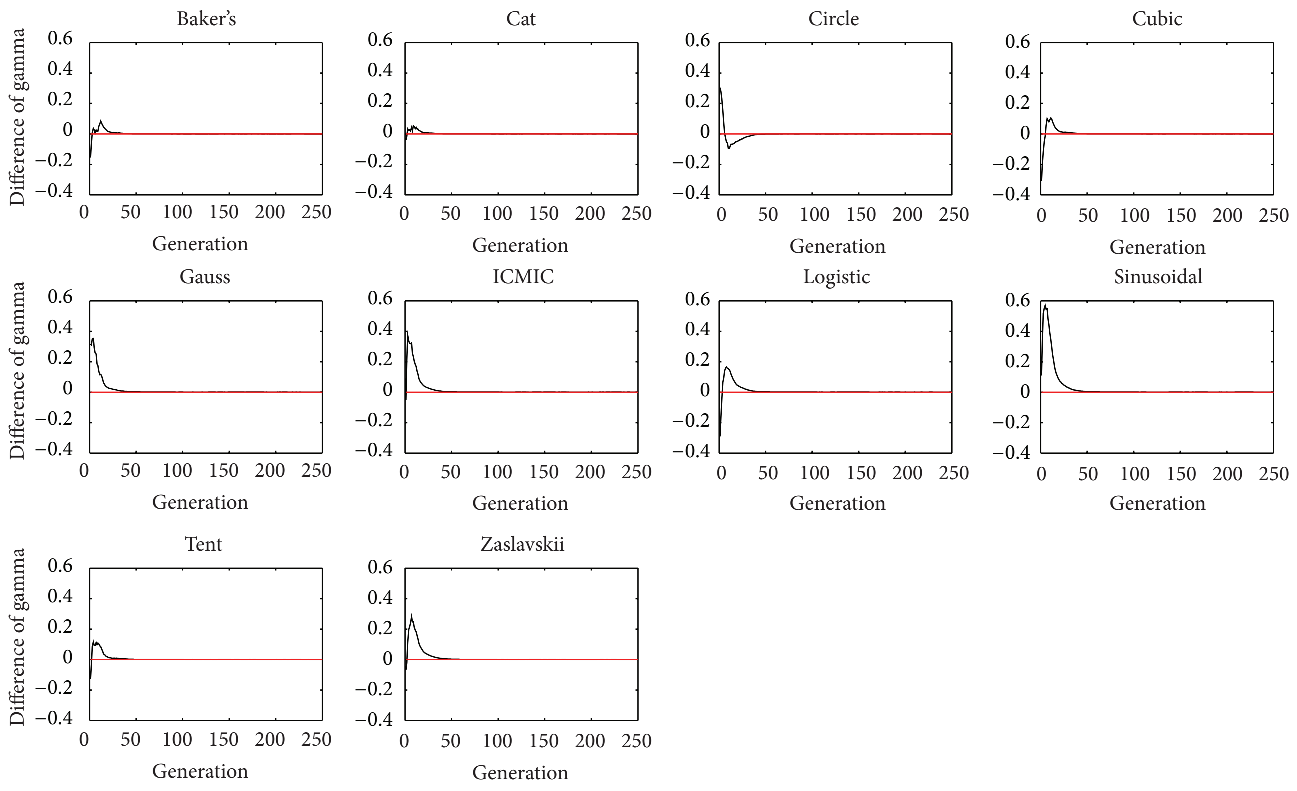

Figure 2: Performance of chaotic maps in initial population in solving ZDT1.

