PAPER

A Multi-Learning Immune Algorithm for Numerical Optimization

Shuaiqun WANG[†], ***Nonmember***, **Shangce GAO**[††,†††a)], ***Member***, **Aorigele**[†††], **Yuki TODO**[††††], ***Nonmembers***, ***and* Zheng TANG**[†††], ***Member***

SUMMARY The emergence of nature-inspired algorithms (NIA) is a great milestone in the field of computational intelligence community. As one of the NIAs, the artificial immune algorithm (AIS) mimics the principles of the biological immune system, and has exhibited its effectiveness, implicit parallelism, flexibility and applicability when solving various engineering problems. Nevertheless, AIS still suffers from the issues of evolution premature, local minima trapping and slow convergence due to its inherent stochastic search dynamics. Much effort has been made to improve the search performance of AIS from different aspects, such as population diversity maintenance, adaptive parameter control, etc. In this paper, we propose a novel multi-learning operator into the AIS to further enrich the search dynamics of the algorithm. A framework of embedding multiple commonly used mutation operators into the antibody evolution procedure is also established. Four distinct learning operators including baldwinian learning, cauchy mutation, gaussian mutation and lateral mutation are selected to merge together as a multi-learning operator. It can be expected that the multi-learning operator can effectively balance the exploration and exploitation of the search by enriched dynamics. To verify its performance, the proposed algorithm, which is called multi-learning immune algorithm (MLIA), is applied on a number of benchmark functions. Experimental results demonstrate the superiority of the proposed algorithm in terms of convergence speed and solution quality.

key words: *artificial immune algorithm, mutation operators, immune algorithm, hybridization, multi-learning*

1. Introduction

Nature-inspired algorithms (NIA, including evolutionary algorithms, swarm intelligence, artificial immune systems, simulated annealing, etc.) have received much attention regarding their potential as complex problem-solving techniques [1], [2]. The main idea of NIA is to develop computational algorithms by taking inspiration from nature (either biology or physics) for the solution of complex problems. Among all nature-inspired algorithms, the artificial immune system (AIS) [3]–[6], which is inspired by the theory of biological immune system, is one of the recently developed population based problem-solving techniques. AIS mimics the mechanisms of the biological (either innate or adaptive) immune response, which depicts the procedures of responses when a biological immune system is exposed to an antigen. The most commonly used mechanisms of the biological immune response are clonal proliferation [7], negative selection [8], immune network [9], danger theory [10], and dendritic cell model [11]. Until now, AIS has been successfully applied on various complex problems, such as traveling salesman problems [12], [13], automatic clustering [14], pattern classification [15], graph drawing problems [16], mobil robot control [17], numerical optimization problems [18], and so on.

For solving optimization problems, AIS utilizes a collective learning process of a population of antibodies, and undergoes a cycle process of clonal proliferation, maturation and antibody selection. According to a fitness function, the clonal proliferation in AIS favors better antibodies to reproduce more often than those that are relatively worse. During the period of maturation, descendants of antibodies are generated using randomized learning operators. Thereafter, fitter antibodies are selected to be reserved to enter into the next generation. Although AIS has exhibited its promising applicability in solving difficult optimization problems, its performance is limited with the increment of dimension or multimodality of the problem [19], indicating that AIS still suffers from the traditional drawbacks of optimization algorithms, such as search stagnation, evolution premature, local minima trapping, parameter tuning and slow convergence [20].

To alleviate these drawbacks of AIS, much effort has been made and a number of AIS variants have been proposed. By revising the representation method of solutions, Jiao et al. [21] proposed a quantum encoding-based AIS using quantum rotation gates to avoid premature convergence. Aiming to resolve the parameter tuning problem, Garrett [22] proposed a parameter-free AIS by removing the user-defined parameters of population size, clonal size, and the amount of mutations of clones. To enhance the exploration of global and local optima, especially to lead antibodies to unexplored areas, a vaccine injected AIS was proposed in [23] where vaccines are extracted from equally divided decision space of the problem. To speed up the convergence of each antibody and reduce the computational effort necessary to simulate the whole population, a cluster and gradient based AIS was proposed in [18]. Other attempts to develop AIS in optimization scenarios are made to com-

Manuscript received August 22, 2014.

[†]The author is with the College of Electronics & Information Engineering, Tongji University, No. 4800 Cao An Road, 201804, Shanghai, China.

[††]The author is with the College of Information Sciences and Technology, Donghua University, No. 2999 North Renmin Road, 201620, Shanghai, China.

[†††]The authors are with the Faculty of Engineering, University of Toyama, Toyama-shi, 930-8555 Japan.

[††††]The author is with the School of Electrical and Computer Engineering, Kanazawa University, Kanazawa-shi, 920-1192 Japan.

a) E-mail: gaosc@dhu.edu.cn

DOI: 10.1587/transfun.E98.A.362

bine AIS with one of the other intelligent algorithms, such as simulated annealing [24], [25], ant colony optimization [26], genetic algorithm [27], fuzzy self-organized network [28], particle swarm optimization [29], and so on.

In this paper, we tried to develop AIS by enriching its searching dynamics, i.e. developing a novel learning operator to mature the antibodies in the population. By doing so, it was expected that the learning capacity of AIS can be improved. In optimization scenarios, learning capacity is the most critical component of an optimization algorithm [30]. It determines the search scope and search efficiency of the current population. A well-designed learning operator can not only guide the search to promising areas which are near to the global optima with a high probability, but also reduce the number of useless or redundant search times. In the literature, there are various learning (mutation) operators [31], wherein gaussian mutation [32], cauchy mutation [32], [33], lateral mutation [34], [35] and baldwinian learning [36] are commonly used by many researchers due to their simplicities and easy implementations. However, the efficiencies of these operators are problem-independent. In other words, each learning operator is designed for specific problems with distinct characteristics. For example, the gaussian mutation is more capable of exploitation in small regions of a smooth decision space, while cauchy mutation enables the search to carry out long distance jumps, thus specializing in exploring unvisited regions [32], [37]. As a result, cauchy mutation is more suit to optimize the problems with plenty of local optima. After realizing this, Khilwani et al. [38] has proposed a fast AIS by combining gaussian mutation with cauchy mutation for an expectation of mixing the search dynamics of both ones. But the question remains if such fusion of two different learning operators is sufficient to improve the search performance of AIS.

Intuitively, the combination of more than two different learning operators is a feasible method to further enhance the learning capacity of the algorithm. Based on the analysis of the properties of each learning operator (i.e. the above mentioned four operators), we propose a framework to embed all selected operators into a union, thus forming a novel multi-learning operator to perform the search procedure of the algorithm. In this framework, each selected single operator is assigned a probability of being implemented, and the cumulative probability is used to control which operator should be carried out to perform the search. It can be expected that the multi-learning operator can take advantage of the search properties of each single learning operator, thus possessing a more abundant search dynamics. Therefore, the resultant multi-learning immune algorithm (MLIA) is more capable of balancing the exploration and exploitation of the search. Experimental results based on fifteen numerical benchmark functions verified the superiority of MLIA.

The rest of the paper is organized as follows: Sect. 2 describes the generic immune algorithm and four commonly used learning operators. Section 3 elaborates more about the multi-learning operator and MLIA. Section 4 describes the experiment based on several benchmark functions including comparative results between MLIA and other variants of AIS. Finally, concluding remarks are presented in Sect. 5.

2. Learning Operators in AIS

Artificial immune algorithms are a special class of biologically inspired algorithms, which are based on the biological immune system of vertebrates and derive from various immunological theories, namely the clonal selection principle, negative selection, immune networks or the danger theory [4], [5]. Besides the natural tasks of anomaly detection and classification, they are often applied to function optimization. In this context, mostly algorithms are based on the principles of clonal selection and antibody maturation in the adaptive immune response [39].

The generic computational framework of AIS is depicted in Fig. 1. From this figure, it is clear that the quality of mutated populations is determined by the capacity of learning operators [31], [40], thus directly influencing the performance of the algorithm. Many researchers have designed and investigated an amount of learning operators, some of which are widely used in evolutionary algorithms [32], [33], while others are specifically designed based on the mechanisms in biological immune systems [34], [36], [41]. It should be noted that it is not our task in this work to make a comprehensive review of all learning operators used in AIS. Instead, some of the typical operators are selected and investigated to be used in the proposed MLIA. The rules to select learning operators in AIS are two: first, it should be widely used in AIS community and also has been demonstrated to be effective; second, sophisticated operators are ignored due to their hard programming and implementation.

In this section, four learning operators including gaussian mutation (*GM*), cauchy mutation (*CM*), lateral mutation (*LM*) and baldwinian learning (*BL*) are briefly introduced. For the sake of simplification of the description, we unify the representation of the current population of antibodies to be $\{X_1, ..., X_i, ..., X_N\}$, where the i-th antibody

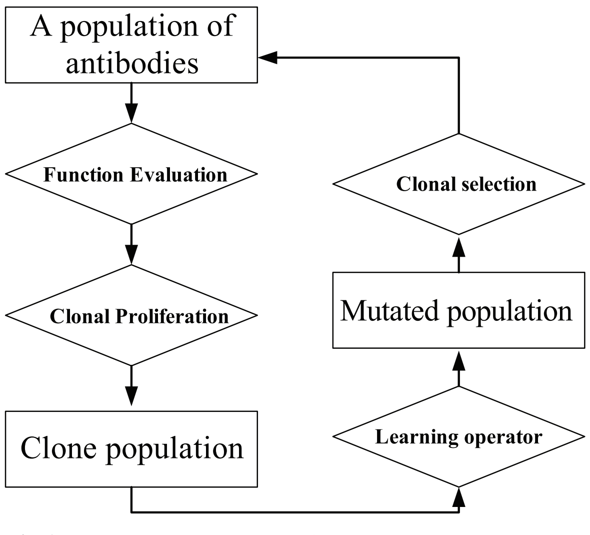


Fig. 1 Computational procedure for a generic immune algorithm.

$X_i = (x_{i1}, ..., x_{ij}, ..., x_{iD})$, N is the number of antibodies in the population, and D denotes the dimension of the optimization problem. Manipulated by learning operators, the mutated population is represented by $\{X'_1, ..., X'_i, ..., X'_N\} = L\{X_1, ..., X_i, ..., X_N\}$, where $X'_i = (x'_{i1}, ..., x'_{ij}, ..., x'_{iD})$, L represents the learning operator, and $L \in \{GM, CM, LM, BL\}$.

