A New Clonal Selection Immune Algorithm with Perturbation Guiding Search and Non-uniform Hypermutation

Xinchao Zhao
Department of Mathematics, School of Science,
Beijing University of Posts and Telecommunications, Beijing 100876, China
Email: xcmmrc@gmail.com

Guoli Liu, Huqiu Liu, Guoshuai Zhao, Shaozhang Niu
School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract

A new clonal selection immune algorithm with perturbation guiding search and non-uniform hypermutation (nCSIA) is proposed based on the idea of perturbed particle swarm algorithm and non-uniform mutation. The proposed algorithm proportional clones antibody based on the affinity, adaptively adjusts the searching steps of antibodies with hypermutation according to the adaptive variation rule of non-uniform mutation and chooses the promising antibody based on the affinity by clonal selection principle. In order to keep the balance of exploration/exploitation better, perturbation guiding search strategy is presented, which is actually an elitist learning mechanism and is borrowed from the perturbed particle swarm algorithm. In order to validate the effectiveness of nCSIA, comprehensive experiments and analysis are done based on fifteen unimodal or multimodal benchmark functions. Compared with standard and the recent algorithms, it indicates that the proposed algorithm is feasible, effective and has better performance in terms of convergence, accuracy and stability. More evident predominance emerges from further experimental comparisons with expanding search space and increasing dimensions.

Keywords: Clonal selection, perturbation guiding search, particle swarm algorithm, non-uniform mutation, artificial immune system.

1. Introduction

The immune system (IS)[1] is a complex of cells, molecules and organs that represent an identification mechanism capable of perceiving and combating dysfunction from our own cells (*infectious self*) and the action of exogenous infectious microorganisms (*infectious nonself*). The emphasis is on a systemic view of the immune system, with a focus on the clonal selection principle, the affinity maturation of the immune response, and the immune network theory[2]. Immune algorithm[3] (IA) is a heuristic optimization algorithm which was inspired by biological immune system's character. The majority immune system inspired optimization algorithms which are based on the applications of the clonal selection and hypermutation[4]. Clonal selection algorithm[5-7] is characterized by cloning and mutating to produce an offspring population around the candidates. It expands the searching range by a combination of antibodies and antigens, namely, calculating the fitness, select the best antibody and memorize it. By the death of inactive cell and abandoning antibodies which have low affinity, the generation of memory cell can maintain the antibodies diversity. Though it has so many advantages, it still needs further improvements[8]: it is costly in terms of the number of

evaluations of the objective function, it is not adaptive to variations in the topology of the response surface, and the evolution is chiefly accomplished by random mutation, so the speed of convergence is slowed down.

Particle swarm optimization (PSO) algorithm, which was introduced by Kennedy and Eberhart in 1995[9], is an intelligent optimization algorithm that mimics swarm behavior in birds flocking and fish schooling to guide the particle population to search for global optimal solution. As PSO is easy to implement, it is rapidly successfully applied in many areas, such as function optimization, network training, fuzzy system control and other fields[10-13]. However, every coin has two sides. The rapid convergence speed of the standard PSO also means to be easily trapped into the local optima with the decreasing diversity of swarm during population evolution[14].

Perturbed particle swarm optimization (pPSO)[15] offers a new way to keep population diverse and to escape from the local optimal trap. The best location (solution) found by the particle population is denoted as $gbest$ whose property and behavior has an important effect on the PSO's final performance. The perturbed $gbest$ updating strategy is based on the concept of possibility measure[16] to model the lack of information about the true optimality of the $gbest$. The $gbest$ in pPSO is denoted as "possibly at $gbest$" ($pgbest$) which is characterized by a normal distribution around the $gbest$ and it also provides a simple and efficient exploration at the early stage and encourages local fine-tuning at the latter stage. Its function is to reduce the likelihood of premature convergence and to guide the search towards the promising space.

Mutation operation is a main operator in evolutionary algorithm (EA)[17], and various mutations have been incorporated into EA, such as Gaussian[18], Cauchy[19] and Lévy probability distribution-based[20] mutations, non-uniform mutation[21] and some mixed mutation strategy[22]. Non-uniform mutation has the merits of even "longer jumps" than Cauchy mutation at the early stage of the algorithm and much "finer-tunings" than Gaussian mutation operator at the later stage. The basic idea of mixed strategy[22] is that different mutation operators have some types of optimization problems that cannot be solved efficiently and integrate several mutation operators into a single algorithm can overcome this problem. Inspired by evolutionary game theory, Dong et al. presented a mixed strategy evolutionary programming algorithm[22] that employs the Gaussian, Cauchy, Lévy, and single-point mutation operators. Experimental results show that the mixed strategy performs equally well or better than the best of the four pure strategies does.

In this paper, a new clonal selection immune algorithm with perturbation guiding search and non-uniform hypermutation (nCSIA) is proposed to integrate the advantages of perturbing the global best antibody for the guided search and non-uniform mutation. The perturbation guiding search idea in AIS is borrowed from the perturbed particle swarm optimization algorithm[15], which features in keeping population diverse and elitist learning mechanism along with the global best solution with a slight perturbation. The algorithmic analysis and experimental results show that nCSIA has excellent performance with good convergence, stability and application potentials.

2. The Artificial Immune System and Particle Swarm Optimization

2.1. *Artificial Immune System and Inspired Optimization Algorithms*

The human immune system (HIS) is a highly evolved, parallel and distributed adaptive system. The information processing abilities of HIS provide important aspects in the field of computation. This emerging field is referring to as the Artificial Immune Systems[23]. The immune system's ability to adapt its B-cells to new types of antigens is powered by processes known as clonal selection and affinity maturation by hypermutation[24]. In fact, besides the clonal selection, during the initial expansion of clones, some of the progeny cells neither went on dividing nor developed into plasma cells. Instead, they reverted to small lymphocytes bearing the same B-cell receptor on their surface that their ancestors had. This lays the foundation for a more rapid and massive response the next time when the antigen enters the body, i.e. immune memory. The majority immune-inspired optimization algorithms are mainly concentrated on the clonal selection while the immune memory is only a concomitant which is simply modeled as an elitist selection.

AIS can be defined as computational systems inspired by theoretical immunology and observed immune functions, principles and models, which are applied to problem solving[8]. The first immune optimization algorithm[25] may be the work of Fukuda et al. that included an abstraction of clonal selection to solve computational problems[26]. But the AIS for optimization have been popularized mainly by de Castro and Von Zuben's CLONALG[5]. CLONALG selects part fittest antibodies to clone proportionally to their antigenic affinities. The hypermutation operator performs an affinity maturation process inversely proportional to the fitness values generating the matured clone population.

After computing the antigenic affinity of the matured clone population, CLONALG creates randomly part new antibodies to replace the lowest fitness antibodies in current population and retain best antibodies to recycle.

In recent years, AIS have received significant amount of interests from researchers and industrial sponsors[27,28]. Some of the first work in applying immune system paradigms was undertaken in the area of fault diagnosis[29]. Later work applied immune system paradigms to the field of computer security[30,31], which seemed to act as a catalyst for further investigation of the immune system as a metaphor in many areas, such as anomaly detection[23,32], pattern recognition[33,34], sensor fusion and configuration[35], rule extraction[36], and optimization[5,7,37-39].