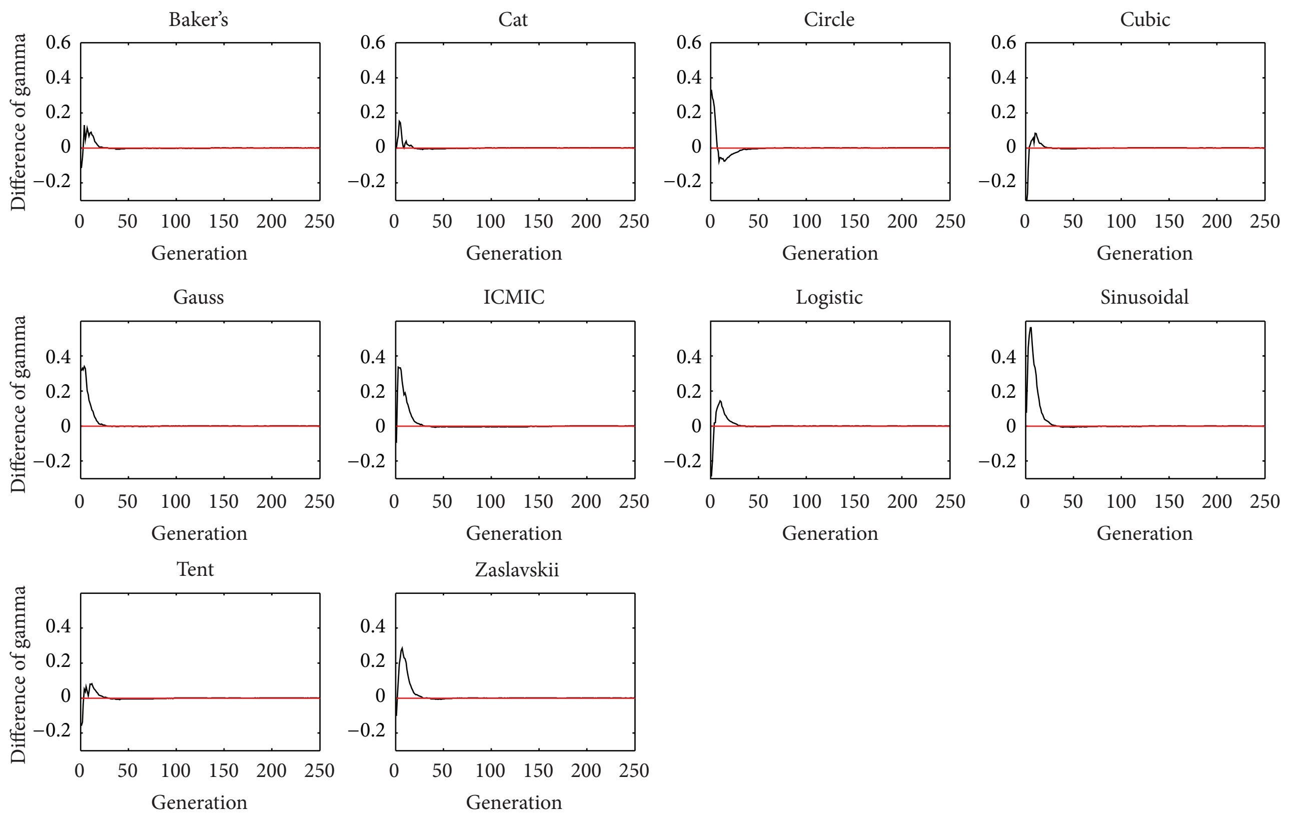

Figure 3: Performance of chaotic maps in initial population in solving ZDT3.

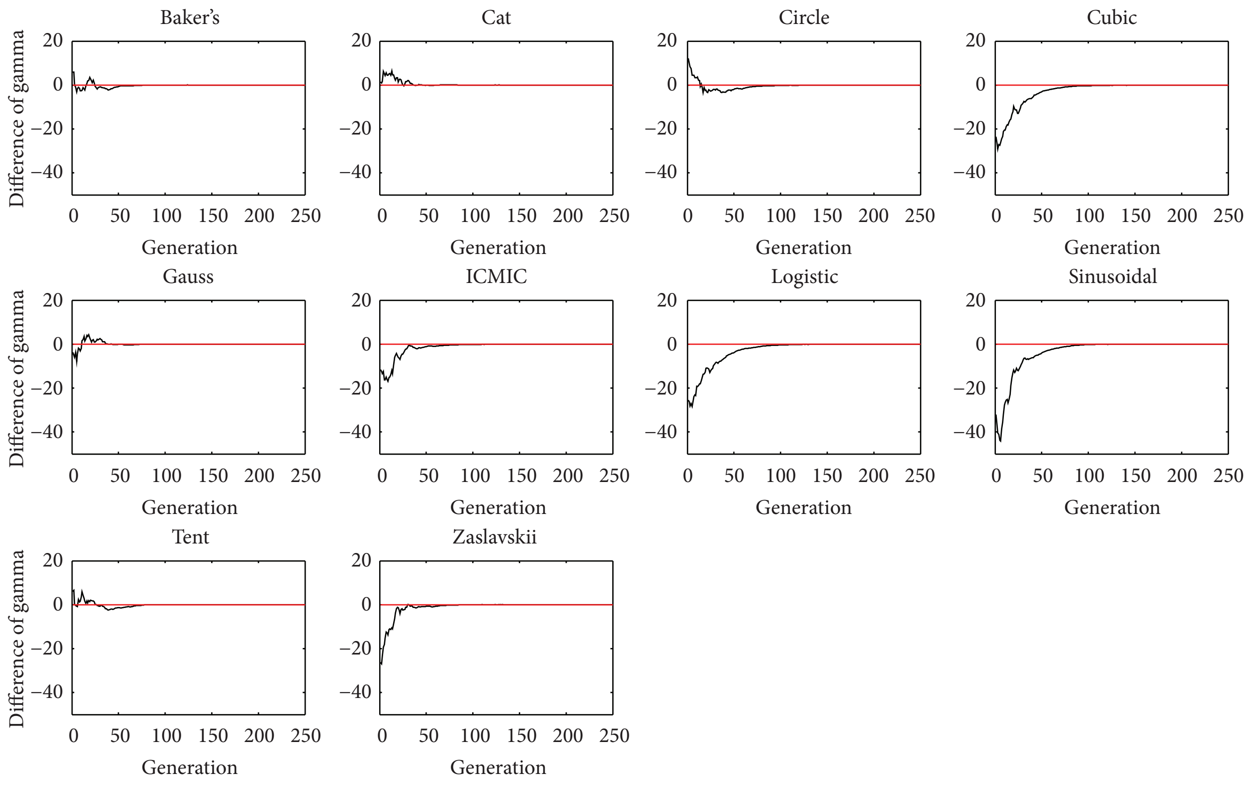

Figure 4: Performance of chaotic maps in initial population in solving ZDT4.

