Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Swarm and Evolutionary Computation 27 (2016) 31–67

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Artificial infectious disease optimization: A SEIQR epidemic dynamic model-based function optimization algorithm

Guangqiu Huang

School of Management, Xi'an University of Architecture and Technology, Xi'an 710055, China

ARTICLE INFO

Article history:
Received 10 February 2015
Received in revised form
5 August 2015
Accepted 21 September 2015
Available online 9 October 2015

Keywords:
Function optimization
Intelligent optimization computation
Epidemic dynamics
SEIQR epidemic model
Artificial infectious disease optimization

ABSTRACT

To solve some complicated function optimization problems, an artificial infectious disease optimization algorithm based on the SEIQR epidemic model is constructed, it is called as the SEIQR algorithm, or SEIQRA in short. The algorithm supposes that some human individuals exist in an ecosystem; each individual is characterized by a number of features; an infectious disease (SARS) exists in the ecosystem and spreads among individuals, the disease attacks only a part of features of an individual. Each infected individual may pass through such states as susceptibility (S), exposure (E), infection (I), quarantine (Q) and recovery (R). State S, E, I, Q and R can automatically and dynamically divide all people in the ecosystem into five classes, it provides the diversity for SEIQRA; that people can be attacked by the infectious disease and then transfer it to other people can cause information exchange among people, information exchange can make a person to transit from one state to another; state transitions can be transformed into operators of SEIQRA; the algorithm has 13 legal state transitions, which corresponds to 13 operators; the transmission rules of the infectious disease among people is just the logic to control state transitions of individuals among S, E, I, Q and R, it is just the synergy of SEIQRA, the synergy can be transformed into the logic structure of the algorithm. The 13 operators in the algorithm provide a native opportunity to integrate many operations with different purposes; these operations include average, differential, expansion, chevy, reflection and crossover. The 13 operators are executed equi-probably; a stable heart rhythm of the algorithm is realized. Because the infectious disease can only attack a small part of organs of a person when it spreads among people, the part variables iteration strategy (PVI) can be ingeniously applied, thus enabling the algorithm to possess of high performance of computation, high suitability for solving some kinds of complicated optimization problems, especially high dimensional optimization problems. Results show that SEIQRA has characteristics of strong search capability and global convergence, and has a high convergence speed for some complicated functions optimization problems.

1. Introduction

Infectious diseases can transmit from a person or any other species to another person or other species through various channels. Usually susceptible individuals can be infected with infectious diseases by directly contacting with infected individuals, body fluids and excreta of infected individuals, or materials polluted by infected individuals; infectious diseases can spread through air, water, food, skin contact, soil, blood, and etc [1]. In nature, there exist a large number of infectious viruses, humans, animals and plants are subjected to their threat all the time. Behind almost every infectious disease, a sad story is always implied. Ebola, outbreak in early 2014, and still raging in Africa, has made thousands of Africans died [2–4]; in early 2002, outbreak of SARS had made tens of thousands of Chinese people died, more Chinese people disabled [5–8]; H7N9 [9–12] or H5N6 [13–14], which visits southern regions of China every year, makes a lot of poultry slaughtered.

However, the purpose of this article is not to introduce a model to describe an infectious disease, but reveals an important application hiding in the spreading mechanism of each infectious disease, namely behind each infectious disease, even it hides an optimization algorithm which can solve some complicated function optimization problems! In other words, an infectious disease actually corresponds to an optimization algorithm. The task of this article is to reveal how this correspondence happens.

Suppose the optimization problem we want to solve is as follows:

$$\min f(\boldsymbol{X}) \quad \text{s.t.} \quad \begin{cases} g_i(\boldsymbol{X}) \geq 0, i \in I \\ \boldsymbol{X} \in S \subset R^n \end{cases} \tag{1}$$

where R^n is a n-dimensional Euclidean space; $\boldsymbol{X}=(x_1,x_2,\ldots,x_n)$ is an n-dimensional decision vector; S is a search space; $f(\boldsymbol{X})$ is an objective function; $g_i(\boldsymbol{X}) \geq 0$ is the ith inequality constraint, $i \in I$, I is the set of inequality constraints.

http://dx.doi.org/10.1016/j.swevo.2015.09.007
2210-6502/

If $f(\boldsymbol{X})$ is neither a concave function nor a convex function, or $f(\boldsymbol{X})$ and $g_i(\boldsymbol{X})$ are discontinuous or non-differentiable, or even their mathematical expressions don't know, then optimization problem (1) can only be solved by heuristic search methods [15–20], one of which is the population-based intelligence optimization method [21–24]. For a population-based intelligence optimization algorithm, we always assume that for given $\boldsymbol{X}$, $f(\boldsymbol{X})$ and $g_i(\boldsymbol{X})$ can be calculated, while $f(\boldsymbol{X})$ and $g_i(\boldsymbol{X})$ are always without any restrictions [25–27].

Up to now, many population-based intelligence optimization algorithms have been developed, for example, genetic algorithm (GA) [21,22,25,28–32], ant colony algorithm (ACA) [33,34], particle swarm optimization (PSO) [35–41], biogeography-based optimization (BBO) [42], differential evolution (DE) [43–45], artificial bee colony (ABC) [46–49], artificial immunity algorithm (AIA) [50–56] and evolutionary strategy (ES) [57,58] and so on.

Because a population-based intelligence optimization algorithm generally doesn't require special restrictions on objective function and constraints of an optimization problem, they have broad suitability and applicability [35]. A common feature of these algorithms is that evolutionary scene is very simple and corresponding operators are very few [35].

NFL [59,60] has pointed out, there is not an algorithm that can solve all optimization problems within finite time, but there is an algorithm that can solve some classes of optimization problems, for example, the simplex method can solve all linear programming problems [61]. Though operators contained in a population-based intelligence optimization algorithm are very simple, they are widely researched and applied in the wake of the corresponding algorithm [35].

Each algorithm can solve some kinds of optimization problems. If cores of these algorithms are extracted and combined into some new operators, then these new operators may have better suitability and wider application. For example, suppose we extract A cores from A algorithms, each core is called as an operation, if we select randomly a operations from the A cores to combine a new operator, then we can obtain C_A^a new operators. Obviously, the suitability of a new algorithm that possesses of the C_A^a new operators may be better than that of anyone of the A algorithms; applicable scope of the new algorithm may be wider than that of anyone of the A algorithms.

The above-mentioned strategy is called as integration in the article.

When a population-based intelligence optimization algorithm makes iteration, all variables in an individual, which is the biological explanation of an alternative solution of optimization problem (1), take part in computation simultaneously, we call the iteration strategy as all variables iteration (AVI). AVI means all dimensions of an optimization problem take part in computation simultaneously. The algorithms applying AVI include: GA, PSO, AFSA [62], BAT [63], Cuckoo Search (CS) [64–66], Glowworm-inspired Agent Swarms [67–68], AIA and its variations [50–56], ES [57,58], and so on. Because all variables of each alternative solution take part in computation during iteration, a population-based intelligence algorithm applying AVI may consume more CPU time, its efficiency of computation may be low relatively. Therefore it is not suitable to solve high dimensional optimization problems.

When a population-based intelligence optimization algorithm makes iteration, only a very small part of variables in an individual take part in computation simultaneously, we call the iteration strategy as part variables iteration (PVI). PVI means only a few of dimensions of an optimization problem take part in computation simultaneously. The algorithms applying PVI include: DE, BBO, ABC and so on. Because only a small part of variables of an alternative solution during iteration take part in computation, a population-based intelligence algorithm applying PVI may consume a little of CPU time, its efficiency of computation may be high comparatively, consequently it is suitable to solve some high dimensional optimization problems.

Therefore, developing a PVI-based population-based intelligence algorithm may be a good developing direction.

If a population-based intelligence optimization algorithm has its heart rhythm, then the algorithm may behave like a live animal. If the heart rhythm of the algorithm is stable, then it means that all operators contained in the algorithm are executed equi-probably; if the heart rhythm throbs (heart palpitation), then it means that a special operator is executed with higher probability under special conditions, a targeted exploration may be realized.

In a population-based intelligence optimization algorithm, many individuals work together, it is the basic property that a population-based intelligence optimization algorithm differs from a traditional optimization algorithm. In the article, we call the property as synergy, but we assign it much wider implications: synergy may be cooperation, competition, interaction, role changing, state transition and so on [35,86–88].

Information exchange among individuals is always carried out during iteration in a population-based intelligence optimization algorithm. By information exchange a new search strategy is formed. If information exchange of an algorithm is very sufficient, then the algorithm's performance may be good [35,46,57,62–64,87,88].

Diversity of individuals in a population-based intelligence optimization algorithm can make individuals to evolve vividly, and then reducing the probability that evolution drops into local pitfalls [69,70,87,88].

Evolution of population with the survival of the fittest is always the basis of a population-based intelligence optimization algorithm. If there exists evolution in a population, each individual has instincts to make itself to grow better or become stronger [25,71–74], then it means that the algorithm can converge to global optima of optimization problem (1) with higher probability [26,35,51,61,83,86–88].

Synergy can be transferred into the logic structure of a population-based intelligent optimization algorithm; information exchange can be described into operators of the algorithm; diversity can make search to develop along different directions; while the evolution and instincts of each individual will enable each individual to evolve toward better fitness so as to arrive at global optima at higher probability [26,35,51,61,74,83,86–88].

In nature, is there a scenario which can reflect the above-mentioned 8 properties simultaneously, namely integration, PVI, heart rhythm and heart palpitation, synergy, information exchange, diversity, and evolution? The answer is YES.

In the article, we introduce a new population-based intelligence optimization algorithm by telling the following sad story.

There are many people in an ecosystem; people always like to eat wild animals. One day, SARS (severe acute respiratory syndromes, SARS [5–8]) broke out suddenly within the ecosystem. At first, the SARS virus attacked some people because these people ate some wild animals that had infected with the SARS virus; hence these people were exposed, but did not come on.

Because SARS can spread by air, and people have the habitual nature of living together in the ecosystem, so the exposed people can easily transmit their diseases to other people. After a period of latency, the exposed people became ill. Because people did not know SRAS at that time, it provided a good opportunity for the virus to spread among people easily and widely. Through closely

contacting with the exposed or sick people, many people got infected. After several months, thousands and thousands of people had been infected with SARS, many people died, many people became maimed. A huge fear spread in the ecosystem, all kinds of social and economic activities stopped unavoidably.

From the sad story described above, we can find that at some time, some people were not infected with SARS, they are healthy (susceptible) (we call it state S); some people are exposed (we call it state E) because they are just infected with SARS but do not come on after making closely contacting with some exposed or sick people that had been infected with SARS or ate some wild animals infected with SARS; some people became ill (we call it state I) after they had been exposed for a period of time; some people were quarantined (we call it state Q) once they were found to have infected with SARS and became ill; some people were recovered (we call it state R) after they were exposed, infected, quarantined or vaccinated by medical treatment. Each person can only stay at one state at one time, this is to say, all people in the ecosystem were divided into five classes S, E, I, Q and R automatically and dynamically.

The aforementioned story can be described by an epidemic dynamic model call the SEIQR epidemic model [75–79], which is established base on the famous Kermack–Mckendrick bin model [80,81]. The SARS-based SEIQR model describes the SARS virus attacks human population; any individual in the population may pass through five states: susceptible (S), exposed (E), infected (I), quarantined (Q) and recovered (R).

State S, E, I, Q and R can automatically and dynamically divide all people in the ecosystem into five classes. It is just the diversity the SEIQR epidemic model can provide for us. That people can be attacked by SARS and then transfer their diseases to other people can cause information exchange among people. Information exchange can make a person to transit from one state to another; state transitions can be transformed into operators of the algorithm we introduce in the article. The rules of SARS transmission among people is just the logic to control state transitions of individuals among S, E, I, Q and R, it is just the synergy the SEIQR epidemic model can provide for us. The synergy can be transformed into the logic structure of the algorithm.

State transitions among state S, E, I, Q and R can produce 13 legal state transitions, which will be described in Section 2.4, if these legal state transitions are triggered evenly, then the 13 operators the 13 legal state transitions correspond to are executed equi-probably, a stable heart rhythm is realized. The 13 operators contained in the algorithm provide a native opportunity to integrate many operations with different purposes, and then integration is realized.

Because SARS can only attack a small part of organs of a person when it spreads among people, an organ corresponds with a variable of an alternative solution of optimization problem (1), PVI can be ingeniously applied, thus enabling the proposed algorithm to possess of high performance of computation, high suitability for solving some kinds of complicated optimization problems, especially high dimensional optimization problems.

Based on the aforementioned discussions, we summarize the following key points:

(1) *Synergy*. It is realized by the rules of SARS transmission which are described by the Kermack–Mckendrick bin model.
(2) *Information exchange*. When SARS spreads among people, exposed or infected people transmit their virus to other people, making their states to transfer among S, E, I, Q and R, and then information exchange is realized automatically and dynamically.
(3) *Diversity*. Individuals in the SEIQR epidemic model are automatically and dynamically divided into 5 classes, namely class S, E, I, Q and R, individuals in each class can be considered into a subclass of population, and then diversity is realized naturally.
(4) *Evolution*. Each individual in the ecosystem has instincts to survive, which makes itself to grow better or become stronger.
(5) *Integration*. There are 13 operators in the proposed algorithm, thus it provides a good opportunity to integrate some cores of some optimization algorithms together.
(6) *Heart rhythm*. 13 legal state transitions can lead the corresponding 13 operators to execute equi-probably; the curve of a state transition is very similar to that of a person's heart rhythm. If a special operator is found to have high performance when SEIQRA solves a problem, then the operator will be executed with higher probability, a targeted exploration is realized, it is just the heart palpitation.
(7) *PVI*. When SARS attacks people, only a very small part of organs are infected, through test we find that only 1/1000–1/10 of all variables take part in computation, the proposed algorithm can obtain high performance, thus a native reduction of dimensionality is realized.

The SEIQR epidemic model built based on the Kermack–Mckendrick bin model [32–33] is an nonlinear mathematical model that describes dynamic behaviors of individuals in random state transition among susceptibility (S), exposure (E), infection (I), quarantine (Q) and recovery (R) under the action of an infectious disease, the model considers an infectious disease not from the perspective of pathological knowledge of the infectious disease, but from the description of spread of the infectious disease, analysis of quantitative change rules of infected individuals and revelation of developmental state of the infectious disease according to general mechanism of transmission of the infectious disease and through the process of quantitative relation [33–35], which means that state transitions among state S, E, I, Q and R can be described by mathematical equations based on the Kermack–Mckendrick bin model. That an infectious disease spreads among individuals enables interaction among individuals to be reflected incisively and vividly, the biological meaning of the interaction can be clearly illustrated; what an infectious disease attack is a few of organs of an individual, the phenomenon, when mapped to the situation of searching optimum solutions of an optimization problem, is that the number of variables to be processed at each time deals with only a very small part of all variables. Therefore, the biological meaning of the variables treatment strategy in the algorithm is very clear.

Since the SEIQR epidemic model can appropriately describe epidemic rules of infectious diseases among individuals, which to a large extent, are beneficial to help depict information exchange among many individuals; while an individual is just a biologic explanation of an alternation solution of optimization problem (1); information exchange among many individuals means information exchange among many alternative solutions of optimization problem (1). Therefore the model has unique advantages in solving some complicated function optimization problems. Based on the SEIQR epidemic model in which individuals interact each other, this paper presents a new algorithm for function optimization, namely the SEIQR algorithm, or SEIQRA in short. The paper focuses on solving the following 6 problems:

(1) How to transfer a SEIQR epidemic model into a SEIQR function optimization algorithm that can solve some complicated optimization problems.

(2) How to make the operators in SEIQRA fully reflect the ideas of the SEIQR epidemic model.
(3) How to illustrate the convergence of SEIQRA.
(4) How to determine the suitable setting of parameters in SEIQRA and analyze the stability of parameter setting in SEIQRA.
(5) How to trace dynamic behaviors of the operators in SEIQRA.
(6) How to analyze the exploration and exploitation ability of SEIQRA.

The structure of the article is illustrated in Fig. 1. In nature, there are many stories that epidemic diseases spread among animals and/or humans, these stories can be described by epidemic dynamic models based on the Kermack and Mckendrick Assumptions [80,81]. Besides SARS, the SEIQR epidemic model can describe many other infectious diseases; each infectious disease can be transferred into a population-based intelligence optimization algorithm. SEIQRA proposed in the paper provides a reference to convert these stories of infectious diseases into population-based intelligence optimization algorithms.

2. The algorithm design based on the SEIQR epidemic model

To enable SEIRQA to adapt many kinds of optimization problems, the objective function of optimization problems (1) is rewritten as follows:

$$F(\boldsymbol{X}) = \begin{cases} f(\boldsymbol{X}) & \forall i \in \{1, 2, \cdots, I\}, g_i(\boldsymbol{X}) \geq 0; \boldsymbol{X} \in S \\ F_{\max} & \text{otherwise} \end{cases} \quad (2)$$

where $F_{\max}$ is a very large real number used to punish the alternative solutions that does not satisfy the constraints of optimization problems (1).

2.1. SARS

The SEIQR epidemic model [75–79] can be used to describe diseases like Ebola, SARS, H7N9, H5N6, Dengue and so on. Here we use the model to describe SARS, SARS is an acute respiratory infectious disease caused by the SARS corona virus (SARS-CoV), WHO named it as the severe acute respiratory syndrome (SARS) [5]. The main way to spread this disease is flying saliva

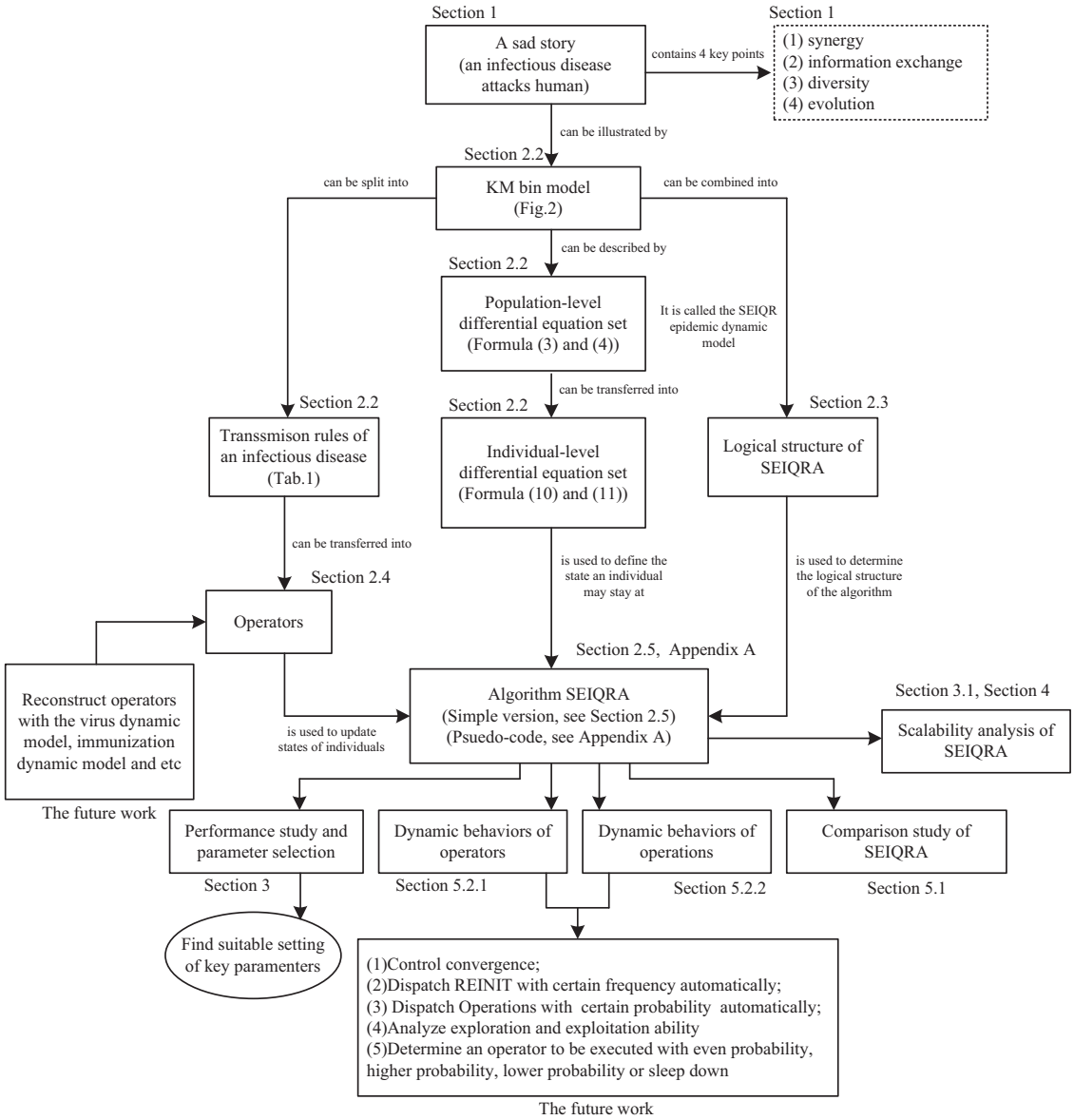

Fig. 1. The structure of the article.

transmission within short distance or close contact with patients' respiratory secretions. Fever, headache, muscle aches, fatigue, dry cough phlegm of patients are main clinical manifestations; respiratory distress may appear in severe cases. The disease has stronger infectivity, significant gathering phenomena happen in families and hospitals. Global initial cases occurred in Foshan, Guangdong Province of China, in November 2002, and quickly formed a popular trend.

The SARS virus is discharged out of body through respiratory tract secretions such as oral saliva, sneezing and so on, and spreads through the air; peaks of infection are in the autumn, winter and early spring of a year. The latency period of SARS is 3–5 days; after 10–14 days the disease outbursts, poisoning symptoms such as fever and fatigue worsen, and frequent cough, short breathing or difficulty breathing happens; only with slightly activity, asthma and heart palpitations may occur, patients are forced to stay in bed. The secondary infections of respiratory tract are prone to happen at this period. After 2–3 weeks, heat fades, other signs and symptoms relieve and even disappear; Absorption and recovery of pulmonary inflammatory changes are relatively slow, they need 2 weeks to be absorbed completely back to normal level after body temperature returns to normal.

2.2. SEIQR epidemic model

When people establish epidemic dynamic models for some infectious diseases, the bin modeling method represented by Kermack and McKendrick [80,81] is always used. Based on the method, the SEIQR epidemic model gives the following assumptions [75–79]:

(1) An human population in an ecosystem is divided into 5 bins (classes): susceptible individuals form class S; the exposed individuals (All those have closely contacted with carriers in the ecosystem, but not come on yet, they are potential onsets) class E; infected but not isolated individuals class I, isolated individuals class Q; recovered individuals that possess of short term of immunity class R; the number of individuals at time t in class S, E, I, Q and R is expressed by $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$.
(2) Infected individuals are forcibly isolated once they are found to be infected; the quarantine rate is δ.
(3) Only susceptible individuals are pulsatingly vaccinated, the vaccination rate is p, $0<p<1$, the cycle of pulsating vaccination is T.
(4) The standard transmission rate $\beta S(t)I(t)$ is applied, β is the contacting coefficient, $0<\beta<1$.
(5) The latency period of the infectious disease is τ, recovered individuals can obtain immunity. But after a period of time λ, the recovered individuals may lose immunity and transfer into susceptible individuals again, the losing rate of immunity is ρ, $0<\rho<1$.

According to the aforementioned assumptions, the flowchart of the SEIQR epidemic model with time lag and quarantine is shown in Fig. 2(a) [1,75–79].

In Fig. 2(a), A expresses the constant immigration of individuals; μ the normal mortality rate of individuals; α_1, α_2 and α_3 the mortality rate of infected, quarantined and exposed individuals because of illness respectively; γ_1, γ_2 and γ_3 the recovery coefficient of exposed, infected and quarantined individuals respectively. Under the impact of pulsating immunization, the corresponding epidemic model is as follows:

$$\begin{cases} \frac{dS(t)}{dt} = A-\mu S(t)-\beta S(t)I(t)+\rho R(t) \\ \frac{dE(t)}{dt} = \beta S(t)I(t)-(\mu+\gamma_1)E(t)-\beta e^{-\mu\tau}S(t-\tau)I(t-\tau)-\alpha_3 E(t) \\ \frac{dI(t)}{dt} = \beta e^{-\mu\tau}S(t-\tau)I(t-\tau)-(\alpha_1+\gamma_2+\mu+\delta)I(t) \qquad t\neq nT \\ \frac{dQ(t)}{dt} = \delta I(t)-(\alpha_2+\gamma_3+\mu)Q(t) \\ \frac{dR(t)}{dt} = \gamma_1 E(t)+\gamma_2 I(t)+\gamma_3 Q(t)-\mu R(t)-\rho R(t) \end{cases} \tag{3}$$

$$\begin{cases} S(t^+) = (1-p)S(t) \\ E(t^+) = E(t) \\ I(t^+) = I(t) \qquad t=nT \\ Q(t^+) = Q(t) \\ R(t^+) = R(t)+pS(t) \end{cases} \tag{4}$$

Suppose the number of individuals is $N(t)$, then $N(t)=S(t)+E(t)+I(t)+Q(t)+R(t)$. For the sake of briefness and quick computation, we do not consider the normal mortality rate of individuals and the mortality rate of infected, quarantined and exposed individuals because of illness, namely let $\mu=0$, $\alpha_1=0$, $\alpha_2=0$, $\alpha_3=0$, $A=0$, then $N(t)$ is constant N, namely $N(t)=N$, N is the number of individuals in an ecosystem. The flowchart of the simplified SEIQR epidemic model is shown in Fig. 2(b), its corresponding epidemic dynamic model is simplified into the following form:

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t)+\rho R(t) \\ \frac{dE(t)}{dt} = \beta S(t)I(t)-\gamma_1 E(t)-\beta S(t-\tau)I(t-\tau) \\ \frac{dI(t)}{dt} = \beta S(t-\tau)I(t-\tau)-(\gamma_2+\delta)I(t) \qquad t\neq nT \\ \frac{dQ(t)}{dt} = \delta I(t)-\gamma_3 Q(t) \\ \frac{dR(t)}{dt} = \gamma_1 E(t)+\gamma_2 I(t)+\gamma_3 Q(t)-\rho R(t) \end{cases} \tag{5}$$

In formula (4) and (5), there are population-level state variables $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$, there is no individual information in such a description. The method that population-level state variables $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ are transferred into individual-level state variables $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$ ($i=1, 2, \ldots, N$) is as follows:

For population-level state variables $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$, we have

$$S(t)+E(t)+I(t)+Q(t)+R(t)=N \tag{6}$$

Formula (6) means that there are $S(t)$ susceptible individuals, $E(t)$ vaccinated individuals, $I(t)$ infected individuals, $Q(t)$ quarantined individuals, $R(t)$ recovered individuals among N individuals. $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ are integer number, they are very difficult to process in solving differential equations.