As far as multiobjective optimization is concerned, MISA[40] is the first attempt to solve general multiobjective optimization problems using artificial immune systems. A vector Artificial Immune System (VAIS)[41] is proposed for solving multiobjective optimization problems based on the opt-aiNet. NNIA[42] is proposed for multiobjective optimization based on its unique selection technique, which only selects minority isolated nondominated individuals based on their crowding-distance values. The selected individuals are then cloned proportionally to their crowding-distance values before heuristic search. By using the nondominated neighbor-based selection and proportional cloning, the new algorithm realizes the enhanced local search in the less-crowded regions of the current trade-off front. Chen et al. proposed a hybrid mutation operator (GP-HM operator)[43] with the combination of Gaussian and polynomial mutations. The GP-HM operator adopts an adaptive switching parameter to control the mutation process, which uses relative large steps in high probability for boundary individuals and less-crowded individuals. With the program run, the probability of performing relative large steps is reduced gradually. By this means, the exploratory capabilities are enhanced by keeping a desirable balance between global and local search.

2.2. *Antigen, Antibody and Antibody Population*

In this paper, we follow the nomenclature of immunology and define the terms as follows.

For the minimization problem $\min y = f(x), x \in \Omega$, where $x = \{x_1, x_2, \cdots, x_n\}^T$, Ω is the feasible region. The optimization problem $y = f(x), x \in \Omega$ is antigen and the decision variable $x = \{x_1, x_2, \cdots, x_n\}^T$ is the antibody. The set of the antibody is called antibody population.

By the way, as the idea of perturbation guiding search in this paper is borrowed from the perturbed particle swarm algorithm[15], the terms of "antibody" from AIS and "particle" from PSO are indiscriminating in this paper. Both of them are solution variable $x = \{x_1, x_2, \cdots, x_n\}^T$.

2.3. *Particle Swarm Optimization Algorithm*

In PSO[44,45], a potential solution is viewed as a particle without weight and volume who can move in the search space at a certain speed to some direction. Each particle tracks two extrema to update its velocity and location. One extremum is the best location found by itself, denoted as $pbest$. Another one is the best location found by all particles, denoted as $gbest$.

In the *n*-dimension search space, the particle swarm $X = \{x_1, x_2, \cdots, x_m\}$ is composed of *m* particles, and the *i*-th particle's location is $x_i = \{x_{i1}, x_{i2}, \cdots, x_{in}\}^T$, its velocity is $v_i = \{v_{i1}, v_{i2}, \cdots, v_{in}\}^T$. The best location found by itself is $p_i = \{p_{i1}, p_{i2}, \cdots, p_{in}\}^T$ and the best location $(gbest)$ found by all the particles is $p_g = \{p_{g1}, p_{g2}, \cdots, p_{gn}\}^T$. During the evolutionary process, the velocity and position of *i*-th particle on *d*-th dimension are updated as

$$v_{id}^{t+1} = w * v_{id}^t + c_1 * r_1 * (p_{id}^t - x_{id}^t) + c_2 * r_2 * (p_{gd}^t - x_{id}^t) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^t \quad (2)$$

where $d = 1, 2, \cdots, n$, $i = 1, 2, \cdots, m$. n is the dimension of the search space, m is the scale of the swarm, t stands for the current generation, c_1, c_2 are the acceleration coefficients, r_1 and r_2 are two random values uniformly distributed in the range of [0, 1]. w is the inertia weight. In order to prevent the particle from exceeding the search space, velocity is restricted in $[-v_{\max}, v_{\max}]$, where $v_{\max} = k * x_{\max}$ and $\{0.1 \le k \le 1.0\}$.

In PSO, the swarm converges rapidly within the intermediate vicinity of the $gbest$. However, such a high convergence speed often results in Ref. 45: 1) the lost of diversity and 2) premature convergence if the $gbest$ wrongly guides to a local optimum. This motivates the

development of a perturbed particle swarm algorithm[15] based on the perturbed $gbest$ updating strategy, which is based on the concept of possibility measure[16] to model the lack of information about the true optimality of the $gbest$ [46]. In contrast to conventional approaches, the $gbest$ in pPSO is denoted as "possibly at $gbest$ $pgbest = \{p'_{g1}, p'_{g2}, \cdots, p'_{gn}\}^T$", instead of a crisp location, where p'_{gd} is the perturbation result of p_{gd} of p_g ($d = 1, 2, \cdots, n$).

3. nCSIA Algorithm

3.1. *Algorithm Composition*

3.1.1. *Generation and Evaluation of Initial Antibody Population*

The initial antibody population is generated randomly in the range of $[X_{\min}, X_{\max}]$. Every antibody is evaluated for its affinity with optimization problem (fitness). The current optimal solutions $pbest$ are assigned as the initial antibodies, and locate the best antibody as the global best individual $gbest$.

3.1.2. *Proportional Cloning*

In immunology, cloning means asexual propagation so that a group of identical cells can be descended from a single common ancestor, such as a bacterial colony whose members arise from a single original cell as the result of mitosis. Clone is implemented proportionally according to the affinity of antibodies in order to self adaptively explore the total search space. Combined with the fitness value and position of antibody, the affinity of *i*-th antibody can be defined as follows:

$$affinity_i = \frac{fitness_i}{dis_i + 1} \tag{3}$$

Where $fitness_i$ denotes the fitness value of *i*-th antibody, and dis_i denotes the distance between *i*-th antibody and global optimal antibody $gbest$:

$$dis_i = \sqrt{\sum_{d=1}^{n} (x_{id} - gbest_d)^2} \tag{4}$$

where x_{id} and $gbest_d$ denote the *d*-th dimension components of *i*-th antibody and global optimal antibody respectively, and n denotes the dimension of the decision variable. We can conclude from (3) that the larger the particle' fitness value is and the closer the particle to the global optimum, the larger the affinity is.

Assume that m antibodies are cloned proportionally, and produce the clonal sets $S_i, i = 1 \cdots m$. During the clonal process, the number of clone of *i*-th antibody is:

$$num_i = \left\lfloor \frac{affinity_i}{\sum_{j=1}^{m} affinity_j} * m \right\rfloor \tag{5}$$

where $|S_i| = num_i, i = 1 \cdots m$.

From (5) we can conclude that the clones of the antibody are in proportion to the antibody's affinity. Through this method, antibodies with larger affinity are cloned more and then accelerated to explore the even larger domain space with hypermutation operation. That is, their excellent properties are fully explored for even more promising search area until the global best neighborhood and locate the global solution exactly. The antibodies with low affinity are maintained a little for population diversity and some possible good genes may be used.