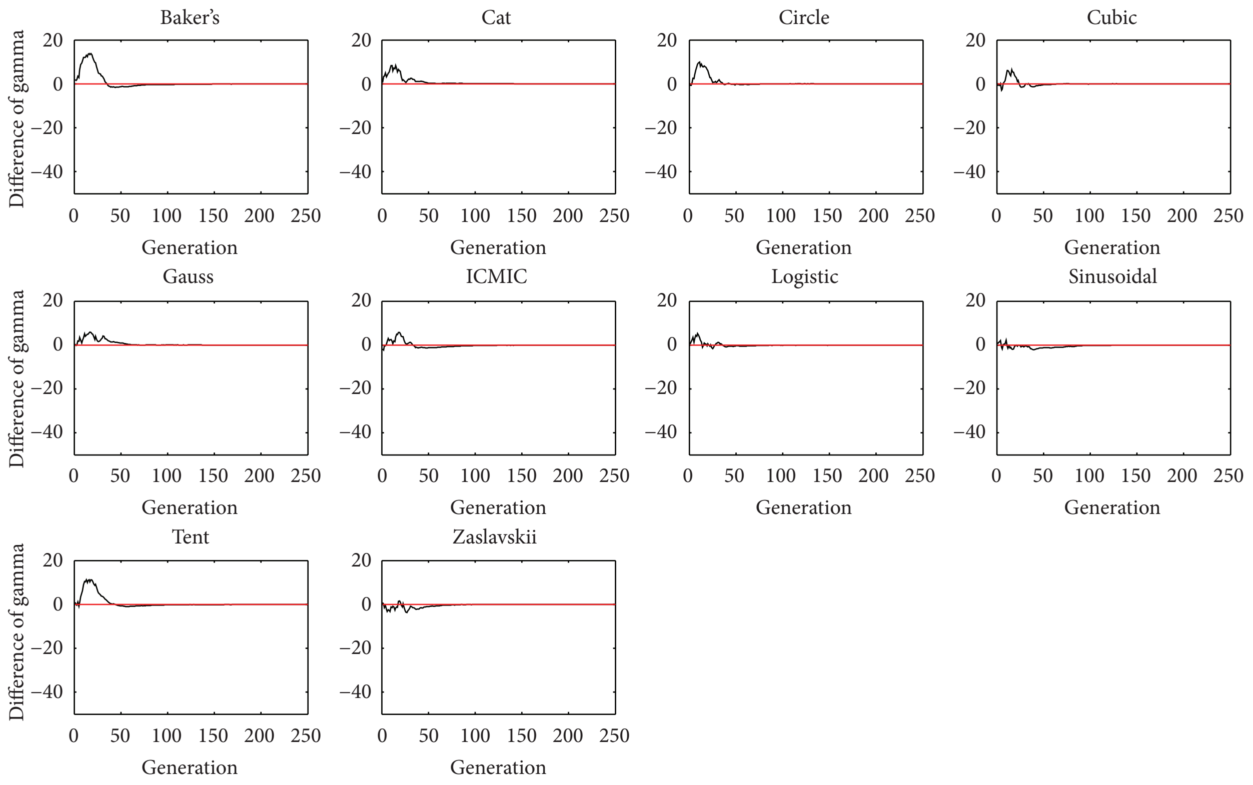

Figure 5: Performance of chaotic maps in crossover operator in solving ZDT4.

Table 3: The normalized results of γ.