2.1 Gaussian Mutation

The gaussian mutation (*GM*) has two parameters: the mean value μ and the standard deviation σ, which are used to determine the step size for the mutation. The one-dimension gaussian density function with the mean value μ and the standard deviation σ is defined by:

$$f_{gau}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{1}$$

In most studies, the simplified mutation strategy of *GM* based on the origin-centered gaussian density function with $\mu = 0, \sigma = 1$ is used [32], [33], [38]. The *GM* is implemented in the learning procedure based on the following equations:

$$x'_{ij} = x_{ij} + s_g N_j(0, 1), \quad \text{for } j = 1, ..., D \tag{2}$$

where $N_j(0, 1)$ is the normally distributed gaussian random number with the mean value of 0 and the standard deviation of 1. The symbol i denotes the serial number of the antibody in the population, while j denotes the dimension. s_g is the step size of mutation, and it is generated by [32], [38]:

$$s_g = random(+, -)\sqrt{2\ln(w_g\sqrt{2\pi})} \tag{3}$$

where w_g is a random number uniformly generated in the range of $[0, f_{gau}(0)]$, and $random(+, -)$ returns a uniformly generated negative or positive sign.

2.2 Cauchy Mutation

The cauchy mutation (*CM*) is mainly based on the cauchy distribution which is shown in Eq. (4).

$$f_{cauchy}(x) = \frac{1}{\pi}\frac{1}{1 + x^2} \tag{4}$$

The important characteristic of cauchy distribution is that its expectation does not exist [32], [38]. The shape of $f_{cauchy}(x)$ is exactly similar with the gaussian density function but approaches the axis very slowly. The *CM* is implemented as follows [38]:

$$x'_{ij} = x_{ij} + s_c\delta_j, \quad \text{for } j = 1, ..., D \tag{5}$$

where δ_j is a cauchy random variable, s_c is the step size of cauchy mutation which is generated by:

$$s_c = random(+, -)\sqrt{\frac{1}{w_c\pi} - 1} \tag{6}$$

where s_c is a random number uniformly generated in the range of $[0, f_{cauchy}(0)]$.

2.3 Lateral Mutation

In immune systems, the lateral interaction during different antibodies usually takes place according to the idiotypic network theory [9], [42]. In other words, each paratope on an antibody can not only recognize a foreign antigen, but also can be recognized by external idiotopes. Motivated by this mechanism, the lateral mutation (*LM*) [34], [35], [43] is implemented as:

$$x'_{ij} = (1 - \beta)x_{ij} + \beta x_{kj}, \quad \text{for } j = 1, ..., D \tag{7}$$

where $k \in \{1, 2, ..., N\}$ and $k \neq i$. The learning rate $\beta \in (0, 1)$ is a randomly generated real number.

2.4 Baldwinian Learning

Learning mechanism can provide an easy evolutionary path towards co-adapted alleles in environments, by means of employing differential information during other antibodies [36]. It is realized as: for $j = 1, ..., D$

$$x'_{ij} = \begin{cases} x_{ij} + s \cdot (x_{pj} - x_{qj}) & \text{if } rand() \leq Q \\ x_{ij} & otherwise \end{cases} \tag{8}$$

where $p, q \in \{1, 2, ..., n\}$, $p \neq q \neq i$, s indicates the strength of baldwinian learning, $rand()$ is a random number uniformly generated in the interval of $[0, 1]$. $Q \in (0, 1]$ controls the probability of the *BL* to be implemented. In our experiments, the value of Q is set to be 0.8 as suggested in [36].

2.5 Search Dynamics Analysis

Intuitively, all the above four learning operators are able to evolve antibodies into matured ones in semi-blind manners, although some of the matured ones might possess lower affinities. However, due to the parallel feature of the immune algorithm, there does exist a probability of making progress to improve the affinity of antibodies. After the clonal selection progress, the most improved antibodies are reserved and enter into the next generation of evolution. In Fig. 2, we summarize the characteristics of the learning operators. The solid rectangle S shows the solution space of the optimization problem. The dashed circles denote contour lines of affinity, and the inner circles indicate that they represent higher affinities than the outer ones. From Fig. 2, we can see that the learning mechanisms used in *GM* and *CM* on the antibody X_i only utilize random perturbation on the antibody itself, while those in *LM* and *BL* make use of information in the environment.

As noticed by Yao et al. [32], *CM* is more likely to generate an offspring that is far away from its parent than *GM* due to its long flat tails of the density function. It is more likely to carry out large variations in antibodies due

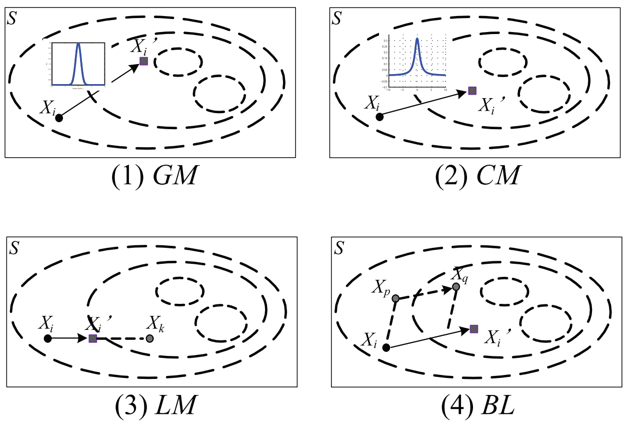


Fig. 2 The learning characteristics of the operators.

to its chance to execute longer jumps with higher probability, thus often being used to act as a learning method that can solve the local optima trapping problem. On the other hand, as reported in [34], [36], learning from the environment provides an encouraging alternative method, probably a more easy way to achieve better search performance. In details, the mechanism in *LM* uses the information of a randomly selected antibodies in the population to guide the current search. A successful guide is strongly depending on the quality of the selected guiding antibody X_j, implying that there must be amount of redundant search if the guiding antibody is far away from the global optimal solution. Instead, the mechanism in *BL* utilizes the differential information between two other antibodies X_p and X_q in the population. The learning acting on this differential information might have ability to use the mutual beneficial components, thus exhibiting more promising properties for searching.

3. Multi-Learning Immune Algorithm

The analysis of the search characteristics of the four learning operators indicates that different learning strategy strongly influences the performance of the algorithm. In order to improve the search effectiveness of the learning operator, a multi-learning operator which actually is a hybridization of different existing learning operators is proposed. Due to the fact that different learning operators have distinct search dynamics, the multi-learning operator can be expected to possess a more abundant search dynamics. Before describing the proposed multi-learning immune algorithm (MLIA), some immunological concepts and metaphors are introduced to make this paper self-explanatory.

3.1 Immunological Inspirations

The biological immune system is a complex pattern recognition and optimization device with the main goal of protecting our body from malefic external invaders, called *antigens*. The primary elements are the *antibodies*, which bind to antigens for their posterior destruction by other cells.

Table 1 Corresponding relationships (i.e. Metaphors) between MLIA and inspirations by immune system.

MLIA	Immune Inspirations
Problem	Antigen
Solution (Candidate Solution)	Antibody (B Cell)
Population Diversity	Immune Diversity
Quality of Solutions	Affinity
Solution Replication	Clonal Proliferation
Solution Improvement	Affinity Maturation
Learning Operators	Antibody Mutations

Affinity is the key measure to represent the fitness of antibody to antigen. When there are detected antigens, the immune system will choose B cells with higher affinity to proliferate, which is called clonal selection and proliferation. When the antigen are eliminated, the B cells with lower affinity will be chosen for elimination. The two procedures make the antibody population stable. Although the repertoire of antibodies in the immune system is limited; though affinity maturation, it is capable of evolving antibodies to successfully recognize and bind with known and unknown antigens, leading to their eventual elimination. Thus, a rapid accumulation of mutations of the antibody is necessary for a fast maturation of the immune response. Table 1 depicts the corresponding relationships between MLIA and inspirations by the biological immune system.

Mathematically, antigens which refer to the optimization problem itself can be formulated as (without loss of generality, a minimization problem is considered in this study):

$$minimize : f(X) \tag{9}$$

where $X \in \Omega$, X is a variable vector in R^D, $\Omega \sqsubseteq R^D$, Ω defines the feasible solution space and $f(X)$ is the objective function which calculates out the quality of the candidate solutions. An antibody represents a candidate solution of the problem, and its representation $X_i = (x_{i1}, ..., x_{ij}, ..., x_{iD})$ can be in forms of binary strings, symbolic sequences, real-valued number sequences [38] or quantized values [21]. In this paper, we adopt the real-valued number representation.

3.2 Framework of the Multi-Learning Operator

In order to merge multiple different learning operators together, we propose a general framework to realize this. Assume that there are K learning operators which will be combined together, represented by LO_i ($i = 1, 2, ..., K$). First, each embedded single learning operator LO_i is assigned a probability $Prob_i$ of being implemented, then the cumulative probability $\sum_{j=1}^{i} Prob_j$ is used to control which operator should be carried out to perform the search. The general framework can be illustrated as in Fig. 3, where $q = rand() \in (0, 1)$ is a random number, and $\sum_{j=1}^{K} Prob_j = 1$. In this framework, each embedded single learning operator LO_i will be carried out if its corresponding probability $Prob_i \neq 0$, inferring that the search dynamic of this operator contributes to the search of the whole algorithm. On the

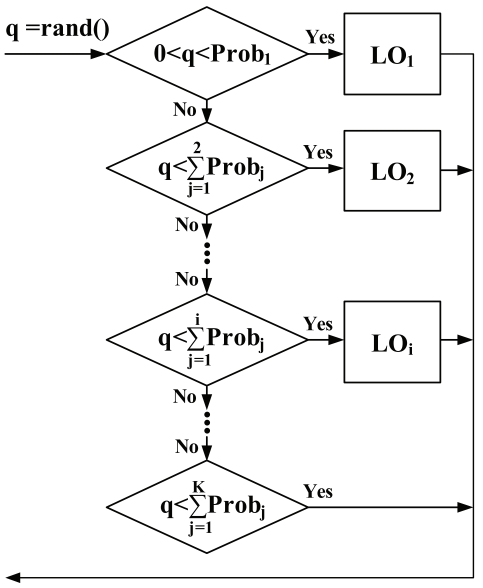


Fig. 3 General framework of the multi-learning operator.

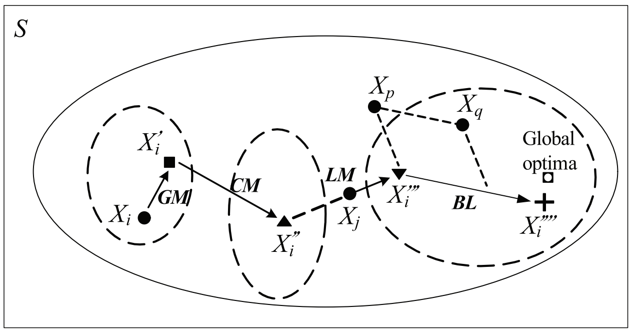


Fig. 4 The learning characteristics of the multi-learning operator.

other hand, due to the stochastic nature of q, the single operator is implemented randomly rather than deterministically, in spite of the deterministic setting values of implementation probabilities. Additionally, it is flexible to adjust the implementation probabilities of each single learning operator, suggesting that the search dynamics of the resultant multi-learning operator can be well tuned by resetting the values of these probabilities. As discussed at the previous section, the adopted four learning operators (*GM*, *CM*, *LM*, *BL*) have their distinct search dynamics. It worth emphasizing that any other learning operator in AIS or evolutionary computation communities can also be employed as a embedded single learning operator in the general multi-learning operator construction framework. In this research, only four typical ones (i.e., $K = 4$) are investigated and utilized to merger together as a novel multi-learning operator (*ML*). The basic idea behind this hybridization is illustrated as in Fig. 4.