3.1.3. *Non-uniform HyperMutation*

Hypermutation is the key operation to implement evolution in the antibody population. Non-uniform hypermutation applies different perturbation vector to the cloned offspring antibodies and makes antibodies evolve continuously. The principle is defined as follows: assume that we need to mutate the *d*-th dimension of antibody $x_i = \{x_{i1}, \cdots, x_{id}, \cdots, x_{in}\}^T$ whose upper and lower bounds are denoted as UB, LB respectively. Then, the *d*-th dimension after mutation of the antibody is:

$$x'_{id} = \begin{cases} x_{id} + \Delta(t, UB - x_{id}), & \xi(0,1) = 0 \\ x_{id} - \Delta(t, x_{id} - LB), & \xi(0,1) = 1 \end{cases} \tag{6}$$

where

$$\Delta(t, y) = y * (1 - r^{(1 - t/T)^b}) \tag{7}$$

and random variable $\xi(0,1)$ will be 0 or 1 with equal probability. t denotes the current cycle variable, T denotes the maximum generation number, r is a uniform random real number in the range of (0, 1), and b denotes the system parameter which determines the

degree of dependency on iteration number (non-uniformity)[17].

3.1.4. *Clonal Selection*

The function of clonal selection is to make antibody population evolve, which is similar with the selection operation of evolutionary algorithm[17]. The average population affinity is improved between the pre-mutation and the post-mutation antibody population to avoid algorithm degradation.

The schematic diagram is revealed as Figs. 1.①, ②, ③ represent proportional clone, non-uniform hypermutation and clonal selection, respectively. x_i denotes the *i*-th antibody, who has num_i copies. After non-uniform hypermutation and clonal selection, the new individual x' is obtained.

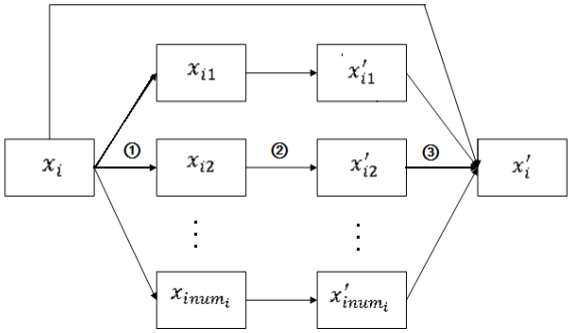


Fig. 1. Clone, HyperMutation and Clonal Selection Process

3.1.5. *Perturbation Guiding Search*

The velocity and position updating equation of PSO is actually an elitist learning mechanism from the historical experience of individual and population. The premature occurrence is possible to appear if all the solutions learn from the same global best individual, which is the inherent motivation of pPSO[15].

This idea is applied to the clonal selection immune algorithm in this paper, that is to say, an elitist learning strategy is adopted by nCSIA as follows.

After clonal selection operation, every antibody of nCSIA will begin a learning process from two extrema. One extremum is the best antibody found by itself, denoted as *pbest* . Another one is the (perturbed) best solution found by antibody population. nCSIA perturbs the global best antibody *gbest* , gets a perturbed global best *pgbest* , and update the antibody's (particle's) flying velocity as follows:

$$p'^{t}_{gd} = N(p^{t}_{gd}, \sigma) \tag{8}$$

$$\sigma = p(t) \tag{9}$$

$$v^{t+1}_{id} = w*v^{t}_{id} + c_1*r_1*(p^{t}_{id} - x^{t}_{id}) + c_2*r_2*(p'^{t}_{gd} - x^{t}_{id}) \tag{10}$$

where p'^{t}_{gd} denotes the *d*-th dimension of *pgbest* in the *t*-th generation. From (8) it can be observed that the *pgbest* is characterized by a normal distribution $N(p^{t}_{gd}, \sigma)$, where σ represents the degree of uncertainty about the optimality of the *gbest* . In order to account for the information received over time that reduces uncertainty about the *gbest* position, σ is modeled as a non-increasing function of the generation number as (11). In this paper the update formula of σ is defined as follows:

$$p(t) = \begin{cases} \sigma_{\max} & , \ t < \alpha * T \\ \sigma_{\min} / 10^{s} & , \ others \end{cases} \tag{11}$$

where

$$s = \left\lfloor \frac{(t - \alpha * T)}{interval} \right\rfloor \tag{12}$$

In the above formulas, $\sigma_{\max}$, $\sigma_{\min}$, α and *interval* are manually set parameters. Parameter *interval* in (12) indicates that σ should be updated every *interval* generations. *t* denotes the current iteration number. T is the maximal generation number. During the perturbing process, if one dimension of an antibody exceeds the definition domain, the antibody remains unchanged. Otherwise, its velocity will be updated according to Eq.(10). Its position is updated with Eq.(2).

The perturbation guiding search strategy (8-11) should be distinguished from conventional velocity and position update equations (1) and (2) of PSO, which applies a random perturbation to the antibodies. The function of *pgbest* is to encourage the antibody to explore an even larger region beyond that defined by the search trajectory. By considering the uncertainty associated with each *gbest* as a function of time, *pgbest* provides a simple and efficient exploration at the early stage when σ is large and encourages local fine-tuning at the latter stage when σ is small. Subsequently, this approach helps to reduce the likelihood of premature convergence, and simultaneously guides the search toward the promising search area.

3.2. *Algorithm Composition*

A new clonal selection immune algorithm with perturbation guiding search and non-uniform hypermutation (nCSIA) is proposed in this paper. Reference[15] exhibited an enhanced perturbed particle swarm optimization algorithm based on the perturbation idea. As Ref. 21 analyzed, non-uniform mutation has self-adaptive property and evolutionary programming based on non-uniform mutation shows encouraging performance. Algorithm nCSIA combines the strategies of proportional clone, affinity proliferation, non-uniform hypermutation and perturbation guiding search. Therefore, the balance of exploration and exploitation is easy to be kept and even satisfactory performance is expected.

The algorithmic flowchart is presented as Fig. 2. The detailed description is as follows:

Step1: Initialize antibody population and related parameters;
Step2: Evaluate antibodies;
Step3: Calculate the affinity of antibodies;
Step4: Determine the clone number of antibody according to its affinity;
Step5: Implement non-uniform hypermutation and obtain the offspring antibody.
Step6: Select the antibodies whose affinities are largest from the parent antibodies and offspring population as the new antibody population.
Step7: Perturb the global best antibody for perturbation guiding search with equations (8-11).
Step8: Algorithm terminates if termination conditions meet. Otherwise, go to Step2.

3.3. *Computational Complex*

Analyzing nCSIA's computational complexity is revealing. The related parameters are referred to Section IV.B. Then the time complexity of one generation for the algorithm can be calculated as follows:

The time complexity for evaluating fitness of population is $O(m)$; the time complexity for calculating affinity is $O(mn)$; the time complexity for proportional cloning is $O(2m)$; the worst time complexity for hyperMutation is $O(mn)$; the time complexity for clonal selection is $O(2m)$; the time complexity for perturbation guiding search is $O(m+mn)$. So the worst total time complexity is

$$O(m)+O(mn)+O(2m)+O(mn)+O(2m)+O(m+mn).$$

According to the operational rules of the symbol O, the worst time complexity of one generation for nCSIA can be simplified as

$$O(mn) \tag{13}$$

So the time complexity of nCSIA is $O(mnT)$ and the costs of calculating affinity and hypermutation dominate the computational complexity of nCSIA.