	ZDT1			ZDT2			ZDT3			ZDT4			ZDT6		
	c	i	m	c	i	m	c	i	m	c	i	m	c	i	m
Baker	0.617	0.080	0.049	0.682	0.149	0.094	0.668	0.109	0.003	0.220	−0.040	0.167	0.567	0.105	−0.090
Cat	−0.060	0.076	0.101	0.013	0.084	0.024	−0.007	0.090	0.174	0.191	0.109	0.158	−0.028	0.022	−0.096
Circle	−0.142	−0.040	−0.016	0.013	−0.121	−0.007	−0.153	0.028	−0.016	0.151	−0.069	0.098	−0.176	−0.072	−0.002
Cubic	−0.544	0.064	−0.025	−0.626	−0.041	0.006	−0.334	−0.049	0.077	0.032	−0.781	0.071	−0.288	0.040	0.008
Gauss	0.307	0.513	0.089	0.306	0.507	0.070	0.454	0.585	0.037	0.159	0.005	0.191	0.114	−0.010	0.132
ICMIC	−0.415	0.558	0.132	−0.280	0.609	0.144	−0.295	0.510	0.004	0.003	−0.380	0.088	−0.162	0.252	0.163
Logistic	0.070	0.242	0.189	0.072	0.204	−0.031	0.012	0.158	0.127	0.017	−0.819	0.183	0.152	0.129	0.132
Sinusoidal	0.077	1	0.616	0.121	1	0.742	0.169	1	0.734	−0.091	−1	−0.138	0.148	0.688	1
Tent	0.655	0.177	0.062	0.704	0.103	−0.005	0.731	0.043	−0.088	0.190	−0.008	−0.058	0.569	0.124	−0.003
Zaslavskii	−0.051	0.462	0.032	−0.150	0.518	0.064	−0.086	0.499	0.110	−0.103	−0.339	0.108	−0.060	0.174	0.183

Table 4: The visualized results of γ.

	ZDT1			ZDT2			ZDT3			ZDT4			ZDT6		
	c	i	m	c	i	m	c	i	m	c	i	m	c	i	m
Baker	++	0	0	++	+	0	++	+	0	+	0	+	++	+	0
Cat	0	0	+	0	0	0	0	0	+	+	+	+	0	0	0
Circle	−	0	0	0	−	0	−	0	0	+	0	0	−	0	0
Cubic	—	0	0	—	0	0	—	0	0	0	—	0	−	0	0
Gauss	++	++	0	++	++	0	++	++	0	+	0	+	+	0	+
ICMIC	—	++	+	−	++	+	−	++	0	0	—	0	−	+	+
Logistic	0	+	+	0	+	0	0	+	+	0	—	+	+	+	+
Sinusoidal	0	++	++	+	++	++	+	++	++	0	—	−	+	++	++
Tent	++	+	0	++	+	0	++	0	0	+	0	0	++	+	0
Zaslavskii	0	++	0	−	++	0	0	++	+	−	—	+	0	+	+

Table 5: Results of Gauss mutation.

	ZDT1		ZDT2		ZDT3		ZDT4		ZDT6	
	c	i	c	i	c	i	c	i	c	i
Circle	−0.4886	−0.0170	1.9394	−0.3130	0.0971	−0.4370	185.6683	75.0688	2.8693	−0.5529
Cubic	−2.4674	−0.0589	−6.2676	−0.0630	−2.5351	−1.2051	49.1193	−448.985	−2.3071	5.7609
Sinusoidal	1.0277	6.0982	0.4343	9.4173	−0.0396	4.5810	−106.768	−525.536	5.84566	28.5351
Tent	3.5035	0.6784	5.9502	1.1086	1.8616	0.0623	−272.428	101.4576	15.0061	11.2005

Table 6: Results of Cauchy mutation.

	ZDT1		ZDT2		ZDT3		ZDT4		ZDT6	
	c	i	c	i	c	i	c	i	c	i
Circle	1.3978	0.5012	1.3997	0.7723	0.7187	0.4043	135.4511	−136.938	0.1105	−2.6454
Cubic	−2.6201	−0.4932	−4.2148	−0.5317	−2.4831	−0.7487	−77.1621	−456.939	−5.7560	5.4533
Sinusoidal	0.1470	4.6977	0.8327	8.2820	0.3179	4.5376	−202.995	−468.243	6.5462	23.8545
Tent	2.7613	0.3380	5.2687	0.9699	2.6813	−0.1916	−283.172	57.9606	12.7523	5.2806

with worst performance, are chosen to be used in the experiments. These chaotic maps are circle map, cubic map, sinusoidal map, and tent map. The values of γ differences are shown in Tables 5 and 6. As seen from Tables 5 and 6, the performance of sinusoidal map and tent map is better than the performance of circle map and cubic map. Sinusoidal map in initial population is better than that in crossover operation, and tent map in crossover operation is better than that in initial population. This means the rules of combinations of chaotic maps and phases in solving MOPs are almost the same as in the previous observations. So the rules based on the framework of NSGA-II algorithm are applicable to other MOEAs.