As shown in Fig. 4, the multi-learning operator *ML* consists of four single learning operators, i.e., *GM*, *CM*, *LM*, *BL*. The current antibody X_i mutates to X_i' by *GM*, to X_i'' by *CM*, to X_i''' by *LM*, and finally to X_i'''' by *BL*. In each iteration of the antibody maturation, only one of the learning operators is carried out; however, after a large number of iterations, each single learning operator will be implemented with plenty times, implying that *ML* will possess all the search dynamics of embedded single learning operators. To conclude, the search characteristics of *ML* can be remarked as:

- *ML* has a capacity of carrying out fine-grained searches within a relative small neighborhood of the antibody, which inherits from the exploitation ability of *GM*.
- *ML* is capable to perform coarse-grained searches in a wider neighborhood of the antibody due to occasional long-distance mutations from the *CM*, and sometimes these long-distance mutations are beneficial to make the algorithm jump out local optima.
- *ML* is able to use information from the environment by *LM*, i.e., from a guided antibody X_j. If the guide antibody is already near the global optima, such learning is very effective to accelerate the convergence speed, and meanwhile to greatly improve the average quality of the solutions.
- *ML* can also utilize some specific differential information from the environment by *BL*, i.e. by the differential information of other two trial antibodies $X_q - X_p$, indicating that a more effective way to learn from the environment can be realized [36].

The pseudocode of implementing the multi-learning operator *ML* is shown in Algorithm 1.

Algorithm 1– **Implementation of *ML* in MLIA**

```
Begin-learning: Input an antibody $X_i$
initialize $Prob_{GM}$, $Prob_{CM}$, $Prob_{LM}$ and $Prob_{BL}$
set $q = rand()$
If $0 < q < Prob_{GM}$, execute the gaussian mutation GM
Else-If $q < Prob_{GM} + Prob_{CM}$, execute the cauchy mutation CM
Else-If $q < Prob_{GM} + Prob_{CM} + Prob_{LM}$, execute the literal mutation LM
Else execute the Baldwinian learning BL
End-If
End-learning: Output the matured antibody $X_i'$
```

3.3 The Whole Procedure of MLIA

The complete procedure of the proposed algorithm MLIA can be represented as in Fig. 5.

Step 1. Set all user-defined parameters in MLIA;

Step 2. A number of N antibodies X_i ($i = 1, 2, ..., N$) are randomly generated within the search space $[low_i, up_i]^D$, where D is the dimension of the optimization problem. Each antibody is represented by a real-valued vector $X_i = (x_{i1}, ..., x_{ij}, ..., x_{iD})$ using the following formula:

$$x_{ij} = low_i + \mu(up_i - low_i) \tag{10}$$

where $\mu \in [0, 1]$ is a random number with the uniform distribution. low_i and up_i is the lower and upper bounds of the search space respectively.

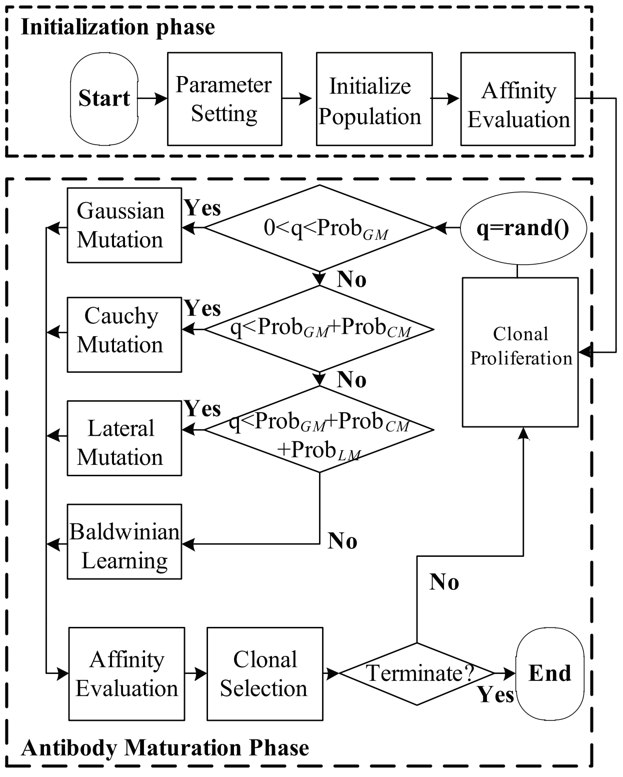


Fig. 5 Flowchart of the proposed algorithm.

Step 3: Evaluate affinity $f(X_i)$, $i = 1, ..., N$, for each antibody in the antibody population.

Step 4: All antibodies are undergone the clonal proliferation procedure. First, sort all antibodies in an ascending order, i.e., $f(X_1) < f(X_2) < ... < f(X_N)$, for the minimization problems. Then, for each antibody X_i, asexually produce m_i clones where

$$m_i = \left\lceil M * \frac{N-i}{N} \right\rceil \tag{11}$$

Here, M is the multiplying factor which determines the scope of the clone and $\lceil . \rceil$ is the operator which returns the nearest integer greater than or equal to the argument. It is clear that the amount of generated clones of an antibody is inversely proportional to its affinity. In other words, the better the antibody, the more clones it produces. After this step, we can obtain $\sum m_i$ antibodies just as $(X_{1,1}, X_{1,2}, ..., X_{1,m_1}; ...; X_{N,1}, X_{N,2}, ..., X_{N,m_N})$;

Step 5: Apply the multi-learning operator *ML* on each clone of the antibody using the procedures in Algorithm 1. After learning, $\sum m_i$ mutated antibodies are generated.

Step 6: Evaluate the mutated antibodies according to the affinity function.

Step 7: The clonal selection procedure utilizes the elite reservation strategy. The fittest antibodies Y_i $(i = 1, 2, ..., N)$ of all the clones of each parent antibody are firstly selected, i.e., $Y_i = X_{i,j} = min_j\{f(X_{i,1}), ..., f(X_{i,j}), ..., f(X_{i,m_i})\}$. Then, a hill climbing update rule [3], [42] is used to replace the parent antibodies X_i with selected clones Y_i according to a updating probability $P(X_i \leftarrow Y_i)$:

$$P = \begin{cases} 1 & \text{if} \quad f(Y_i) < f(X_i) \\ 0 & \text{if} \quad f(Y_1) \geq f(X_1) \\ \exp(\frac{f(X_i)-f(Y_i)}{\alpha}) & \text{otherwise} \end{cases} \tag{12}$$

Based on Eq. (12), if the fittest antibody of the clones Y_i has a smaller affinity (i.e. better quality for the minimization problems) than its parent antibody X_i, update it with probability "1". As a result, the elites in the offspring have been preserved and enter into the following generation. On the contrary, if Y_i is not better than its parent, then update according to an exponential function to maintain the diversity of the population, where α is a positive value related to the diversity. Generally, the better the diversity is, the bigger α is, and vice versa. Furthermore, in order to save the information of the original population such that the best antibody during the parents could not be replaced, the exponential function is not used for X_1.

Step 8: If the termination condition (in most cases the maximum iteration number T_{max}; or some special ones which will be described in Sect. 4.5) is not fulfilled, goto Step 4. Otherwise, output the best antibody in the current population.

4. Experimental Studies

To evaluate the performance of MLIA, fifteen benchmark functions are chosen. The algorithms are implemented in Microsoft Visual Studio 2005 and run on a personal computer with Intel Core(TM)2 Duo CPU 2.10 GHz and 3 GB memory. All simulation results are based on 30 runs to make some statistical analyses. In this section, the benchmark functions are presented. Then, the parameter settings and the benefit of *ML* for MLIA are discussed. Finally, the algorithms chosen for comparison are given, and the simulation results obtained from different experimental studies are also analyzed.

4.1 Benchmark Functions

The description of the selected fifteen benchmark functions is given in Table 2, where the function definition gives the affinity formula of the optimization problem. The symbol "Dim." denotes the dimension of the function. The domain $S = [low, up]^D$ represents the search space of feasible solutions for the D dimension problem. f_{min} is the known affinity of the global optimal solution.

The selected functions are widely used in evaluating global numerical optimization algorithms [32], [44], [45]. Furthermore, these functions are used to evaluate the performance of the algorithm for giving a generalized conclusion [46]. The first eight functions (f_1 to f_8) are high-dimension problems, where functions f_1 and f_2 are unimodal functions; the function f_3 is a step function; functions f_4 to f_8 are multimodal functions with plenty of local minima and the number of the local minima in these functions increases exponentially with the dimension of the function. In addition, functions f_9 to f_{15} are low-dimension functions that only

Table 2 Benchmark problems used in the experiments.

		Function Definition	Dim.	Domain $[low, up]^D$	f_{min}
$f_1(X)$	=	$\sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^D$	0
$f_2(X)$	=	$\sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	$[-10, 10]^D$	0
$f_3(X)$	=	$\sum_{i=1}^{n}(\lfloor x_i + 0.5\rfloor)^2$	30	$[-100, 100]^D$	0
$f_4(X)$	=	$\sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i\rvert})$	30	$[-500, 500]^D$	$-418.9829D$
$f_5(X)$	=	$\sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]^D$	0
$f_6(X)$	=	$-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})$ $-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	30	$[-32, 32]^D$	0
$f_7(X)$	=	$\frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600, 600]^D$	0
$f_8(X)$	=	$\frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})]$ $+(y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \ge a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	30	$[-50, 50]^D$	0
$f_9(X)$	=	$[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}]^{-1}$	2	$[-65.536, 65.536]^D$	0.998
$f_{10}(X)$	=	$4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^D$	−1.0316285
$f_{11}(X)$	=	$(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi}\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
$f_{12}(X)$	=	$[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]\times$ $[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2]$	2	$[-2, 2]^D$	3
$f_{13}(X)$	=	$-\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]^D$	−10.1422
$f_{14}(X)$	=	$-\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]^D$	−10.3909
$f_{15}(X)$	=	$-\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]^D$	−10.53

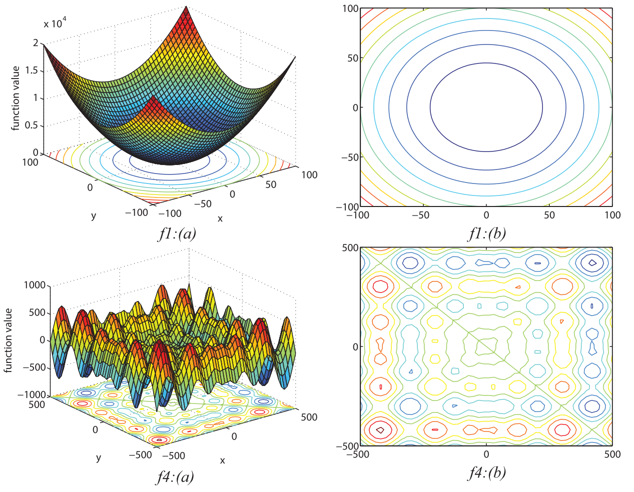


Fig. 6 The 2-dimension sketch (a) and the contour (b) for the unimodal function f_1 and the multimodal function f_4 respectively.

have a few local minima. Figure 6 depicts the characteristics of the unimodal function f_1 and the multimodal function f_4 respectively, in terms of the two-dimension sketch and the contour. The different types of benchmark functions can test the searching ability of learning operators from different aspects: unimodal functions trend to reflect the convergence speed of the operator in a direct manner, while multimodal functions are likely to estimate the operator's capacity of escaping from local optima.