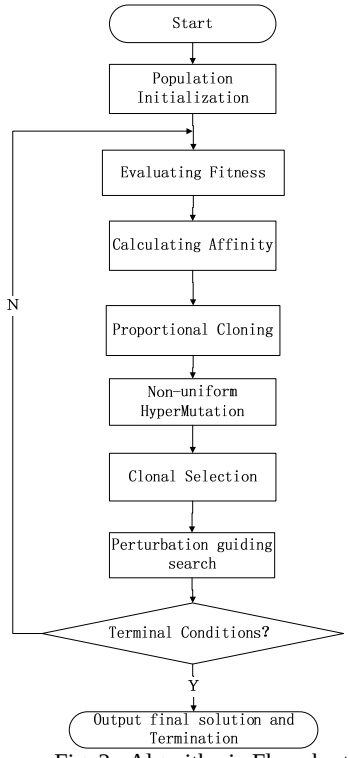


Fig. 2. Algorithmic Flowchart

4. Experimental comparison and algorithmic analysis

In order to validate the necessity and good performance of nCSIA, 15 benchmarks (in Table-1) are adopted to do numerical experiments and comparison, in which first seven functions $f_1 \sim f_7$ are unimodal and the others are multimodal. The statistical results of the Best, Median, Mean and standard deviation (STD) are presented and compared each other among PSO, pPSO, FEP, ImPSO and nCSIA, respectively.

4.1. *Algorithms to Compare*

The comparing algorithms in Tables 2-4 are specified as follows. PSO is standard particle swarm optimization algorithm. pPSO[15] is a hybrid algorithm of standard

particle swarm optimization algorithm and a perturbation strategy. FEP is a faster evolutionary programming[19] with *Cauchy* mutation, which has a larger search step than *Guassan* mutation. ImPSO is a clonal selection immune algorithm whose hupermutation is based on Gaussian mutation. The nCSIA algorithm is proposed in this paper and its operations are given in Section 3.

4.2. *Parameter Setting*

The parameters in algorithm are set as follows: the dimension of search space n = 30 for functions $f_1 \sim f_{12}$. The antibody population size m = 30, the maximal generation number T = 2000, the inertia weight w =0.9, the acceleration constant $c_1 = c_2 = 0.5$ and the degree of uncertainty parameters $\sigma_{\max} = 0.4$, $\sigma_{\min} = 0.001$, $\alpha = 0.2$ and $interval = T/10$, system parameter b = 5. For functions $f_{13} \sim f_{15}$, the dimension of the search space n = 4, and the other parameters are the same to the functions $f_1 \sim f_{12}$.

4.3. *The Experimental Results and Analysis*

The experimental results of PSO, pPSO, FEP, ImPSO, nCSIA are summarized in Tables 2-4. All the experimental data are obtained from the statistical results of 30 independent runs except FEP. The experimental results of FEP are fetched from Ref. 19 immediately. FEP does not provide the best and median experimental items and the corresponding items are filled with the character of slash "/" in Tables. Furthermore, the maximal generation numbers of FEP vary with questions in the range of 1 500 to 20 000. That is to say, the comparison between FEP and other algorithms is unfair to a certain extent, which is merely taken as a reference.

Generally speaking, nCSIA algorithm illustrates encouraging performance as Tables 2-4 show. Firstly, nCSIA found the most truly optimal solutions than all other algorithms except for FEP (13 from 15 functions) observed from the "best" items in Tables. Secondly, it has obviously better performance than PSO on all of the test functions, which shows steady and robust performance of the 30 independent tests. Thirdly, nCSIA outperforms pPSO greatly for high dimensional functions $f_1, f_2, f_4, f_6, f_9, f_{10}, f_{11}, f_{12}$ and the solution accuracy has also been raised drastically. Both of them have comparative performance for functions f_5, f_7, f_8 and three low dimensional multimodal functions. For function f_3, the best item of pPSO is better than that of nCSIA, however, the items of median, mean and STD of nCSIA are markedly better than those of pPSO. Fourthly, nCSIA has better or similar performance with
FEP for 10 functions and is slightly worse than FEP for other functions. For immunity based algorithms ImPSO and nCSIA, the latter has more satisfactory performance than the former for functions $f_2, f_4, f_6, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}$, performs comparatively for functions $f_3, f_5, f_7, f_8, f_9, f_{10}$ and is worse than the former for function f_1. In a word, nCSIA has an encouraging performance in terms of convergence, accuracy and stability comparing with four other algorithms for various different benchmark functions.

4.4. *Online Performance Comparison and Analysis*

In order to graphically present the comparison in terms of average convergence characteristics of the evolutionary processes in solving different problems, now we show the online performance comparison of four algorithms with fourteen benchmark functions. The abscissa stands for the evolutionary generations and the vertical axis is the logarithmic plot of the average function values of the 30 simulations for the first twelve functions. For functions f_{14}, f_{15}, they are the average function values of the 30 simulations.

Observed from Fig. 3, we can get the conclusion that nCSIA has stronger exploration and exploitation abilities and can reach satisfactory evolutionary behaviors for most functions. There is a common and interesting phenomenon that the online performance of nCSIA is worse than other algorithms at the early stage in all the figures, however, it will preponderate over them at about the middle or later stage. This situation is coincident with the features of the non-uniform mutation[21]. Non-uniform mutation does not focus on the exploitation, but on the exploration for the promising search area at the early stage, which results in an apparently inferior performance to other algorithms. But with the progress of algorithm, the predominance of non-uniform mutation operation emerges, namely to surpass or to come up with other algorithms. For example, functions f_4, f_9, f_{14} are three representatives for three kinds of benchmark functions of featuring these evolutionary behaviors.

This mechanism is to keep nCSIA not easy to be trapped by the local optimal area with the algorithm run. Furthermore, it is explicit that nCSIA still has the trend to find even better solutions if the maximal generation numbers were increased for functions $f_1, f_2, f_3, f_4, f_{11}$ and f_{12}.

5. Experimental Comparison with Expanding Spaces and Increasing Dimensions

Encouraging results have demonstrated through numerical experiments in last Section when comparing with standard and recent algorithms. However, can the excellent performance of nCSIA remain when the benchmark functions become more difficult? Expanding their search spaces and increasing their dimensions are considered in this paper.

Three unimodal functions f_1, f_4, f_6 and three multimodal functions f_{10}, f_{11}, f_{12}, which have different characteristics, are selected as representatives to do the further comparable experiments due to computational costs. f_1 is a typical unimodal function which is used to validate the quick convergence ability of algorithm. f_6 is a step function, which is characterized by plateaus and discontinuity. It is well-known that the numbers of local minima of f_{10}, f_{11}, f_{12} increase exponentially with the dimension increasing. They are the most difficult class of problems for many optimization algorithms[19] and they are used to check if the performance of algorithms is dramatically affected by the enlarging spaces or not.

Algorithms pPSO, ImPSO and nCSIA are considered in these two groups of experiments.