We rewrite formula (6) as follows:

$$\frac{S(t)}{N}+\frac{E(t)}{N}+\frac{I(t)}{N}+\frac{Q(t)}{N}+\frac{R(t)}{N}=1 \tag{7}$$

Formula (7) means that if that the total number of individuals in the ecosystem is 1 unit, then $\frac{S(t)}{N}$, $\frac{E(t)}{N}$, $\frac{I(t)}{N}$, $\frac{Q(t)}{N}$ and $\frac{R(t)}{N}$ are the percentage of susceptible, exposed, infected, quarantined and recovered individuals respectively, they are continuous real number, they are very convenient to process in solving differential equations.

If we redefine state variables $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ as the percentage of susceptible, exposed, infected, quarantined and recovered individuals in an ecosystem respectively, then we can

rewrite formula (7) as follows:

$$S(t)+E(t)+I(t)+Q(t)+R(t)=1 \tag{8}$$

Formula (8) means that suppose that the total number of individuals in the ecosystem is 1 unit, then $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ represent the proportion of individuals in class S, E, I, Q and R respectively. So, for each individual in the ecosystem, $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ represent the probability that the individual belongs to class S, E, I, Q and R respectively, or the probability that the individual stays at state S, E, I, Q and R respectively.

Therefore formula (8) can be understood into the individual-level state equation:

$$S_i(t)+E_i(t)+I_i(t)+Q_i(t)+R_i(t)=1 \ (i \text{ can be any individual}) \tag{9}$$

where $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$ are the probability that an individual stays at state S, E, I, Q and R or belongs to class S, E, I, Q and R.

A similar analysis can be made from formula (3) to formula (5).

At last, which state an individual will stay at? It is defined by the probability distribution of $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$. For example, if $R_i(t)$ is the maximum one among $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$, then the probability of the individual staying at state R is highest.

Using formula (8) and (9), $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$ are changed into continuous real number, population-level state variables $S(t)$, $E(t)$, $I(t)$, $Q(t)$ and $R(t)$ are transferred into individual-level state variables $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$, they can be easily processed in mathematical computation, especially in solving differential equations, the method is widely used in Computational Biology and Population Dynamics.

Based on the aforementioned analysis, we can apply formula (4) and (5) to any individual in an ecosystem, say individual i, formula (4) and (5) can be rewritten into the following discrete iterating form, considering that β, γ_1, γ_2, γ_3, δ and p vary with time:

$$\begin{cases} S_i(t+1)=S_i(t)-\beta^t S_i(t)I_i(t)+\rho^t R_i(t) \\ E_i(t+1)=E_i(t)+\beta^t S_i(t)I_i(t)-\gamma_1^t E_i(t)-\beta^t S_i(t-\tau)I_i(t-\tau) \\ I_i(t+1)=I_i(t)+\beta^t S_i(t-\tau)I_i(t-\tau)-(\gamma_2^t+\delta^t)I_i(t) & t\neq nT \\ Q_i(t+1)=Q_i(t)+\delta^t I_i(t)-\gamma_3^t Q_i(t) \\ R_i(t+1)=1-S_i(t+1)-E_i(t+1)-I_i(t+1)-Q_i(t+1) \end{cases} \tag{10}$$

$$\begin{cases} S_i(t+1)=(1-p^t)S_i(t) \\ E_i(t+1)=E_i(t) \\ I_i(t+1)=I_i(t) & t=nT \\ Q_i(t+1)=Q_i(t) \\ R_i(t+1)=R_i(t)+p^t S_i(t) \end{cases} \tag{11}$$

$R_i(t+1)$ can be expressed as follows:

$$R_i(t+1)=R_i(t)+\gamma_1^t E_i(t)+\gamma_2^t I_i(t)+\gamma_3^t Q_i(t)-\rho^t R_i(t)$$

This is because:

$$\begin{aligned} R_i(t+1) &= 1-[S_i(t+1)+E_i(t+1)+I_i(t+1)+Q_i(t+1)] \\ &= 1-\Big[S_i(t)-\beta^t S_i(t)I_i(t)+\rho^t R_i(t)+E_i(t) \\ &\quad +\beta^t S_i(t)I_i(t)-\gamma_1^t E_i(t)-\beta^t S_i(t-\tau)I_i(t-\tau)\Big] \\ &\quad -\Big[I_i(t)+\beta^t S_i(t-\tau)I_i(t-\tau)-(\gamma_2^t+\delta^t)I_i(t) \\ &\quad +Q_i(t)+\delta^t I_i(t)-\gamma_3^t Q_i(t)\Big] \\ &= 1-\big[S_i(t)+E_i(t)+I_i(t)+Q_i(t)+\rho^t R_i(t) \\ &\quad -\gamma_1^t E_i(t)-\gamma_2^t I_i(t)-\gamma_3^t Q_i(t)\big] \\ &= R_i(t)+\gamma_1^t E_i(t)+\gamma_2^t I_i(t)+\gamma_3^t Q_i(t)-\rho^t R_i(t) \end{aligned} \tag{12}$$

In formula (12), we use $S_i(t)+E_i(t)+I_i(t)+Q_i(t)+R_i(t)=1$. At time t, in formula (10) and (11), all parameters take their values by $\beta^t=Rnd(0,\beta)$, $\gamma_1^t=Rnd(0,\gamma_1)$, $\gamma_2^t=Rnd(0,\gamma_2)$, $\gamma_3^t=Rnd(0,\gamma_3)$, $p^t=Rnd(0,p)$, $\delta^t=Rnd(0,\delta)$, $\rho^t=Rnd(0,\rho)$. $Rnd(a,b)$ means to generate a random real number with even distribution within interval $[a,b]$.

2.3. Scenario design of the algorithm

Suppose there are N human individuals in an ecosystem, these individuals are numbered with 1, 2, …, N, let $P=\{1, 2, \ldots, N\}$. Each individual is characterized by n features, namely for individual i, its features are $(x_{i1}, x_{i2}, \ldots, x_{in})$. There exists an infective disease called SARS in the ecosystem, which spreads among some individuals. Individuals will be infected with SARS through closely contacting with flying saliva or respiratory secretions of carriers. The virus just attacks a small part of features of an individual. In order to protect human individuals from SARS, susceptible individuals can be vaccinated. The vaccinated individuals will not be infected during a certain period of time, nor do they disseminate SARS to any other individuals. But the immunity of individuals is temporary, and it will lose effectiveness after a period of time. Then the human individuals without immunity will be infected with the infective disease again. The rules of SARS spreading among individuals are as follows:

(1) If an individuals who stays at susceptibility state S makes close contact with some individuals who have been exposed or infected with SARS, then it will catch the infectious disease. Individuals who are just infected with SARS do not come on at once; their diseases transfer firstly into the latency period, which is called exposure state E. The individuals staying within the latency period can transmit their diseases to other individuals also; these individuals at exposure state E can be cured if they accept medical treatment.
(2) After a period of time, some individuals who have stayed at exposure state E become ill, they transfer into infected state I, the individuals stayed in infection state I can transmit their virus to other individuals when they make close contact with them; these individuals at infection state I can be cured if they accept medical treatment.
(3) Once an infected individual is found, it will be quarantined at once, namely it moves into quarantine state Q; an infected individual who stays at quarantine state Q will stay at the state until the individual is cured. An individual who stays at state Q cannot spread its disease to any other individuals.
(4) If individuals who stay at quarantine state Q are cured, they will enter into recovery state R and obtain a certain period of immunity. Individuals who stay at state R will be not infected within a period of time, so they cannot transmit the infectious disease to other individuals at this period of time.
(5) The immunity of individuals who stay at recovery state R may be lost after a period of time, the individuals who lose their immunity will enter into susceptibility state S again; when these individuals contact closely with the exposed or infected individuals, they will become exposed again.
(6) In order to prevent SARS to spread casually and widely, some susceptible individuals may be vaccinated cyclically, which is called pulsating vaccination.

Step (1)–(6) are derived from Fig. 2(b), namely Step (1) means state transition S→E and E→R; Step (2) E→I and I→R; Step (3) I→Q; Step (4) Q→R; Step (5) R→S; Step (6) S→R. The other 5 state

Table 1
Legal state transition of the SEIQR epidemic model.

State at time $t-1$	State at time t	Type of state transition	Operator
S	S	S→S	S–S
S	E	S→E	S–E
S	R	S→R	S–R
E	E	E→E	E–E
E	I	E→I	E–I
E	R	E→R	E–R
I	I	I→I	I–I
I	Q	I→Q	I–Q
I	R	I→R	I–R
Q	Q	Q→Q	Q–Q
Q	R	Q→R	Q–R
R	R	R→R	R–R
R	S	R→S	R–S

transitions S→S, E→E, I→I, Q→Q and R→R are no necessary to be included into Step (1)–(6).

There are 8 state transitions in Step (1)–(6), 5 state transitions which keep their states unchanged are S→S, E→E, I→I, Q→Q and R→R. Therefore there are 8+5=13 state transitions, as listed in Table 1.

The above-mentioned scenario is mapped onto the process of searching the global optimum solutions of optimization problem (1), its meaning is as follows.

The search space of optimization problem (1) corresponds to the ecological system, an individual in the ecological system corresponds to an alternative solution of optimization problem (1), the set of alternative solutions N individuals correspond to is $X=\{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_N\}$. A feature of individual i ($i=1, 2, \ldots, N$) corresponds to a variable of alternative solution $\boldsymbol{X}_i$ of the optimization problem, namely feature j of individual i corresponds to variable x_{ij} of alternative solution $\boldsymbol{X}_i$, so the number of features of individual i is equal to the number of variables of alternative solution $\boldsymbol{X}_i$, they are both n. Therefore, individual i and alternative solution $\boldsymbol{X}_i$ are equivalent.

The physical strength of an individual is represented by individual physique index (IPI), which exactly corresponds to the objective function value of optimization problem (1). Good alternative solutions correspond to individuals with higher IPI values which represent these individuals with strong physique, while bad alternative solutions correspond to individuals with lower IPI values which represent these individuals with weak physique. For optimization problem (1), IPI of individual i is calculated as follows:

$$IPI(\boldsymbol{X}_i)=\begin{cases}\frac{1}{1+F(\boldsymbol{X}_i)}, \text{if } F(\boldsymbol{X}_i)\geq 0\\ 1+|F(\boldsymbol{X}_i)|, \text{if} F(\boldsymbol{X}_i)<0\end{cases}, i=1,\ 2, \ldots, N \tag{13}$$

At time t, randomly generate β^t, γ_1^t, γ_2^t, γ_3^t, p^t, δ^t, ρ^t of the ecosystem, formula (10) and (11) is used to calculate susceptible probability $S_i(t)$, exposure probability $E_i(t)$, infection probability $I_i(t)$, quarantine probability $Q_i(t)$ and recovery probability $R_i(t)$ of individual i. At time t, which state individual i stays at among state S, E, I, Q and R is determined by the probability distribution of $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$ and $R_i(t)$. Legal state transitions of any individual listed in Table 1 meet the situation depicted in Fig. 2(b).

For state S, E, I, Q and R, there are $5\times 5=25$ possible state transitions among individuals, but legal state transitions that satisfy Fig. 2(b) are 13, as shown in Table 1. Except the 13 legal state transitions in Table 1, other state transitions are illegal. The 13 legal state transitions can be described by 13 operators, as shown in Table 1. From formula (10) and (11) we can know that susceptible probability $S_i(t)$, exposure probability $E_i(t)$, infection probability $I_i(t)$, quarantine probability $Q_i(t)$ and recovery probability $R_i(t)$ of individual i vary with time, so the growth state of individual i will transfer randomly among state S, E, I, Q and R. This state transition, when mapped onto the solution space of optimization problem (1), means that each alternative solution in the solution space moves from one location to another, then the random search in the solution space is realized.

During the random search process, if IPI of individual i at time t is higher than that at time $t-1$, individual i will continue to grow, which means that individual i is getting closer to global optimum solutions. On the contrary, if IPI of individual i at time t is less than or equal to that at time $t-1$, individual i will stop growing, which means that individual i stays at the position of time $t-1$ and does not move. The random search strategy is called "each-step-is-not-bad" [83,86–88], it allows the algorithm to converge.

According to the scenario design of SEIQRA, the implementation process of SEIQRA is described as follows.

2.3.1. Logical structure of algorithm SEIQRA

(1) At time t, β^t, γ_1^t, γ_2^t, γ_3^t, p^t, δ^t of a population are generated randomly. At the beginning, $t=0$.
(2) Use formula (10) and (11) to calculate the susceptibility probability $S_i(t)$, exposure probability $E_i(t)$, infection probability $I_i(t)$, quarantine probability $Q_i(t)$ and recovery probability $R_i(t)$ of individual i at time t respectively.
(3) Determine which state is selected according to probability distribution $\{S_i(t), E_i(t), I_i(t), Q_i(t), R_i(t)\}$ of individual i. There are five cases as follows:
(31) If state S is selected, then it means that at time t individual i stays at state S. At this time we judge at which state individual i stayed at time $t-1$.There are two legal cases:

(a) If at time $t-1$ individual i stayed at state S, then we can know that now individual i stays still at the susceptible state S, namely S→S. In this case we use the S–S operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(b) If at time $t-1$ individual i stayed at state R, then we can know that now individual i loses its immunity and transfers into susceptible state S, namely R→S. In this case we use the R–S operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.

(32) If state E is selected, then it means that at time t individual i stays at state E. At this time we judge at which state individual i stayed at time $t-1$.There are two legal cases:

(a) If at time $t-1$ individual i stays at state E, then we can know that now individual i still stays at the exposed state E, namely E→E. In this case we use the E–E operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(b) If at time $t-1$ individual i stayed at state S, then we can know that now individual i is infected with SARS but does not come on, namely S→E. In this case we use the S–E operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.

(33) If state I is selected, then it means that at time t individual i stays at state I. At this time we judge at which state individual i stayed at time $t-1$. There are two legal cases:

(a) If at time $t-1$ individual i stayed at state E, then we can know that now individual i becomes ill after it is exposed (transferring from state E into state I), namely E→I. In this case we use the E–I operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(b) If at time $t-1$ individual i stayed at state I, then we can know that now individual i continues to be sick, namely I→I. In this case we use the I–I operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.

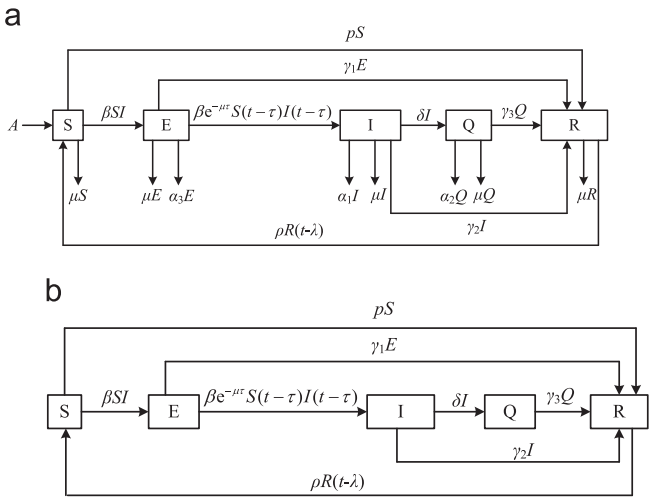

Fig. 2. The flowchart of the SEIQR epidemic model: (a) before simplification and (b) after simplification.

(34) If state Q is selected, then it means that at time t individual i stays at state Q. At this time we judge at which state individual i stayed at time $t-1$.There are two legal cases:

(a) If at time $t-1$ individual i stayed at state I, then we can know that now individual i is quarantined after it is infected, namely I→Q. In this case we use the I–Q operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(b) If at time $t-1$ individual i stayed at state Q, then we can know that now individual i is still in quarantine state, namely Q→Q. In this case we use the Q–Q operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.

(35) If state R is selected, then it means that at time t individual i stays at state R. At this time we judge at which state individual i stayed at time $t-1$.There are five legal cases:

(a) If at time $t-1$ individual i stayed at state S, then we can know that now individual i is vaccinated, and it cannot be infected within a time of time, namely S→R. In this case we use the S–R operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(b) If at time $t-1$ individual i stays at state E, then we can know that now individual i is cured after it is exposed, namely E→R. In this case we use the E–R operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(c) If at time $t-1$ individual i stayed at state I, then we can know that now individual i is cured after it is infected, namely I→R. In this case we use the I–R operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(d) If at time $t-1$ individual i stayed at state Q, then we can know that now individual i is cured and obtains a certain period of immunity simultaneously after it is isolated, namely Q→R. In this case we use the Q–R operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.
(e) If at time $t-1$ individual i stayed at state R, then we can know that now individual i does not lose its immunity and still stays at state R, namely R→R. In this case we use the R–R operator to calculate the growth state $\boldsymbol{X}_i^t$ of individual i.

(4) If at time t the state transition of individual i is illegal, then individual i keeps its original state unchanged.
(5) Repeat Step (2) to (4) until each individual is processed, then go to Step (6).

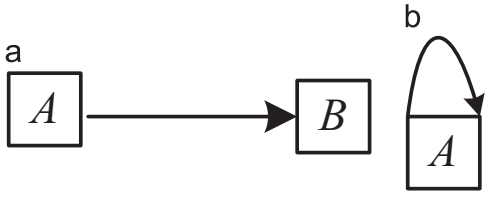

Fig. 3. Two kinds of state transition of an individual at time t.

(6) Let $t=t+1$.Repeat Step (1) to (5) until the global optimum solution is found.

2.4. Design of ecological operators

At time t, the initial values of N individuals in the ecosystem are $\boldsymbol{X}_i^t = (x_{i1}^t, x_{i1}^t, \cdots x_{in}^t)$, $i=1,2, \ldots, \boldsymbol{N}$, we give the design method of the 13 ecological operators listed in Table 1.

In GA [21,22,25,28–32], when two chromosomes carry out crossover operation, male and female chromosome are randomly selected from many chromosomes; position of crossover in each chromosome is randomly selected also. The idea that results from genetics, but do not fully abide by genetics can greatly simplify the design of GA, the method is widely used in population-based intelligence optimization algorithms.

Similarly, in SEIQRA, we assume that

(1) SARS can only attack a small part of features of an individual each time; these features are randomly selected from all features of the individual. Once an individual is attacked by SARS, its all features possess automatically of the SARS virus.
(2) When an individual stays at certain state, its all features possess automatically of the corresponding properties acclimatizing to the state. For example, when an individual stays at state S, its all features can be attacked by SARS; when the individual stays at state E, its all features contain the SARS virus, but do not come on; when the individual stays at state I, its all features are infected with the SARS virus, and become ill; when the individual stays at state Q, its all features are immunized, but do not recover; when the individual stays at

state R, its all features are not only immunized, but also are cured.

2.4.1. The biological meaning of transition types

Individual i can be described with the following structure:

$$\langle SEIQR_i(t), \boldsymbol{X}_i^t \rangle, i = 1, 2, \cdots, N$$

Where $SEIQR_i(t)$ expresses the state of individual i at time t, $SEIQR_i(t) \in \{S, E, I, Q, R\}$; $\boldsymbol{X}_i^t$ expresses the state value of features of individual i at time t.

Based on the Fig. 2(b), we can find the general method to explain each state transition and update $SEIQR_i(t)$ and $\boldsymbol{X}_i^t$ of individual i, and define the biological meaning of transition types. By decomposing Fig. 2(b), we obtain the following two cases:

(1) At time t, an individual can transfer from state A to state B, as shown in Fig. 3(a), where A and $B \in \{S, E, I, Q, R\}$, but $A \neq B$. A large number of state transitions belong to the case, for example, S→E, S→R, E→I, E→R, I→Q, I→R, Q→R and R→S. When an individual transfers from state A to state B, or an individual at class A becomes an individual at class B, we will let the composite state values of some features of several individuals who stay at state B endow to the corresponding features of the individual who stays at state A, namely we enable some features of the individual to possess of similar state values of the corresponding features of some individuals who have stayed at state B. This method realizes that an individual transfers from class A to class B really. For example, for state transition S→E, we let the average state values of some features of several exposed or infected individuals who have stayed at state E or state I endow to the corresponding features of a susceptible individual who stays at state S, then we can make the individual to get exposed, namely state transition S→E is achieved. This strategy means that some exposed or infected individuals transfer their own something with virus to a susceptible individual, making it to get exposed.

(2) At time t, when an individual stays at state A, $A \in \{S, E, I, Q, R\}$, any state transition does not take place, it is equivalent to $A \to A$, as shown in Fig. 3(b). Each node in Fig. 2(b) actually implies the case shown in Fig. 3(b), for example, S→S, E→E, I→I, Q→Q and R→R.

When an individual stays at state A, in order to enable the individual to evolve towards good direction, but its state remains the same, we will let the composite state values of some features of several strong individuals whose IPI values are higher than that of the individual transfer to the corresponding features of the individual, namely strong individuals with higher IPI transfer strong feature information to weak individuals with lower IPI so as to make these weak individuals to grow for the better direction. The reason is that when individuals stay at susceptible, exposed, infected, isolated or recovery state, they always makes their own physique to be improved by keeping in good health, health care, training or other ways; when individuals stay at state of illness, they always make their own physique to be enhanced through medical treatment and nutritional supplement so as to achieve the goal of conquer disease.

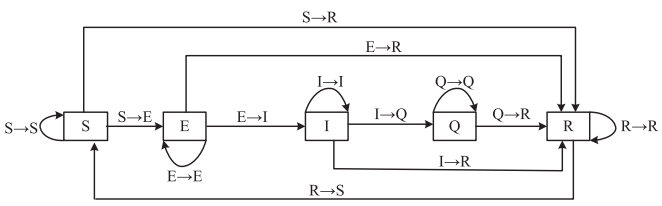

Fig. 4. State transitions of the SEIQR epidemic model.

After state transitions are explicitly expressed according to Fig. 3, the flowchart shown in Fig. 2(b) can be illustrated as Fig. 4.

For state transition $A \to B$, we randomly select L individuals from those who have stayed at state B, let the composite state value of a randomly selected feature j of the L individuals be transferred to the corresponding feature j of current individual i who stays at state A, then the individual obtains the similar properties with the individuals staying at state B. Thus, state transition $A \to B$ is realized, namely

$$f\left(x_{i_1 j}^{t-1}, x_{i_2 j}^{t-1}, \cdots, x_{i_L j}^{t-1}\right) \overset{A \to B}{\Longrightarrow} x_{ij}^t$$

Where $i_1, i_2, \ldots, i_L$ are the numbers of the L randomly selected individuals who have stayed at state B; function f expresses the method of combining the state value of feature j of the L individuals, it produces a new state value of feature j, the new state value will be transferred to the corresponding feature j of current individual i staying at state A; f is equivalent to a mixer, and can be defined according to Section 2.4.2.

For state transition $A \to A$, we randomly select L strong individuals from those who have stayed at state A but their IPI values are higher than that of current individual i; let the composite state value of a randomly selected feature j of the L strong individuals be transferred to the corresponding feature j of current individual i who stays at state A, then the individual may become stronger also, but its state does not transfer. Thus, state transition $A \to A$ is realized. Namely

$$f\left(x_{s_1 j}^{t-1}, x_{s_2 j}^{t-1}, \cdots, x_{s_L j}^{t-1}\right) \overset{A \to A}{\Longrightarrow} x_{ij}^t$$

where $s_1, s_2, \ldots, s_L$ are the numbers of the L strong individuals, but IPI values of the L strong individuals are higher than that of individual i.

2.4.2. Method of operators design

Function f mentioned in Section 2.4.1 is defined as average, differential, expansion, chevy, reflection and crossover operation.

2.4.2.1. Design of operations. At time $t-1$, suppose the set C^{t-1} of individuals that satisfy given requirements, the current individual is i, its affected feature is j, then the design methods of average, differential, expansion, chevy, reflection and crossover operation are as follows:

(1) Average Operation $AVG(C^{t-1}, i, j)$. Randomly select L individuals from C^{t-1}, let the state values of feature j of the L individuals are averaged to produce a new compound value by use of average operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individuals in C^{t-1}, namely

$$\begin{cases} v_{ij}^t = \frac{1}{\left|C^{t-1}\right|} \sum\limits_{k \in C^{t-1}} x_{kj}^{t-1} & \left|C^{t-1}\right| > 0 \\ v_{ij}^t = x_{ij}^{t-1} & \left|C^{t-1}\right| = 0 \end{cases}$$

where, $i_1, i_2, \ldots, i_L$ are the numbers of the L individuals selected randomly from C^{t-1}; $\boldsymbol{V}_i^t = (v_{i1}^t, v_{i2}^t, \cdots, v_{in}^t)$, $\boldsymbol{X}_{i_k}^{t-1} = (x_{i_k 1}^{t-1}, x_{i_k 2}^{t-1}, \cdots, x_{i_k n}^{t-1})$, v_{ij}^t and $x_{i_k j}^{t-1}$ are the state values of feature j of individual i at time t and $t-1$ respectively.

(2) Differential Operation $DE(C^{t-1}, i, j)$. Randomly select 3 individuals from C^{t-1}, let the state values of feature j of the 3 individuals are differentiated to produce a new compound value by use of differential operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individuals in

Table 2
Design of operators in SEIQRA.