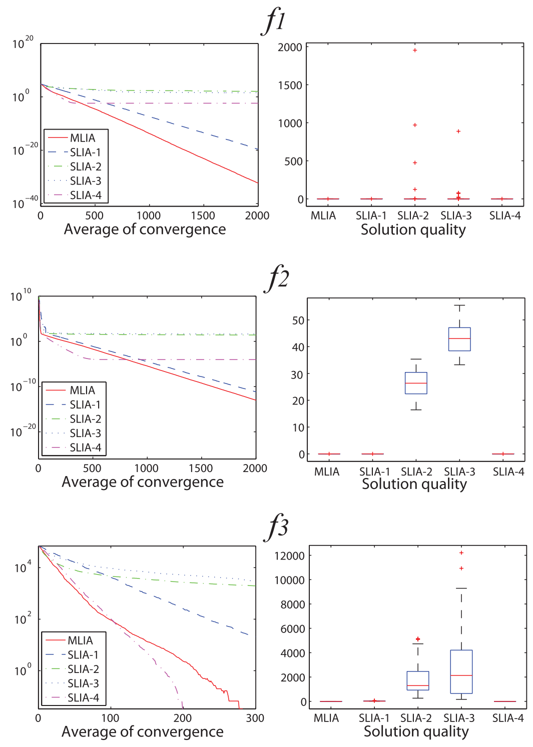


Fig. 7 The comparative results of the convergence graphs and box-and-whisker diagrams of solutions during MLIA, SLIA-1, SLIA-2, SLIA-3 and SLIA-4 for high dimensional unimodal benchmark functions f_1, f_2 and the step function f_3.

4.2 Verification of Search Dynamics in Learning Operators

The first set of experiments was aimed to compare the search dynamics of the multi-learning operator and single learning operators. To realize this, we constructed four variants of MLIA using only one single learning operator by turns. The resultant variants including SLIA-1, SLIA-2, SLIA-3, and

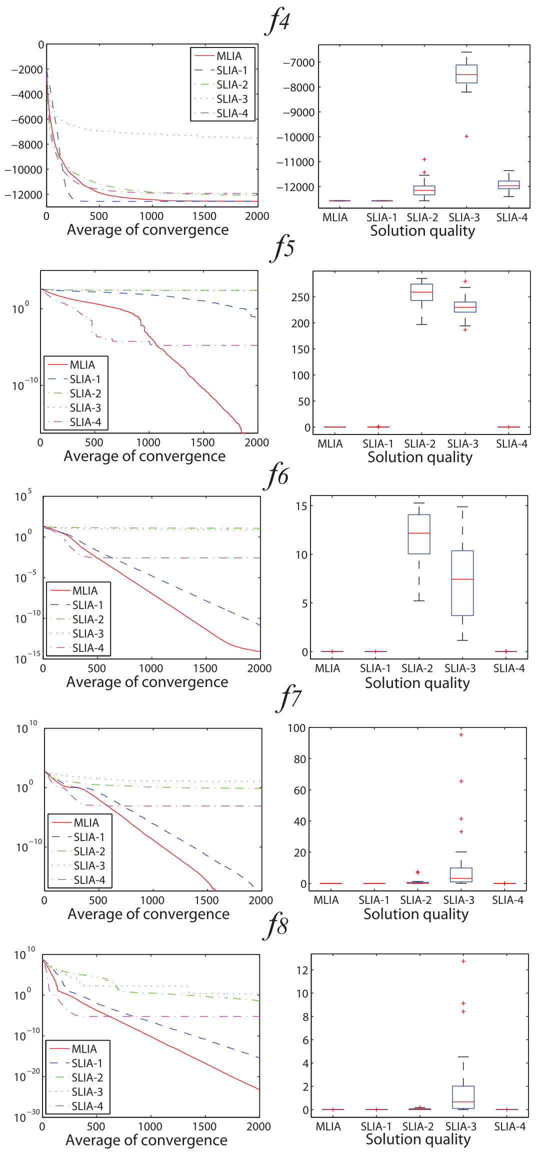


Fig. 8 The comparative results of the convergence graphs and box-and-whisker diagrams of solutions during MLIA, SLIA-1, SLIA-2, SLIA-3 and SLIA-4 for high dimensional multimodal benchmark functions f_4 - f_8.

SLIA-4 denote the utilization of *BL*, *CM*, *GM* and *LM* in the algorithm respectively. Figures 7–9 depict the comparative results by means of convergence graphs and box-and-whisker diagrams of solutions for all tested problems. Table 3 summarizes the statistical results obtained by the compared algorithms. It should be noted that the results obtained are under the settings for parameters by $Prob_{GM} = 0.1$, $Prob_{CM} = 0.1$, $Prob_{LM} = 0.4$, $Prob_{BL} = 0.4$ in MLIA, and for the rest parameters by $N = 30$, $M = 5$, $\alpha = 100$, and T_{max} equals to 2000 for high-dimensional problems, to 100 for low-dimensional problems in all compared algorithms. In addition, the parameter settings $Prob_{GM} = 0.1$, $Prob_{CM} = 0.1$, $Prob_{LM} = 0.4$, and $Prob_{BL} = 0.4$ is a reasonable choice which will be explained in Sect. 4.3.2 in detail.

From Figs. 7–9, it is apparent that MLIA performs bet-

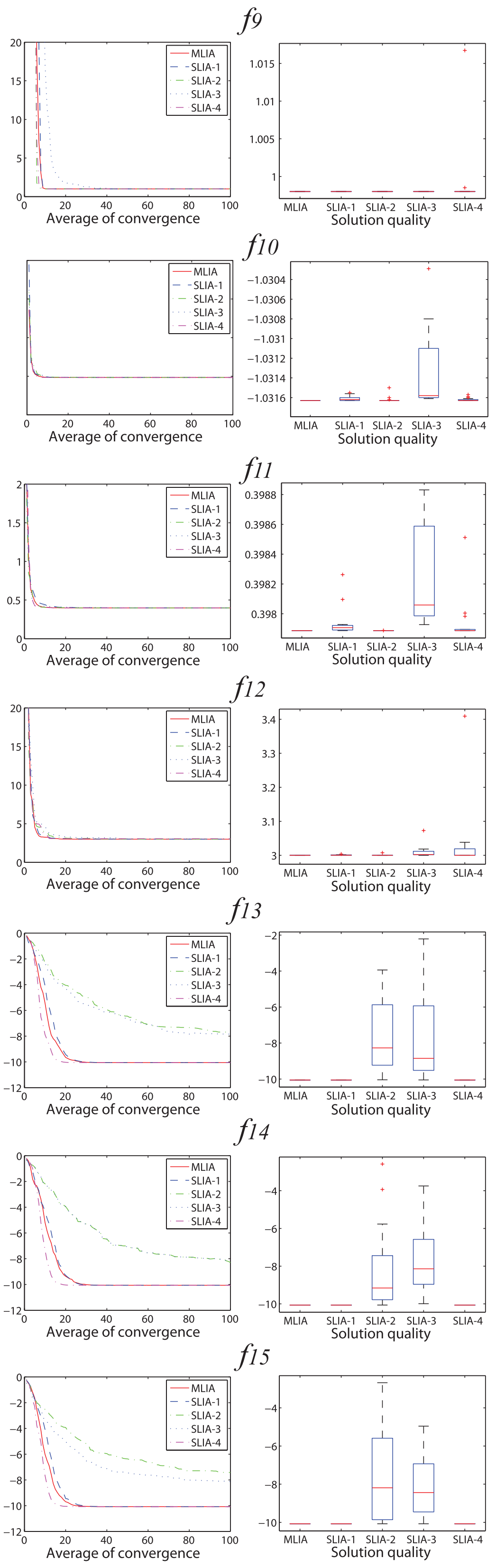


Fig. 9 The comparative results of the convergence graphs and box-and-whisker diagrams of solutions during MLIA, SLIA-1, SLIA-2, SLIA-3 and SLIA-4 for low dimensional benchmark functions f_9 - f_{15}.

Table 3 The statistical results obtained by MLIA, SLIA-1, SLIA-2, SLIA-3 and SLIA-4 algorithms for all benchmark functions.

Func.	MLIA	SLIA-1	SLIA-2	SLIA-3	SLIA-4
	Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std
f_1	**4.87E-33±8.43E-33**	5.62E-20±7.56E-20	2.39E+02±5.96E+02	1.23E+02±4.09E+02	4.19E-05±1.54E-04
f_2	**1.05E-13±7.36E-14**	6.81E-12±9.76E-12	2.59E+01±5.41E+00	4.31E+01±5.78E+00	9.54E-05±2.22E-04
f_3	**0±0**	2.08E+01±1.28E+01	1.96E+03±1.51E+03	3.05E+03±3.15E+03	**0 ± 0**
f_4	**−12569.5±0.00000**	**−12569.5±0.00000**	−12078.93±3.40E+02	−7522.94±6.36E+02	−11934.03±2.64E+02
f_5	**0±0**	3.90E-02±2.04E-01	2.54E+02±2.25E+01	2.32E+02±2.16E+01	1.66E-05±4.73E-05
f_6	**8.38E-15±2.41E-15**	1.32E-11±8.94E-12	1.18E+01±2.68E+00	7.82E+00±4.24E+00	2.71E-03±8.12E-03
f_7	**0±0**	**0±0**	7.26E-01±1.780000	1.15E+01±2.14E+01	8.02E-04±2.29E-03
f_8	**6.14E-24±9.58E-24**	4.27E-16±8.10E-16	4.02E-02±6.59E-02	1.96±3.05	5.90E-06±2.69E-05
f_9	**0.998004±6.78E-16**	0.998004±6.78E-16	0.998004±6.78E-16	0.998004±6.78E-16	0.998645±3.42E-03
f_{10}	**−1.031628±0**	−1.03161±2.50E-05	−1.03162±3.32E-05	−1.03133±3.88E-04	−1.03162±1.81E-05
f_{11}	**0.397887±1.69E-16**	0.397934±9.69E-05	0.397887±3.46E-07	0.398229±3.19E-04	0.397963±1.89E-04
f_{12}	**3±0**	3.001010±1.40E-03	3.001012±2.62E-03	3.011812±2.15E-02	3.061650±1.39E-01
f_{13}	**−10.0535±9.03E-15**	**−10.0535±9.03E-15**	−7.7341±1.9900	−7.84994±2.24000	**−10.0535±9.03E-15**
f_{14}	**−10.0637±1.81E-15**	**−10.0637±1.81E-15**	−8.3001±1.9600	−7.7082±1.8100	**−10.0637±1.81E-15**
f_{15}	**−10.0750±5.42E-15**	**−10.0750±5.42E-15**	−7.420±2.460	−8.126±1.600	−10.075±5.42E-15

ter than its four variants in terms of the convergence speed and solution quality for all tested problems, indicating that the best learning capacity is possessed by the multi-learning operator *ML* rather than any of the single learning operators. On the other hand, it also suggests that the hybridization of multiple single learning operators using the proposed framework shown in Fig. 3 is a promising method to enrich and promote the searching performance of learning operators.