5.1. *Comparison for Expanding Domains*

Eq. (6) requires setting the lower bound and the upper bound. It is no problem for the present benchmarks. However, it is difficult to do so for real world problems, for we don't know where the optima situate. The possible guidelines for setting the variable bounds are the personal or expert experience and multiple trials starting from a large domain which is smaller and smaller. It is pivotal that whether algorithms have similar performance with expanding domains. For this reason, this group of experiment is done to verify whether nCSIA has steady and consistent performance when facing this situation. Four different domains are adopted, which are the initial, 10 times, 100 times and 1000 times larger domains. Function f_1 is taken as an example, namely four groups of experiments are conducted with the initial domain $x \in [-10^2, 10^2]^n$, 10 times domain $x \in [-10^3, 10^3]^n$, 100 times domain $x \in [-10^4, 10^4]^n$ and 1000 times domain $x \in [-10^5, 10^5]^n$, where n is the dimension of benchmark. In order to make the simulation more persuasion, all the parameters are the same to the above experimental settings and the statistical results are also obtained from 30 independent runs.

There is something to say that the search space is drastically enlarged as we did. For example, the expanded search space will be $1000^n \cdot \Omega$ if the initial space of a function is Ω when 1000 times domain is considered. The search space is expanded 10^{90} times which is an astronomical datum when n is 30.

The items "1, 10, 100, 1000" at the first column of Tables 5-10 mean the expanding times of the initial domain of the function. For example, "1" in Table 5 means the initial domain and "1000" means the domain of expanding 1000 times of function 1. Item "Algo." is the abbreviation of "algorithm".

Generally speaking, the results of three algorithms for these selected functions are worsening with the domains expanding observed from the Tables 5-10. The expanding domains have most evident effects on pPSO whose results are heavily affected, and ImPSO is next. Observing from the "Best" items of nCSIA of all six benchmarks, it can be seen that the performance of nCSIA is slightly worsening to some extent with the domain expanding. Similar situation occurs to the "Mean" items of six functions except for f_{11}. The mean value of f_5 is 2.83 for the initial domain, however, it is 0.669 and 0.77 for the domains of 10 and 100 times larger. Comparatively saying, the superiority of nCSIA over pPSO and ImPSO are more evident with larger search domains as Tables 5-10 show.

The obvious performance difference of three algorithms with different search spaces maybe should attribute to the non-uniform hypermutation of nCSIA because pPSO and ImPSO lack this operation, which is introduced to AIS in this paper. Therefore we conclude that expanding domain has a little influence on nCSIA, however, it is much smaller than that of other algorithms.

5.2. *Comparison for Increasing Dimensions*

The algorithms will be further compared each other with higher dimensional benchmarks which are 30, 60, 90 and 120. The comparison results are given in Tables 11-16.

Observed from Tables 11-16, similar conclusions can be also reached as the above experiment. Three algorithms have worse and worse performance with dimension increasing, however the worsening speeds are different. On the whole, pPSO and ImPSO perform comparatively. However, nCSIA still shows a steady and robust performance with the increasing dimensions when comparing with two other algorithms. The effects of the proposed strategies are verified again from these numerical experiments. It also can be concluded that the higher dimension has some influence on nCSIA, however, it is the least impact on the performance.

Table 1. Benchmark functions, search spaces and the known optima.

$f_1 = \sum_{i=1}^{n} x_i^2$	$[-100,100]^n$	0
$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-100,100]^n$	0
$f_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100,100]^n$	0
$f_4 = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	$[-100,100]^n$	0
$f_5 = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30,30]^n$	0
$f_6 = \sum_{i=1}^{n} \left(\lfloor x_i + 0.5 \rfloor \right)^2$	$[-100,100]^n$	0
$f_7 = \sum_{i=1}^{n} i * x_i^4 + random[0,1)$	$[-1.28,1.28]^n$	0
$f_8 = \sum_{i=1}^{n} \left[x_i^2 - 10 * \cos(2\pi x_i) + 10 \right]$	$[-5.12,5.12]^n$	0
$f_9 = -20 \exp\left[-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right] - \exp(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e$	$[-32,32]^n$	0
$f_{10} = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600,600]^n$	0
$f_{11} = \frac{\pi}{n} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 * \left[1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	$[-50,50]^n$	0
$f_{12} = 0.1 \left\{ 10 \sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 * \left[1 + 10 \sin^2(\pi x_{i+1}) \right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_n) \right] \right\}$ $+ \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50,50]^n$	0
where $y_i = 1 + \frac{1}{4}(1 + x_i)$, $u(x_i, a, k, s) = \begin{cases} k(x_i - a)^s & , \text{if } x_i > a \\ 0 & , \text{if } -a < x_i < a \\ k(-x_i - a)^s & , \text{if } x_i < -a \end{cases}$		
$f_{13} = -\sum_{i=1}^{5} \left[(x - a_i)(x - a_i)^T + c_i \right]^{-1}$	$[0,10]^4$	-10.1532
$f_{14} = -\sum_{i=1}^{7} \left[(x - a_i)(x - a_i)^T + c_i \right]^{-1}$	$[0,10]^4$	-10.4028
$f_{15} = -\sum_{i=1}^{10} \left[(x - a_i)(x - a_i)^T + c_i \right]^{-1}$	$[0,10]^4$	-10.5362

Table 2. Results comparison among PSO, pPSO, FEP, ImPSO, nCSIA for high dimensional unimodal functions. "/" means the item not provided.

Function	Algorithm	Best	Median	Mean	STD
f_1	PSO	2.12E-01	8.94E+00	1.81E+01	5.22E+02
	pPSO	5.32E-06	6.89E-06	6.95E-06	7.95E-13
	FEP	/	/	5.74E-4	1.3E-4
	ImPSO	8.39E-14	4.04E-12	2.96E-11	1.11E-20
	nCSIA	1.10E-09	3.89E-09	5.88E-09	6.34E-17
f_2	PSO	3.53E+00	1.35E+01	1.44E+01	7.79E+01
	pPSO	1.02E-02	1.81E-01	3.17E-01	2.48E-01
	FEP	/	/	8.4E-3	7.7E-4
	ImPSO	4.01E-05	7.74E-04	2.76E-03	2.81E-05
	nCSIA	2.71E-05	2.36E-04	5.41E-04	5.47E-07
f_3	PSO	7.20E+02	3.46E+03	4.17E+03	4.66E+06
	pPSO	9.28E-05	2.04E+01	2.63E+02	5.51E+05
	FEP	/	/	1.6E-2	1.4E-2
	ImPSO	8.96E-03	5.17E-02	5.91E-02	2.91E-03
	nCSIA	1.01E-02	2.75E-02	3.51E-02	5.37E-04
f_4	PSO	8.33E+00	2.00E+01	2.03E+01	2.90E+01
	pPSO	1.79E+00	6.39E+00	6.52E+00	1.37E+01
	FEP	/	/	0.3	0.5
	ImPSO	1.36E-01	8.74E-01	1.74E+00	3.88E+00
	nCSIA	1.68E-02	4.30E-02	7.05E-02	4.03E-03
f_5	PSO	1.20E+02	7.94E+02	1.36E+03	1.44E+06
	pPSO	1.89E+01	2.49E+01	1.15E+02	9.77E+04
	FEP	/	/	5.06	5.87
	ImPSO	1.44E+01	2.53E+01	3.35E+01	5.54E+02
	nCSIA	2.24E+01	2.68E+01	3.67E+01	7.31E+02
f_6	PSO	7.40E+01	2.85E+02	3.44E+02	7.42E+04
	pPSO	0	4.00	4.53	6.53
	FEP	/	/	0	0
	ImPSO	0	0	3.33E-02	3.33E-02
	nCSIA	0	0	0	0
f_7	PSO	6.09E-01	1.20E+00	1.32E+00	2.40E-01
	pPSO	1.96E-01	4.79E-01	5.73E-01	1.52E-01
	FEP	/	/	7.6E-3	2.6E-3
	ImPSO	2.01E-01	7.35E-01	6.90E-01	8.57E-02
	nCSIA	2.43E-01	6.70E-01	7.06E-01	8.77E-02