Type of state transition	Operator	Match of operation A and B	Function of operation A and B
S→S	S–S	Average operation/differential operation	$AVG(C_{PS}^{t-1}, i, j)/DE(C_{PS}^{t-1}, i, j)$
S→E	S–E	Average operation/expansion operation	$AVG(C_{E}^{t-1} \cup C_{I}^{t-1}, i, j)/EPN(C_{E}^{t-1} \cup C_{I}^{t-1}, i, j)$
S→R	S–R	Differential operation/expansion operation	$DE(C_{R}^{t-1}, i, j)/EPN(C_{R}^{t-1}, i, j)/CHV(C_{R}^{t-1}, i, j)/RFL(C_{R}^{t-1}, i, j)/CRS(C_{R}^{t-1}, i, j)/AVG(C_{R}^{t-1}, i, j)$
E→E	E–E	Differential operation/chevy operation	$DE(C_{PE}^{t-1}, i, j)/CHV(C_{PE}^{t-1}, i, j)$
E→I	E–I	Expansion operation/chevy operation	$EPN(C_{I}^{t-1}, i, j)/CHV(C_{I}^{t-1}, i, j)$
E→R	E–R	Expansion operation/reflection operation	$EPN(C_{R}^{t-1}, i, j)/RFL(C_{R}^{t-1}, i, j)$
I→I	I–I	Chevy operation/reflection operation	$CHV(C_{PI}^{t-1}, i, j)/RFL(C_{PI}^{t-1}, i, j)$
I→Q	I–Q	Chevy operation/crossover operation	$CHV(C_{Q}^{t-1}, i, j)/CRS(C_{Q}^{t-1}, i, j)$
I→R	I–R	Reflection operation/ crossover operation	$RFL(C_{R}^{t-1}, i, j)/CRS(C_{R}^{t-1}, i, j)$
Q→Q	Q–Q	Reflection operation/average operation	$RFL(C_{PQ}^{t-1}, i, j)/AVG(C_{PQ}^{t-1}, i, j)$
Q→R	Q–R	Crossover operation /differential operation	$CRS(C_{R}^{t-1}, i, j)/DE(C_{R}^{t-1}, i, j)$
R→R	R–R	Crossover operation/average operation	$CRS(C_{PR}^{t-1}, i, j)/AVG(C_{PR}^{t-1}, i, j)$
R→S	R–S	Differential operation/crossover operation	$DE(C_{S}^{t-1}, i, j)/EPN(C_{S}^{t-1}, i, j)$

C^{t-1}, namely

$$v_{ij}^t = x_{i_1}^{t-1} + 0.5\left(x_{i_2}^{t-1} - x_{i_3}^{t-1}\right)$$

If $\left|C^{t-1}\right| \geq 3$, then 3 individuals i_1, i_2 and i_3 are randomly selected from C^{t-1}, namely $i_1, i_2, i_3 \in C^{t-1}$; if $\left|C^{t-1}\right| < 3$, then 3 individuals i_1, i_2 and i_3 are randomly selected from N individuals, namely $i_1, i_2, i_3 \in P$. Obviously, the differential operation is the core of DE [43–45].

(3) Expansion Operation $EPN(C^{t-1}, i, j)$. Randomly select an individual from C^{t-1}, let the state value of feature j of the individual is expanded to produce a new compound value by use of expansion operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individual in C^{t-1}, namely

$$\begin{cases} v_{ij}^t = x_{ij}^{t-1} + r_1\left(x_{ij}^{t-1} - x_{kj}^{t-1}\right) & k \in C^{t-1}, \left|C^{t-1}\right| > 0 \\ v_{ij}^t = x_{ij}^{t-1} + r_1\left(x_{ij}^{t-1} - x_{i_{\mathrm{gBest}}j}^{t-1}\right) & \left|C^{t-1}\right| = 0 \end{cases}$$

where if $|C^{t-1}| > 0$, then individual k is randomly selected from C^{t-1}, namely $k \in C^{t-1}$; if $\left|C^{t-1}\right| = 0$, then let $k = i_{\mathrm{gBest}}$, i_{gBest} is the number of the best individual up to time $t-1$; $r_1 = Rnd(-1,1)$. Obviously, the expansion operation is the core of ABC [46–49].

(4) Chevy Operation $CHV(C^{t-1}, i, j)$. Find the best individual i_{lBest} in C^{t-1}, let the state value of feature j of the best individual is chevyed to produce a new compound value by use of chevy operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individual in C^{t-1}, namely

$$\begin{cases} v_{ij}^t = r_2 x_{ij}^{t-1} + r_3\left(x_{i_{\mathrm{lBest}}j}^{t-1} - x_{ij}^{t-1}\right) + r_4\left(x_{i_{\mathrm{gBest}}j}^{t-1} - x_{ij}^{t-1}\right) & \left|C^{t-1}\right| > 0 \\ v_{ij}^t = x_{i_{\mathrm{gBest}}j}^{t-1} & \left|C^{t-1}\right| = 0 \end{cases}$$

where $i_{\mathrm{lBest}} \in C^{t-1}$; $r_2 = Rnd(0.4, 0.6)$, $r_3 = 1.3Rnd(0,1)$, $r_4 = 1.8Rnd(0,1)$. Obviously, the chevy operation is the core of PSO [35–41].

(5) Reflection Operation $RFL(C^{t-1}, i, j)$. Randomly select 2 individuals from C^{t-1}, let the state values of feature j of the 2 individual are reflected to produce a new compound value by use of reflection operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individuals in C^{t-1}, namely

$$\begin{cases} v_{ij}^t = x_{i_1 j}^{t-1} + x_{i_2 j}^{t-1} - x_{ij}^{t-1} & \left|C^{t-1}\right| > 0 \\ v_{ij}^t = x_{i_{\mathrm{gBest}}j}^{t-1} & \left|C^{t-1}\right| = 0 \end{cases}$$

If $|C^{t-1}| > 1$, then i_1 and i_2 are randomly selected from C^{t-1}, namely $i_1, i_2 \in C^{t-1}$; if $|C^{t-1}| = 1$, then i_1 is randomly selected from C^{t-1}, namely $i_1 \in C^{t-1}$, $i_2 = i_{\mathrm{gBest}}$; if $|C^{t-1}| = 0$, then $v_{ij}^t = x_{i_{\mathrm{gBest}}j}^{t-1}$.

(6) Crossover Operation $CRS(C^{t-1}, i, j)$. Randomly select 2 individuals from C^{t-1}, let the state values of feature j of the 2 individual mate to produce a new compound value by use of crossover operation, the new value is transferred to the corresponding feature j of individual i at time t, enabling individual i to have similar properties with the individuals in C^{t-1}, namely

$$\left.\begin{cases} v_{ij}^t = a x_{i_1 j}^{t-1} + (1-a) x_{i_2 j}^{t-1} & Rnd(0,1) < 0.5 \\ v_{ij}^t = a x_{i_2 j}^{t-1} + (1-a) x_{i_1 j}^{t-1} & Rnd(0,1) \geq 0.5 \end{cases}\right\} \quad \left|C^{t-1}\right| \geq 2 \quad v_{ij}^t$$

$$= (1+a) x_{i_{\mathrm{gBest}}j}^{t-1} - a x_{i_1 j}^{t-1} \left|C^{t-1}\right| = 1 \; v_{ij}^t = x_{i_{\mathrm{gBest}}j}^{t-1} \qquad \left|C^{t-1}\right| = 0$$

If $|C^{t-1}| > 1$, then i_1 and i_2 are randomly selected from C^{t-1}, namely $i_1, i_2 \in C^{t-1}$; if $|C^{t-1}| = 1$, then i_1 is randomly selected from C^{t-1}, namely $i_1 \in C^{t-1}$, $i_2 = i_{\mathrm{gBest}}$; if $|C^{t-1}| = 0$, then $v_{ij}^t = x_{i_{\mathrm{gBest}}j}^{t-1}$; $a = Rnd(-0.5, 1.5)$. Obviously, the crossover operation is the core of RC-GA [32].

2.4.2.2. Design of operators. At time $t-1$, randomly select L individuals from the susceptible, exposed, infected, quarantined and recovered individuals respectively to form the following sets:

The set of susceptible individuals:

$$C_{\mathrm{S}}^t = \left\{\boldsymbol{X}_{i_1}^{\mathrm{S}}(t), \boldsymbol{X}_{i_2}^{\mathrm{S}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{S}}(t)\right\}$$

The set of exposed individuals:

$$C_{\mathrm{E}}^t = \left\{\boldsymbol{X}_{i_1}^{\mathrm{E}}(t), \boldsymbol{X}_{i_2}^{\mathrm{E}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{E}}(t)\right\}$$

Table 3
The strategy of taking values of the parameters in SEIQRA.

Name of parameter	Strategy of initialization
The maximum periods: G	G is used to prevent the iteration from dropping into infinite looping when the convergence condition is not satisfied, generally G=8000–300,000.
The lowest error: ε	$\varepsilon > 0$, the smaller the ε is, the higher the precision of the found optimum solution will be, but the longer the calculation time will be, generally the scope of value is $\varepsilon = 10^{-5}$–10^{-10}.
The number of variables: n	n is defined by the actual optimization problem to be solved.
The number of work individuals: N	Although N taking larger value can expand the search scope, the overall time complexity of the algorithm is proportional to N, therefore N cannot be made too large, the precision of the parameter value is without too high, simply on the basis of the specific optimization problem and the speed of your computer, and range of value for N is 100–2000.
The number of individuals that join in information exchange: L	$L \geq 1$, the parameter is not sensitive, generally L=6
The biggest probability that a feature of an individual is attacked by an infectious disease: E_0	$0 < E_0 < 1$, at every step of evolution, only a part of features are allowed to be attacked by an infectious disease. if n is smaller, then E_0=1/200–1/20; if n is bigger, then E_0=1/500–1/1000
Latency period: τ	Always set τ=1
Immunity period: λ	Always set λ=1
Vaccination cycle: T	Always set T=4
The upper limit of the SEIQR epidemic model's parameters β, γ_1, γ_2, γ_3, p, δ: d_0	$0 < d_0 < 1$, always set d_0=0.1
The initialization number of individuals: N_0	$N_0 \geq$ 10,000
The allowable maximum generations the global optimization solution keeps unchanged: $t_{\max}$	$t_{\max}$=10–100
The shrinking factor: ρ	$0 < \rho < 1$, generally ρ=0.8
The lower bound of interval shrinkage: LU_0	$LU_0 > 0$, generally $LU_0 = \varepsilon$
The ratio of individuals being replaced: r	r=0–1, generally r=0.5
The parameters of the SEIQR epidemic model: β_0, γ_1, γ_2, γ_3, p, δ	β_0, γ_1, γ_2, γ_3, p and $\delta = Rnd(0,d_0)$, d_0=0.1

The set of infected individuals:

$$C_{\mathrm{I}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{I}}(t), \boldsymbol{X}_{i_2}^{\mathrm{I}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{I}}(t) \right\}$$

The set of quarantined individuals:

$$C_{\mathrm{Q}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{Q}}(t), \boldsymbol{X}_{i_2}^{\mathrm{Q}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{Q}}(t) \right\}$$

The set of recovered individuals:

$$C_{\mathrm{R}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{R}}(t), \boldsymbol{X}_{i_2}^{\mathrm{R}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{R}}(t) \right\}$$

At time $t-1$, randomly select L individuals whose IPI values are higher than that of current individual i from the susceptible, exposed, infected, quarantined and recovered individuals respectively to form the following sets:

The set of strong susceptible individuals:

$$C_{\mathrm{PS}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{PS}}(t), \boldsymbol{X}_{i_2}^{\mathrm{PS}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{PS}}(t) \right\}$$

The set of strong exposed individuals:

$$C_{\mathrm{PE}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{PE}}(t), \boldsymbol{X}_{i_2}^{\mathrm{PE}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{PE}}(t) \right\}$$

The set of strong infected individuals:

$$C_{\mathrm{PI}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{PI}}(t), \boldsymbol{X}_{i_2}^{\mathrm{PI}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{PI}}(t) \right\}$$

The set of strong quarantined individuals:

$$C_{\mathrm{PQ}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{PQ}}(t), \boldsymbol{X}_{i_2}^{\mathrm{PQ}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{PQ}}(t) \right\}$$

The set of strong recovered individuals:

$$C_{\mathrm{PR}}^t = \left\{ \boldsymbol{X}_{i_1}^{\mathrm{PR}}(t), \boldsymbol{X}_{i_2}^{\mathrm{PR}}(t), \cdots, \boldsymbol{X}_{i_L}^{\mathrm{PR}}(t) \right\}$$

For each operator, its standard structure is as follows:

Operator*X*
  IF*Rnd*(0,1) < 0.5 **THEN**
    Execute operation *A*;
  ELSE
    Execute operation *B*;
  END IF
END

For each state transition listed in Table 1, its corresponding operator and the match of operation *A* and *B* are shown in Table 2.

By careful arrangement, each operation in Table 2 has the same probability 1/6 to be executed. The meaning of elements in Table 2 is explained as follows:

For state transition S→S, its corresponding operator is S–S, the structure of the operator is as follows:

Operator S–S
  IF*Rnd*(0,1) < 0.5 **THEN**
    Execute average operation: $AVG(C_{\mathrm{PS}}^{t-1}, i, j)$;
  ELSE
    Execute differential operation: $DE(C_{\mathrm{PS}}^{t-1}, i, j)$;
  END IF
END

For state transition S→R, its corresponding operator is S–R, the structure of the operator is as follows:

Operator S–R
  IF*Rnd*(0,1) < 1/6 **THEN**
    Execute average operation: $DE(C_{\mathrm{R}}^{t-1}, i, j)$;
  ELSE IF*Rnd*(0,1) < 2/6 **THEN**
    Execute differential operation: $EPN(C_{\mathrm{R}}^{t-1}, i, j)$;
  ELSE IF*Rnd*(0,1) < 3/6 **THEN**
    Execute differential operation: $CHV(C_{\mathrm{R}}^{t-1}, i, j)$;
  ELSE IF*Rnd*(0,1) < 4/6 **THEN**
    Execute differential operation: $RFL(C_{\mathrm{R}}^{t-1}, i, j)$;
  ELSE IF*Rnd*(0,1) < 5/6 **THEN**
    Execute differential operation: $CRS(C_{\mathrm{R}}^{t-1}, i, j)$;

Table 4
The time complexity of SEIQRA.

Operation	Average time complexity	Maximum loop times
Computation of $S_i(t)$, $E_i(t)$, $I_i(t)$, $Q_i(t)$, $R_i(t)$, $SEIQR_i(t)$	$O(12)$	$(G+N+9)N$
Operators: S–S,S–E,E–E,E–I,E–R,I–I,I–Q,I–R,Q–Q,Q–R,R–R,R–S	$O(3nE_0/13)$	$(G+N+9)(N+3)$
S–R	$O(6nE_0/13)$	$(G+N+9)(N+3)$
Operation *AVG*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Operation *DE*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Operation *EPN*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Operation *CHV*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Operation *RFL*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Operation *CRS*	$O((5N+5L+3)nE_0/6)$	$(G+N+9)(N+3)$
Maintenance of states	$O((1-7E_0/10)n)$	$(G+N+9)(N+3)$
Calculation of objective function	$O(n)$–$O(n^2)$	$(G+N+9)(N+3)$
Operator growth	$O(3n)$	$(G+N+9)(N+3)$
Operator REINIT	$O(3n+7(n+1)N+n^2N)$	$G+N+9$
Output of results	$O(n)$	1

ELSE

Execute differential operation: $AVG(C_R^{t-1}, i, j)$;

END IF

END

2.4.3. Design of other operators

2.4.3.1. Operator growth. For an individual, compare its new generation with its current generation, replace the current generation with the new generation if the latter are better than the former; otherwise keep the former unchanged. For optimization problem (1), operator growth can be described as follows:

$$\boldsymbol{X}_i^t = \begin{cases} \boldsymbol{V}_i^t & \text{If } IPI(\boldsymbol{V}_i^t) > IPI(\boldsymbol{X}_i^t) \\ \boldsymbol{X}_i^{t-1} & \text{otherwise} \end{cases}, i = 1, 2, \ldots, N \tag{14}$$

Where function IPI($\boldsymbol{V}_i^t$) and IPI($\boldsymbol{X}_i^t$) are calculated according to formula (13). The operator growth possesses of the property of "each-step-is-not-bad", ensuring the algorithm to converge globally [83,86–88].

2.4.3.2. Operator INIT. Suppose that the dimensionality of the search space of optimization problem (1) is n, the search interval for each variable is $[l_i, u_i]$, $i=1, 2, \ldots, n$, the constructing operator INIT of the orthogonal table $L_N(N^n)$ that uses the orthogonal Latin squares generating method [82] to produce N initial solutions is as follows:

2.4.3.2.1. Operator INIT

Step 1: Calculate discrete points for each variable y_{ij}:

$$y_{ij} = l_i + (j-1)(u_i - l_i)/(N-1), i = 1, 2, \ldots\ldots, n; j = 1, 2, \ldots\ldots, N$$

Step 2: Use the orthogonal Latin squares generating method to produce initial solution x_{ij}:

$$x_{ij} = y_{jk}, i = 1, 2, \ldots\ldots, N, j = 1, 2, \ldots\ldots, n$$

where $k=(i+j-1+m)$ mod N; if $k=0$, then $k=N$; if $j=1$ then $m=0$, else $m=n_0$; n_0 is a random number selected randomly from [1, N], namely $n_0=Rnd(1, N)$.

N initial solutions $\boldsymbol{X}_i=(x_{i1}, x_{i2}, \ldots, x_{in})$, $i=1, 2, \ldots, N$, determined by operator INIT have good balance dispersion and neat comparability.

2.4.3.3. Operator REINIT. When an individual makes search, if the current position of the individual keeps unchanged for a long time, we think that the individual has stayed into sticky state. Two situations may cause sticky state: one is that an individual drops into a local optimum solution; the other is that current position of an individual cannot be updated by the information thrown out by other individuals.

When all individuals drop into stick state, search will stop automatically; global optimization solution will keep unchanged forever. On the other way, when some problems have high condition numbers, enhancing precision of global optima is always difficult when individuals easily drop into sticky state. At this time, a reinitialization–redistribution of individuals is necessary.

Suppose from time t_0 to time t, global optimization solution keeps unchanged, we say that evolution has dropped into stick state. Reinitialization–redistribution of individuals will make search to escape from stick state or get high precision global optima. The operator of reinitialization–redistribution of individuals (Operator REINIT) is designed as follows:

2.4.3.3.1. Operator REINIT. If $t-t_0 > t_{\max}$, where $t_{\max}$ is the allowable maximum periods that global optimization solution keeps unchanged, then execute the following operations:

Step 1: (Decrease the interval of each decision variable): Decrease the interval LU_i of decision variable x_i ($i=1, 2, \ldots, n$) by $LU_i=\rho LU_i$, where ρ is the shrinking factor of the interval of decision variable x_i, $0<\rho<1$; LU_i is its change limit. At start, $LU_i=\frac{u_i-l_i}{2}$; afterwards, if $LU_i<LU_0$, then $LU_i=\frac{u_i-l_i}{2}$, LU_0 is the lower bound of interval shrinkage; the smaller LU_0 is, the narrower the scope of search is, and the higher the precision of optimum solutions will be, but the scope of search may not cover global optimum solutions.

Step 2: (Operation reinitialization: Reinitialize a very big number of individuals): Take each variable x_i^* ($i=1, 2, \ldots, n$) of the current global solution $\boldsymbol{X}^*=(x_1^*, x_2^*, \cdots, x_n^*)^T$ as center, namely for x_i^* ($i=1, 2, \ldots, n$), its change extent is within $[x_i^*-LU_i, x_i^*+LU_i]$, then take $[x_i^*-LU_i, x_i^*+LU_i]$ as the interval of variable x_i of individual $\boldsymbol{X}_i$, use the above-mentioned operator INIT to initialize all N_0 individuals; N_0 can assign a very larger number, for example, $N_0=2000$ or even much bigger number.

Step 3: (Operation redistribution: Redistribute a small number of individuals): Find the new best individual $\boldsymbol{X}^{*0}=(x_1^{*0}, x_2^{*0}, \cdots, x_n^{*0})^T$ from the just initialized N_0 Individuals; then redistribute N individuals within $[x_i^{*0}-LU_i, x_i^{*0}+LU_i]$ by the above-mentioned algorithm INIT, where N is the number of work individuals used by SEIQRA, $N \ll N_0$, for example $N=50$–200.

Step 4: (Replace the individuals with lower IPI values according to ratio r): Replace the current individuals whose IPI values are very low with the newly-generated individuals whose IPI values are very high, the ratio of replacement is r. For example, when $r=0.5$, it means that 50% of the current individuals, whose IPI values are very low among all the current individuals, are replaced with 50% of the newly-generated individuals, whose IPI values are very high among all the newly-generated individuals.

Reinitialization of a very big number of individuals is used to find the approximate position of global optima; in the vicinity of the approximate position of global optima, redistribution of a small number of work individuals is easy to find the global optima quickly and accurately by the algorithm SEIQRA.

Conclusively, operation reinitialization is good at enhancing precision of an optimum solution, while operation redistribution is good at breaking sticky state of an individual.

2.5. Construction method of SEIQRA

Based on the logical structure of Algorithm SEIQRA mentioned in Section 2.3, we can design a simple structure of SEIQRA as follows:

Algorithm SEIQRA

Initialize all the parameters SEIQRA involves as specified in Table 3;
FOR $t=1$ **TO** G //Where t is used to count generations or periods of evolution; G is the maximum generations or periods of evolution.
FOR each individual i
Step 1: Calculate the state transition probability of the individual by state transition Eqs. (10) and (11);
Step 2: Determine which state s the individual may transfer to at time t based on its state transition probability computed by Step1;
Step 3: If state transition from time $t-1$ to time t is legal, then carry out information exchange by transferring some features of some randomly-selected individuals at time $t-1$ to the corresponding features of the individual at time t; else the state of the individual keeps unchanged.
Step 4: Compute the IPI value of the individual by formula (13), if its IPI value is improved when comparing to that at time $t-1$, then the individual evolves to the new state s at time t; else the individual still stay at its old state at time $t-1$.
END FOR
Operator REINIT is activated at certain frequency;
END FOR

The psuedo-code of SEIQRA can be seen in Appendix A. The strategy of taking values of the parameters in SEIQRA is shown in Table 3.

2.6. Time complexity of SEIQRA

The time complexity of SEIQRA is evaluated as shown in Table 4, it has relations with the maximum evolution times G, the number of individuals N, the number of variables n, the time complexity of operators: S–S, S–E, S–R, E–E, E–I, E–R, I–I, I–Q, I–R, Q–Q, Q–R, R–R, R–S, Growth and REINIT, operations: *AVG*, *DE*, *EPN*, *CHV*, *RFL*, *CRS* and other auxiliary operations.

3. Performance study and parameter selection of SEIQRA

The computer used to test is a Toshiba notebook computer, its CPU is Intel Core™ I5, M520 @ 2.40 GHz, its memory size is 4 GB, its OS is Windows 7.

3.1. Performance study and parameter selection of SEIQRA

Now we use the Bump function $F_0(\boldsymbol{X})$ [35] and Michalewicz function $F_1(\boldsymbol{X})$ [35] as an example to test the performance of

Table 5
Properties of the Bump function $F_0(\boldsymbol{X})$ and Michalewicz function $F_1(\boldsymbol{X})$.

Property	Bump function $F_0(\boldsymbol{X})$	Michalewicz function $F_1(\boldsymbol{X})$
Optimization problem with constraints?	Yes	No
Multimodal?	Yes	Yes
Rotated and Shifted?	Yes, the original point (0,0,…,0) is not the global optimum solution.	Yes, the original point (0,0,…,0) is not the global optimum solution.
Domain of $\boldsymbol{X}$	$\boldsymbol{X}\in[0,\pi]^n$	$\boldsymbol{X}\in[0,10]^n$
Non-separable?	Yes	No
Global optimum is known?	No, the global optimum objective function value varies with dimensionality n	No, the global optimum objective function value varies with dimensionality n
The function contains a cyclic function?	Yes	Yes
It can be solved easily?	No	No

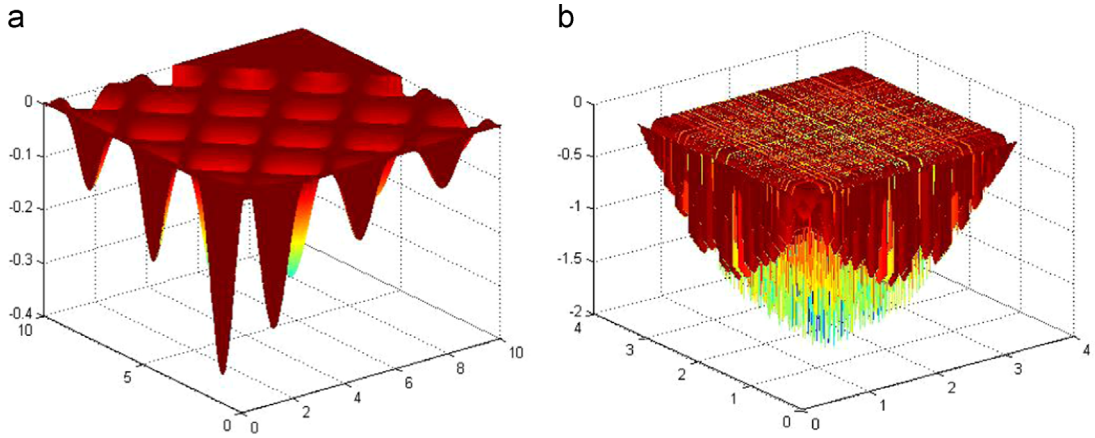

Fig. 5. The graphs of the Bump and Michalewicz function when $n=2$. (a) The Bump function; (b) Michalewicz function.

Table 6
Effect of parameter L when $n=50$, $E_0=0.06$, $N=200$, $N_0=2000$, $T=4$, $\lambda=1$, $\tau=1$, $G=1.0E+5$ and run times=20; REINIT is prohibited.