In details, the algorithms' behaviors on f_1 are quite illuminating to further elaborate the search dynamics of the multi-learning operator *ML* and its four component single learning operators. In the beginning, SLIA-4 displays the fastest convergence speed among all algorithms, suggesting that the lateral mutation operator *LM* possesses the best local exploitation ability. However, it quickly approaches stable solutions that still be far away from the global optimum and cannot improve the solution qualities any further, implying that *LM* is not good at global exploration. The underlying reason for the behaviors of SLIA-4 is the frequently information exchange of search dynamics in *LM*. During the whole search progress, SLIA-1 displays the second best search performance in terms of the solution quality and robustness, and exhibits the third fastest convergence speed. This implies that the operator *BL* can well balance the exploitation and exploration of the search. Compared with SLIA-1 and SLIA-4, SLIA-2 and SLIA-3 show low-grade learning capacities in terms of the convergence speed and solution qualities. Considering the differences between the two kinds of variants, the comparative results demonstrate that the effectiveness of the search driven by *GM* and *CM* which only utilize random perturbations on the antibody itself is not as good as those operators (i.e. *BL* and *LM*) which make use of information in the environment. It further verifies the fact that learning from environment is a more easy way to achieve better search performance as demonstrated in [36]. Especially, we find that SLIA-2 performs better than SLIA-3 on most functions by comparing their convergence and learning performances. It is evident that *GM* in SLIA-3 prefers carrying out fine-grained searches within the small neighborhood of the solution, hence usually suffering from the local minimum trapping problem, while *CM* in SLIA-2 enables the algorithm to perform coarse-grained search and thus to have the capacity of escaping from local optima.

Table 4 User-defined parameters used in MLIA.

population size	N
clone size	M
antibody updating parameter	α
maximum iteration number	T_{max}
probability of implementing *GM*	$Prob_{GM}$
probability of implementing *CM*	$Prob_{CM}$
probability of implementing *LM*	$Prob_{LM}$
probability of implementing *BL*	$Prob_{BL}$

To sum up, a direct conclusion can be drawn: the multi-learning operator is more effective than single learning operators for searching better solutions by using the proposed construction framework illustrated in Fig. 3, wherein the component operators can play complementary search effects to each other no matter how inferiority of their own search capacities.

4.3 Investigation on User-Defined Parameters

Appropriate values of the parameter strongly influence the performance of the algorithm. MLIA has several user-defined parameters needed to be fine-tuned before it can be applied to optimization problems. Table 4 summarizes the parameters used in MLIA, namely, N, M, α, T_{max}, $Prob_{GM}$ and $Prob_{CM}$, $Prob_{LM}$, $Prob_{BL}$. The analyses are undertaken using the unimodal function f_1 and the multimodal function f_6 respectively. Except for the parameter under test, simulations are implemented under the following condition: $N = 30$, $M = 5$, $\alpha = 100$, $T_{max} = 2000$ for high dimensional functions f_1-f_8 and $T_{max} = 100$ for low dimensional functions f_9-f_{15}.

4.3.1 Setup of Learning Irrelevant Parameters

The user-defined parameters N, M, α and T_{max} are commonly used in immune algorithms involving MLIA. MLIA is initialized with N randomly generated B cells. N determines the population size of the antibodies. Intuitively, the larger value of N, the better performance and meanwhile the more computational cost of the algorithm. According to [38], [42], a reasonable value for N is 30. The parameter M is responsible for controlling the clonal size. Large values for M result in large clones produced in the algorithm and more implemental times of learning operators in each generation, thus requiring more computational times. However, as suggested in [23], [36], a trade-off between the quality of solution and computational time is to set M as 5. The parameter α indicates the acceptance probability of a temporary worse antibody, thus maintaining the population diversity to some extent, and it is set to be 100 based on the analyses in [42]. The value of the parameter T_{max} directly influences the generation size of the evolution. To make the further comparisons fair, the values for T_{max} are set the same as in [38], that is, $T_{max} = 2000$ for high dimensional functions and $T_{max} = 100$ for low dimensional functions.

4.3.2 Setup of Learning Relevant Parameters

Further considerations deal with the setting of the implementation probabilities of each single learning operator in *ML*. It is impossible to test all possible combinations of these probabilities under the constrain of $Prob_{GM}+Prob_{CM}+Prob_{LM}+Prob_{BL} = 1$ within reasonable simulation time since the number of combinations is infinite. As a result, finite number of typical combinations should be selected and tested, aiming to find the optimal setting of these probabilities.

Alternatively, experimental design with mixtures [47], [48] can accomplish the above task. Experiments with mixtures are experiments in which the variants are proportions of ingredients in a mixture. An example is an experiment for determining the proportion of ingredients in a polymer mixture that will produce plastics products with the highest tensile strength. Designs for deciding how to mix the ingredients are called experimental designs with mixtures. A design of R runs for mixtures of m ingredients is a set of R points in the domain:

$$T_m=(\lambda_1,...,\lambda_m) : \lambda_j\geq 0, j=1,...,m, \lambda_1+...+\lambda_m=1 \quad (13)$$

In this study, the simplex-lattice design introduced in [49] was adopted. Suppose that the mixture has m components. Let H be a positive integer and suppose that each component takes $(H + 1)$ equally spaced places from 0 to 1, i.e., $\lambda_i = 0, 1/H, 2/H, ..., 1$, $i = 1, ..., m$. Thus, $\{m, H\}$-simplex-lattice can be used to represent this design which has C_{H+m}^{m-1} design points.

In the experiment, fifteen representative design points including ten points of the $\{4, 1\}$-simplex-lattice, four of the $\{4, 2\}$-simplex-lattice, the centrobaric point “0.25/0.25/0.25/0.25”, and an empirical trail point “0.1/0.1/ 0.4/0.4” are used to make a comparison. Tables 5 and 6 record the statistical values of solutions obtained by MLIA with different learning probabilities in *ML* for the unimodal function f_1 and the multimodal function f_6, respectively. These learning probabilities are taken from the above mentioned fifteen representative design points, where a, b, c and d in “a/b/c/d” denote the assigned implementation probabilities of *GM*, *CM*, *LM* and *BL* respectively. Tables 5 and 6 show that the setting of the implementation probabilities in *ML* using “0.1/0.1/0.4/0.4” exhibits the best learning performance in terms of solution qualities. It is worth emphasizing that, although “0.1/0.1/0.4/0.4” might not be the optimal setting of the probabilities, it outperforms most of the other settings due to the usage of experimental design with mixtures. Thus, we can conclude that a reasonable setting of the learning relevant parameters is that $Prob_{GM} = 0.1$, $Prob_{CM} = 0.1$, $Prob_{LM} = 0.4$ and $Prob_{BL} = 0.4$.

4.4 Investigation on Learning Sequences in *ML*

In the Algorithm 1 of MLIA, the sequence of single learning operators to be implemented is *LM-BL-GM-CM*. As discussed in Sect. 3.2, although the setting values of implementation probabilities in *ML* are deterministic, each single operator is carried out with randomness due to the stochastic nature of the control parameter q. Thus, on condition of that these probabilities have been determined, the influence of using different learning sequences on MLIA was investigated. Table 7 summarizes the simulation results of all the twenty-four possible sequences of learning operators on f_1 and f_6 under 30 independent runs, where the Wilcoxon signed ranks test [50]–[52] is used to detect significant differences between the behavior of two algorithms. In Table 7, R^+ denotes the sum of ranks for the problems in which the base algorithm outperformed the competitive one, and R^- the sum of ranks for the opposite. Once the R^+ and R^- related to the comparisons between the sequence *LM-BL-GM-CM* and the rest of sequences are obtained, their associated p-values can be computed. It is worth pointing out that, from the statistical point of view, the Wilcoxon signed ranks test is more sensitive and safer than the t-test, because it does not assume normal distributions and meanwhile the outliers have less effect on the Wilcoxon test than on the t-test [50]. As Table 7 states, *LM-BL-GM-CM* shows no significant improvement over other learning sequences at a level significance $\alpha = 0.05$, with only one exception on f_1. Therefore, we can conclude that the sequence of single learning operators in *ML* makes little influence on the performance of MLIA.

4.5 Performance Comparison

To further validate the proposed MLIA, an intensive comparisons with existing AISs including FCA [38], CLONALG [53], SIA [54], Opt-IMMALG [34], PAISA [55], dopt-aiNet [56], QICA [21] and Vaccine-AIS [23] have been carried out. The specific details of the compared AISs are brief given as follows:

1) FCA [38] is a fast clonal algorithm incorporating

Table 5 The statistical values of solutions obtained by MLIA with different learning probabilities in *ML* for f_1 under 30 runs.

f_1 : Probability	Mean	Std.	Median	Min	Max
1/0/0/0	1.23E+02	4.09E+02	1.13E-02	8.80E-08	1.71E+03
0/1/0/0	2.39E+02	5.96E+02	2.22E-02	1.24E-05	2.34E+03
0/0/1/0	4.19E-05	1.54E-04	1.11E-10	1.10E-39	6.26E-04
0/0/0/1	5.62E-20	7.56E-20	2.06E-20	1.57E-22	3.22E-19
0.5/0.5/0/0	3.31E+01	1.54E+02	8.54E-02	5.94E-06	8.44E+02
0.5/0/0.5/0	3.66E+02	6.70E+02	1.21E+02	1.71E-05	3.23E+03
0.5/0/0/0.5	1.42E+01	1.32E+01	1.13E+01	2.09E-09	5.24E+01
0/0.5/0.5/0	2.00E+01	7.49E+01	5.17E-06	1.00E-12	3.61E+02
0/0.5/0/0.5	4.19E+01	6.51E+01	1.56E-04	2.00E-10	2.16E+02
0/0/0.5/0.5	1.31E-30	1.87E-30	4.52E-31	3.43E-32	9.80E-30
$\frac{1}{3}/\frac{1}{3}/\frac{1}{3}/0$	7.00E+01	2.19E+02	9.13E-02	1.64E-07	9.22E+02
$\frac{1}{3}/\frac{1}{3}/0/\frac{1}{3}$	4.91E+00	1.60E+01	2.06E-02	1.68E-08	8.44E+01
$\frac{1}{3}/0/\frac{1}{3}/\frac{1}{3}$	9.69E+01	3.52E+02	6.85E-02	9.41E-08	1.82E+03
$0/\frac{1}{3}/\frac{1}{3}/\frac{1}{3}$	1.97E+02	8.47E+02	1.71E-01	8.19E-07	4.58E+03
0.25/0.25/0.25/0.25	6.75E+01	3.56E+02	4.92E-03	5.36E-06	1.95E+03
0.1/0.1/0.4/0.4	**4.87E-33**	**8.43E-33**	**1.47E-33**	**5.65E-35**	**3.91E-32**