In general, Baker's map with a phase for crossover operator, sinusoidal map with phases for initial population and mutation operator, and tent map with a phase for crossover operator could be the best choice for improving evolutionary algorithms for MOPs without local optimum. For problems

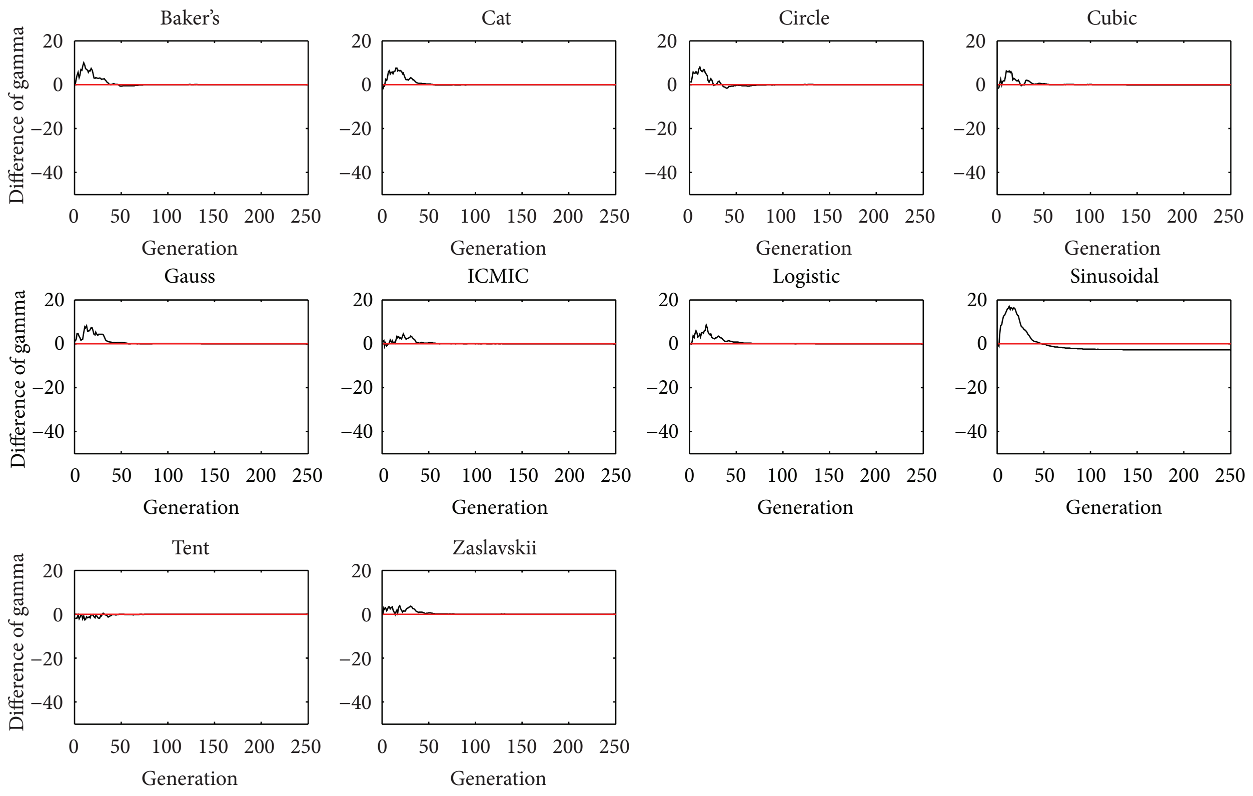

Figure 6: Performance of chaotic maps in mutation operator in solving ZDT4.