L	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
1	−0.827953281950753	1875	−49.5957355933971	502
2	−0.831606013670455	1922	−49.6091287496027	226
3	−0.831606013670455	1922	−49.6154517305820	1166
4	−0.832688629582492	1884	−49.6069852705723	307
5	−0.832970269937945	1981	−49.6148346625030	264
6	**−0.834826053656805**	**1922**	**−49.6161619228999**	**855**
7	−0.833787056092329	1908	−49.6101059078490	514
8	−0.833787056092329	1908	−49.6227683161410	1224
9	−0.834897470880660	1904	−49.6226863498109	262
10	−0.834647833760122	1917	−49.6176302033132	300

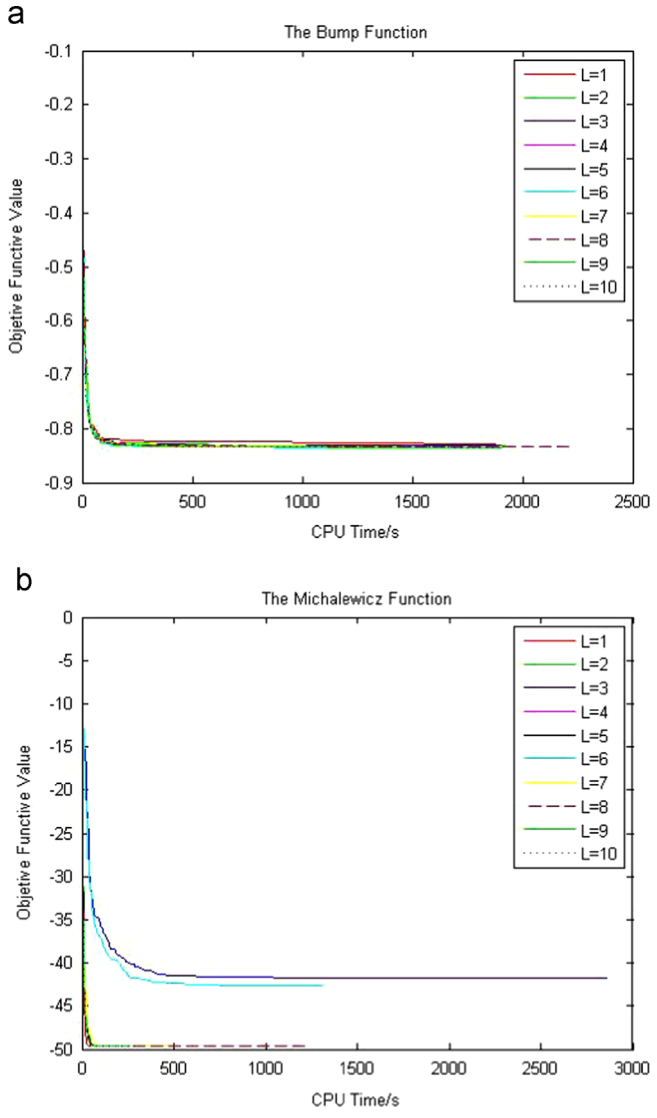

Fig. 6. Relation of objective function value with CPU time when L takes different values. (a) The Bump function; (b) The Michalewicz function.

SEIQRA, and determine the setting of some key parameters. The mathematical models of $F_0(\boldsymbol{X})$ and $F_1(\boldsymbol{X})$ are as follows:

Table 7
Effect of parameter E_0 when $n=50$, $L=6$, $N=200$, $N_0=2000$, $G=1.0E+5$, $T=4$, $\lambda=1$, $\tau=1$, and run times=20; REINT is prohibited.

E_0	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
0.001	−0.687779446604573	898	−49.6155521313138	707
0.002	−0.718737770254323	904	−49.6228212529016	712
0.004	−0.760642273836502	1091	−49.6228212529016	428
0.006	−0.790099822986605	1095	−49.6228212529016	488
0.008	−0.809000316375438	1161	−49.6228212529016	840
0.01	−0.815311674554719	1182	−49.6228212529016	346
0.02	−0.824890036082204	1349	−49.6227922678471	289
0.04	−0.830856211441936	1618	−49.6228212529017	272
0.06	**−0.833803362869570**	**1908**	**−49.6228122188572**	**217**
0.08	−0.831795692900186	2149	−49.6132970801741	217
0.1	−0.828502259790528	2332	−49.589768503263	1719
0.2	−0.818150482452084	3697	−48.8793535376783	396
0.4	−0.697848195144567	6495	−47.7219912917489	1350
0.6	−0.741233476291840	3921	−47.583935279112	1412
0.8	−0.687309180877045	1937	−47.0371451853083	1234
1	−0.666689009262077	1413	−46.4085466749631	3421

Bump function:

$$\begin{cases} \min F_0(\boldsymbol{x}) = -\dfrac{\left|\sum_{i=1}^{n}\cos^4(x_i) - 2\prod_{i=1}^{n}\cos^2(x_i)\right|}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \\ \text{s.t.} \quad \prod_{i=1}^{n} x_i \geq 0.75 \\ \quad \sum_{i=1}^{n} x_i \leq 7.5n \\ \quad 0 < x_i \leq 10 \end{cases}$$

Michalewicz function:

$$\min F_1(\boldsymbol{X}) = -\sum_{i=1}^{n} \sin(x_i)\left[\sin\left(\frac{ix_i^2}{\pi}\right)\right]^{2m}, (m=15)$$

The two functions are carefully selected because they have such properties as shown in Table 5.

The reason we select the Bump and Michalewicz function to determine the setting of parameters in SEIQRA is that properties of the two functions have a typical representative in engineering applications, and hence the generalized configuration of

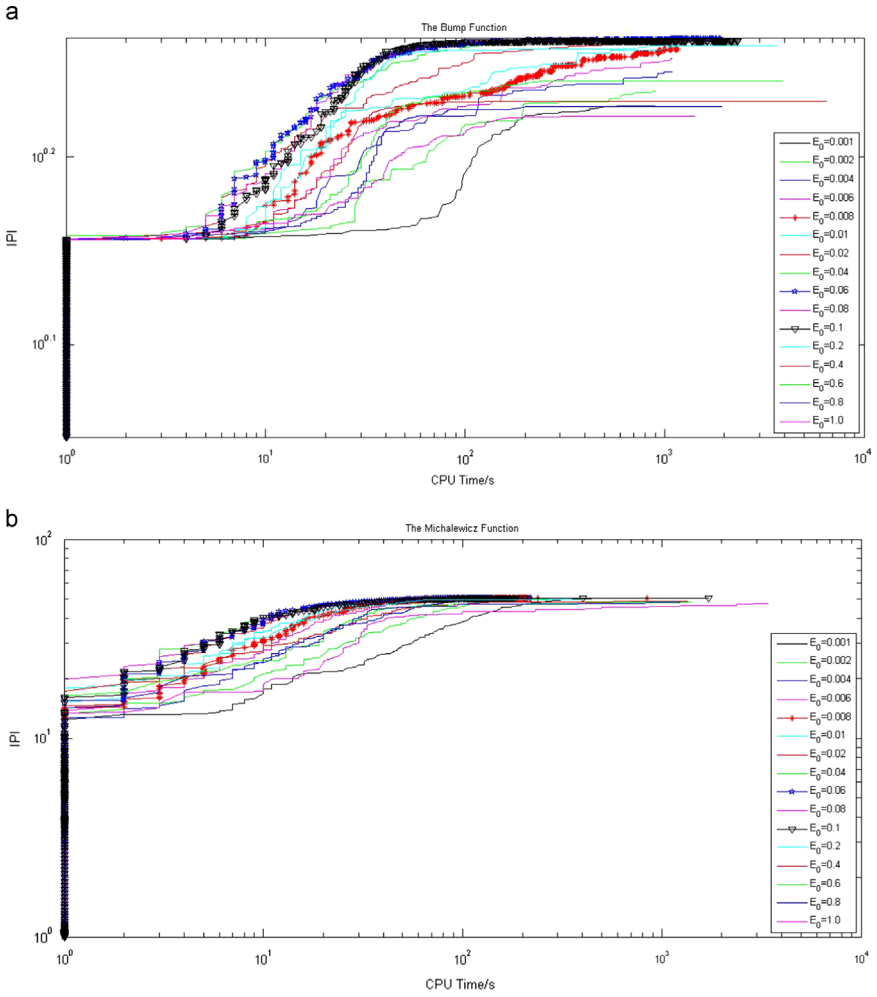

Fig. 7. Relation of IPI value with CPU time when E_0 takes different values. (a) The Bump function; (b) The Michalewicz function.

parameters in SEIQRA can be approximately found by the two functions.

Fig. 5 is the graphs of the Bump and Michalewicz function when $n=2$. When n is greater, the Bump and Michalewicz function are very difficult to optimize.

To make clear that the setting of parameters in SEIQRA gives influence on its evolving process when it solves the Bump and Michalewicz function, a deep investigation is made into the important parameters such as L, E_0, N, T, τ and λ.

Table 6 describes the relationship among average best objective function value, parameter L and average CPU time; Fig. 6 shows that relation of objective function value with CPU time when L takes different values. Table 6 and Fig. 6 show that when L is valued differently, for the Bump function there is little difference in CPU time, for the Michalewicz function there is great difference in CPU time for some Ls; but when $L>1$, there is little difference in precision of average optimum objective function values. Therefore, for SEIQRA, $L=2$–10 is more appropriate, we suggest $L=6$ is the best setting based on comprehensive consideration of precision of objective function value and CPU time.

Table 7 describes the relationship among parameter E_0, average best objective function value and average CPU time. It shows that for the Bump function, when $E_0=0.008$–0.2, the precision is relatively high, but CPU time increases moderately; when $E_0>0.2$, CPU time varies greatly, but the precision decreases greatly also; especially when $E_0=1$, it is unable to obtain the optimum solution; for the Michalewicz function, when $E_0=0.001$–0.1, the precision is relatively high, but CPU time varies greatly; when $E_0>0.1$, CPU time varies greatly, but the precision decreases greatly also; especially when $E_0=1$, it is unable to obtain the optimum solution. Therefore, when $E_0=0.008$–0.1, SEIQRA performs best. When $E_0=0.008$–0.1, it means that there are at most only 0.8%–10% variables to be involved in computation, but the CPU time consumed by SEIQRA is lower than that when E_0 takes other values,

Table 8
Relation of average best objective function value with average CPU time when N takes different values, and $n=50$, $L=6$, $E_0=0.06$, $N_0=2000$, $G=1.0\text{E}+5$, $T=4$, $\tau=1$, $\lambda=1$ and run times=20; REINT is prohibited.

N	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
50	−0.823492909251594	341	−49.4590588403967	37
100	−0.831597657955816	756	−49.6085383998470	57
150	−0.834587575202624	1287	−49.6041680740384	128
200	**−0.834904200815997**	**1889**	**−49.6207649262707**	**115**
250	−0.834368405804599	2464	−49.6193576435355	186
300	−0.834107295117368	4246	−49.6174447715844	238
350	−0.833079266813661	4228	−49.6227683161410	260
400	−0.831765847386432	5358	−49.6221674761665	414
450	−0.832726350044014	6453	−49.6228212529016	608
500	−0.831655895034941	7611	−49.6215008404875	1113

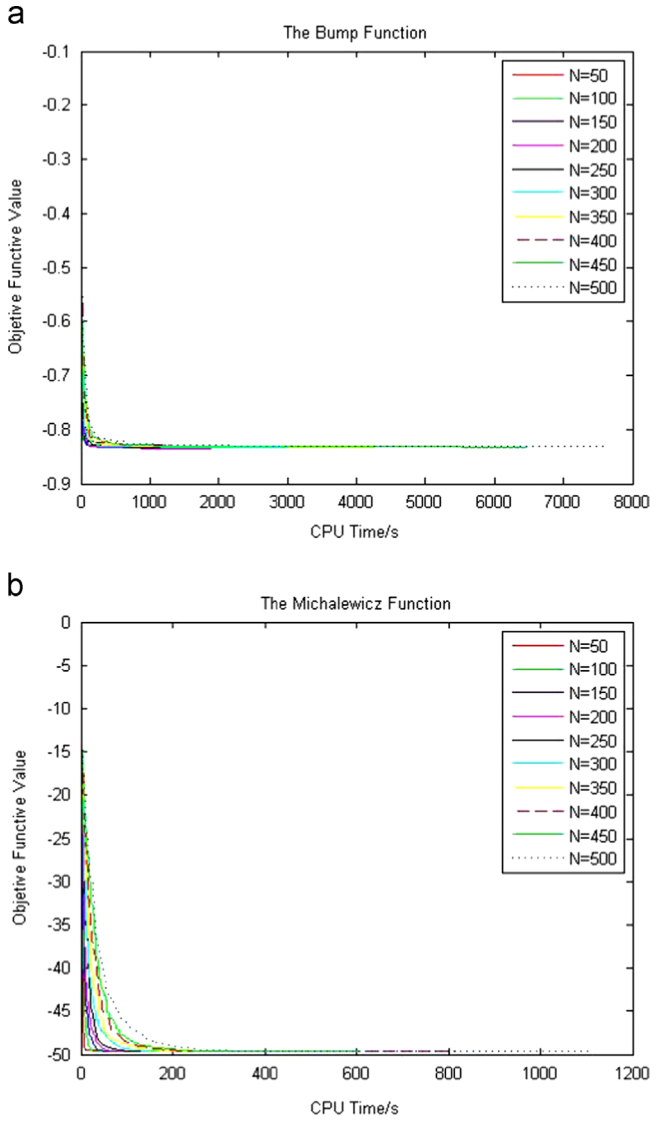

Fig. 8. Relation of objective function value with CPU time when N takes different values. (a) The Bump function; (b) The Michalewicz function.

Table 9
Relation of average best objective function value with average CPU time under different vaccination cycles T when $n=50$, $E_0=0.06$, $N=200$, $N_0=2000$, $L=6$, $\tau=1$, $\lambda=1$, $G=80{,}000$ and run times=20; REINT is prohibited.

T	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
2	−0.832266925553587	375	−49.6169127665783	188
4	**−0.829913555844357**	**372**	**−49.6220370470490**	**226**
6	−0.832552777450192	373	−49.6212884223517	176
8	−0.831113846663504	370	−49.6200359088558	286
10	−0.831731827910368	370	−49.6171262764610	159
12	−0.831162154102009	372	−49.6215181903295	180
14	−0.831960998149279	296	−49.6172562470701	268
16	−0.830513944599602	372	−49.6209997605222	204
18	−0.831892355828711	371	−49.6224605571573	306
20	−0.829500740118085	375	−49.6211005825003	306

SEIQRA can still obtain the optimum solution with relatively high precision.

Table 10
Relation of average best objective function value with average CPU time under different latency periods τ when $n=50$, $E_0=0.06$, $N=200$, $N_0=2000$, $L=6$, $T=4$, $\lambda=1$, $G=80{,}000$ and run times=20; REINT is prohibited.

τ	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
1	**−0.832264972912071**	**374**	**−49.6226411044064**	**208**
2	−0.832089019848813	370	−49.6195936998853	194
3	−0.832106201210614	371	−49.619302645638	196
4	−0.830133500000392	369	−49.6202190684598	200
5	−0.831524847058460	367	−49.6150659802303	317
6	−0.829945399638853	371	−49.6149880010193	135
7	−0.832489306938715	363	−49.621078127580	203
8	−0.832031698894371	354	−49.6162557853017	284
9	−0.830629150090071	288	−49.6164115369954	251
10	−0.829828723701112	371	−49.6125735983749	147

Table 11
Relation of average best objective function value with average CPU time under different immunity periods λ when $n=50$, $E_0=0.06$, $N=200$, $N_0=2000$, $L=6$, $T=4$, $\tau=1$, $G=80{,}000$ and run times=20; REINT is prohibited.

λ	The Bump function		The Michalewicz function	
	Average best objective function value	Average CPU time/s	Average best objective function value	Average CPU time/s
1	**−0.829709813549924**	**373**	**−49.616733008549**	**162**
2	−0.832503794330900	370	−49.6160884750357	237
3	−0.827466380640695	369	−49.6190711754889	133
4	−0.830384696699006	369	−49.6108273829626	238
5	−0.82998943759237	367	−49.6147830621168	212
6	−0.829456526977736	367	−49.6170709900992	174
7	−0.829905775418647	505	−49.6165828146807	317
8	−0.830430239230583	438	−49.6205720290063	262
9	−0.830675022518777	499	−49.6179419076148	189
10	−0.828715260929138	457	−49.6189395939285	232

Fig. 7, which is logarithmic scales on vertical and horizontal axis, shows the relation of IPI (objective function value can be transformed into IPI value with formula (13)) with CPU time when E_0 takes different values. From Fig. 7 we can see that when E_0 increases from 0.0001 to 1, the ascending rate of IPI value, which corresponds to the descending rate of objective function value, increases greatly, but the precision of objective function value decreases moderately; from which we can find that the most suitable value for E_0 is $E_0=0.06$.

Table 8 describes the relation among parameter N, average best objective function value and average CPU time consumed by SEIQRA; Fig. 8 shows the relation of objective function value with CPU time when N takes different values. From Table 8 and Fig. 8, we can see that for average objective function value, when N increases, the precision of the average objective function value increases a little; for average CPU time, when N increases, the average CPU time increases greatly. Therefore, $N=150$–350 is the most suitable setting, we suggest $N=200$ is the suitable setting based on comprehensive consideration of precision of objective function value and CPU time.

Tables 9–11 show relation of average objective function value with average CPU time under different vaccination cycles, latency periods and immunity periods respectively; Figs. 9–11 show relation of objective function value with CPU time under different

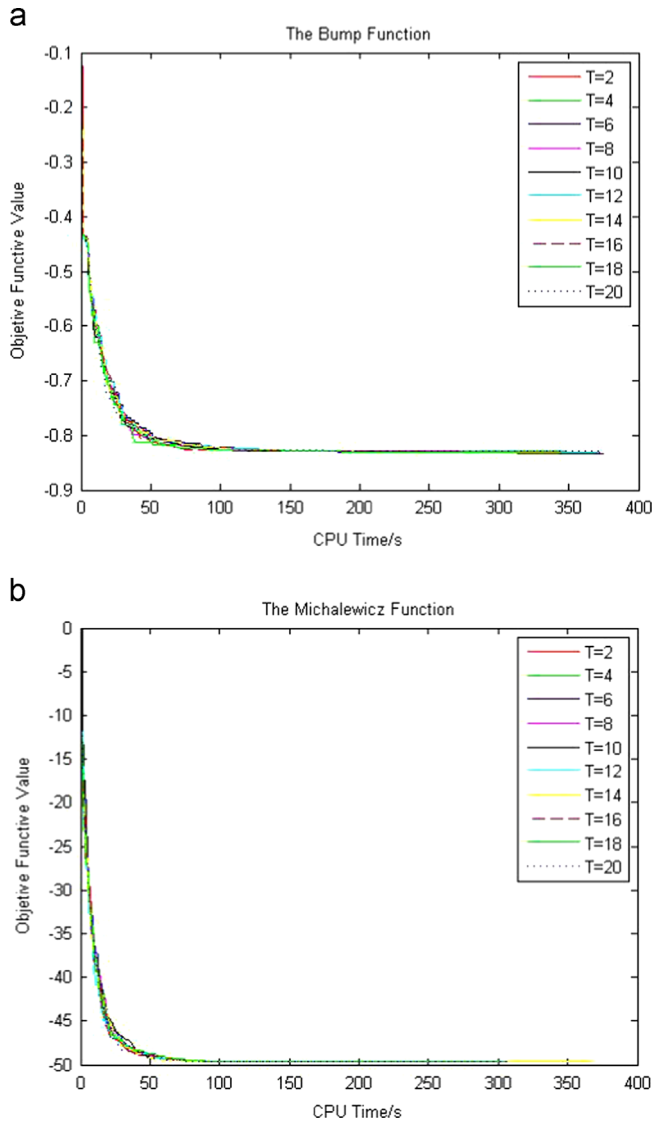

Fig. 9. Relation of objective function value with CPU time when T takes different values. (a) The Bump Function; (b) The Michalewicz function.

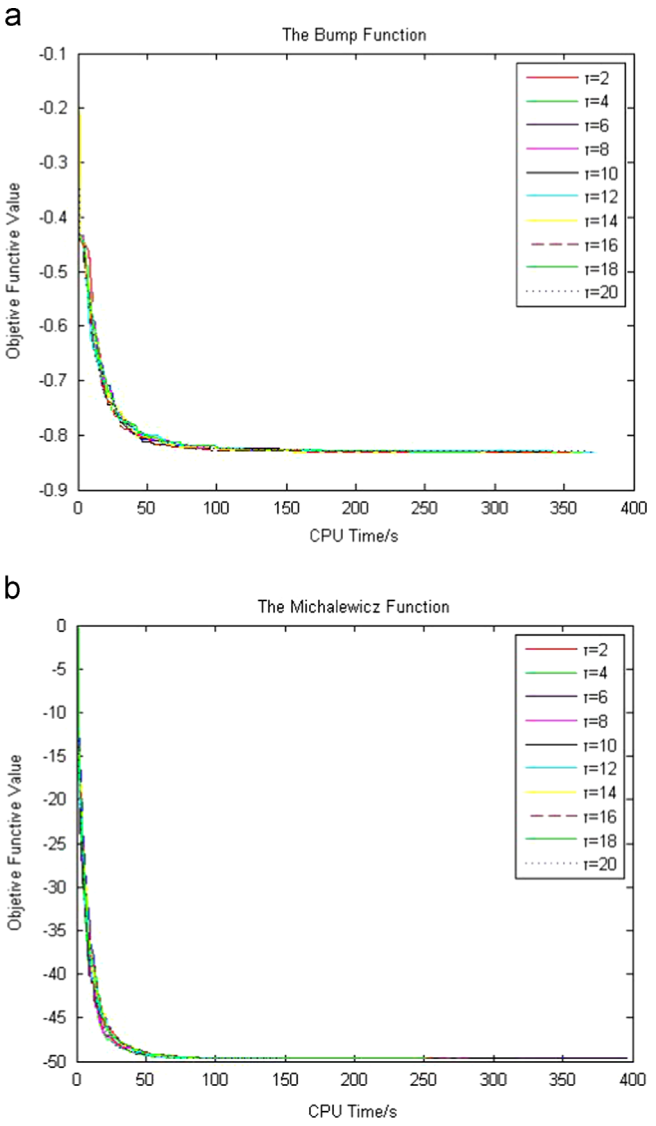

Fig. 10. Relation of objective function value with CPU time when τ takes different values. (a) The Bump Function; (b) The Michalewicz function.

vaccination cycles, latency periods and immunity periods respectively. From Tables 9–11 and Figs. 9–11 we can see that for different vaccination cycles, latency periods and immunity periods, average objective function value and average CPU Time do not change greatly. The reason is that the 6 operations are executed equiprobably, the time consumption is almost the same for different vaccination cycles, latency periods and immunity periods. We suggest $T=4$, $\tau=1$ and $\lambda=1$ are the best setting based on comprehensive consideration of precision of objective function value and CPU time.

Since in an ecosystem, β^t, γ_1^t, γ_2^t, γ_3^t, p^t, δ^t and ρ^t of population fluctuate with time, therefore, optimum objective function value obtained by SEIQRA always fluctuates in the vicinity of theoretical optimum objective function value. In order to increase precision of optimal objective function value, each setting of parameters should be made reasonably and allow SEIQRA to run many times.

Table 12 shows that relation among n (the number of variables of optimization problem (1)), average best objective function value and average CPU time. From Table 12 we can see that when n increases, the consumed CPU time increases greatly, but the precision of objective function value decreases a little also. Fig. 12 describes the convergence graphs when n takes different values. Owing to the "each-step-is-not-bad" search strategy, the convergence process of SEIQRA is fairly smooth [83,86–88].

3.2. Analysis of parameter setting

The parameters in SEIQRA are classified into three classes, namely

(1) Class A: the basic parameters a population-based intelligent optimization algorithms may possess of: G (or ε) and N;
(2) Class B: the parameters associated with the SEIQRA epidemic model: L, E_0, T, τ and λ;
(3) Class C: the parameters associated with operator REINIT: N_0, t_{max}, ρ, LU_0, r;

The parameters in Class A is a basis; the parameters in Class B is very important for SEIQRA, but L, T, τ and λ produce little influence on precision of the best objective function value and CPU time, so L, T, τ and λ can take fixed values; the parameters in Class C is not

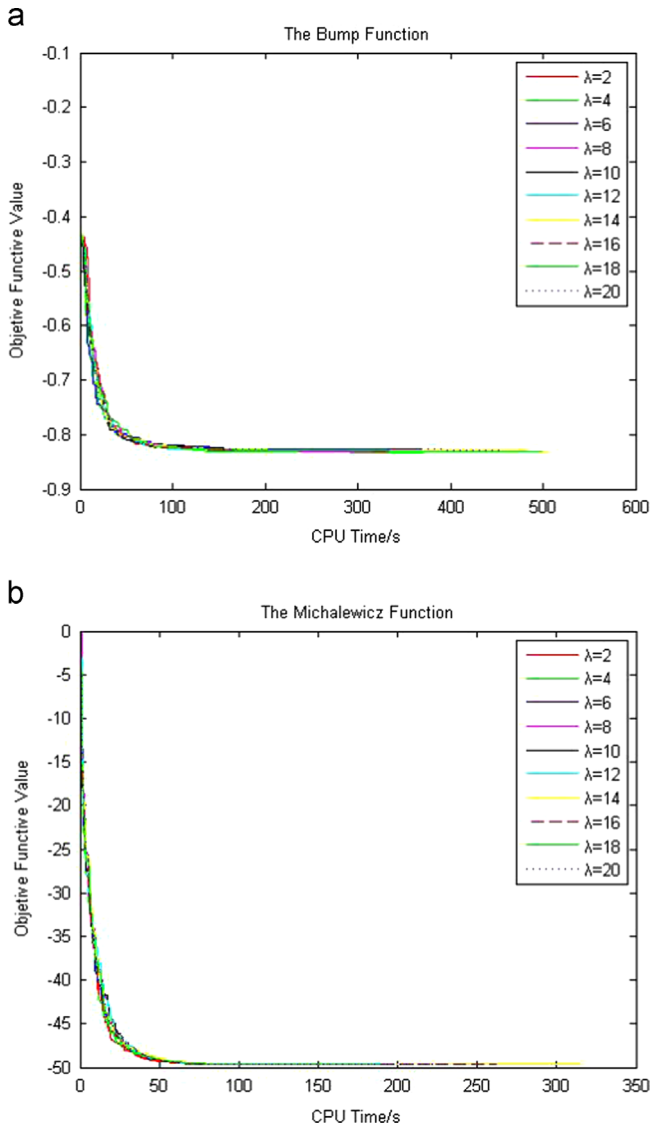

Fig. 11. Relation of objective function value with CPU time when λ takes different values. (a) The Bump Function; (b) The Michalewicz function.