Table 6 The statistical values of solutions obtained by MLIA with different learning probabilities in *ML* for f_6 under 30 runs.

f_6 : Probability	Mean	Std.	Median	Min	Max
1/0/0/0	7.82E+00	4.24E+00	7.44E+00	1.16E+00	1.49E+01
0/1/0/0	1.18E+01	2.68E+00	1.22E+01	5.22E+00	1.53E+01
0/0/1/0	2.71E-03	8.12E-03	2.32E-06	3.24E-14	4.15E-02
0/0/0/1	1.32E-11	8.94E-12	1.05E-11	3.00E-12	4.82E-11
0.5/0.5/0/0	4.21E+00	3.42E+00	3.51E+00	5.08E-02	1.17E+01
0.5/0/0.5/0	4.23E+00	3.83E+00	3.33E+00	5.50E-03	1.24E+01
0.5/0/0/0.5	4.69E+00	8.23E-01	4.53E+00	3.59E+00	7.68E+00
0/0.5/0.5/0	2.70E-01	7.81E-01	1.89E-02	3.91E-03	3.30E+00
0/0.5/0/0.5	8.35E+00	1.05E+00	8.29E+00	6.58E+00	1.09E+01
0/0/0.5/0.5	6.75E-13	3.44E-13	5.96E-13	2.35E-13	1.49E-12
$\frac{1}{3}/\frac{1}{3}/\frac{1}{3}/0$	6.94E+00	4.44E+00	6.49E+00	1.16E+00	1.55E+01
$\frac{1}{3}/\frac{1}{3}/0/\frac{1}{3}$	9.19E+00	4.41E+00	9.60E+00	1.16E+00	1.63E+01
$\frac{1}{3}/0/\frac{1}{3}/\frac{1}{3}$	6.72E+00	4.51E+00	4.88E+00	9.31E-01	1.53E+01
$0/\frac{1}{3}/\frac{1}{3}/\frac{1}{3}$	8.57E+00	3.98E+00	8.47E+00	9.31E-01	1.61E+01
0.25/0.25/0.25/0.25	9.09E-02	5.32E-02	7.32E-02	2.70E-02	2.17E-01
0.1/0.1/0.4/0.4	**8.38E-15**	**2.41E-15**	**7.55E-15**	**4.00E-15**	**1.47E-14**

a parallel mutation operator, comprising of Gaussian and Cauchy mutation strategy. Meanwhile, it utilizes chaotic initialization and elitist immune memory mechanisms to further improve the efficiency of the algorithm. 2) CLONALG [53] is the original clonal selection algorithm which manipulates the affinities of antibodies with proportional cloning and counter-proportional hypermutation. 3) SIA [54] is a simple but effective immunological approach where only two parameters and simple immune operators are used. 4) opt-IMMALG [34] is an extension of SIA to solve more complex and high-dimensional problems. The algorithm uses an inversely proportional hypermutation operator, a novel aging operator together with the $(\mu + \lambda)$-selection to maintain the population diversity. 5) PAISA [55] is a population based immune algorithm using self/non-self learning operators and feedback rules on memory cells. 6) dopt-aiNet [56] is an expanded version of opt-aiNet, which is capable of solving problems in both stationary and dynamic environments. 7) QICA [21] is a quantum-inspired clonal algorithm where the quantum rotation gate strategy and the dynamic adjusting angle mechanism are applied to accelerate convergence. 8) Vaccine-AIS [23] is a diversity enhancement based immune algorithm by proposing a vaccine injection operator for exploring all areas in the search space.

The first performance comparison was carried out during MLIA, FCA, CLONALG, SIA, and opt-IMMALG on all functions with 2, 4 and 30 dimensions. Table 8 summarizes the experimental results of these compared algorithms, where the reported results for FCA, CLONALG, SIA and opt-IMMALG are taken from [34], [38]. Each result of MLIA is obtained from 30 independent runs. As can be seen from Table 8, MLIA outperforms FCA on the functions f_3-f_{15} (13 out of 15 functions), outperforms CLONALG on the functions f_1, f_3-f_{15} (14 out of 15 functions), outperforms SIA on all the 15 functions, outperforms opt-IMMALG on the functions f_3, f_4, f_5, f_7, f_8, and f_{10}-f_{12} (8 out of 15 functions). For functions f_3-f_5, f_7, f_8, and f_{10}-f_{12}, MLIA shows the best performance in finding the global optima. It is worth emphasizing that, although MLIA did not perform significant better than its competitors, it has been demonstrated to be an effective algorithm for solving optimization problems. Furthermore, it is expected the perfor-

Table 7 Results of the Wilcoxon signed ranks test at a level of significance $\alpha = 0.05$ for $LM - BL - GM - CM$ versus the competitors on f_1 and f_6.

Order	f_1				f_6			
	R^+	R^-	p-value	Significant	R^+	R^-	p-value	Significant
$BL - CM - GM - LM$	171.5	263.5	1	No	235.0	230.0	0.950798	No
$BL - CM - LM - GM$	275.5	159.5	0.172398	No	231.0	234.0	1	No
$BL - GM - CM - LM$	270.5	164.5	0.072094	No	253.0	212.0	0.665789	No
$BL - GM - LM - CM$	263.5	201.5	0.36803	No	176.0	289.0	1	No
$BL - LM - CM - GM$	258.0	177.0	0.277395	No	130.0	335.0	1	No
$BL - LM - GM - CM$	207.5	257.5	1	No	266.0	199.0	0.48435	No
$CM - BL - GM - LM$	229.5	235.5	1	No	229.0	236.0	1	No
$CM - BL - LM - GM$	186.5	248.5	1	No	155.0	310.0	1	No
$CM - GM - BL - LM$	229.0	206.0	0.767681	No	242.0	223.0	0.837038	No
$CM - GM - LM - BL$	201.5	263.5	1	No	270.0	195.0	0.434452	No
$CM - LM - BL - GM$	159.5	275.5	1	No	247.0	218.0	0.757683	No
$CM - LM - GM - BL$	191.5	243.5	1	No	224.0	241.0	1	No
$GM - BL - CM - LM$	135.5	299.5	1	No	227.0	238.0	1	No
$GM - BL - LM - CM$	288.5	176.5	0.200319	No	225.0	240	1	No
$GM - CM - BL - LM$	265.5	199.5	0.451498	No	119.0	346.0	1	No
$GM - CM - LM - BL$	208.5	226.5	1	No	230.0	235.0	1	No
$GM - LM - BL - CM$	293.5	141.5	0.042012	Yes	285.0	180	0.275659	No
$GM - LM - CM - BL$	232.0	203.0	0.660224	No	236.0	229.0	0.934429	No
$LM - BL - CM - GM$	228.0	237.0	1	No	229.0	236.0	1	No
$LM - CM - BL - GM$	271.5	193.5	0.386557	No	270.0	195.0	0.434452	No
$LM - CM - GM - BL$	210.5	254.5	1	No	217.0	248.0	1	No
$LM - GM - BL - GM$	183.0	282.0	1	No	287.0	178.0	0.256721	No
$LM - GM - CM - BL$	220.5	214.5	0.907498	No	195.0	270.0	1	No

Table 8 Performance comparison between MLIA and other AISs including FCA, CLONALG, SIA and opt-IMMALG in terms of solution qualities.

Func.	MLIA	FCA	CLONALG	SIA	opt-IMMALG
	Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std
f_1	4.87E-33 ± 8.43E-33	**0 ± 0**	7.36E-08 ± 5.15E-08	2.31E-08 ± 8.97E-09	**0± 0**
f_2	1.05E-13 ± 7.36E-14	**0 ± 0**	**0 ± 0**	6.64E-08 ± 2.42E-08	**0± 0**
f_3	**0 ±0**	**0 ± 0**	**0 ± 0**	**0 ± 0**	**0± 0**
f_4	**−12569.5** ± **0**	−12569.46 ± 0.016	−12528.94 ± 21.48	−12518.19 ± 39.21	−12535.15 ± 62.81
f_5	**0 ± 0**	**0 ± 0**	0.18 ± 0.75	1.12 ± 6.214	0.596 ± 4.178
f_6	6.75E-13 ± 3.44E-13	1.56E-07 ± 3.12E-07	7.15E-04±1.27E-04	1.56E-03± 3.12E-04	**0±0**
f_7	**0 ± 0**	**0 ± 0**	4.18E-04 ± 2.02E-04	6.47E-03 ± 7.04E-04	**0±0**
f_8	**6.14E-24 ± 9.58E-24**	7.94E-11 ± 4.25E-12	3.27E-07 ± 8.54E-08	6.57E-03 ± 5.44E-03	1.77E-21 ± 8.77E-24
f_9	0.998004 ± 0	1.04 ± 3.65E-02	1.21 ± 9.50E-01	1.34 ± 2.81E-01	**0.998±1.1E-03**
f_{10}	**−1.03163 ± 0**	−1.03156 ±8.36E-06	−1.02377±1.58E-04	−1.01281±5.24E-02	−1.013±2.21E-02
f_{11}	**0.397887 ± 1.69E-16**	0.401 ± 1.16E-03	0.501 ± 2.87E-03	0.527 ± 8.54E-01	0.423 ± 3.21E-02
f_{12}	**3± 0**	3.0129 ± 2.15E-04	3.8791 ±6.86E-03	6.15813 ± 4.0252	5.837 ± 3.742
f_{13}	−10.0535±9.03E-15	−9.9244 ± 0.0452	−9.8817 ± 0.1521	−9.2677± 0.2252	**−10.153±1.03E-07**
f_{14}	−10.0637 ± 1.81E-15	−9.9438 ± 0.0384	−9.2691 ± 0.284	−8.9342 ± 0.8972	**−10.402±1.03E-05**
f_{15}	−10.075± 5.42E-15	−9.9622 ± 0.0503	−9.1528 ± 0.2987	−8.6654 ±1.1503	**−10.536±1.16E-03**

mance of MLIA can be further improved by incorporating the unique mechanisms proposed in its competitors, such as the chaotic initialization in [38], the aging operator in [34], the vaccine injection in [23], etc.