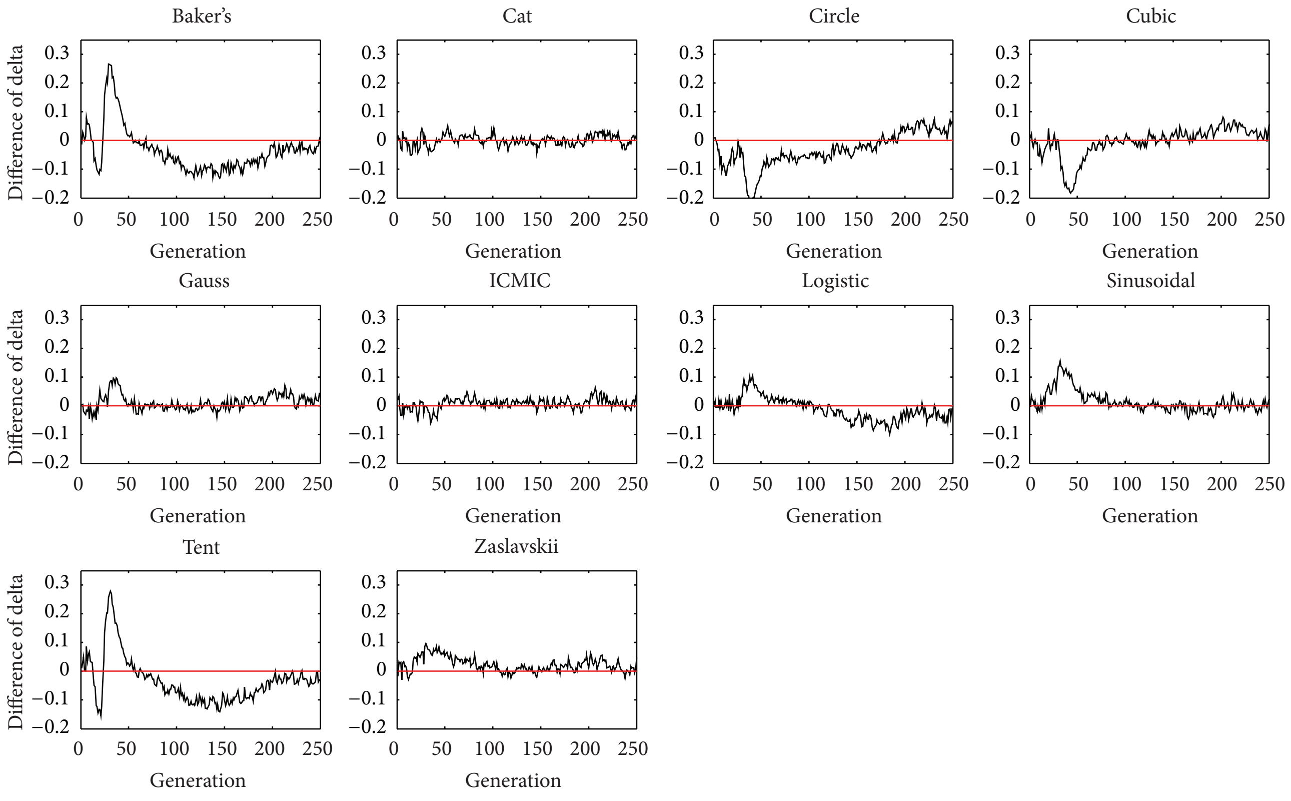

Figure 7: Performance of chaotic maps in crossover operator in solving ZDT1 with metric Δ.

Table 7: The average results of Δ.

	ZDT1			ZDT2			ZDT3			ZDT4			ZDT6		
	c	i	m	c	i	m	c	i	m	c	i	m	c	i	m
Baker	−0.0337	−0.0026	0.0015	0.0226	0.0310	0.0024	−0.0096	0.0069	0.0006	−0.0603	0.0343	−0.0513	0.0125	0.0024	−0.0015
Cat	0.0005	0.0017	0.0042	0.0234	−0.1021	−0.0145	−0.0019	−0.0052	0.0006	0.0277	−0.0090	−0.0148	−0.0001	−0.0002	−0.0068
Circle	−0.0325	−0.0035	−0.0064	−0.2019	−0.0632	−0.1410	−0.0052	−0.0065	0.0010	−0.0255	−0.0259	−0.0136	0.0031	−0.0020	0.0064
Cubic	−0.0047	0.0121	−0.0022	−0.0439	−0.0148	0.0361	0.0000	−0.0006	0.0031	−0.0052	−0.0279	0.0238	−0.0170	0.0054	−0.0017
Gauss	0.0121	0.0075	−0.0180	−0.1075	0.0329	−0.0711	−0.0024	−0.0032	0.0019	0.0330	−0.0214	0.0155	0.0180	0.0003	0.0064
ICMIC	0.0081	0.0092	0.0002	−0.0734	−0.0850	−0.0917	−0.0036	0.0008	−0.0054	−0.0356	−0.0497	0.0238	0.0123	0.0001	0.0085
Logistic	−0.0118	0.0076	0.0075	−0.1935	0.0239	0.0509	−0.0084	−0.0014	0.0080	0.0275	−0.0100	−0.0232	−0.0069	0.0020	0.0077
Sinusoidal	0.0149	0.0069	−0.0564	−0.1750	−0.0067	−0.2902	−0.0014	0.0029	−0.0286	−0.0390	−0.0164	−0.1945	0.0129	0.0088	−0.0488
Tent	−0.0350	0.0093	0.0016	0.0215	−0.0173	−0.0261	−0.0059	−0.0168	0.0071	−0.0761	−0.0139	0.0075	0.0125	0.0028	−0.0002
Zaslavskii	0.02307	0.0056	−0.0014	−0.0292	−0.0189	−0.0013	0.0099	0.0040	−0.0037	0.0201	−0.0056	0.0090	0.0038	0.0027	0.0084

Table 8: Statistical data for combinations of chaotic maps and phases on different problems.