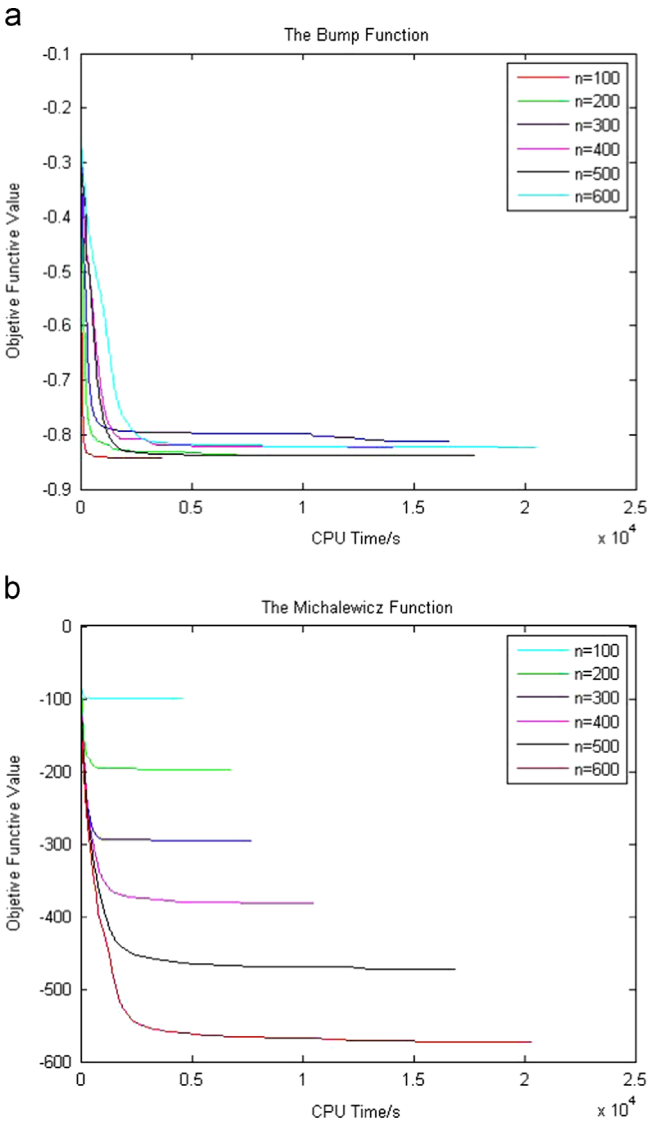

Fig. 12. The convergence process of SEIQRA. (a) The Bump Function; (b) The Michalewicz Function.

Table 12
Relation among n, N, average objective function value and average CPU time when $E_0=0.06$, $L=6$, $N_0=2000$, $G=1.0E+5$, $T=4$, $\tau=1$, $\lambda=1$ and run times=20; REINT is invoked.

n	The Bump Function		The Michalewicz Function	
	Average Objective Function Value	Average CPU Time/s	Average Objective Function Value	Average CPU Time/s
100	−0.842074788323724	3639	−99.5852089056127	4629
200	−0.835425654584833	7026	−197.590732022399	6764
300	−0.812782304160569	16,557	−296.228141716020	7647
400	−0.823113677539460	14,066	−382.365976205754	10,489
500	−0.838622479339227	17,710	−472.813854215453	16,892
600	−0.823781525014319	20,485	−573.012868088253	20,338

important for SEIQRA. If REINIT is prohibited, then the parameters in Class C do not need to be set.

The standard parameter configuration in SEIQRA is as follows:

(1) $G=1.0E+5$, $N=150$–350, generally $N=200$, ε is defined by the engineering problem to be solved.
(2) Always set $L=6$, $T=4$, $\tau=1$ and $\lambda=1$.
(3) Set $E_0=0.06$(for low dimensional problems) or $E_0=0.001$(for high dimensional problems). Conclusively, when SEIQRA solves a problem, only does 1/10–1/1000 of variables of the problem need to take part in computation in each iteration.
(4) Always set $N_0=2000$, $t_{max}=30$, $\rho=0.8$, $LU_0=\varepsilon$, $r=0.8$.

If we want to enhance the probability of finding global optima, we only increase N_0, for example, set $N_0=20{,}000$, the probability of finding global optima will be increased greatly; if we want to enhance the precision of global optima, we can only decrease LU_0 to much lower level than ε.

4. Scalability of SEIQRA

We use 28 benchmark functions listed in CEC'2013 [85] to test the scalability of SEIQRA, these benchmarks are F1–F28, as shown in Table 13. The benchmark functions that we minimized are functions that are representative of those used in the literature of performance analysis of populations-based optimization algorithms. More information about these functions can be found in [85].

Table 13
28 Benchmark function optimization problems [85].

Benchmark function		Range of each variable	Theoretical global optimum solution	Theoretical global optimum objective function value		
				$D=10$	$D=30$	$D=50$
F1	Sphere function	[−100,100]	$\boldsymbol{O}$	−1400	−1400	−1400
F2	Rotated high conditioned elliptic function	[−100,100]	$\boldsymbol{O}$	−1300	−1300	−1300
F3	Rotated Bent cigar function	[−100,100]	$\boldsymbol{O}$	−1200	−1200	−1200
F4	Rotated Discus function	[−100,100]	$\boldsymbol{O}$	−1100	−1100	−1100
F5	Different Powers function	[−100,100]	$\boldsymbol{O}$	−1000	−1000	−1000
F6	Rotated Rosenbrock function	[−100,100]	$\boldsymbol{O}$	−900	−900	−900
F7	Rotated Schaffers F7 function	[−100,100]	$\boldsymbol{O}$	−800	−800	−800
F8	Rotated Ackley's function	[−100,100]	$\boldsymbol{O}$	−700	−700	−700
F9	Rotated Weierstrass function	[−100,100]	$\boldsymbol{O}$	−600	−600	−600
F10	Rotated Griewank's function	[−100,100]	$\boldsymbol{O}$	−500	−500	−500
F11	Rastrigin's function	[−100,100]	$\boldsymbol{O}$	−400	−400	−400
F12	Rotated Rastrigin's function	[−100,100]	$\boldsymbol{O}$	−300	−300	−300
F13	Non-continuous rotated Rastrigin's function	[−100,100]	$\boldsymbol{O}$	−200	−200	−200
F14	Schwefel's function	[−100,100]	Unknown	Unknown (maybe −804.4948)	Unknown (maybe −2262.9856)	Unknown (maybe −3721.4764)
F15	Rotated Schwefel's function	[−100,100]	Unknown	Unknown (maybe −604.4948)	Unknown (maybe −2062.9856)	Unknown (maybe −3521.4764)
F16	Rotated Katsuura function	[−100,100]	$\boldsymbol{O}$	200	200	200
F17	Lunacek bi-Ratrigin function	[−100,100]	$\boldsymbol{O}$	300	300	300
F18	Rotated Lunacek bi-Ratrigin function	[−100,100]	$\boldsymbol{O}$	400	400	400
F19	Rotated Expanded Griewank's plus Rosenbrock's function	[−100,100]	$\boldsymbol{O}$	500	500	500
F20	Rotated Expanded Scaffer's F6 function	[−100,100]	$\boldsymbol{O}$	600	600	600
F21	Composition function 1	[−100,100]	$\boldsymbol{O}$	900	900	900
F22	Composition function 2	[−100,100]	Unknown	Unknown (maybe 440)	Unknown (maybe −498.2879)	Unknown (maybe −1965.7202)
F23	Composition function 3	[−100,100]	Unknown	Unknown (maybe 390)	Unknown (maybe −1200.3297)	Unknown (maybe −2721.4763)
F24	Composition function 4	[−100,100]	Unknown	Unknown (maybe 800)	Unknown (maybe 1200)	Unknown (maybe 631.2470)
F25	Composition function 5	[−100,100]	Unknown	Unknown (maybe 1200)	Unknown (maybe 1200)	Unknown (maybe 1200.000)
F26	Composition function 6	[−100,100]	Unknown	Unknown (maybe 1000)	Unknown (maybe 650)	Unknown (maybe 294.1417)
F27	Composition function 7	[−100,100]	Unknown	Unknown (maybe 1600)	Unknown (maybe 1600)	Unknown (maybe 1600.000)
F28	Composition function 8	[−100,100]	Unknown	Unknown (maybe 1700)	Unknown (maybe 1700)	Unknown (maybe −4744.3417)

In Table 13, D is dimensions of an optimization problem, here $D=n$; $\boldsymbol{O}$ is an n-dimensional decision vector. Table 14 shows biases of theoretical global optimum objective function values of F1–F28. We use SEIQRA to solve 28 benchmark functions listed in Table 13, the results are shown in Table 15, the setting of parameters used in SEIQRA are $D=n=10, 30, 50$, $\varepsilon=1.0\text{E}-12$, $E_0=0.06$, $L=6$, $T=4$, $\tau=1$, $\lambda=1$, $N=200$, $N_0=2000$, $t_{\max}=30$, $\rho=0.8$, $LU_0=\varepsilon$, $r=0.8$, $G=2.0\text{E}+5$ and run times=20; parameter$\boldsymbol{M}1$ and $\boldsymbol{M}2$ in F1–F28 are set according to the method mentioned in article [85]; For convenience of analysis, let $\boldsymbol{O}$ is randomly generated; REINIT is invoked for some benchmark functions.

From Tables 15–17 we can see that SEIQRA can solve F1–F28 when $n=10$, 30 and 50, it means the algorithm has good scalability.

5. Comparison of SEIQRA with other population-based optimization algorithms

5.1. Comparison study of SEIQRA

We choose another 7 population-based optimization algorithms to make comparison to solve F1–F28; these algorithms include Real Code Genetic Algorithm (RC-GA) [32], Differential Ant-Stigmergy Algorithm (DASA) [84], Non-parametric Particle Swarm Optimization (NP-PSO) [38], Biogeography-based Optimization (BBO) [42], Differential Evolution (DE) [89], Self-adaptive Differential Evolution (SaDE) [43], and Artificial Bee Colony (ABC) [47].

When calculating, the setting of parameters in the 7 population-based optimization algorithms is initialized according to Table 18.

The algorithms are run independently 20 times for each benchmark function and average best objective function value, standard deviation (STD), median of the best objective function values, CPU time for finding the best objective function value, and rank of each algorithm are reported in Table 19. The algorithms are ranked based on Rank 1 and Rank 2 in Table 19. Rank 1 is based on precision of average best objective function value; Rank 2 is based on precision of average best objective function value and CPU time. From Table 19 we can see that the ranks of SEIQRA, RC-GA, NP-PSO, BBO, DE, SaDE and ABC based on precision or precision and CPU time are as follows:

$$\text{SEIQRA} > \text{SaDE} > \text{DE} > \text{ABC} > \text{NP}-\text{PSO} > \text{DSDA} > \text{RC}-\text{GA} > \text{BBO}$$

Also, the non-parametric Wilcoxon rank sum test [38,89] is conducted between the SEIQRA's result and the best results achieved by the other 7 algorithms listed in Table 18 for each benchmark function to determine whether the results generated by SEIQRA are statistically different from the results obtained by the other algorithms. With the help of the famous statistical

Table 14
Biases of theoretical global optimum objective function values of F1–F28.

Benchmark function	Bias of theoretical global optimum objective function value		
	$D=10$	$D=30$	$D=50$
F1	1400	1400	1400
F2	1300	1300	1300
F3	1200	1200	1200
F4	1100	1100	1100
F5	1000	1000	1000
F6	900	900	900
F7	800	800	800
F8	700	700	700
F9	600.0022	600.0022	600.0022
F10	500	500	500
F11	400	400	400
F12	300	300	300
F13	200	200	200
F14	804.4948	2262.9856	3721.4765
F15	604.4948	2062.9856	3521.4764
F16	−200	−200	−200
F17	−300	−300	−300
F18	−400	−400	−400
F19	−500	−500	−500
F20	−600	−600	−600
F21	−900	−900	−900
F22	−440	498.287	1965.7202
F23	−390	1200.3297	2721.4763
F24	−800	−1200	−631.2470
F25	−1200	−1200	−1200.000
F26	−1000	−650	−294.1417
F27	−1600	−1700	−1600
F28	−1700	-1500	4744.3417

Table 15
Result deviation of 28 benchmark functions solved by SEIQRA ($D=n=10$).

Benchmark function	Objective function value				CPU time/s			
	Best	Worst	Mean	Standard deviation	Smallest	Biggest	Mean	Standard deviation
F1	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	7	9	8	1.1920E−01
F2	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	37	45	41	4.7655E−01
F3	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	10	15	13	3.3065E−01
F4	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	607	635	621	1.7664E+00
F5	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	118	154	134	2.1151E+00
F6	**0.0000E+00**	5.2400E−02	2.8500E−02	3.6065E−03	443	487	467	3.0284E+00
F7	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	44	51	48	4.0527E−01
F8	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	27	34	30	5.0380E−01
F9	**0.0000E+00**	0.0000E+00	−1.2000E−03	1.4027E−04	127	146	136	1.2114E+00
F10	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	24	35	28	7.6681E−01
F11	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	14	18	16	3.0878E−01
F12	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	35	44	41	5.6196E−01
F13	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	10	17	14	4.0010E−01
F14	**0.0000E+00**	0.0000E+00	0.0000E+00	5.0842E−14	145	186	162	2.0160E+00
F15	**0.0000E+00**	0.0000E+00	0.0000E+00	2.5421E−14	39	47	42	3.2819E−01
F16	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	28	33	30	3.8753E−01
F17	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0.0000E+00
F18/	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	40	48	44	3.7347E−01
F19	**2.5710E−01**	4.7585E+00	2.3441E+00	3.3849E−01	113	126	119	9.7755E−01
F20	**0.0000E+00**	1.0756E+00	5.4020E−01	6.1315E−02	226	270	248	2.5082E+00
F21	**2.0000E+02**	2.0000E+02	2.0000E+02	0.0000E+00	60	68	63	4.5128E−01
F22	**1.2760E−01**	3.7539E+00	2.0645E+00	1.9783E−01	1610	1890	1760	1.5276E+01
F23	**7.4390E−01**	1.1188E+01	6.1489E+00	6.1107E−01	1610	1788	1702	1.0415E+01
F24	**2.0185E+01**	2.8724E+01	2.4406E+01	5.4742E−01	2610	2978	2792	2.3593E+01
F25	**1.7800E−02**	2.3698E+01	1.2443E+01	1.5730E+00	2100	2300	2205	1.3285E+01
F26	**1.4170E−01**	2.3177E+01	1.0148E+01	1.5937E+00	510	600	549	6.2267E+00
F27	**0.0000E+00**	7.8120E−01	3.4080E−01	4.9952E−02	1900	2300	2075	2.5577E+01
F28	**6.5155E+01**	1.6577E+02	1.0063E+02	6.1548E+00	8000	9000	8353	6.1171E+01

Table 16
Result deviation of 28 benchmark functions solved by SEIQRA ($D=n=30$).

Benchmark function	Objective function value				CPU time/s			
	Best	Worst	Mean	Standard deviation	Smallest	Biggest	Mean	Standard deviation
F1	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	20	25	23	2.9800E−01
F2	**0.0000E+00**	8.4800E−02	3.8500E−02	5.0514E−03	178	183	180	2.9784E−01
F3	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	78	82	80	2.6452E−01
F4	**0.0000E+00**	8.9600E−02	4.5800E−02	5.6526E−03	1114	1134	1124	1.2617E+00
F5	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	208	223	215	8.8131E−01
F6	**0.0000E+00**	7.1700E−02	3.9000E−02	4.9349E−03	281	293	288	8.2592E−01
F7	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	498	512	506	8.1054E−01
F8	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	170	183	176	9.3564E−01
F9	**0.0000E+00**	−2.2000E−03	−1.0000E−03	1.4027E−04	82	93	87	7.0136E−01
F10	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	71	80	75	6.2739E−01
F11	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	8	12	10	3.0878E−01
F12	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	177	193	188	9.9904E−01
F13	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	57	65	61	4.5725E−01
F14	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	176	185	180	4.4253E−01
F15	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	76	83	78	2.8717E−01
F16	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	269	283	275	1.0851E+00
F17	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	1	1	1	0.0000E+00
F18/	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	143	153	148	4.6683E−01
F19	**5.5170E−01**	2.4742E+01	1.1767E+01	1.8190E+00	598	613	605	1.1279E+00
F20	**1.2670E−01**	1.0756E+00	6.0330E−01	5.4092E−02	316	329	323	7.4107E−01
F21	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	309	321	314	6.7692E−01
F22	**−9.0000E−04**	2.1607E+00	1.1536E+00	1.1793E−01	12,319	12,593	12,465	1.4948E+01
F23	**0.0000E+00**	−4.4897E+01	−2.3236E+01	2.6269E+00	3179	3356	3271	1.0356E+01
F24	**−8.5270E−01**	2.3780E−01	−3.1360E−01	6.9914E−02	7320	7410	7364	5.7700E+00
F25	**1.7800E−02**	5.4738E+01	2.8730E+01	3.6349E+00	2619	2703	2663	5.5798E+00
F26	**1.4170E−01**	9.2781E+00	3.5569E+00	3.4264E+00	5688	5876	5770	1.3007E+01
F27	**1.0720E−01**	1.3459E+01	5.9322E+00	8.5373E−01	4801	4903	4846	6.5222E+00
F28	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	16,023	16,745	16,278	4.4165E+01

Table 17
Results of 28 benchmark functions solved by SEIQRA ($D=n=50$).

Benchmark function	Objective function value				CPU time/s			
	Best	Worst	Mean	Standard deviation	Smallest	Biggest	Mean	Standard deviation
F1	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	50	54	51	1.1048E+00
F2	**0.0000E+00**	8.4800E−02	6.5300E−02	2.3556E−02	11,604	11,677	11,628	2.1543E+01
F3	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	194	199	197	1.3809E+00
F4	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	7711	7723	7719	3.5904E+00
F5	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	1348	1353	1351	1.3809E+00
F6	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	2268	2278	2274	2.7619E+00
F7	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	350	355	353	1.3809E+00
F8	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	157	162	160	1.6571E+00
F9	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	485	491	489	1.4235E+00
F10	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	243	250	246	1.9333E+00
F11	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	58	61	60	8.2860E−01
F12	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	453	460	456	1.8392E+00
F13	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	322	331	326	2.4857E+00
F14	**1.0000E−05**	0.0000E+00	0.0000E−04	4.4714E−08	230	235	232	1.2812E+00
F15	**0.0000E+00**	0.0000E+00	0.0000E+00	5.4464E−08	185	190	188	1.1810E−01
F16	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	1239	1245	1242	1.6571E+00
F17	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0.0000E+00
F18/	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	269	278	274	2.4857E+00
F19	**2.5710E−01**	4.7585E+00	1.1733E+00	2.9003E+00	15,543	15,712	15,665	4.6676E+01
F20	**3.2770E−01**	1.0756E+00	9.7850E−01	2.0660E−01	3847	3872	3860	6.9046E+00
F21	**0.0000E+00**	0.0000E+00	0.0000E+00	0.0000E+00	703	714	711	3.0381E+00
F22	**−8.1980E−01**	1.0392E+01	0.0000E+00	1.1563E+01	4104	5274	4187	3.2314E+02
F23	**−9.8350E−01**	8.1736E+00	0.0000E+00	1.2205E+01	674	712	683	1.0495E+01
F24	**3.5650E+02**	3.5792E+02	3.5742E+02	1.1659E+00	11,046	11,287	11,152	6.6561E+01
F25	**5.9018E+01**	6.4738E+01	6.2533E+01	1.5797E+00	2231	2276	2249	1.2428E+01
F26	**0.0000E+00**	1.0351E+00	4.9730E−01	3.9630E−01	6101	6159	6137	1.6019E+01
F27	**4.0176E+00**	1.2781E+01	1.1446E+01	2.4204E+00	1759	1781	1765	6.0761E+00
F28	**−7.8500E−01**	1.6261E+00	0.0000E+00	4.4910E−01	110,212	110,786	110,475	1.5853E+02

software package SPSS (version 19.0) and Excel (version 2003) which is used to transfer z-value into p-value, the Wilcoxon rank sum test is achieved, the results are shown in Table 20, where h-value=1 indicates that the performances of SEIQRA is statistically different with 99% certainty, h-value=−1 represents that the compared algorithm are significantly better than SEIQRA, and h-value=0 denotes that the results of the two considered algorithms are not significantly different. In Table 20, rows 1(Better), 0 (Same),

Table 18
The setting of parameters in the 7 population-based optimization algorithms.

Optimization algorithm	Parameters
RC-GA [32]	The number of chromosomes $N=100$, mutation probability=0.01, the number of parents=0.5 N, $G=100{,}000$
DASA [84]	The number of ants $m=30$, discrete base $b=10$, pheromone decay rate $\rho=0.2$, global scale-increasing factor $s_+=0.02$, global scale-decreasing factor $s_-=0.01$, the maximum precision of variables $\varepsilon=1.0e\text{-}15$, $G=300{,}000$
NP-PSO [38]	$N=100$, $G=100{,}000$
BBO [42]	Habitat modification probability=1, immigration probability bounds per gene=[0, 1], step size for numerical integration of probabilities=1, maximum immigration and migration rates for each island=1, and mutation probability=0.02, $N=200$, elitism=2, $G=100{,}000$
DE [89]	Weighting factor $F=0.5$, crossover constant $CR=0.9$, $N=100$, $G=1.0E+5$
SaDE [43]	Interval of weighting factor=[0.45,0.55]; crossover constant=[0.85,0.95], $N=100$, $G=1.0E+5$
ABC [47]	Employed bees or onlookers=300, trying times=300 n, $G=8.0E+5$

and −1 (Worse) give the number of benchmark functions that SEIQRA performs significantly better than, almost the same as, and significantly worse than the compared algorithm, respectively.

From Table 20 we can know that SEIQRA performs significantly better than the other 7 algorithms.

Appendix B illustrates the sample convergence curves of SEIQRA, RC-GA, NP-PSO, BBO, DE, SaDE and ABC when they solve benchmark F1–F28. In order to highlight the change of these sample convergence curves, the horizontal and vertical axis are illustrated with logarithmic scale.

5.2. Discussion

5.2.1. Dynamic behaviors of operators in SEIQRA

Analysis of dynamic behaviors of operators in SEIQRA is as follows:

(1) Dynamic behaviors of the ecological operators: S–S, S–E, S–R, E–E, E–I, E–R, I–I, I–Q, I–R, Q–Q, Q–R, R–R and R–S. Fig. 13 describes that the relationship between times of state transition S→S, S→E, S→R, E→E, E→I, E→R, I→I, I→Q, I→R, Q→Q, Q→R, R→R and R→S being triggered and CPU time when SEIQRA solves F1. It shows that triggered times of these state transitions vary stochastically with time, but the average triggered times are stable for each state transition because the average triggered times approximate to level for all state transitions. Because each state transition corresponds to an operator, these ecological operators are triggered evenly.

(2) The heart rhythm of SEIQRA. SEIQRA possesses of 5 states which divide a population into 5 classes dynamically and automatically. The number of individuals in each class varies with time. The dynamic change of the number of individuals in 5 classes can be thought as the heart beating of SEIQRA, or the heart rhythm of SEIQRA. Fig. 14 describes that the relationship of the number of individuals staying at state S, E, I, Q and R with CPU time when SEIQRA solves F1. It shows that the number of individuals staying at state S, E, I, Q and R vary stochastically with time, but the average number of individuals staying at each state is also stable because the average number of individuals staying at each state approximates to level.

Conclusively, the stability of the heart rhythm of SEIQRA can result in equi-probably triggering of the ecological operators S–S, S–E, S–R, E–E, E–I, E–R, I–I, I–Q, I–R, Q–Q, Q–R, R–R and R–S, because each state transition is equivalent to a triggering of the corresponding operator.

The stability of the heart rhythm of SEIQRA is controlled by parameters β_0, γ_1, γ_2, γ_3, p and δ; when these parameters are set according to Table 3, the stability of the heart rhythm of SEIQRA is guaranteed.

5.2.2. Dynamic behaviors of operations in SEIQRA

SEIQRA possesses of 6 operations *AVG*, *DE*, *EPN*, *CHV*, *RFL* and *CRS*. The times of each operation being executed varies with time. Fig. 15 describes that the relationship of the times of each operation being executed with the CPU time when SEIQRA solves F1. It shows that the times of each operation being executed vary stochastically with time, but the average times of each operation being executed is also stable because the average times of each operation being executed approximates to level.

5.2.3. Dynamic behavior of operator REINIT in SEIQRA

The dynamic behavior of REINIT is controlled by N_0, t_{max}, ρ, LU_0 and r. REINIT is used to enhance the exploration and exploitation ability of SEIQRA. REINIT includes two operations: reinitialization and redistribution, as mentioned in Section 2.4.3.3.

At stage of operation reinitialization, a large number of individuals are used to reinitialize a search space, After reinitialization, if a new global optimum solution is found, it means that the activation of REINIT is effective, we call it the effective triggering point, the current global optimum solution is updated into the newly-found global optimum solution; else the current global optimum solution keeps unchanged. The property of REINIT is called as the exploitation ability of REINIT.

At stage of operation redistribution, a small number of individuals are used to reinitialize the search space determined by the current global optimum solution. After redistribution, a normal search is made by SEIQRA. The positions of the work individuals are always updated no matter whether the current global optimum solution is updated or not. Therefore sticky states of search may be broken. The property of REINIT is called as the exploration ability of REINIT.

During searching process of SEIQRA, the identification of effectiveness of REINIT is illustrated in Fig. 16, where ALM is the IPI convergence curve when SEIQRA solves an optimization problem; the series of points marked by circle is the assigned triggering points, namely at that time REINIT is invoked; point K is the starting time that REINIT is allowed to invoke; KN is a line constructed by use of the effective triggering points; point B is the first assigned triggering point, point C is the second assigned triggering point, …, point J is the last assigned triggering point; the series of the effective triggering points is marked by numbers 1, 2, 3, 4 and 5. Number 1 means the assigned triggering point C takes effect because the current global optimum solution is updated. Similarly, Numbers 2, 3, 4 and 5 mean that points D, E, G and I take effect respectively. Therefore, C, D, E, G and I are effective triggering points, while points B, F, H, J are ineffective triggering points.

It is noteworthy that all assigned triggering points can always change positions of work individuals. Therefore it can help a search to overcome sticky states.

Table 19
The minimization results of benchmark functions of 8 population-based optimization algorithms.