The second performance comparison was carried out during MLIA, PAISA, dopt-aiNet, and Vaccine-AIS on functions with 30 dimensions. Table 9 records the mean of the function values found in 30 trials and the mean number of function evaluations of the algorithms. The data of PAISA, dopt-aiNet and Vaccine-AIS were directly taken form the reported tables, while there were not record for the function f_3. To obtain the mean number of function evaluations for the algorithm, the termination criterion of MLIA is one of the objectives, $|f_{best} - f_{min}| < \varepsilon \cdot |f_{min}|$ or $|f_{best}| < \varepsilon$ if $f_{min} = 0$, is achieved, where f_{best} and f_{min} denote the best solution found until the current generation and the global optimum, respectively. $\varepsilon = 10^{-4}$ is used for all tested functions, which is the same as that in the compared algorithms.

Compared with PAISA, MLIA can find better solutions on all tested six functions in fewer evaluations. When implemented on f_4, unlike the competitors, MLIA produces the global optimal value (i.e. −12569.5) of the function. For f_5 and f_7, MLIA locates the global optimum value, whereas PAISA fails in doing so while dopt-aiNet and Vaccine-AIS take more fitness evaluations to find it. Although MLIA cannot find the global optima solution for f_1, f_2 and f_6 as dopt-aiNet and Vaccine-AIS, it can find close-to-optimal solutions (fitness errors ≤ 6.75E-13) with very few of evaluations. In addition, considering the convergence performance of MLIA on f_1, f_2 and f_6 as shown in Figs. 7 and 8, we find that MLIA is capable of finding better solutions if more fitness evaluations are allowed, due to the fact that MLIA has

Table 9 Performance comparisons between MLIA and other AISs including PAISA, dopt-aiNet, and Vaccine-AIS in terms of solution qualities and the number of fitness evaluations on functions with 30 dimensions.

Func.	Number of fitness evaluations				Mean affinity value			
	MLIA	PAISA	dopt-aiNet	Vaccine-AIS	MLIA	PAISA	dopt-aiNet	Vaccine-AIS
f_1	**2836**	3956	6183	72633	4.87E-33	2.51E-17	0	0
f_2	**2590**	3206	406150	51267	1.05E-13	7.E-14	0	0
f_4	**1952**	1984	4169	172342	−12569.5	−12569.49	−12569.49	−12567.8
f_5	**1884**	2528	3379	84339	0	1.71E-12	0	0
f_6	**2086**	2774	5564	93363	6.75E-13	3.51E-16	0	0
f_7	**1582**	2612	7276	57054	0	1.02E-15	0	0

Table 10 Mean number of function evaluations of MLIA, PAISA and QICA on f_4-f_6 with 20-1000 dimensions.

f_4				f_5			
D	MLIA	PAISA	QICA	D	MLIA	PAISA	QICA
20	**331**	957	2301	20	**365**	1497	4021
100	3010	**2787**	5295	100	**2608**	4914	9391
200	**2074**	3618	7083	200	**5515**	8937	13671
400	20154	**6285**	12,161	400	**11,626**	13,728	16421
1000	**3400**	10,920	21,729	1000	27,588	**17,585**	19091
f_6				f_7			
D	MLIA	PAISA	QICA	D	MLIA	PAISA	QICA
20	**397**	1372	3216	20	**313**	1018	2541
100	**1812**	3036	5137	100	**1487**	3990	4397
200	**2777**	4664	5746	200	**2509**	4982	5345
400	**3976**	6126	6372	400	**3693**	5740	6187
1000	**4909**	7192	6949	1000	**5100**	6988	7249

not been trapped in local minima. In general, MLIA obtains better or competitive solutions than its competitors in AISs at a lower computational cost, and is suitable for the numerical optimization problems.

Finally, the performance of MLIA on four test functions with 20–1000 dimensions was evaluated and compared with those of PAISA and QICA. As can be seen from Table 10, the number of evaluations of MLIA is smaller than that of PAISA and QICA for all the functions except f_4 and f_5 with specific dimensions. In other words, MLIA performs better than PAISA for 17 (out of 20) instances, QICA for 18 instances, in terms of the number of fitness evaluations. The convergence graphs of MLIA on the four functions with 20–1000 dimensions are depicted in Fig. 10, which shows that MLIA obtains acceptable solutions $\varepsilon = 10^{-4}$ at reasonable computational cost, and displays good performance in solving high-dimensional problems.

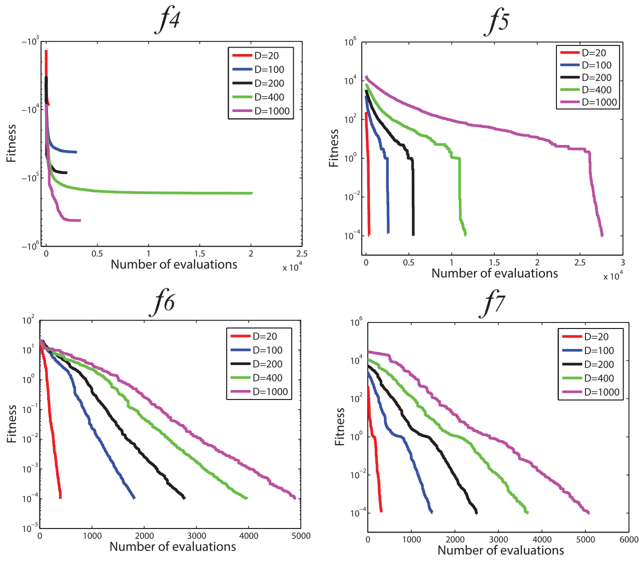


Fig. 10 The convergence graphs of MLIA on functions f_4-f_7 with 20, 100, 200, 400, and 1000 dimensions respectively.

5. Conclusions

This paper has proposed a multi-learning immune algorithm (MLIA) to solve global optimization problems. Our objective is to devise an effective learning operator for AIS to facilitate the evolution of antibodies. The multi-learning operator (*ML*) proposed herein is capable of meeting this challenge. By incorporating four distinct single learning operators including baldwinian learning, gaussian mutation, cauchy mutation and lateral mutation, the multi-learning operator can inherit the search characteristics from its components, and thereby it has a powerful balance ability between exploration and exploitation. We executed MLIA to solve fifteen benchmark problems, with dimensions from 2 to 1000, and numerous local minima. The results show that MLIA can find optimal or near-optimal solutions, exhibiting superior or competitive performance than other immune algorithms in terms of solutions quality and convergence speed. Statistical analyses regarding the user-defined parameters as well as the learning sequence demonstrated that MLIA can be well fine-tuned to a stable performance and is insensitive to learning sequences.

In addition, it should be pointed out that the learning capacity of *ML* can be further enhanced by incorporating more single learning operators, and the associated implementation probabilities can be self-adapted to improve the applicability of the algorithm. In our future work, we plan to verify the performance of MLIA on functions described in CEC 2005 [57], especially on rotated or shifted functions [58]. It is also meaningful to extend the utility of MLIA to handle both single-objective and multi-objective optimization problems under constrained or dynamic environments.

Acknowledgment

The authors would like to thank the Associate Editor and the anonymous reviewers for their great comments. This work is partially supported by the National Natural Science Foundation of China (Grants No. 61203325), Shanghai Rising-Star Program (No. 14QA1400100), "Chen Guang" project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation (No. 12CG35), Ph.D. Program Foundation of Ministry of Education of China (No. 20120075120004), the Fundamental Research Funds for the Central Universities (No. 2232013D3-39).

References

[1] L.N. de Castro, "Fundamentals of natural computing: An overview," Physics of Life Reviews, vol.4, no.1, pp.1–36, 2007.

[2] K. Worden, W.J. Staszewski, and J.J. Hensman, "Natural computing for mechanical systems research: A tutorial overview," Mechanical Systems and Signal Processing, vol.25, no.1, pp.4–111, 2011.

[3] S.C. Gao, Z. Tang, H. Dai, and J. Zhang, "An improved clonal algorithm and its application to traveling salesman problems," IEICE Trans. Fundamentals, vol.E90-A, no.12, pp.2930–2938, Dec. 2007.

[4] D. Dasgupta, S. Yu, and F. Nino, "Recent advances in artificial immune systems: Models and applications," Applied Soft Computing, vol.11, pp.1574–1587, 2011.

[5] A.S. Muhamad and S. Deris, "An artificial immune system for solving production scheduling problems: A review," Artificial Intelligence Review, pp.1–12, 2013.

[6] R. Xiao and T. Chen, "Relationships of swarm intelligence and artificial immune system," Int. J. Bio-Inspired Computation, vol.5, no.1, pp.35–51, 2013.

[7] F.M. Burnet, The Clonal Selection Theory of Acquired Immunity, Cambridge Press, 1959.

[8] G.J.V. Nossal, "Negative selection of lymphocytes," Cell, vol.76, pp.229–239, 1994.

[9] A. Perelson, "Immune network theory," Immunological Review, vol.110, pp.5–36, 1989.

[10] P. Matzinger, "The danger model: A renewed sense of self," Science, vol.296, pp.301–305, 2002.

[11] F. Gu, J. Greensmith, and U. Aickelin, "Theoretical formulation and analysis of the deterministic dendritic cell algorithm," Biosystems, vol.111, no.2, pp.127–135, 2013.

[12] S.C. Gao, H.W. Dai, G. Yang, and Z. Tang, "A novel clonal selection algorithm and its application to traveling salesman problems," IEICE Trans. Fundamentals, vol.E90-A, no.10, pp.2318–2325, Oct. 2007.

[13] S.C. Gao, W. Wang, H. Dai, F. Li, and Z. Tang, "Improved clonal selection algorithm combined with ant colony optimization," IEICE Trans. Inf. & Syst., vol.E91-D, no.6, pp.1813–1823, June 2008.

[14] R. Liu, L. Jiao, X. Zhang, and Y. Li, "Gene transposon based clone selection algorithm for automatic clustering," Inf. Sci., vol.204, pp.1–22, 2012.

[15] G. Dudek, "An artificial immune system for classification with local feature selection," IEEE Trans. Evol. Comput., vol.6, no.16, pp.847–860, 2012.

[16] S.C. Gao, R.L. Wang, H. Tamura, and Z. Tang, "A multi-layered immune system for graph planarization problem," IEICE Trans. Inf. & Syst., vol.E92-D, no.12, pp.2498–2507, Dec. 2009.

[17] A.M. Whitbrook, U. Aickelin, and J.M. Garibaldi, "Idiotypic immune networks in mobile-robot control," IEEE Trans. Syst. Man. Cybern. B, Cybern., vol.37, no.6, pp.1581–1598, 2007.

[18] L. de Mello Honorio, A.M.L. Da Silva, and D.A. Barbosa, "A cluster and gradient-based artificial immune system applied in optimization scenarios," IEEE Trans. Evol. Comput., vol.16, no.3, pp.301–318, 2012.