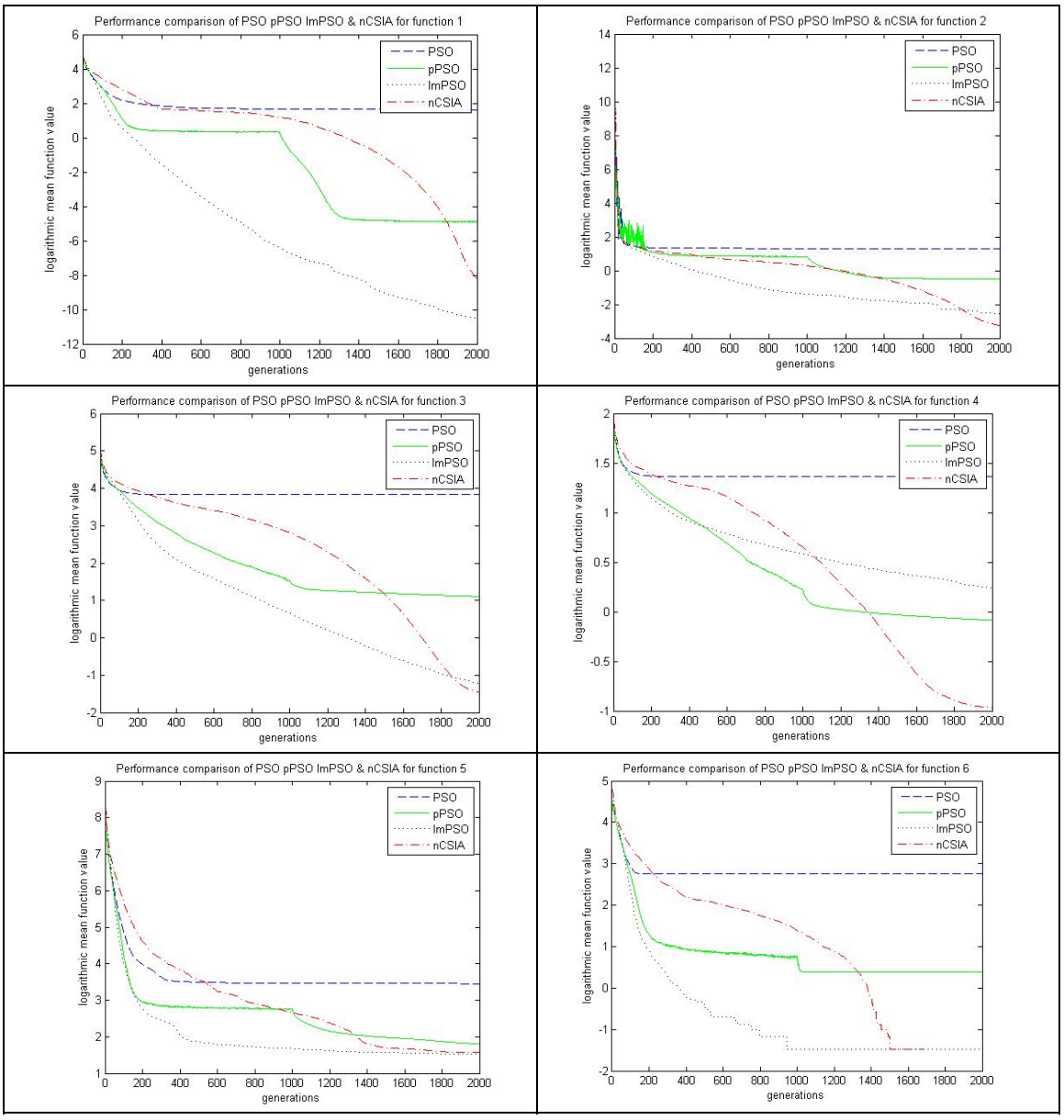
Table 3. Results comparison among PSO, pPSO, FEP, ImPSO, nCSIA for high dimensional multimodal functions. "/" means the item not provided.

Function	Algorithm	Best	Median	Mean	STD
f_8	PSO	5.11E+01	7.83E+01	8.57E+01	7.33E+02
	pPSO	4.58E+01	8.36E+01	8.50E+01	4.77E+02
	FEP	/	/	4.6E-2	1.2E-2
	ImPSO	4.48E+01	6.91E+01	7.26E+01	3.02E+02
	nCSIA	3.08E+01	5.22E+01	5.42E+01	2.41E+02
f_9	PSO	3.99E+00	7.79E+00	8.11E+00	5.81E+00
	pPSO	2.03E-03	1.42E+00	1.59E+00	4.54E+00
	FEP	/	/	1.8E-2	2.1E-3
	ImPSO	4.25E-07	1.34E+00	9.72E-01	6.36E-01
	nCSIA	7.48E-06	1.63E-05	1.57E-01	2.39E-01
f_{10}	PSO	1.03E+00	1.27E+00	1.55E+00	7.58E-01
	pPSO	3.12E-07	9.86E-03	1.21E-02	1.27E-02
	FEP	/	/	1.6E-2	2.2E-2
	ImPSO	2.02E-14	7.40E-03	9.36E-03	8.20E-05
	nCSIA	2.04E-08	1.30E-07	6.23E-03	1.23E-04
f_{11}	PSO	6.71E+00	1.33E+01	1.57E+01	5.86E+01
	pPSO	3.84E+00	9.47E+00	9.78E+00	2.04E+01
	FEP	/	/	9.6E-6	3.6E-6
	ImPSO	2.22E-06	2.08E+00	3.07E+00	1.37E+01
	nCSIA	1.61E-11	2.64E+00	2.83E+00	5.44E+00
f_{12}	PSO	1.82E+01	4.00E+01	4.18E+01	1.27E+02
	pPSO	8.05E-07	1.10E-02	1.29E-02	5.27E-04
	FEP	/	/	1.6E-4	7.3E-5
	ImPSO	7.71E-13	1.11E-11	7.32E-04	7.77E-06
	nCSIA	3.81E-11	2.30E-10	3.06E-10	9.19E-20

Table 4. Results comparison among PSO, pPSO, FEP, ImPSO, nCSIA for low dimensional multimodal functions. "/" means the item not provided.