Threshold	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
0	18	9	15	10	20
0.001	16	9	9	10	18
0.002	13	9	7	10	18
0.003	13	8	6	10	14
0.004	13	8	5	10	12
0.005	12	8	4	10	12
0.006	11	8	4	10	11

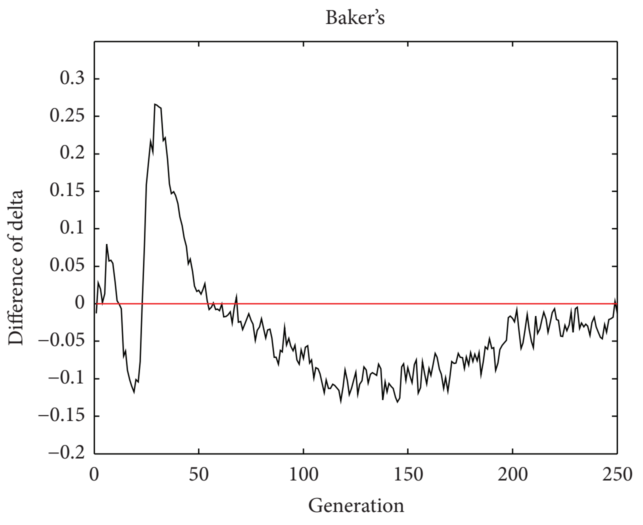

Figure 8: Performance of Baker's maps in crossover operator in solving ZDT1 with metric Δ.

with local optimum, cat map has good performance on improving evolutionary algorithms.

7.3. Diversity Performance. Similar to the convergence metric γ, the differences of the diversity metric Δ between the results of the new algorithm with chaotic maps and the original NSGA-II algorithm are used to measure the performance. Figure 7 shows the performance of chaotic maps in crossover operator in solving ZDT1 with metric Δ. Each subgraph shows the effect of one chaotic map. To be seen more clearly, the first subgraph in Figure 7 is shown in Figure 8.

As seen from Figures 7 and 8, the Δ difference is not stable in 250 generations. The average values of Δ difference in 250 generations are calculated to represent the effect of the new algorithms. For brevity, the rest of results are not shown in graphs but in Table 7.

As seen from Table 7, the Δ values have little difference. We count the number of combinations of chaotic maps and phases for solving one problem in different threshold values. For example, there are 18 combinations whose values of Δ are greater than zero. Based on the number of the combinations of chaotic maps and phases in different threshold values, the values of diversity metric Δ are summarized in Table 8.

In Table 8 the rank of the number of Δ in different threshold values is ZDT1 > ZDT6 > ZDT4 > ZDT2 > ZDT3, especially for larger threshold. ZDT1 problem, which is a convex function and has no local optima, is a relatively easy problem. Chaotic maps bring the biggest improvements on solving ZDT1. Though the solutions of ZDT6 are nonuniformly spaced, chaotic maps can find better spread of solutions. While ZDT4 problem is a complex problem and the solutions are easily trapped into local optima, chaotic maps can improve the distribution of the solutions. ZDT2 problem is a convex function, and the solutions sometimes fall into the local optimum. The effects of chaotic maps can be generalized. The Pareto front of ZDT3 problem is segmented, so the Δ value of ZDT3 is larger and the ranking of ZDT3 is lower. It is our observation that Δ is not fit for evaluating the solutions to problems which are disconnected.

Based on the diversity metric Δ, chaotic maps have the best improvement on solving convex problems without local optima and have better effect on solving problems which have nonuniform solutions. For problems with local minimum, chaotic maps embedded algorithms can improve the performance with regard to metric Δ.

A short summary can be given according to the above experiments. First, chaotic maps can improve the performance of MOEAs, but the results showed that no one chaotic map outperforms other maps for all of the problems. The results in this paper give some guidance on how to choose a chaotic map and a phase in MOEAs. Second, an interesting discovery is that cat map has best performance on solving problems with local optima. Uniformity of cat map may be one of the reasons for the good performance of solving ZDT4.

8. Conclusion

The focus of this paper is to explore the relationships of chaotic maps and phases in MOEAs in solving MOPs. The main framework of algorithms in experiments is the NSGA-II algorithm. The combinations of ten chaotic maps and three phases are chosen in the experiments. Two metrics, convergence metric γ and diversity metric Δ, are used to evaluate the convergence and diversity properties of the algorithms with chaotic maps. The test problems are ZDT series which were all MOPs. The ergodicity and initial value sensitivity of chaotic maps can help evolutionary algorithms avoid solutions from falling into local optimal and get better convergence. In the experimental results, almost all chaotic maps have good effects on improving the performance of evolutionary algorithms to solve MOPs without local optima.

Cat map has best performance on solving problems with local optimum. This work gives insight on choosing chaotic maps and phases for different problems. Our future work will perform further experiments with more chaotic maps on other MOEAs and formulate the theory analysis.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This research is supported by the National Natural Science Foundation of China under Grant no. 61101153 and Prof. Qiu is supported by NSF CNS 1359557.

References

[1] H. Lu, R. Y. 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, pp. 2790–2802, 2013.

[2] J. A. Adeyemo, "Reservoir operation using multi-objective evolutionary algorithms-a review," *Asian Journal of Scientific Research*, vol. 4, no. 1, pp. 16–27, 2011.

[3] H. Hu, L. Xu, R. Wei, and B. Zhu, "Multi-objective control optimization for greenhouse environment using evolutionary algorithms," *Sensors*, vol. 11, no. 6, pp. 5792–5807, 2011.