Benchmark function	SEIQRA	RC-GA	DSDA	NP-PSO	BBO	DE	SaDE	ABC
F1								
Average	**0.0000E+00**	2.5279E+02	8.5400E−02	6.5880E+00	4.2350E−01	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
Median	**0.0000E+00**	2.5122E+02	7.5700E−02	6.5823E+00	4.1110E−01	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
STD	0.0000E+00	3.2044E−01	4.1660E−03	5.2464E−03	3.8377E−05	0.0000E+00	0.0000E+00	0.0000E+00
Time/s	51	2538	147	6119	638	5	7	787
Rank1	1	8	5	7	6	1	1	1
Rank2	3	8	5	7	6	1	2	4
F2								
Average	**6.5300E−02**	8.3237E+06	2.8254E+06	3.1024E+08	1.0862E+07	6.3938E+00	5.3910E−01	3.0186E+08
Median	**4.3600E−02**	8.3237E+06	2.8254E+06	3.1024E+08	1.0862E+07	6.0822E+00	4.7850E−01	3.0186E+08
STD	3.2652E−03	1.8185E+01	1.7340E+01	1.9922E+01	1.9121E+01	4.1929E−03	3.5835E−03	1.5725E+01
Time/s	11,628	12,141	810	2053	1669	9551	5755	51
Rank1	1	5	4	8	6	3	2	7
Rank2	1	5	4	8	6	3	2	7
F3								
Average	**0.0000E+00**	2.3867E+01	5.6600E−01	1.6895E+07	3.5110E−01	**0.0000E+00**	**0.0000E+00**	2.4823E+04
Median	**0.0000E+00**	2.3841E+01	4.7530E−01	1.6895E+07	2.8070E−01	**0.0000E+00**	**0.0000E+00**	2.4817E+04
STD	0.0000E+00	4.0653E−03	3.4497E−03	1.6168E+01	3.9527E−03	0.0000E+00	0.0000E+00	1.4357E+01
Time/s	197	20,789	645	65	1103	53	43	1121
Rank1	1	5	5	8	4	1	1	7
Rank2	3	6	5	8	4	2	1	7
F4								
Average	**0.0000E+00**	9.5427E+04	7.9466E+02	4.1266E+03	2.9388E+03	**0.0000E+00**	**0.0000E+00**	4.9956E+04
Median	**0.0000E+00**	9.5429E+04	7.9456E+02	4.1270E+03	2.9386E+03	**0.0000E+00**	**0.0000E+00**	4.9956E+04
STD	0.0000E+00	2.1040E+01	3.7581E−03	4.1422E+00	3.7088E+00	0.0000E+00	4.0371E−05	2.1602E+01
Time/s	7719	58	5113	3427	3141	515	1178	297
Rank1	1	8	4	6	5	1	1	7
Rank2	3	8	4	6	5	1	2	7
F5								
Average	**0.0000E+00**	5.3927E+02	3.5660E−01	7.5855E+03	1.5110E−01	**0.0000E+00**	**0.0000E+00**	8.9469E+03
Median	**0.0000E+00**	5.3902E+02	3.6630E−01	7.5848E+03	2.1880E−01	**0.0000E+00**	**0.0000E+00**	8.9472E+03
STD	0.0000E+00	3.8820E−03	3.2402E−02	3.8064E+00	3.3413E−03	3.6289E−05	3.6234E−05	3.1642E+00
Time/s	1351	6205	1887	3598	773	1069	1086	80
Rank1	1	6	5	7	4	1	1	8
Rank2	3	6	5	7	4	1	2	8
F6								
Average	**0.0000E+00**	3.0151E+02	2.2495E+02	3.6243E+03	6.8390E+01	**0.0000E+00**	**0.0000E+00**	3.8696E+02
Median	**0.0000E+00**	3.0134E+02	2.2436E+02	3.6251E+03	6.8276E+01	**0.0000E+00**	**0.0000E+00**	3.8659E+02
STD	0.0000E+00	3.5841E−02	4.5944E−02	3.9308E+00	3.9877E−02	3.5218E−05	3.2570E−05	3.7471E−02
Time/s	2274	9999	244	3892	3276	1372	1058	760
Rank1	1	6	5	8	4	1	1	7
Rank2	3	6	5	8	4	2	1	7
F7								
Average	**0.0000E+00**	**0.0000E+00**	3.0632E+02	5.4000E−03	8.8506E+00	1.1000E−03	1.6400E−02	2.5230E−01
Median	**0.0000E+00**	**0.0000E+00**	3.0632E+02	1.1400E−02	8.8327E+00	1.1000E−03	2.8000E−03	1.5470E−01
STD	0.0000E+00	0.0000E+00	3.0426E−03	4.6054E−04	4.0900E−04	2.8791E−05	4.1191E−04	3.1404E−03
Time/s	353	7	2927	77	32	457	2086	5597
Rank1	1	1	8	4	7	3	5	6
Rank2	2	1	8	4	7	3	5	6
F8								
Average	**0.0000E+00**	**0.0000E+00**	4.7000E−03	**0.0000E+00**	4.4000E−03	1.0000E−04	1.3000E−03	2.3000E−03
Median	**0.0000E+00**	**0.0000E+00**	2.5000E−03	**0.0000E+00**	2.8000E−03	1.0000E−04	3.0000E−04	1.5000E−03
STD	0.0000E+00	0.0000E+00	3.6148E−04	0.0000E+00	3.3552E−04	3.3273E−05	4.0662E−05	4.4275E−05
Time/s	160	6	16	122	360	2115	491	328
Rank1	1	1	8	1	7	4	5	6
Rank2	3	1	8	2	7	4	5	6
F9								
Average	**0.0000E+00**	**0.0000E+00**	6.9000E−03	2.9100E−02	3.3000E−03	2.7000E−03	5.5000E−03	3.3000E−03
Median	**0.0000E+00**	**0.0000E+00**	4.6000E−03	1.9100E−02	3.9000E−03	2.6000E−03	3.7000E−03	3.7000E−03
STD	3.0479E−05	4.6563E−05	3.2479E−04	3.9442E−04	4.3589E−04	3.8901E−05	3.4213E−04	3.8233E−05
Time/s	489	11	1171	48	99	1615	469	472
Rank1	1	1	7	8	4	3	6	4
Rank2	2	1	7	8	4	3	6	5
F10								
Average	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	5.0000E−04	**0.0000E+00**	**0.0000E+00**
Median	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	4.0000E−04	**0.0000E+00**	**0.0000E+00**
STD	0.0000E+00	0.0000E+00	3.5156E−05	0.0000E+00	2.9912E−05	3.4835E−05	0.0000E+00	2.7635E−05
Time/s	246	2	0	24	11	99	263	137
Rank1	1	1	1	1	1	1	1	1
Rank2	8	2	1	4	3	5	7	6

Table 19 (*continued*)

Benchmark function	SEIQRA	RC-GA	DSDA	NP-PSO	BBO	DE	SaDE	ABC
F11								
Average	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**1.6300E−02**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
Median	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**1.5500E−02**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
STD	0.0000E+00	0.0000E+00	3.8533E−05	0.0000E+00	3.8176E−04	2.2409E−05	0.0000E+00	2.4235E−05
Time/s	60	2	544	736	14	383	251	247
Rank1	1	1	1	1	8	1	1	1
Rank2	2	1	6	7	8	5	4	3
F12								
Average	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	1.0000E−04	**0.0000E+00**
Median	**0.0000E+00**	**0.0000E+00**	**1.0000E−04**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	1.0000E−04	**0.0000E+00**
STD	0.0000E+00	0.0000E+00	4.3581E−05	4.0126E−05	3.2116E−05	3.5867E−05	3.5737E−05	4.2985E−05
Time/s	456	5	0	51	88	1408	347	140
Rank1	1	1	1	1	1	1	8	1
Rank2	7	2	1	3	4	6	8	5
F13								
Average	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
Median	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
STD	0.0000E+00	0.0000E+00	3.9454E−05	3.8178E−05	4.4735E−05	3.7556E−05	3.7292E−05	3.7416E−05
Time/s	326	3	1	1895	85	2604	482	199
Rank1	1	1	1	1	1	1	1	1
Rank2	5	2	1	7	3	8	6	4
F14								
Average	**0.0000E+00**	**0.0000E+00**	1.1000E−03	1.5100E−02	2.8841E+03	9.7000E−03	7.7000E−03	0.0000E+00
Median	**0.0000E+00**	**0.0000E+00**	1.1000E−03	1.5000E−02	2.8841E+03	9.6000E−03	7.7000E−03	0.0000E+00
STD	4.1160E−05	3.2139E−05	3.1498E−05	4.2935E−05	3.5650E−05	3.3448E−05	3.4866E−05	3.4194E−05
Time/s	232	8	1272	249	599	77	130	99
Rank1	1	1	4	7	8	6	5	1
Rank2	3	1	4	7	8	6	5	2
F15								
Average	**0.0000E+00**	**0.0000E+00**	1.0000E−04	5.0000E−04	1.4000E−03	3.0000E−04	**0.0000E+00**	5.0000E−04
Median	**0.0000E+00**	**0.0000E+00**	1.0000E−04	5.0000E−04	1.3000E−03	1.0000E−04	−1.0000E−04	5.0000E−04
STD	2.8171E−05	3.3254E−05	4.1701E−05	3.5130E−05	3.5641E−05	3.4427E−05	4.9930E−05	4.0094E−05
Time/s	188	9	327	445	121	444	504	93
Rank1	1	1	1	5	8	7	1	5
Rank2	2	1	3	6	8	7	4	5
F16								
Average	**0.0000E+00**	**0.0000E+00**	1.0000E−04	1.0000E−04	3.0000E−04	1.0000E−04	**0.0000E+00**	1.0000E−04
Median	**0.0000E+00**	**0.0000E+00**	1.0000E−04	1.0000E−04	3.0000E−04	1.0000E−04	**0.0000E+00**	0.0000E+00
STD	0.0000E+00	0.0000E+00	3.7005E−05	2.6888E−05	4.4005E−05	3.1010E−05	0.0000E+00	2.6570E−05
Time/s	1242	6	1946	1844	141	394	552	479
Rank1	1	1	1	5	8	5	1	5
Rank2	3	1	4	7	8	5	2	6
F17								
Average	**0.0000E+00**	5.1000E−03	3.8000E−03	1.8000E−03	2.0247E+00	4.3200E−02	9.6000E−03	**0.0000E+00**
Median	**0.0000E+00**	5.4000E−03	3.7000E−03	1.9000E−03	2.0246E+00	4.3300E−02	9.6000E−03	**0.0000E+00**
STD	0.0000E+00	3.7792E−05	4.0089E−05	4.4945E−05	3.1251E−05	4.0504E−05	4.1728E−05	0.0000E+00
Time/s	0	7	653	552	489	239	625	0
Rank1	1	5	4	3	8	7	6	1
Rank2	1	5	4	3	8	7	6	1
F18								
Average	**0.0000E+00**	**0.0000E+00**	2.0000E−04	0.0000E+00	2.8252E+02	1.2140E−01	**0.0000E+00**	1.3000E−03
Median	**0.0000E+00**	**0.0000E+00**	2.0000E−04	0.0000E+00	2.8252E+02	1.2560E−01	**0.0000E+00**	1.3000E−03
STD	0.0000E+00	0.0000E+00	3.4344E−05	0.0000E+00	3.6611E−03	4.1844E−03	0.0000E+00	3.8839E−05
Time/s	274	3	1	570	188	18	647	26
Rank1	1	1	5	1	8	7	1	6
Rank2	2	1	5	3	8	7	4	6
F19								
Average	**1.1733E+00**	4.1841E+02	5.3082E+01	1.0809E+04	3.8540E+01	2.1548E+00	2.0825E+00	2.8233E+01
Median	**1.1159E+00**	4.1841E+02	5.3087E+01	1.0810E+04	3.8548E+01	2.1445E+00	2.0447E+00	2.8226E+01
STD	2.9033 E+00	4.4896E−04	4.1200E−04	2.3979E+01	4.1444E−03	2.7829E−03	4.1410E−03	3.6863E−03
Time/s	15,665	10,150	3123	10,328	2351	199	1296	720
Rank1	1	7	6	8	5	3	2	4
Rank2	1	7	6	8	5	3	2	4
F20								
Average	**9.7850E−01**	8.5818E+00	3.6954E+00	1.9024E+00	5.1151E+00	1.3028E+01	1.3953E+01	1.1739E+01
Median	**5.1430E−01**	8.5827E+00	3.6938E+00	1.9012E+00	5.1134E+00	1.3021E+01	1.3952E+01	1.1738E+01
STD	2.0661E−01	4.1805E−03	4.0235E−03	2.9896E−03	3.5409E−03	4.3044E−053	3.2601E−03	5.0311E−03
Time/s	3860	3701	716	499	4546	400	99	553
Rank1	1	5	3	2	4	7	8	6
Rank2	1	5	3	2	4	7	8	6

Table 19 (*continued*)

Benchmark function	SEIQRA	RC-GA	DSDA	NP-PSO	BBO	DE	SaDE	ABC
F21								
Average	**0.0000E+00**	5.2485E+03	1.4045E+01	7.7911E+00	2.1558E+01	**0.0000E+00**	**0.0000E+00**	4.7579E+00
Median	**0.0000E+00**	5.2483E+03	1.4044E+01	7.7902E+00	2.1551E+01	**0.0000E+00**	**0.0000E+00**	4.7513E+00
STD	0.0000E+00	9.6462E−01	4.0378E−03	2.9258E−03	3.6489E−03	0.0000E+00	0.0000E+00	3.4583E−03
Time/s	711	4543	2581	12,506	475	197	225	5379
Rank1	1	8	6	5	7	1	1	4
Rank2	3	8	6	5	7	1	2	4
F22								
Average	**0.0000E+00**	2.9864E+05	−1.1441E+02	1.3629E+03	1.8143E+03	5.9107E+03	1.5730E+02	5.3596E+02
Median	**1.1284E+00**	2.9865E+05	−1.1441E+02	1.3629E+03	1.8143E+03	5.9105E+03	1.5730E+02	5.3596E+02
STD	1.1563E+01	2.0103E+01	3.7340E−05	3.6630E−05	5.4300E−05	4.1423E+00	5.1860E−05	3.9080E−05
Time/s	4187	249	1570	1931	2611	331	3998	1577
Rank1	2	8	1	5	6	7	3	4
Rank2	2	8	1	5	6	7	3	4
F23								
Average	**0.0000E+00**	1.9792E+03	2.0031E+02	1.4306E+01	2.0492E+02	3.6572E+02	2.1345E+02	6.3643E+01
Median	**8.3770E−01**	1.9792E+03	2.0030E+02	1.4298E+01	2.0492E+02	3.6572E+02	2.1344E+02	6.3639E+01
STD	1.2205E+01	4.2570E−04	3.4260E−03	3.8950E−03	3.4220E−03	4.6370E−03	3.9050E−03	3.5760E−03
Time/s	683	2001	2660	1149	6804	654	2210	2720
Rank1	1	8	4	2	5	7	6	3
Rank2	1	8	4	2	5	7	6	3
F24								
Average	3.5742E+02	5.7192E+02	5.6956E+02	4.3088E+02	5.6900E+02	4.8506E+02	**0.0000E+00**	7.9204E+01
Median	3.5804E+02	5.7184E+02	5.6938E+02	4.3090E+02	5.6914E+02	4.8518E+02	**0.0000E+00**	7.9205E+01
STD	1.1659E+00	1.4154E+00	1.2980E+00	1.0883E+00	1.2643E+00	1.2272E+00	3.8259E−05	3.8754E−03
Time/s	11,152	946	1599	1164	1170	433	2567	1772
Rank1	3	8	7	4	6	5	1	2
Rank2	3	8	7	4	6	5	1	2
F25								
Average	**6.2533E+01**	7.9420E+01	8.9847E+01	6.7286E+01	9.7890E+01	6.2932E+01	8.7304E+01	7.3093E+01
Median	**6.1276E+01**	7.9719E+01	9.0006E+01	6.7203E+01	9.7872E+01	6.2600E+01	8.7701E+01	7.3341E+01
STD	1.5797E+00	1.1098E+00	1.1801E+00	1.0510E+00	1.2081E+00	1.6040E+00	9.3316E−01	1.2855E+00
Time/s	2249	581	7	2623	888	1136	588	1805
Rank1	1	5	7	3	8	2	6	4
Rank2	1	5	7	3	8	2	6	4
F26								
Average	**0.0000E+00**	8.0700E−01	1.2060E+03	4.9700E−01	8.3540E+01	8.6240E−01	1.3381E+00	4.9690E−01
Median	**0.0000E+00**	8.0790E−01	1.2063E+03	4.9810E−01	8.3539E+01	8.6240E−01	1.3327E+00	4.9710E−01
STD	3.9631E−01	3.4304E−03	1.1094E+00	4.7387E−04	2.7967E−03	4.4850E−03	3.9554E−03	3.4717E−03
Time	6137	555	319	5649	575	3321	737	1272
Rank1	1	4	8	3	7	5	6	2
Rank2	1	4	8	3	7	5	6	2
F27								
Average	1.1446E+01	2.1338E+01	**1.5291E+00**	6.4663E+00	8.7212E+00	9.7071E+00	3.8324E+01	3.5825E+00
Median	1.2770E+01	2.0967E+01	**9.7730E−01**	6.4517E+00	8.6188E+00	1.0310E+01	3.8167E+01	3.6585E+00
STD	2.4204E+00	1.3671E+00	1.5052E+00	1.4331E+00	1.0570E+00	1.5268E+00	9.4364E−01	1.1675E+00
Time	1765	4155	3520	4285	120	3553	1162	3063
Rank1	6	7	1	3	4	5	8	2
Rank2	6	7	1	3	4	5	8	2
F28								
Average	**0.0000E+00**	1.2589E+04	8.7236E+03	2.2516E+03	8.7726E+03	7.8117E+03	7.7779E+03	8.8673E+03
Median	**1.8380E−01**	1.2589E+04	8.7232E+03	2.2516E+03	8.7728E+03	7.8116E+03	7.7780E+03	8.8671E+03
STD	4.4912E−01	1.2685E+00	1.4660E+00	4.1750E−06	1.2277E+00	1.3148E+00	1.3627E+00	1.3491E+00
Time/s	110,475	5910	2565	15,121	2083	3167	4760	1438
Rank 1	1	8	5	2	6	4	3	7
Rank 2	1	8	5	2	6	4	3	7
Sum Rank 1	36	122	118	119	156	100	92	113
Sum Rank 2	76	126	128	142	163	122	117	133
Final Rank 1	1	7	6	5	8	3	2	4
Final Rank 2	1	7	6	5	8	3	2	4

Fig. 17 illustrates a general case described in Fig. 16 when SEIQRA solves benchmark function F14 and F6. When SEIQRA solves benchmark function F14, from Fig. 17(a) we can see that at 3 s, REINIT is invoked; at 95 s, the first assigned triggering point is produced; at 99 s, the first effective triggering point is produced. When search is approximate to the global optimum solution of F14, three effective triggering points are produced; when SEIQRA solves benchmark function F6, from Fig. 17(b) we can see that at 2 s, REINIT is invoked; at 592 s, the first assigned triggering point is produced; at 1009 s, the first effective triggering point is produced. When search is approximate to the global optimum solution of F6, a larger number of effective triggering points are produced; it means that REINIT continues to enhance the precision of the global optimum solution.

Fig. 18 illustrates a special case when SEIQRA solves basic benchmark function F18. From Fig. 18 we can see that at 3 s, REINIT is invoked; at 57 s, the first assigned triggering point is

Table 20
Comparative results of SEIQRA for the benchmark functions using Wilcoxon's rank sum test (α=0.01).

Benchmark function	Wilcoxon's rank sum test	SEIQRA vs.						
		RC-GA	DSDA	NP-PSO	BBO	DE	SaDE	ABC
F1	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	1	1	1
	h-value	1	1	1	1	0	0	0
	*z*val	−3.920	−3.920	−3.920	−3.920	0.000	0.000	0.000
F2	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8182E−05	8.8549E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	−3.921	−3.920
F3	*p*-value	8.8182E−05	8.8182E−05	8.8549E−05	8.8182E−05	1	1	8.8549E−05
	h-value	1	1	1	1	0	0	1
	*z*val	−3.921	−3.921	−3.920	−3.921	0.000	0.000	−3.920
F4	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	1	2.3959E−01	8.8549E−05
	h-value	1	1	1	1	0	0	1
	*z*val	−3.920	−3.920	−3.920	−3.920	0.000	−1.176	−3.920
F5	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	1.2349E−02	1.4539E−01	8.8549E−05
	h-value	1	1	1	1	0	0	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−2.502	−1.456	−3.920
F6	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	9.9951E−03	1.6728E−01	8.8549E−05
	h-value	1	1	1	1	1	0	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−2.576	−1.381	−3.920
F7	*p*-value	1	8.8549E−05	8.8549E−05	8.8182E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	0	1	1	1	1	1	1
	*z*val	0.000	−3.920	−3.920	−3.921	−3.920	−3.920	−3.920
F8	*p*-value	1	8.8549E−05	1	8.8549E−05	8.8549E−05	8.8549E−05	8.8182E−05
	h-value	0	1	0	1	1	1	1
	*z*val	0.000	−3.920	0.000	−3.920	−3.920	−3.920	−3.921
F9	*p*-value	3.3155E−01	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	0	1	1	1	1	1	1
	*z*val	−0.971	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920
F10	*p*-value	1	1.6894E−02	1	8.2276E−01	8.8549E−05	1	2.9602E−01
	h-value	0	0	0	0	1	0	0
	*z*val	0.000	−2.389	0.000	−0.224	−3.920	0.000	−1.045
F11	*p*-value	1	8.8155E−01	1	8.8549E−05	1.6728E−01	1	8.2564E−02
	h-value	0	0	0	1	0	0	0
	*z*val	0.000	−0.149	0.000	−3.920	−1.381	0.000	−1.736
F12	*p*-value	1	1.0162E−03	3.9034E−01	6.8107E−01	8.5166E−01	8.8549E−05	8.5979E−02
	h-value	0	1	0	0	0	1	0
	*z*val	0.000	−3.286	−0.859	−0.411	−0.187	−3.920	−1.717
F13	*p*-value	1	1.6728E−01	5.2258E−02	3.7025E−01	8.8155E−01	9.7049E−01	2.7881E−01
	h-value	0	0	0	0	0	0	0
	*z*val	0.000	−1.381	−1.941	−0.896	−0.149	−0.037	−1.083
F14	*p*-value	6.1902E−02	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8182E−05
	h-value	0	1	1	1	1	1	1
	*z*val	−1.867	−3.920	−3.920	−3.920	−3.920	−3.920	−3.921
F15	*p*-value	3.7025E−01	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	7.7948E−01	8.8549E−05
	h-value	0	1	1	1	1	0	1
	*z*val	−.896	−3.920	−3.920	−3.920	−3.920	−0.280	−3.920
F16	*p*-value	1	1.4010E−04	8.8549E−05	8.8549E−05	8.8549E−05	1	8.8549E−05
	h-value	0	1	1	1	1	0	1

Table 20 (*continued*)

Benchmark function	Wilcoxon's rank sum test	SEIQRA vs.						
		RC-GA	DSDA	NP-PSO	BBO	DE	SaDE	ABC
	*z*val	0.000	−3.808	−3.920	−3.920	−3.920	0	−3.920
F17	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	1
	h-value	1	1	1	1	1	1	0
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	-3.920	0.000
F18	*p*-value	1	8.8549E−05	1	8.8549E−05	8.8549E−05	1	8.8549E−05
	h-value	0	1	0	1	1	0	1
	*z*val	0	−3.920	−3.920	−3.920	−3.920	0	−3.920
F19	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920
F20	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920
F21	*p*-value	8.6370E−05	8.6370E−05	8.6370E−05	8.6370E−05	1	1	8.6370E−05
	h-value	1	1	1	1	0	0	1
	*z*val	−3.920	−3.920	−3.920	−3.920	0.000	0.000	−3.920
F22	*p*-value	8.8549E−05	8.8182E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.7817E−05	8.8182E−05
	h-value	1	−1	1	1	1	1	1
	*z*val	−3.920	−3.921	−3.920	−3.920	−3.920	−3.922	−3.921
F23	*p*-value	8.8549E−05	8.7453E−05	8.7453E−05	8.6370E−05	8.6730E−05	8.7091E−05	8.7091E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.920	−3.923	−3.923	−3.926	−3.925	−3.924	−3.924
F24	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	1	1	1	−1	−1
	*z*val	−3.92	−3.92	−3.92	−3.92	−3.92	−3.92	−3.92
F25	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920
F26	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	−1	1	1	1	−1
	*z*val	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920	−3.920
F27	*p*-value	1.7127E−03	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	−1	−1	−1	−1	1	−1
	*z*val	−3.136	−3.92	−3.92	−3.92	−3.92	−3.92	−3.92
F28	*p*-value	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05	8.8549E−05
	h-value	1	1	1	1	1	1	1
	*z*val	−3.92	−3.92	−3.92	−3.92	−3.92	−3.92	−3.92
1 (Better)		17	23	20	24	19	15	19
0 (Same)		11	3	6	3	8	12	6
−1 (Worse)		0	2	2	1	1	1	3

produced. But no effective triggering points are generated during the whole process of search, it means that REINIT is only used to break sticky states of work individuals, namely operation reinitialization does not take effect, while operation redistribution takes effect.

When an optimization problem can be solved without the help of REINIT, if the operator is invoked forcibly, then the convergence rate of solving the problem may be decreased because continuous triggering of REINIT can consume much CPU time. Fig. 19 illustrates a comparison of the convergence rate between "REINIT is not invoked" and "REINIT is invoked" when SEIQRA solves basic benchmark function F3. From Fig. 19 we can see that the convergence rate when REINIT is not invoked is faster than that when REINIT is invoked; as time lapses, triggering of REINIT become more and more frequent, much time is consumed for triggering REINIT.