Function	Algorithm	Best	Median	Mean	STD
f_{13}	PSO	-10.2	-5.10	-6.15	13.2
	pPSO	-10.2	-7.63	-6.98	11.2
	FEP	/	/	-5.52	1.59
	ImPSO	-9.88	-5.02	-5.43	1.94
	nCSIA	-10.2	-5.10	-5.89	3.25
f_{14}	PSO	-10.4	-3.72	-5.91	12.5
	pPSO	-10.4	-3.72	-6.10	13.2

Function	Algorithm	Best	Median	Mean	STD
	FEP	/	/	-5.52	2.12
	ImPSO	-5.13	-5.03	-5.01	1.30E-02
	nCSIA	-10.4	-5.13	-6.49	5.30
	PSO	-10.5	-3.84	-4.91	10.5
	pPSO	-10.5	-10.5	-7.08	14.6
f_{15}	FEP	/	/	-6.57	3.14
	ImPSO	-5.17	-5.09	-5.06	1.27E-02
	nCSIA	-10.5	-5.18	-6.07	4.12



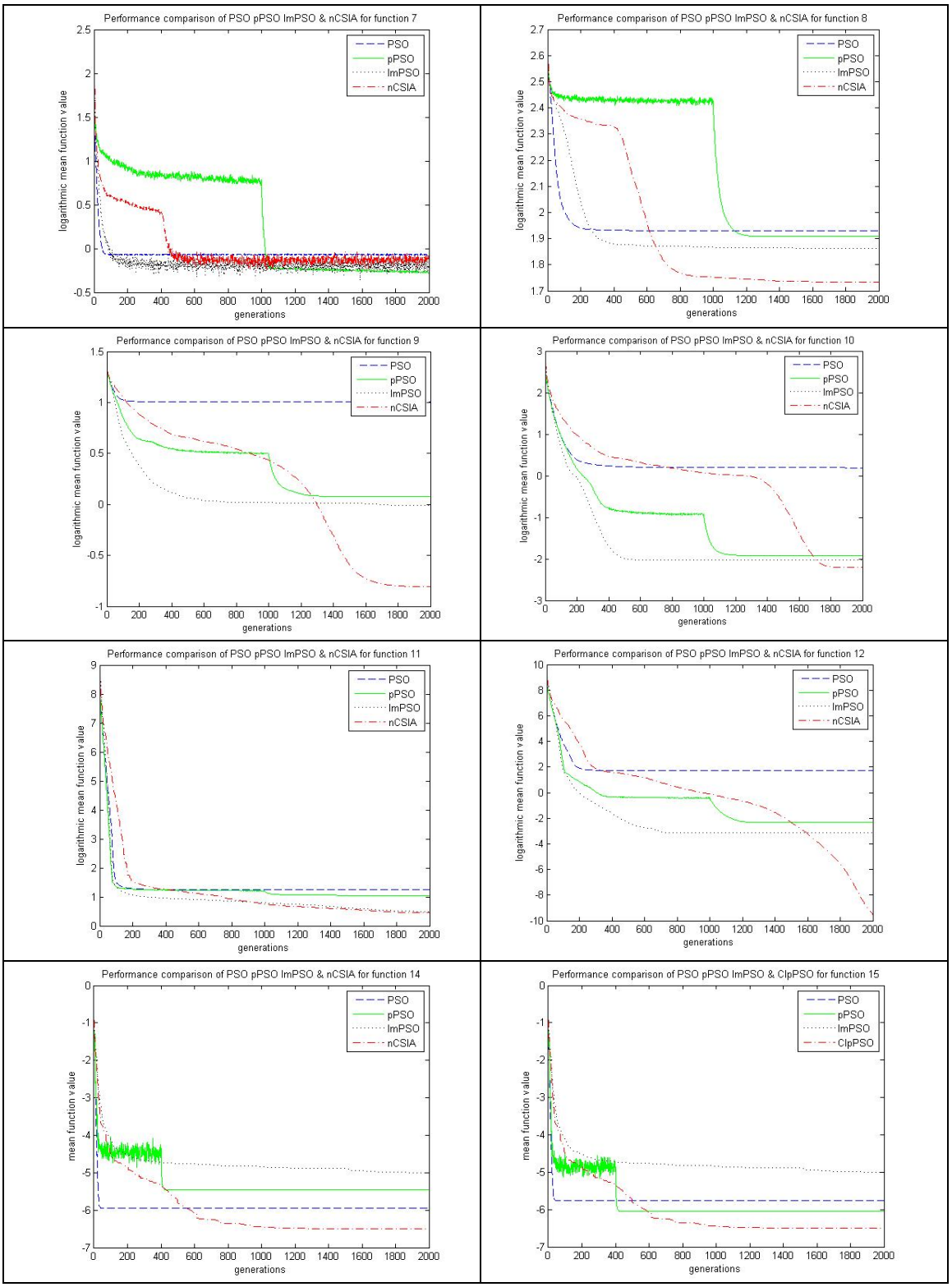


Fig. 3. PSO, pPSO, ImPSO & nCSIA online performance analysis

6. Conclusion and Future studies

A new clonal selection immune algorithm (nCSIA) is proposed with the ideas of non-uniform hypermutation and perturbation guiding search based on the clonal selection principle. In immunology, affinity is the fitness measurement for an antibody. In the paper it is computed in direct proportion to its fitness and in inverse proportion to its distance to the global best individual which coincides with the inherent exploration and exploitation tradeoff requirement of swarm intelligence. Proportional clone and non-uniform hypermutation adaptively adjusts the exploration or exploitation radius at the neighborhood of the promising individuals for the balance of population diversity and selection pressure. An elitist learning strategy of perturbation guiding search is also proposed and for better balance of global search and local search.

In order to validate the performance of nCSIA a lot of experiments have been done and compared with other four algorithms including standard PSO, perturbed PSO (pPSO), immune based PSO (ImPSO) and fast evolutionary programming (FEP). Comparison with pPSO is to verify the effect of proportional cloning and affinity proliferation for hybrid algorithm. Comparison with ImPSO is to prove that the idea of perturbation guiding search is good for the new algorithm. Experimental results show that nCSIA algorithm has even better performance in terms of convergence, robustness and stability for most benchmark functions. Thus it indicates that the proposed strategies are effective and promising.

Further experiments are conducted when functions are becoming more difficult with much larger search spaces. Simulation results illustrate that the superiority of nCSIA are more evident with larger search space although the expanding domain and increasing dimension have some influence on algorithms for a certainty. That is to say, it has a little influence on nCSIA, however, it is much smaller than that of other algorithms.

HyperMutation and perturbation guiding search have important effects on the algorithms. Better hypermutation operation for adaptive exploration and exploitation and even more intelligent elitist learning strategy for perturbation guiding search deserve further research.