[19] F. Freschi, C.A.C. Coello, and M. Repetto, "Multiobjective optimization and artificial immune systems: A review," Handbook of Research on Artificial Immune Systems and Natural Computing: Applying Complex Adaptive Technologies, Medical Information Science Reference, pp.1–21, Hershey, New York, 2009.

[20] X. Yao and Y. Xu, "Recent advances in evolutionary computation," J. Computer Science and Technology, vol.21, no.1, pp.1–18, 2006.

[21] L. Jiao, Y. Li, M. Gong, and X. Zhang, "Quantum-inspired immune clonal algorithm for global optimization," IEEE Trans. Syst. Man. Cybern. B, Cybern., vol.38, no.5, pp.1234–1253, 2008.

[22] S.M. Garrett, "Parameter-free, adaptive clonal selection," IEEE Congress on Evolutionary Computation, pp.1052–1058, 2004.

[23] K.M. Woldemariam and G.G. Yen, "Vaccine-enhanced artificial immune system for multimodal function optimization," IEEE Trans. Syst. Man. Cybern. B, Cybern., vol.40, no.1, pp.218–228, 2010.

[24] R. Zhang and C. Wu, "A hybrid immune simulated annealing algorithm for the job shop scheduling problem," Applied Soft Computing, vol.10, no.1, pp.79–89, 2010.

[25] B. Naderi, M. Mousakhani, and M. Khalili, "Scheduling multi-objective open shop scheduling using a hybrid immune algorithm," Int. J. Advanced Manufacturing Technology, vol.66, no.5-8, pp.895–905, 2013.

[26] Z.J. Lee, C.Y. Lee, and S.F. Su, "An immunity-based ant colony optimization algorithm for solving weapon-target assignment problem," Applied Soft Computing, vol.2, no.1, pp.39–47, 2002.

[27] L. Jiao and L. Wang, "A novel genetic algorithm based on immunity," IEEE Trans. Syst. Man. Cybern. A, Syst. Humans, vol.30, no.5, pp.552–561, 2000.

[28] M.R. Widyanto, B. Kusumoputro, H. Nobuhara, K. Kawamoto, and K. Hirota, "A fuzzy-similarity-based self-organized network inspired by immune algorithm for three-mixture-fragrance recognition," IEEE Trans. Ind. Electron., vol.53, no.1, pp.313–321, 2006.

[29] M.M. El-Sherbiny and R.M. Alhamali, "A hybrid particle swarm algorithm with artificial immune learning for solving the fixed charge transportation problem," Computers & Insustrial Engineering, vol.64, no.2, pp.610–620, 2013.

[30] J. Chen, B. Xin, Z. Peng, L. Dou, and J. Zhang, "Optimal contraction theorem for exploration-exploitation tradeoff in search and optimization," IEEE Trans. Syst. Man. Cybern. A, Syst. Humans, vol.39, no.3, pp.680–691, 2009.

[31] V. Cutello, G. Nicosia, and M. Pavone, "Exploring the capability of immune algorithms: A characterization of hypermutation operators," Third International Conference on Artificial Immune Systems, ICARIS 2004, pp.263–276, 2004.

[32] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," IEEE Trans. Evol. Comput., vol.3, no.2, pp.82–102, 1999.

[33] X. Xu and J. Zhang, "An improved immune evolutionary algorithm for multimodal function optimization," Third International Conference on Natural Computation, ICNC 2007, pp.641–646, 2007.

[34] V. Cutello, G. Nicosia, and M. Pavone, "Real coded clonal selection algorithm for unconstrained global optimization using a hybrid inversely proportional hypermutation operator," Proc. 2006 ACM

Symposium on Applied Computing, pp.950–954, 2006.

[35] M. Pavone, G. Narzisi, and G. Nicosia, "Clonal selection: An immunological algorithm for global optimization over continuous spaces," J. Global Optimization, pp.1–40, 2011.

[36] M. Gong, L. Jiao, and L. Zhang, "Baldwinian learning in clonal selection algorithm for optimization," Inf. Sci., vol.180, no.8, pp.1218–1236, 2010.

[37] S. Gao, T. Gong, W. Zhong, F. Wang, and B. Chen, "Enhancing the learning capacity of immunological algorithms: A comprehensive study of learning operators," Advances in Artificial Life, pp.876–883, 2013.

[38] N. Khilwani, A. Prakash, R. Shankar, and M.K. Tiwari, "Fast clonal algorithm," Engineering Applications of Artificial Intelligence, vol.21, no.1, pp.106–128, 2008.

[39] B. Haktanirlar Ulutas and S. Kulturel-Konak, "A review of clonal selection algorithm and its applications," Artificial Intelligence Review, vol.36, no.2, pp.117–138, 2011.

[40] T. Jansen and C. Zarges, "Analyzing different variants of immune inspired somatic contiguous hypermutations," Theor. Comput. Sci., vol.412, no.6, pp.517–533, 2011.

[41] M. Gong, L. Jiao, L.N. Zhang, and H. Du, "Immune secondary response and clonal selection inspired optimizers," Progress in Natural Science, vol.19, no.2, pp.237–253, 2009.

[42] S.C. Gao, H. Dai, J. Zhang, and Z. Tang, "An expanded lateral interactive clonal selection algorithm and its application," IEICE Trans. Fundamentals, vol.E91-A, no.8, pp.2223–2231, Aug. 2008.

[43] V. Cutello, G. Nicosia, M. Pavone, and J. Timmis, "An immune algorithm for protein structure prediction on lattice models," IEEE Trans. Evol. Comput., vol.11, no.1, pp.101–117, 2007.

[44] T. Bäck and H.P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," Evol. Comput., vol.1, no.1, pp.1–23, 1993.

[45] H.P. Schwefel, Evolution and optimum seeking, Wiley, New York, 1995.

[46] D.H. Wolpert and W.G. Macready, "No free lunch theorems for optimization," IEEE Trans. Evol. Comput., vol.1, no.1, pp.67–82, 1997.

[47] D.C. Montgomery, Design and analysis of experiments, 5th ed., John Wiley & Sons, 2001.

[48] J.A. Cornell, "Experiments with mixtures: A review," Technometrics, vol.15, no.3, pp.437–455, 1973.

[49] H. Scheffe, "Experiments with mixtures," J. Royal Statistical Society, Series B (Methodological), vol.20, pp.344–360, 1958.

[50] J. Derrac, S. Garcia, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," Swarm and Evolutionary Computation, vol.1, pp.3–18, 2011.

[51] S. Garcia, A. Fernandez, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," Inf. Sci., vol.180, no.10, pp.2044–2064, 2010.

[52] J. Alcala-Fdez, L. Sanchez, S. Garcia, M.J. Del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, and V.M. Rivas, "Keel: A software tool to assess evolutionary algorithms for data mining problems," Soft Computing, vol.13, no.3, pp.307–318, 2009.

[53] L.N. De Castro and F.J. Von Zuben, "Learning and optimization using the clonal selection principle," IEEE Trans. Evol. Comput., vol.6, no.3, pp.239–251, 2002.

[54] V. Cutello and G. Nicosia, "An immunological approach to combinatorial optimization problems," Advances in Artificial Intelligence — IBERAMIA 2002, pp.361–370, 2002.

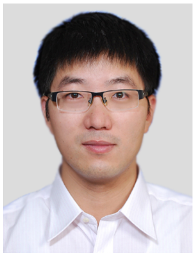
[55] M. Gong, L. Jiao, and X. Zhang, "A population-based artificial immune system for numerical optimization," Neurocomputing, vol.72, no.1-3, pp.149–161, 2008.

[56] F.O. de Franca, F.J. Von Zuben, and L.N. de Castro, "An artificial immune network for multimodal function optimization on dynamic environments," Proc. 2005 Conference on Genetic and Evolutionary Computation, pp.289–296, 2005.

[57] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," KanGAL Report, vol.2005005, 2005.

[58] S.Z. Zhao, P.N. Suganthan, Q.K. Pan, and M. Fatih Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with harmony search," Expert Systems with Applications, vol.38, no.4, pp.3735–3742, 2011.

Shuaiqun Wang received the B.S. degree from Inner Mongolia Normal University, Huhehaote, China in 2007, and the M.S. degree from Shanghai Maritime University, Shanghai, China in 2010. Now, she is working toward the D.E. degree at Tongji University, Shanghai, China. Her main research interests are artificial immune system and optimization problems.



Shangce Gao received the B.S. degree from Southeast University, Nanjing, China in 2005, and the M.S. and Ph.D. degrees in intellectual information systems and innovative life science from University of Toyama, Toyama, Japan in 2008 and 2011, respectively. He is currently an Associate Professor with the Faculty of Engineering, University of Toyama, Toyama, Japan. From 2011 to 2012, he was an associate research fellow with the Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai, China. From 2012 to 2014, he was an associate professor with the College of Information Sciences and Technology, Donghua University, Shanghai, China. His main research interests include computational intelligence, nature-inspired technologies, swarm intelligence, and neural networks for optimization and real-world applications. He was a recipient of the Shanghai Rising-Star Scientist award, the Chen-Guang Scholar of Shanghai award, the Outstanding Academic Performance Award of IEICE, the Outstanding Self-financed Students Abroad Award of Chinese Government, the Outstanding Academic Achievement Award of IPSJ, and the Outstanding Doctoral Award of University of Toyama.

Aorigele received the B.S. degree in Computer Science and Technology from Inner Mongolia Normal University, Huhehaote, China in 2007. Now, he is working toward the D.E. degree at Toyama University, Toyama, Japan. His main research interests are artificial immune system and optimization problems.

Yuki Todo received the B.S. degree from Zhejiang University, Zhejiang, China, the M.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, and the D.E. degree from Kanazawa University, Kanazawa, Japan, in 1983, 1986, and 2005, respectively. From 1987 to 1989, she was an Assistant Professor with the Institute of Microelectronics, Shanghai Jiaotong University, Shanghai, China. From 1989 to 1990, she was a research student at Nagoya University, Nagoya, Japan. From 1990 to 2000, she was a Senior Engineer with Sanwa Newtech Inc., Miyazaki, Japan. From 2000 to 2011, she worked with Tateyama Systems Institute, Toyama, Japan. In 2012, she joined Kanazawa University, where she is now an Associate Professor with the School of Electrical and Computer Engineering. Her current research interests include multiple-valued logic, neural networks, and optimization.

Zheng Tang received the B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Tshinghua University, Beijing, China in 1984 and 1988, respectively. From 1988 to 1989, he was an Instructor in the Institute of Microelectronics at Tshinhua University. From 1990 to 1999, he was an associate professor in the Department of Electrical and Electronic Engineering, Miyazaki University, Miyazaki, Japan. In 2000, he joined Toyama University, Toyama, Japan, where he is currently a professor in the Department of Intellectual Information Systems. His current research interests include intellectual information technology, neural networks, and optimizations.