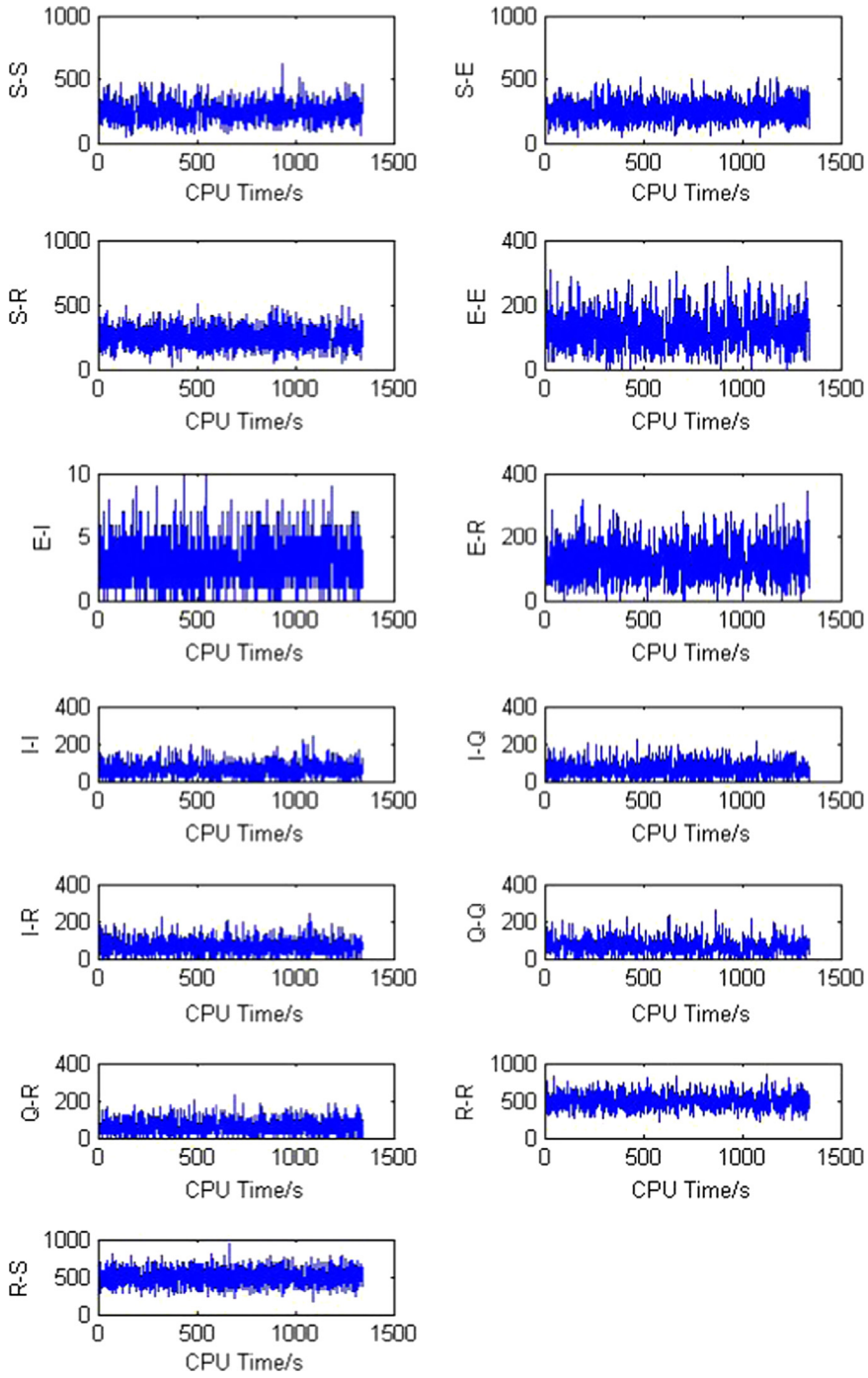

Fig. 13. Times of state transition S→S, S→E, S→R, E→E, E→I, E→R, I→I, I→Q, I→R, Q→Q, Q→R, R→R and R→S being triggered during evolution of individuals when SEIQRA solves F1.

Conclusively, REINIT has two effects of the pros and cons, it can enhance the exploration and exploitation ability of SEIQRA, but it may consume much CPU time. Therefore, we suggest that if an optimization problem can be solved without REINIT, REINIT should be prohibited; if sticky states always occur when SEIQRA solves an optimization problem, REINIT should be invoked.

6. Conclusions

In SEIQRA, Individuals come from one population; operators S–S, S–E, S–R, E–E, E–I, E–R, I–I, I–Q, I–R, Q–Q, Q–R, R–R and R–S are constructed based on the SEIQR epidemic model, they are not related to any actual optimization problems to be solved. Because the SEIQR epidemic model does not require the support of pathological knowledge, SEIQRA does not require the support of pathological knowledge also, this characteristic is beneficial to the research and improvement of SEIQRA.

In conclusion, SEIQRA has the following properties:

1. The stable rhythm of heart: the stable rhythm of heart in SEIQRA can ensure the ecological operators to be activated evenly.
2. Exploration and exploitation ability: it is represented by operator REINIT, which is activated at certain frequency to improve exploration and exploitation ability.

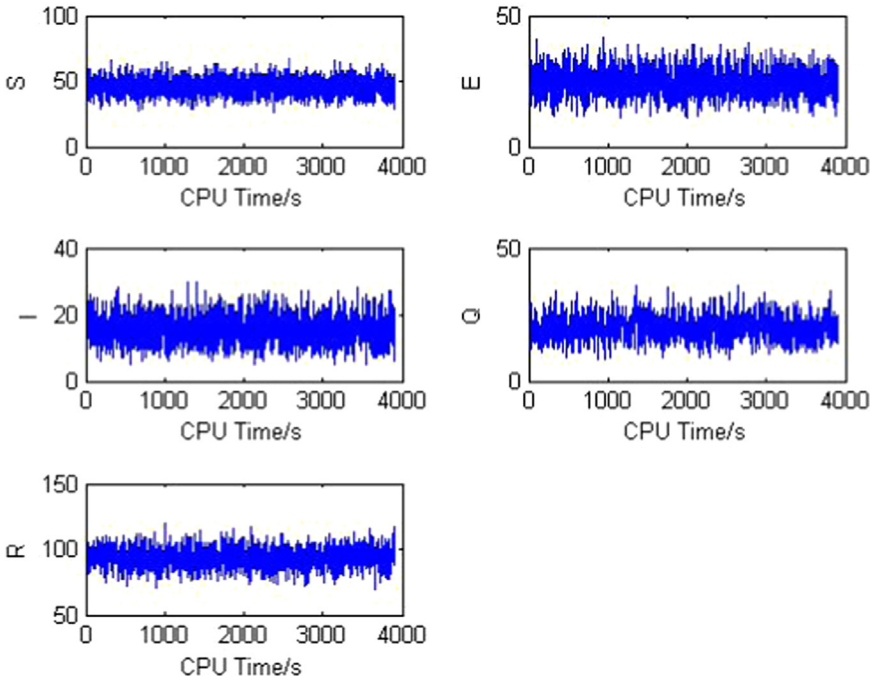

Fig. 14. The number of individuals staying at state S, E, I, Q and R during evolution of individuals when SEIQRA solves F1.

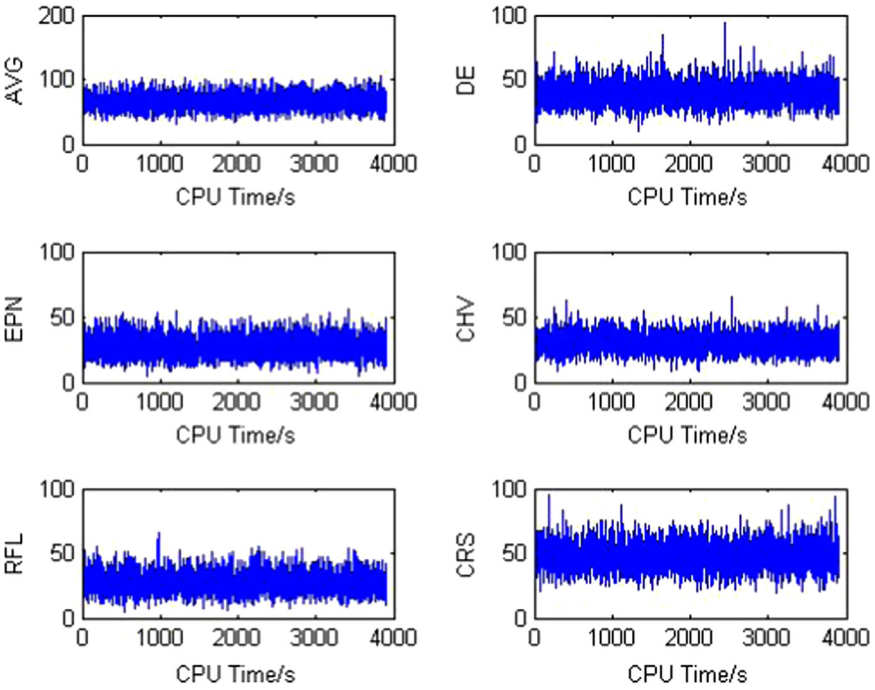

Fig. 15. Relationship of the times of each operation being executed with the CPU time when SEIQRA solves F1.

3. Wide adaptability: if an operator can solve one type of optimization problems, then synergy of many operators can solve many types of optimization problems. Because SEIQRA has 13 operators, it has wider adaptability to solve different types of optimization problems.
4. A small part of variables being involved in each iteration: Since a virus attacks only a small part of individual features every time, when individuals in different states exchange their feature information, there is just a small portion of features used in computation, even so, the individual's IPI can still be well improved. Owing to the substantial reduction of features in calculation, when solving complicated optimization problems, especially high-dimensional optimization problems, SEIQRA can converge significantly and rapidly.
5. The stable setting of parameters: Although SEIQRA uses more parameters to control its running, the setting of its parameters is stable.
6. Multiple operations: Because SEIQRA possesses of 13 operators, it provides opportunities to contain many operations simultaneously.

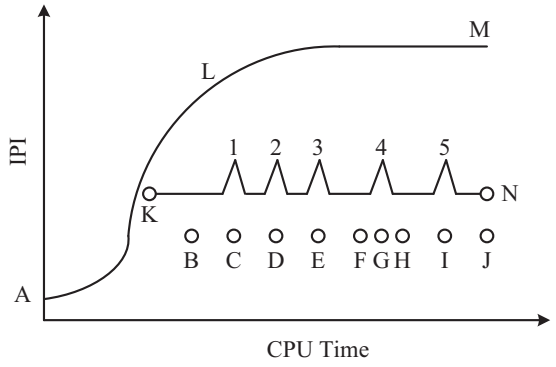

Fig. 16. Identification of working effectiveness of REINIT.

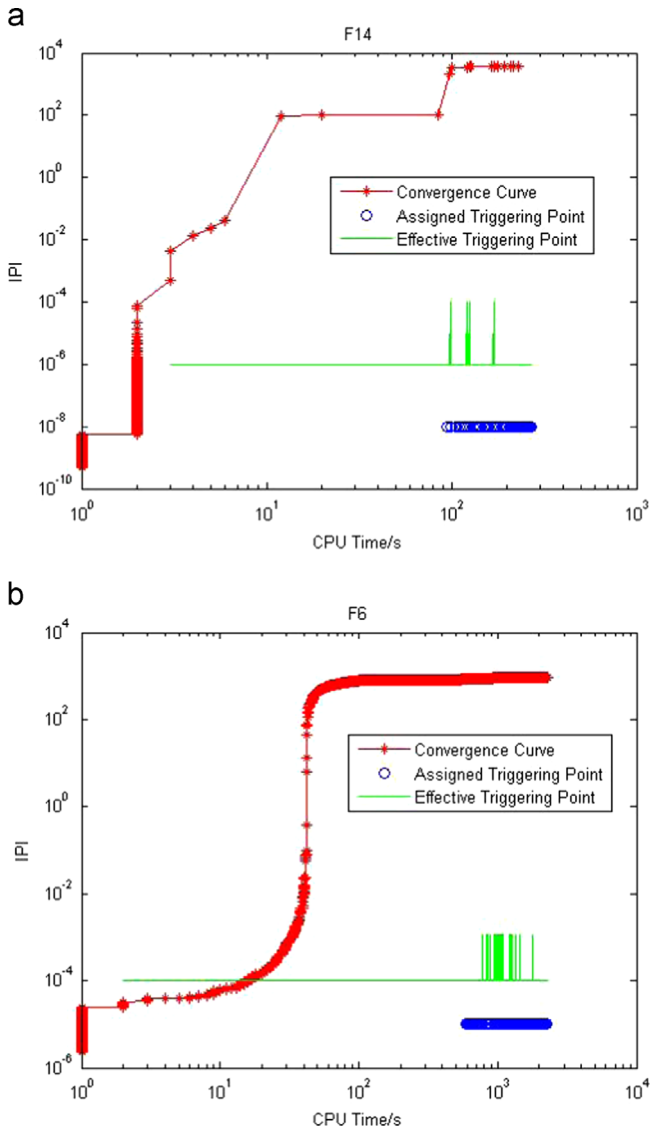

Fig. 17. Identification of working effectiveness of REINIT when SEIQRA solves benchmark functions F14 and F6, (a) F14 and (b) F6.

7. Convergence: When evolving, the individuals with strong physique can continue to grow, while the individuals with weak physique will stop growing. This kind of search strategy can ensure global optima to be found at higher probability.

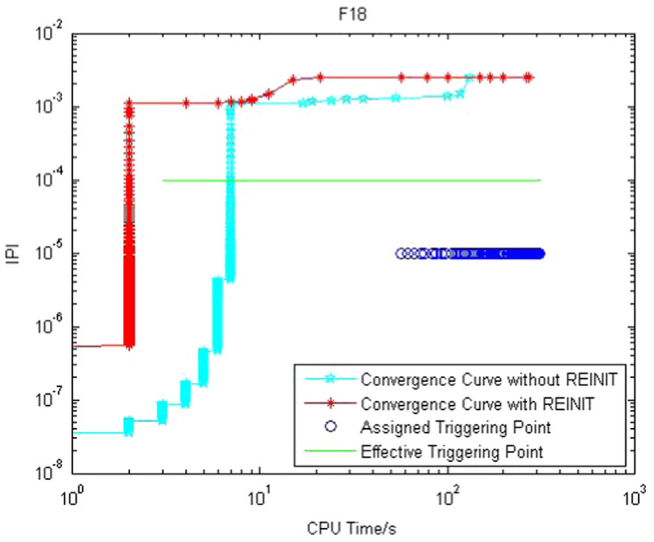

Fig. 18. Identification of working effectiveness of REINIT when SEIQRA solves basic benchmark function F18.

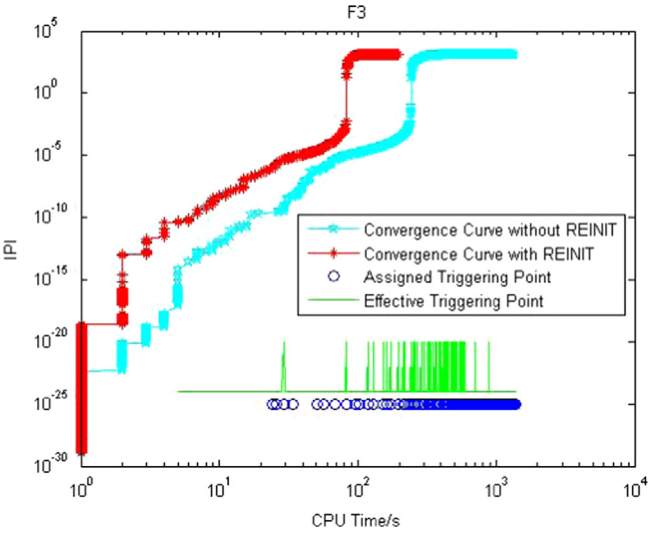

Fig. 19. A comparison of the convergence rate between "REINIT is not invoked" and "REINIT is invoked" when SEIQRA solves basic benchmark function F3.

8. Visualization: The story-based design of SEIQRA makes it easy to visualize its process of evolution when solving an optimization problem.

SEIQRA has characteristics of high adaptability, fast solving speed and convergence for complicated function optimization problems. Besides these, precision of global optimal solutions can also be improved by operator REINIT. These characteristics of SEIQRA may facilitate practical applications. Further study in the future includes:

(1) How to use the mechanisms of immunity to construct immunity-based operators;
(2) How to use the mechanisms of virus attacking cells to construct virus transmission operators;
(3) How to use the mechanisms of infectious disease treatment to construct infectious disease treatment operators;
(4) How to implant independent virus description factor into features of an individual;
(5) How to embed various kinds of viruses in SEIQRA;
(6) How to bring more different types of individuals into SEIQRA;

(7) How to reflect time-delay differences of outbreak of different viruses in SEIQRA.

In nature, there are millions of infectious diseases with different transmission mechanisms, infection mechanisms, immunity mechanisms and medical treatment methods, each infectious disease can be transformed into a population-based optimization algorithm with different performance. Besides these, the epidemic dynamics and the virus dynamics, which are built on mathematical models, provide solid mathematical foundations for developing artificial infectious disease optimization algorithms. Therefore, artificial infectious disease optimization may be form a new idea to carry out function optimization, which has rich connotations and magnanimous natural materials.

Acknowledgment

The article is supported by the Foundation Research Project of Natural Science of Shaanxi Province-Key Project (2015JZ010) and Society Science foundation of Shaanxi Province (2014P07).

Appendix A

Psuedo-code of SEIQRA

(1) Initialize the following parameters: (a) Let $t=0$, initialize the parameters SEIQRA involves as specified in Table 3; (b) Take N_0 initialization individuals, N work individuals as input parameters, use operator REINIT to produce N individuals $\boldsymbol{X}_1^0, \boldsymbol{X}_2^0, \cdots, \boldsymbol{X}_N^0$; (c) Let *LatentPeriod*$(i)=\infty$, *ImmunityPeriod*$(i)=\infty$, $i=1, 2, \ldots, N$; // *LatentPeriod*(i) is the starting time of latency of individual i; *ImmunityPeriod*(i) is the starting time of vaccination of individual i.

(2) Produce 5 random numbers: $a_1^i=Rnd(0,1)$, $a_2^i=Rnd(0,1)$, $a_3^i=Rnd(0,1)$, $a_4^i=Rnd(0,1)$, $a_5^i=Rnd(0,1)$; compute $S_i(0)=a_1^i/\left(\sum_{j=1}^{5} a_j^i\right)$, $E_i(0)=a_2^i/\left(\sum_{j=1}^{5} a_j^i\right)$, $I_i(0)=a_3^i/\left(\sum_{j=1}^{5} a_j^i\right)$, $Q_i(0)=a_4^i/\left(\sum_{j=1}^{5} a_j^i\right)$, $R_i(0)=1-S_i(0)-E_i(0)-I_i(0)-Q_i(0)$, $i=1, 2, \ldots, N$;

(3) Calculate the state of S, E, I, Q and R of individual i, $SEIQR_i(0)=GetSEIQR\{S_i(0), E_i(0), I_i(0), Q_i(0), R_i(0)\}$, $i=1, 2, \ldots, N$; /* function *GetSEIQR* () is used to determine which state individual i will stay at.*/

(4) Execute the following operations:

FOR t=1 **TO** G /* G is the maximum times of iteration*

$\beta^t=Rnd(0, d_0)$, $\gamma_1^t=Rnd(0, d_0)$, $\gamma_2^t=Rnd(0, d_0)$, $\gamma_3^t=Rnd(0, d_0)$, $p^t=Rnd(0, d_0)$, $\delta^t=Rnd(0, d_0)$, d_0=0.1; //d_0 is the upper limit of the SEIQR epidemic model's parameters $\beta_0, \gamma_1, \gamma_2, \gamma_3, p, \delta$.

FOR i=1 **TO** N

IF t mod $T\neq 0$ **THEN**

Use formula (10) to compute $S_i(t)$, $E_i(t)$, $I_i(t)$, $R_i(t)$和 $Q_i(t)$;

ELSE

Use formula (11) to compute $S_i(t)$, $E_i(t)$, $I_i(t)$, $R_i(t)$和 $Q_i(t)$;

END IF

Compute $SEIQR_i(t)=GetSEIQR\{S_i(t), E_i(t), I_i(t), R_i(t)$和 $Q_i(t)\}$;

FOR j=1 **TO** n

IF $Rnd(0, 1)\leqslant E_0$ **THEN**/* E_0 is the maximum probability that a feature of an individual is attacked by an infectious disease */

IF $SEIQR_i(t\text{-}1)$=S **THEN**

IF $SEIQR_i(t)$=S **THEN**

Use operator S-S to compute v_{ij}^t of state transition S→S;

ELSE IF $SEIQR_i(t)$=E **THEN**

Use operator S-E to compute v_{ij}^t of state transition S→E;

LatentPeriod$(i)=t$; /*Individual i is infected and enters into exposuse state; the starting time of exposure is t; function *LatentPeriod*(i) is used to record the time of individual i being infected at the first time*/

ELSE IF $SEIQR_i(t)$=R **THEN**

Use operator S-R to compute v_{ij}^t of state transition S→R;

ImmunityPeriod$(i)=t$; /*Individual i is cured, and obtains immunity; *ImmunityPeriod*(i) is used to record the time of individual i being immunized at the first time*/

ELSE/*It is an illegal state transition, its state keeps unchanged*/.
$v_{ij}^t = x_{ij}^{t-1}$, $SEIQR_i(t)=SEIQR_i(t\text{-}1)$;
EXIT FOR;
END IF
ELSE IF $SEIQR_i(t\text{-}1)$=E **THEN**
IF $SEIQR_i(t)$=E **THEN**
Use operator E-E to compute v_{ij}^t of state transition E→E;
ELSE IF $SEIQR_i(t)$=I **AND** $t\text{-}LatentPeriod(i) \geqslant \tau$ **THEN**
Use operator E-I to compute v_{ij}^t of state transition E→I;
$LatentPeriod(i)=\infty$;
ELSE IF $SEIQR_i(t)$=R **THEN**
Use operator E-R to compute v_{ij}^t of state transition E→R;
IF $ImmunityPeriod(i)>t$ **THEN**
$ImmunityPeriod(i)=t$;
END IF
ELSE/*It is an illegal state transition, its state keeps unchanged*/.
$v_{ij}^t = x_{ij}^{t-1}$, $SEIQR_i(t)=SEIQR_i(t\text{-}1)$;
EXIT FOR;
END IF
ELSE IF $SEIQR_i(t\text{-}1)$=I **THEN**
IF $SEIQR_i(t)$=I **THEN**
Use operator I-I to compute v_{ij}^t of state transition I→I;
ELSE IF $SEIQR_i(t)$=Q **THEN**
Use operator I-Q to compute v_{ij}^t of state transition I→Q;
ELSE IF $SEIQR_i(t)$=R **THEN**
Use operator I-R to compute v_{ij}^t of state transition I→R;
$ImmunityPeriod(i)=t$;
ELSE/*It is an illegal state transition, its state keeps unchanged*/
$v_{ij}^t = x_{ij}^{t-1}$, $SEIQR_i(t)=SEIQR_i(t\text{-}1)$;
EXIT FOR;
END IF
ELSE IF $SEIQR_i(t\text{-}1)$=Q **THEN**
IF $SEIQR_i(t)$=Q **THEN**
Use operator Q-Q to compute v_{ij}^t of state transition Q→Q;
ELSE IF $SEIQR_i(t)$=R **THEN**
Use operator Q-R to compute v_{ij}^t of state transition Q→R;
$ImmunityPeriod(i)=t$;
ELSE/*It is an illegal state transition, its state keeps unchanged*/
$v_{ij}^t = x_{ij}^{t-1}$, $SEIQR_i(t)=SEIQR_i(t\text{-}1)$;
EXIT FOR;
END IF
ELSE IF $SEIQR_i(t\text{-}1)$=R **THEN**
IF $SEIQR_i(t)$=R **THEN**
Use operator R-R to compute v_{ij}^t of state transition R→R;
ELSEIF $SEIQR_i(t)$=S AND $t\text{-}ImmunityPeriod(i) \geqslant \lambda$ **THEN**
Use operator R-S to compute v_{ij}^t of state transition R→S;
$ImmunityPeriod(i)=\infty$;
END IF
ELSE/*It is an illegal state transition, its state keeps unchanged*/
$v_{ij}^t = x_{ij}^{t-1}$, $SEIQR_i(t)=SEIQR_i(t\text{-}1)$;
EXIT FOR;
END IF
END IF
ELSE
$v_{ij}^t = x_{ij}^{t-1}$;
END IF
END FOR

Calculate the operator growth according to formula (14) so as to update V_i^t into

X_i^t ; /*fitness evaluation of individual i is computed here*/

END FOR

Get the global optimum solution at time t: $X^{*t} = \min\left\{X_1^t, X_2^t, \cdots, X_N^t\right\}$;

IF X^{*t} is updated, THEN

Let t_0=t; /* where t_0 records the latest update of the global optimum solution*/

END IF

IF t-t_0>$t_{\max}$, **THEN**/*where $t_{\max}$ is the allowable maximum generations the global optimization solution keeps unchanged, if $t_{\max}$ is very large, REINIT will not be activated forever.*/

Execute REINIT to produce new N work individuals through N_0 initial individuals;

Let t_0=t;

END IF

IF the error between the global optimum solution X^{*t} obtained just now and the global optimum solution found most recently meets the minimum requirement of the errors ε **THEN**

GOTO Step (5);

END IF

Save the global optimum solution X^{*t} obtained just now;

END FOR

(5) Ends.

By using the roulette wheel strategy, function *GetSEIQR*(S, E, I, Q, R) is defined as follows:

FUNCTION *GetSEIQR*(S, E, I, Q, R)

p=*Rnd*(0,1); //p is the probability that an individual stays at state S, E, I, Q or R.