[4] T. Niknam, "An efficient hybrid evolutionary algorithm based on PSO and HBMO algorithms for multi-objective Distribution Feeder Reconfiguration," *Energy Conversion and Management*, vol. 50, no. 8, pp. 2074–2082, 2009.

[5] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[6] E. Zitzler, M. Laumanns, L. Thiele et al., *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*, Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), 2001.

[7] 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.

[8] B. Alatas, E. Akin, and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons and Fractals*, vol. 40, no. 4, pp. 1715–1734, 2009.

[9] G. Zilong, W. Sun'an, and Z. Jian, "A novel immune evolutionary algorithm incorporating chaos optimization," *Pattern Recognition Letters*, vol. 27, no. 1, pp. 2–8, 2006.

[10] 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.

[11] G. Yu, X. Wang, and P. Li, "Application of chaotic theory in differential evolution algorithms," in *Proceedings of the 6th International Conference on Natural Computation (ICNC '10)*, pp. 3816–3820, August 2010.

[12] 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.

[13] L. D. S. Coelho and V. C. Mariani, "Chaotic artificial immune approach applied to economic dispatch of electric energy using thermal units," *Chaos, Solitons and Fractals*, vol. 40, no. 5, pp. 2376–2383, 2009.

[14] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[15] B. Alatas, "Chaotic harmony search algorithms," *Applied Mathematics and Computation*, vol. 216, no. 9, pp. 2687–2699, 2010.

[16] L. D. S. Coelho, "A quantum particle swarm optimizer with chaotic mutation operator," *Chaos, Solitons and Fractals*, vol. 37, no. 5, pp. 1409–1418, 2008.

[17] L. S. Dos Coelho and P. Alotto, "Multiobjective electromagnetic optimization based on a nondominated sorting genetic approach with a chaotic crossover operator," *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 1078–1081, 2008.

[18] H. F. Zhang, J. Z. Zhou, Y. C. 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.

[19] Y.-J. Wang and J.-S. Zhang, "Global optimization by an improved differential evolutionary algorithm," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 669–680, 2007.

[20] R. Caponetto, L. Fortuna, S. Fazzino, and M. G. Xibilia, "Chaotic sequences to improve the performance of evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, pp. 289–304, 2003.

[21] M. Ahmadi and H. Mojallali, "Chaotic invasive weed optimization algorithm with application to parameter estimation of chaotic systems," *Chaos, Solitons & Fractals*, vol. 45, no. 9-10, pp. 1108–1120, 2012.

[22] Z. S. Ma, "Chaotic populations in genetic algorithms," *Applied Soft Computing*, vol. 12, pp. 2409–2424, 2012.

[23] S. Talatahari, B. Farahmand Azar, R. Sheikholeslami, and A. H. Gandomi, "Imperialist competitive algorithm combined with chaos for global optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 3, pp. 1312–1319, 2012.

[24] W.-M. Zheng, "Kneading plane of the circle map," *Chaos, Solitons and Fractals*, vol. 4, no. 7, pp. 1221–1233, 1994.

[25] T. D. Rogers and D. C. Whitley, "Chaos in the cubic mapping," *Mathematical Modelling*, vol. 4, no. 1, pp. 9–25, 1983.

[26] M. Bucolo, R. Caponetto, L. Fortuna, M. Frasca, and A. Rizzo, "Does chaos work better than noise?" *IEEE Circuits and Systems Magazine*, vol. 2, no. 3, pp. 4–19, 2002.

[27] D. He, C. He, L.-G. Jiang, H.-W. Zhu, and G.-R. Hu, "Chaotic characteristics of a one-dimensional iterative map with infinite collapses," *IEEE Transactions on Circuits and Systems I*, vol. 48, no. 7, pp. 900–906, 2001.

[28] R. May, "Simple mathematical models with very complicated dynamics," in *The Theory of Chaotic Attractors*, B. Hunt, T. Y. Li, J. Kennedy, and H. Nusse, Eds., pp. 85–93, Springer, New York, NY, USA, 2004.

[29] R. Barton, "Chaos and fractals," *The Mathematics Teacher*, vol. 83, pp. 524–529, 1990.

[30] A. Baykasoglu, "Design optimization with chaos embedded great deluge algorithm," *Applied Soft Computing Journal*, vol. 12, no. 3, pp. 1055–1067, 2012.

[31] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and Chaos in*

Applied Sciences and Engineering, vol. 8, no. 6, pp. 1259–1284, 1998.

[32] V. I. Arnold and A. Avez, *Problèmes ergodiques de la mécanique classique*, Gauthier-Villars, Paris, France, 1967.

[33] G. M. Zaslavsky, "The simplest case of a strange attractor," *Physics Letters A*, vol. 69, no. 3, pp. 145–147, 1978/79.

[34] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, "A practical method for calculating largest Lyapunov exponents from small data sets," *Physica D*, vol. 65, no. 1-2, pp. 117–134, 1993.