IF $p \le S$ **THEN**
RETURN S; //Return state S.
ELSE IF $p \le S+E$ **THEN**
RETURN E; //Return state E.
ELSE IF $p \le S+E+I$ **THEN**
RETURN I;//Return state I.
ELSE IF $p \le S+E+I+Q$ **THEN**
RETURN Q; //Return state Q.
ELSE IF $p \le S+E+I+Q+R=1$ **THEN**
RETURN R; //Return state R.
END IF
END FUNCTION

Appendix B

See Fig. B1

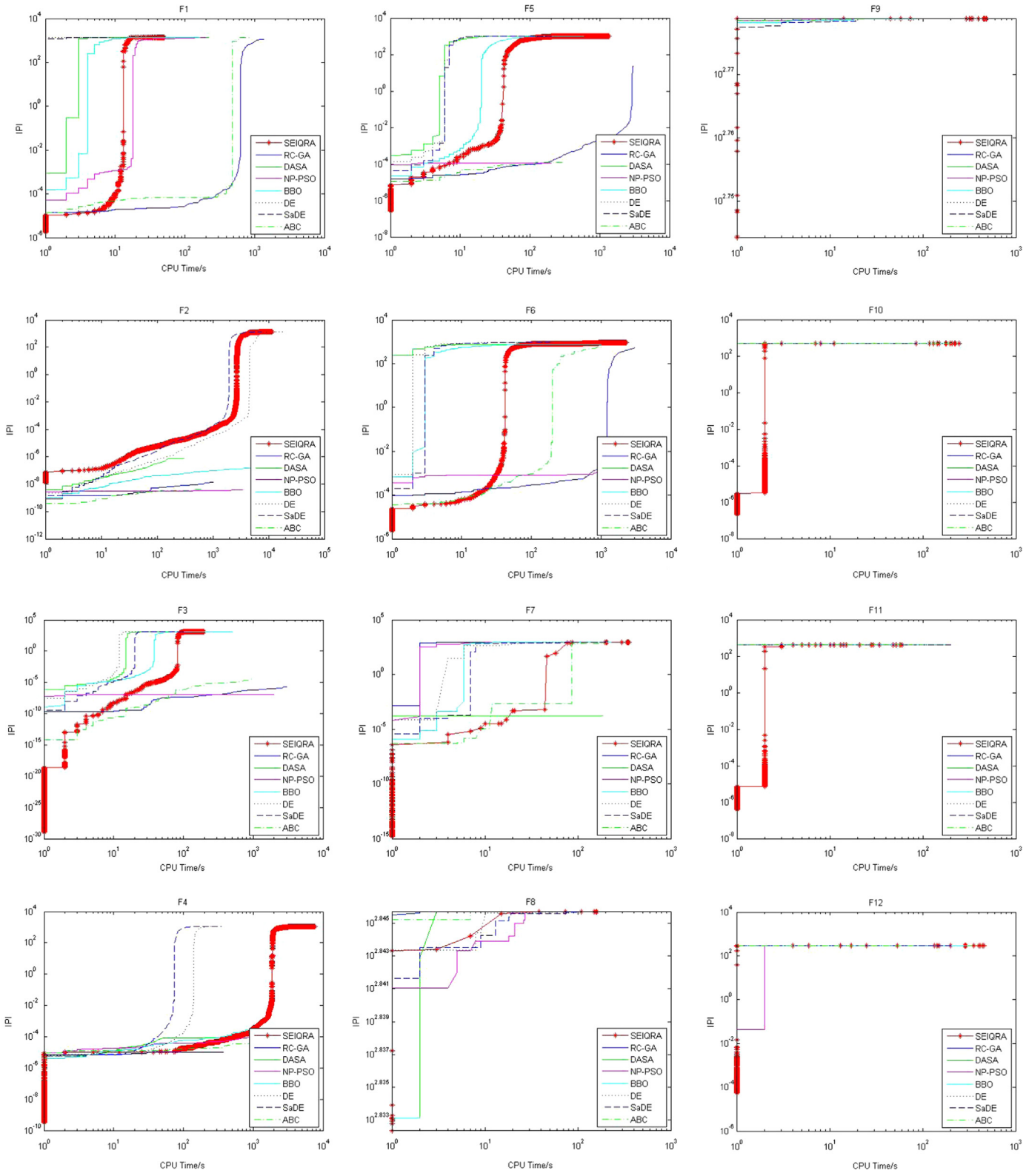

Fig. B1. Convergence graphs of F1–F28.

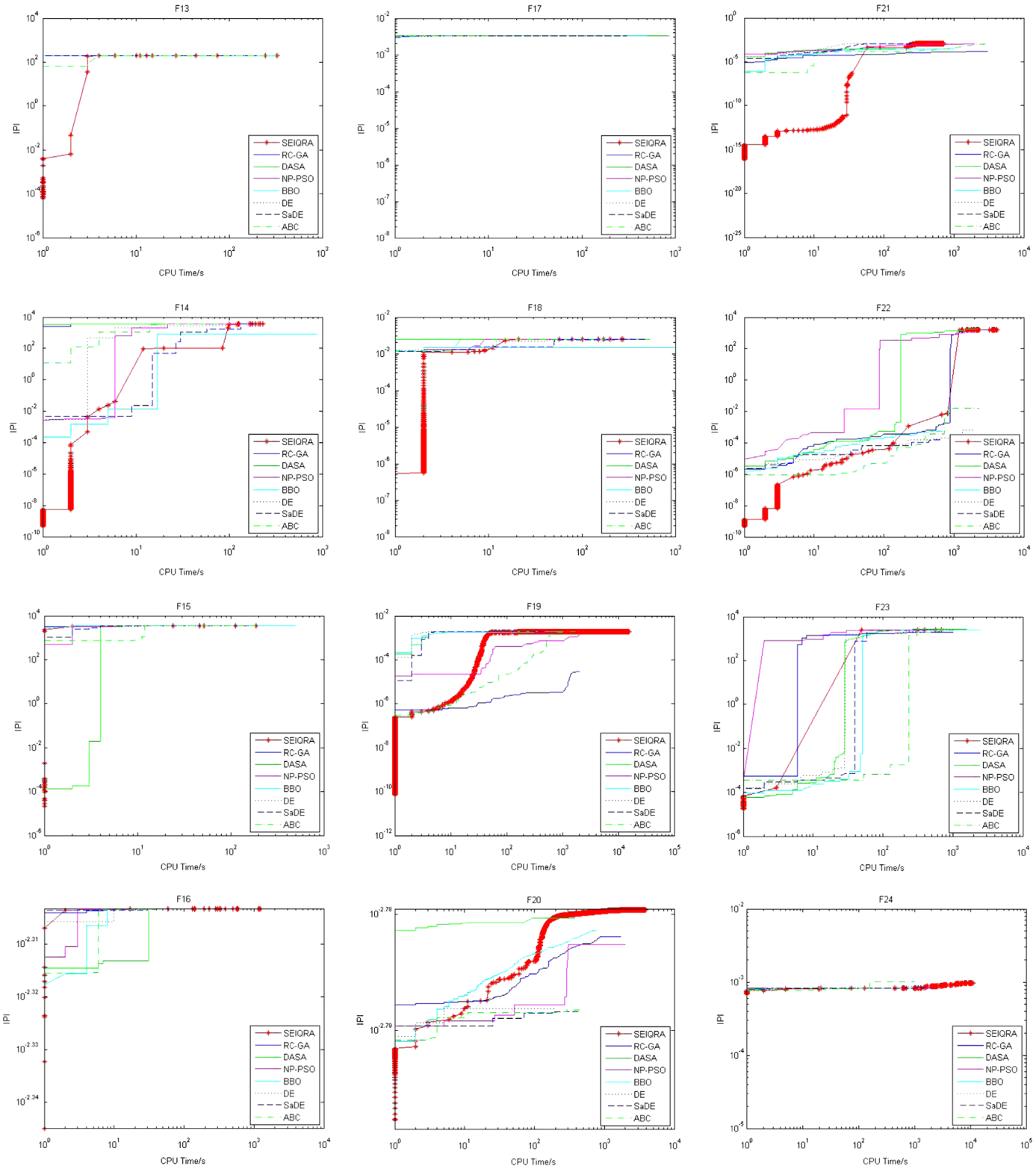

Fig. B1. (*continued*)

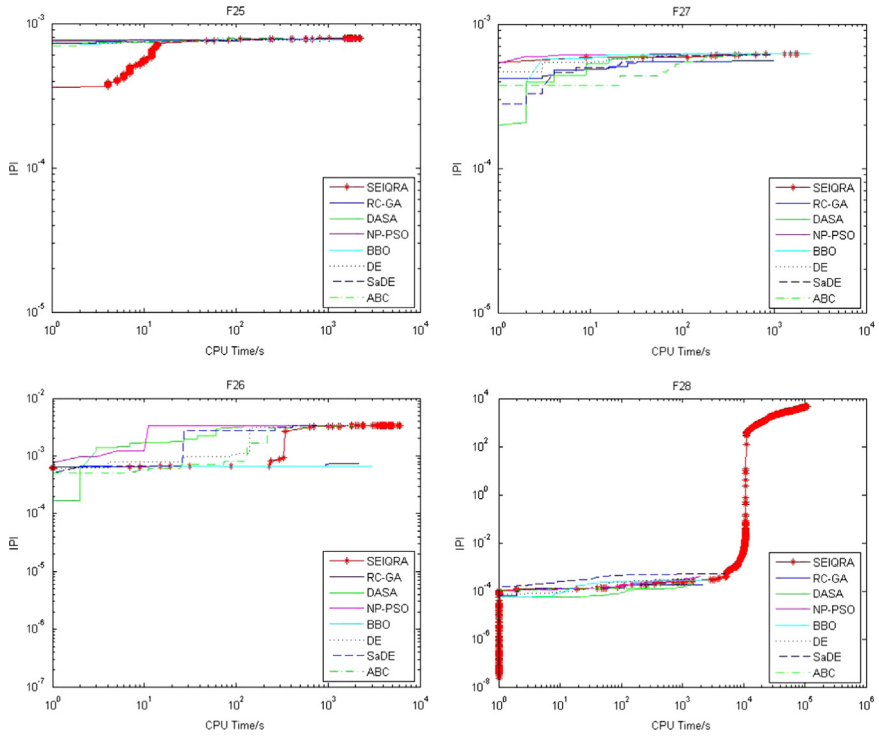

Fig. B1. (*continued*)

References

[1] Wei Yang, Some Mathematic Models of Infectious Diseases and Their Analysis, Fudan University, Shanghai, China, 2010, Doctoral Thesis.

[2] Aaron G. Buseh, Patricia E. Stevens, Mel Bromberg, Sheryl T. Kelber, The Ebola epidemic in West Africa: challenges, opportunities, and policy priority areas, Nurs. Outlook 63 (1) (2015) 30–40.

[3] Haoyang Li, Tianlei Ying, Fei Yu, Lu Lu, Shibo Jiang, Development of therapeutics for treatment of Ebola virus infection, Microbes Infect. 17 (2) (2015) 109–117.

[4] Ling Ye, Chinglai Yang, Development of vaccines for prevention of Ebola virus infection, Microbes Infect. 17 (2) (2015) 98–108.

[5] E. Fenner, D.A. Henderson, Summary of probable SARS cases with onset of illness from 1 November 2002 to 31 July 2003, WHO Retrieved (2008) 10–31.

[6] Yahira M. Báez-Santos, Sarah E. St John, Andrew D. Mesecar, The SARS-coronavirus papain-like protease: structure, function and inhibition by designed antiviral compounds, Antivir. Res. 15 (2015) 21–38.

[7] Mukesh Mahajan, Surajit Bhattacharjya, NMR structures and localization of the potential fusion peptides and the pre-transmembrane region of SARS-CoV: implications in membrane fusion, Biochim. Biophys. Acta (BBA)-Biomembranes 1848 (2) (2015) 721–730.

[8] Yasuhiro Shimamoto, Yasunao Hattori, Kazuya Kobayashi, Kenta Teruya, Akira Sanjoh, Atsushi Nakagawa, Eiki Yamashita, Kenichi Akaji, Fused-ring structure of decahydroisoquinolin as a novel scaffold for SARS 3CL protease inhibitors, Bioorg. Med. Chem. 23 (4) (2015) 876–890.

[9] HaiYan Mao, Bin Guo, FengYing Wang, Yi Sun, XiuYu Lou, Yin Chen, Lei Zhang, XinYing Wang, Zhen Li, SheLan Liu, ShuWen Qin, JunChao Wei, ZhiFeng Pang, ZhiPing Chen, YanJun Zhang, A study of family clustering in two young girls with novel avian influenza A (H7N9) in Dongyang, Zhejiang Province, in 2014, J. Clin. Virol. 63 (2015) 18–24.

[10] Wenfei Zhu, Yuelong Shu, Genetic tuning of avian influenza A (H7N9) virus promotes viral fitness within different species, Microbes Infect. 17 (2) (2015) 118–122.

[11] Erica Spackman, Mary Pantin-Jackwood, David E. Swayne, David L. Suarez, Darrell R. Kapczynski, Impact of route of exposure and challenge dose on the pathogenesis of H7N9 low pathogenicity avian influenza virus in chickens, Virology 477 (2015) 72–81 , Original Research Article.

[12] Wenbiao Hu, Wenyi Zhang, Xiaodong Huang, Archie Clements, Kerrie Mengersen, Shilu Tong, Weather variability and influenza A (H7N9) transmission in Shanghai, China: a Bayesian spatial analysis, Environ. Res. 36 (2015) 405–412.

[13] Haibo Wu, Rufeng Lu, Xiaorong Peng, Lihua Xu, Linfang Cheng, Xiangyun Lu, Changzhong Jin, Tiansheng Xie, Hangping Yao, Nanping Wu, Novel reassortant highly pathogenic H5N6 avian influenza viruses in poultry in China, Infection, Genetics and Evolution, 2015 (in press, accepted manuscript, available online 31 January 2015).

[14] Qunhui Li, Xuan Wang, Zhao Gao, Zhongtao Sun, Zhu Cui, Zhiqiang Duan, Juan Li, Min Gu, Xiaoquan Wang, Jiao Hu, Xiaowen Liu, Xiufan Liu, Novel reassortant H5N5 viruses bind to a human-type receptor as a factor in pandemic risk, Vet. Microbiol. 75 (2–4) (2015) 356–361.

[15] Enrico Angelelli, Renata Mansini, M. Grazia Speranza, Kernel search: a general heuristic for the multi-dimensional knapsack problem, Comput. Oper. Res. 37 (11) (2010) 2017–2026.

[16] Jonghyun Lee, Jin S. Lee, Heuristic search for scheduling flexible manufacturing systems using lower bound reachability matrix, Comput. Ind. Eng. 59 (4) (2010) 799–806.

[17] Hang Lei, Keyi Xing, Libin Han, Fuli Xiong, Zhaoqiang Ge, Deadlock-free scheduling for flexible manufacturing systems using Petri nets and heuristic search, Comput. Ind. Eng. 72 (2014) 297–305.

[18] E. Machuca, L. Mandow, Multiobjective heuristic search in road maps, Expert. Syst. Appl. 39 (7) (2012) 6435–6445.

[19] Seda Yanik, Burcin Bozkaya, Ronan deKervenoael, A new VRPPD model and a hybrid heuristic solution approach for e-tailing, Eur. J. Oper. Res. 236 (3) (2014) 879–890.

[20] Y. Kergosien, Ch Lenté, J.C. Billaut, S. Perrin, Metaheuristic algorithms for solving two interconnected vehicle routing problems in a hospital complex, Comput. Oper. Res. 40 (10) (2013) 2508–2518.

[21] Julia A. Bennell, Lai Soon Lee, Chris N. Potts, A genetic algorithm for two-dimensional bin packing with due dates, Int. J. Prod. Econ. 145 (2) (2013) 547–560.

[22] B. Naderi Hani Pourvaziri, A hybrid multi-population genetic algorithm for the dynamic facility layout problem, Appl. Soft Comput. 24 (2014) 457–469.

[23] G. Kanagaraj, S.G. Ponnambalam, N. Jawahar, A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems, Comput. Ind. Eng. 66 (4) (2013) 1115–1124.

[24] Xinchao Zhao, Wenqiao Lin, Qingfu Zhang, Enhanced particle swarm optimization based on principal component analysis and line search, Appl. Math. Comput. 229 (25) (2014) 440–456.

[25] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introduction Analysis with Application to Biology, Control and Artificial Intelligence, 2nd ed., MIT Press, Cambridge, 1992.

[26] A. Colorni, M. Dorigo, Distributed optimization by ant colonies, in: Proceedings of the 1st Europe Conference on Artificial Life, 1991, pp. 134–142.

[27] R. Eberhart, J. Kennedy, New optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS'95), IEEE, Piscataway, NJ, USA, 1995, pp. 38–43.

[28] Xiang Li, Gang Du, BSTBGA: a hybrid genetic algorithm for constrained multi-objective optimization problems, Comput. Oper. Res. 40 (1) (2013) 282–302.

[29] Julia A. Bennell, Lai Soon Lee, Chris N. Potts, A genetic algorithm for two-dimensional bin packing with due dates, Int. J. Prod. Econ. 145 (2) (2013) 547–560.

[30] B. Naderi Hani Pourvaziri, A hybrid multi-population genetic algorithm for the dynamic facility layout problem, Appl. Soft Comput. 24 (2014) 457–469.

[31] M.M. Raghuwanshi, O.G. Kakde, Distributed quasi steady-state genetic algorithm with niches and species, Int. J. Comput. Intell. Res. 3 (2) (2007) 155–164.
[32] Ioannis G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Appl. Math. Comput. 203 (2008) 598–607.
[33] Fang Gao, Pu Han, Yongjie Zhai, Ant colony algorithm with mutation operation for continuous function optimization, Comput. Eng. Appl. 47 (4) (2011) 5–8.
[34] Peter Korošec, Jurij Šilc, Bogdan Filipic, The differential ant-stigmergy algorithm, Inf. Sci. 192 (2012) 82–97.
[35] Zhihua Cui, Jianchao Zeng, Particle Swarm Optimization, Science Press, Beijing, 2011.
[36] M. Clerc, Particle Swarm Optimization, The Nether-lands: ISTE Publishing, Amsterdam, 2006.
[37] Zihui Ren, Jian Wang, Yuelin Gao, The global convergence analysis of particle swarm optimization algorithm based on Markov chain, Control Theory Appl. 28 (4) (2011) 462–466.
[38] Zahra Beheshti, Siti Mariyam Shamsuddin, Non-parametric particle swarm optimization for global optimization, Appl. Soft Comput. 28 (2015) 345–359.
[39] Yi Xiang, Yuming Peng, Yubin Zhong, et al., A particle swarm inspired multi-elitist artificial bee colony algorithm for real-parameter optimization, Comput. Optim. Appl. 57 (2014) 493–516.
[40] Samir Sayah, Abdellatif Hamouda, A hybrid differential evolution algorithm based on particle swarm optimization for nonconvex economic dispatch problems, Appl. Soft Comput. 13 (4) (2013) 1608–1619.
[41] Sourav Mallick, S.P. Ghoshal, P. Acharjee, S.S. Thakur, Optimal static state estimation using improved particle swarm optimization and gravitational search algorithm, Int. J. Electr. Power Energy Syst. 52 (2013) 254–265.
[42] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[43] A.K. Qin, P.N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, in: Proceedings of the 2005 IEEE Congress on Evolutionary Computation, 2005, pp. 1785–1791.
[44] K. Price, R. Storn, Differential evolution, Dr. Dobb's J. 22 (1997) 18–20,22,24,78.
[45] Wu Zhu, Research and Application of Differential Evolution based on Adaptive Population Tuning Scheme, Donghua University, Doctoral Dissertation, Shanghai, 2013.
[46] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Glob. Optim. 39 (3) (2007) 459–471.
[47] Marjan Mernik, Shih-Hsi Liu, Dervis Karaboga, Matej Črepinšek, On clarifying misconceptions when comparing variants of the Artificial Bee Colony Algorithm by offering a new implementation, Inf. Sci. 291 (2015) 115–127.
[48] D. Karaboga, B. Basturk, Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems, in: Proceedings of the 12th International Fuzzy Systems Association World Congress on Foundations of Fuzzy Logic and Soft Computing, 2007, pp. 789–798.
[49] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 241 (1) (2009) 108–132.
[50] Lei Wang, Jin Pan, Licheng Jiao, The immune algorithm, ACTA Electron. Sin. 28 (7) (2000) 74–78.
[51] Maojun Li, An Luo, Tiaosheng Tong, Artificial immune algorithm and its applications, Control Theory Appl. 21 (2) (2004) 153–157.
[52] Wang Lei, Pan Jin, Jiao Li-cheng, The immune algorithm, ACTA Electron. Sin. 28 (7) (2000) 74–78.
[53] Maojun Li, Jin Pan, Licheng Jiao, Artificial algorithm and its applications, Control Theory Appl. 21 (2) (2004) 153–157.
[54] Burnet, The Clonal Selection Theory of Acquired Immunity, Cambridge University Press, Cambridge, 1959.
[55] Castro De, Learning and optimization using the clone selection principle, IEEE Trans. Evol. Comput. 6 (3) (2002) 239–251.
[56] Licheng Jiao, Haifeng Du, Development and prospect of the artificial immune system, Acta Electron. Sin. 31 (10) (2003) 1540–1548.
[57] H. Beyer., The Theory of Evolution Strategies, Springer, New York, 2001.
[58] E. Mezura-Montes, C. Coello., A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Trans. Evol. Comput. 9 (2) (2005) 1–17.
[59] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.
[60] S. Christensen, F. Oppacher, What Can We Learn From No Free Lunch? A First Attempt to Characterize the Concept of a Searchable Function, Proceedings of GECCO, Morgan Kaufmann (2001), p. 1219–1226.
[61] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming—Theory and Algorithms, John Wiley & Sons, New York, 1993.
[62] Xiaolei Li, Zhijiang Shao, Jixin Qian, An optimizing method based on autonomous animats: fish-swarm algorithm, Syst. Eng. Pract. 22 (11) (2002) 32–38.
[63] X.S. Yang, A New Metaheuristic Bat-Inspired Algorithm, Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Studies in Computational Intelligence 284, Springer-Verlag, Berlin Eidelberg (2010), p. 65–74.
[64] G. Kanagaraj, S.G. Ponnambalam, N. Jawahar, A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems, Comput. Ind. Eng. 66 (4) (2013) 1115–1124.
[65] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, IEEE Publications, Coimbatore, India (2009), p. 210–214 , World Congress on Nature and Biologically Inspired Computing.
[66] X.S. Yang, S. Deb, Engineering optimization by cuckoo search, Int. J. Math. Model. Numer. Optim. 1 (4) (2010) 330–343.
[67] K. Krishnanand, D. Ghose, Theoretical foundations for rendezvous of glowworm-inspired agent swarms at multiple locations, Robot. Auton. Syst. 56 (7) (2008) 549–569.
[68] K.N. Krishnanand, D. Ghose, A glowworm swarm optimization based multi-robot system for signal localization, Des. Control Intell. Robot. Syst. (2009) 53–74.
[69] Vinicius W.C. Morais, Geraldo R. Mateus, Thiago F. Noronha, Iterated local search heuristics for the vehicle routing problem with cross-docking, Expert. Syst. Appl. 41 (16) (2014) 7495–7506.
[70] Nelson Chibeles-Martins, Tania Pinto-Varela, Ana Paula Barbosa-Póvoa, A. Q. Novais, Multi-objective meta-heuristic approach supported by an improved local search strategy for the design and planning of supply chain networks, Comput. Aided Chem. Eng. 33 (2014) 313–318.
[71] T. Back, Evolutionary Algorithms in Theory and Practice, Oxford University Press, Oxford, U.K., 1996.
[72] Z. Cai, Y. Wang, A multiobjective optimization-based evolutionary algorithm for constrained optimization, IEEE Trans. Evol. Comput. 10 (2006) 658–675.
[73] Mohammed El-Abd, Performance assessment of foraging algorithms vs. evolutionary algorithms, Inf. Sci. 182 (2012) 243–263.
[74] Ling Wang, Ruixin Yang, Yin Xu, et al., An improved adaptive binary harmony search algorithm, Inf. Sci. 232 (2013) 58–87.
[75] Zhigang Wang, Xiaoming Fan, Qixing Han, Global stability of deterministic and stochastic multigroup SEIQR models in computer network, Appl. Math. Model. 37 (20–21) (2013) 8673–8686.
[76] Yongzhen Pei, Shaoying Liu, Shujing Gao, Shuping Li, Changguo Li, A delayed SEIQR epidemic model with pulse vaccination and the quarantine measure, Comput. Math. Appl. 58 (1) (2009) 135–145.
[77] Cagri Ozcaglar, Amina Shabbeer, Scott L. Vandenberg, B.ülent Yener, Kristin P. Bennett, Epidemiological models of *Mycobacterium tuberculosis* complex infections, Math. Biosci. 236 (2) (2012) 77–96.
[78] Bimal Kumar Mishra, Navnit Jha, SEIQRS model for the transmission of malicious objects in computer network, Appl. Math. Model. 34 (3) (2010) 710–715.
[79] Herbert Hethcote, Ma Zhien, Liao Shengbing, Effects of quarantine in six endemic models for infectious diseases, Math. Biosci. 180 (1–2) (2002) 141–160.
[80] W.O. Kermack, A.G. Mckendrick. Contributions to the mathematical theory of epidemics, in: Proceedings of the Royal Society of London, 1927, A115, pp. 700–721.
[81] W.O. Kermack, A.G. Mckendrick, Contributions to the mathematical theory of epidemics, in: Proceedings of the Royal Society of London, 1932, A138, pp. 55–83.
[82] Association of China Field statistics Research, The Orthogonal Table Method and Three-time Design, Science Press, Beijing, 1987.
[83] M. Iisufescu, Finite Markov Processes and Their Applications, Wiley, Chichester, 1980.
[84] Peter Korošec, Jurij Šilc, Bogdan Filipic, The differential ant-stigmergy algorithm, Inf. Sci. 192 (2012) 82–97.
[85] J.J. Liang, B.Y. Qu, P.N. Suganthan, Alfredo G. Hernández-Díaz, Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization, Nanyang Technological University, Tech. Rep., 2013, Available in ⟨http://www3.ntu.edu.sg/home/EPNSugan/index_files/CEC2013/Definitions%20of%20CEC%2013%20benchmark%20suite%200117.pdf⟩.
[86] Guangqiu Huang, Tao Li, Qiuqin Lu, Artificial memory-based optimization, Syst. Eng. — Theory Pract. 34 (11) (2014) 2900–2913.
[87] Guangqiu Huang, Siya Sun, Qiuqin Lu, Artificial food chain-based animal swarm optimization algorithm, Appl. Res. Comput. 31 (9) (2014) 2673–2680.
[88] Guangqiu Huang, Xiaolong Xu, Qiuqin Lu, Ecotoxicology dynamics-based optimization with impulsive toxicant input, Comput. Sci. 41 (8) (2014) 254–262.
[89] J. Derrac, S. Garcia, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.