Table 5. Results of function 1 with expanding domain

f1	Algo.	Best	Mean	STD
1	pPSO	5.32E-06	6.95E-06	7.95E-13
	ImPSO	8.39E-14	2.96E-11	1.11E-20
	nCSIA	1.10E-09	5.88E-09	6.34E-17
10	pPSO	5.27E-06	6.91E-06	8.80E-13
	ImPSO	3.13E-03	8.46E-02	6.89E-03
	nCSIA	1.20E-07	7.66E-07	7.26E-13
100	pPSO	7.10E-06	8.00E+04	1.41E+10
	ImPSO	4.26E-03	1.20E-01	2.82E-02
	nCSIA	9.77E-06	5.37E-05	9.71E-10
1000	pPSO	7.28E+05	3.33E+07	1.25E+15
	ImPSO	1.32E-03	1.70E-01	9.54E-02
	nCSIA	1.14E-03	5.12E-03	1.28E-05

Table 6. Results of function 4 with expanding domain

f4	Algo.	Best	Mean	STD
1	pPSO	1.79E+00	6.52E+00	1.37E+01
	ImPSO	1.36E-01	1.74E+00	3.88E+00
	nCSIA	1.68E-02	7.05E-02	4.03E-03
10	pPSO	8.16E+01	1.83E+02	3.08E+03
	ImPSO	3.72E+00	4.09E+01	9.52E+02
	nCSIA	7.37E-02	2.42E-01	1.68E-02
100	pPSO	1.38E+03	2.40E+03	4.24E+05
	ImPSO	7.71E+01	6.26E+02	2.01E+05
	nCSIA	3.20E-01	2.41E+00	9.88E+00
1000	pPSO	1.39E+04	2.46E+04	3.75E+07
	ImPSO	7.70E+03	1.82E+04	2.57E+07
	nCSIA	3.82E+00	2.42E+01	1.11E+03

Table 7. Results of function 6 with expanding domain

f6	Algo.	Best	Mean	STD
1	pPSO	0	4.53	6.53
	ImPSO	0	3.33E-02	3.33E-02
	nCSIA	0	0	0
10	pPSO	0	1.03E+01	5.08E+01
	ImPSO	4	1.03E+01	1.55E+01
	nCSIA	0	1.33E-01	1.20E-01
100	pPSO	1.10E+01	1.55E+05	4.05E+10
	ImPSO	4	1.13E+01	1.71E+01
	nCSIA	0	1.00E+00	1.45E+00
1000	pPSO	1.41E+06	3.04E+07	9.23E+14
	ImPSO	1	1.01E+01	1.88E+01
	nCSIA	0	3.40E+00	7.28E+00

Table 8. Results of function 10 with expanding domain

f10	Algo.	Best	Mean	STD
1	pPSO	3.12E-07	1.21E-02	1.27E-02
	ImPSO	2.02E-14	9.36E-03	8.20E-05
	nCSIA	2.04E-08	6.23E-03	1.23E-04
10	pPSO	3.33E-07	1.09E+01	5.25E+02
	ImPSO	6.59E-04	2.44E-02	3.80E-04
	nCSIA	7.05E-07	7.80E-03	1.01E-04
100	pPSO	8.28E+01	2.18E+03	8.78E+06
	ImPSO	2.12E-04	1.62E-02	1.28E-04
	nCSIA	1.29E-04	1.41E-02	3.10E-04
1000	pPSO	3.00E+01	2.43E+05	8.96E+10
	ImPSO	2.81E-03	1.96E+04	4.44E+09
	nCSIA	3.41E-03	2.28E-02	1.84E-04

Table 9. Results of function 11 with expanding domain

f11	Algo.	Best	Mean	STD
1	pPSO	3.84E+00	9.78E+00	2.04E+01
	ImPSO	2.22E-06	3.07E+00	1.37E+01
	nCSIA	1.61E-11	2.83E+00	5.44E+00
10	pPSO	3.75E+00	1.06E+01	2.15E+01
	ImPSO	6.11E+00	1.67E+01	3.45E+01
	nCSIA	6.64E-10	6.69E-01	1.31E+00
100	pPSO	1.19E+01	6.56E+10	5.54E+22
	ImPSO	4.48E+00	1.48E+01	3.76E+01
	nCSIA	3.64E-07	7.70E-01	1.56E+00
1000	pPSO	1.02E+12	4.40E+15	2.22E+32
	ImPSO	7.30E+00	1.52E+01	5.20E+01
	nCSIA	1.56E-04	1.94E+00	2.50E+00

Table 10. Results of function 12 with expanding domain

f12	Algo.	Best	Mean	STD
1	pPSO	8.05E-07	1.29E-02	5.27E-04
	ImPSO	7.71E-13	7.32E-04	7.77E-06
	nCSIA	3.81E-11	3.06E-10	9.19E-20
10	pPSO	5.95E-07	2.06E+00	5.18E+01
	ImPSO	1.94E-03	2.18E+01	4.06E+02
	nCSIA	7.40E-20	6.18E-03	3.21E-04
100	pPSO	1.68E+01	9.71E+10	4.82E+22
	ImPSO	1.59E-01	2.62E+01	5.67E+02
	nCSIA	1.25E-19	1.31E-02	1.32E-03
1000	pPSO	1.78E+12	6.93E+15	2.33E+32
	ImPSO	2.39E-01	2.93E+01	4.94E+02
	nCSIA	1.10E-19	1.25E-02	3.94E-04

Table 11. Results of function 1 with increasing dimension

f1	Algo.	Best	Mean	STD
30	pPSO	5.32E-06	6.95E-06	7.95E-13
	ImPSO	8.39E-14	2.96E-11	1.11E-20
	nCSIA	1.10E-09	5.88E-09	6.34E-17
60	pPSO	3.66E-05	4.37E-05	4.26E-11
	ImPSO	7.19E+00	1.59E+01	3.64E+01
	nCSIA	1.99E-05	5.21E-05	9.62E-10
90	pPSO	1.13E+00	9.26E+01	1.04E+04
	ImPSO	3.04E+01	6.58E+01	3.45E+02
	nCSIA	3.03E-03	6.55E-03	5.43E-06
120	pPSO	5.81E+02	1.12E+03	1.86E+05
	ImPSO	7.70E+01	1.45E+02	1.82E+03
	nCSIA	6.48E-02	1.19E-01	6.93E-04

Table 12. Results of function 4 with increasing dimension

f4	Algo.	Best	Mean	STD
30	pPSO	1.79E+00	6.52E+00	1.37E+01
	ImPSO	1.36E-01	1.74E+00	3.88E+00
	nCSIA	1.68E-02	7.05E-02	4.03E-03
60	pPSO	2.18E+01	2.77E+01	1.56E+01
	ImPSO	1.96E+01	3.10E+01	2.30E+01
	nCSIA	2.97E+00	6.41E+00	3.98E+00
90	pPSO	3.10E+01	3.97E+01	2.64E+01
	ImPSO	2.99E+01	4.19E+01	3.51E+01
	nCSIA	8.46E+00	1.19E+01	4.65E+00
120	pPSO	3.27E+01	4.51E+01	1.70E+01
	ImPSO	3.98E+01	4.87E+01	2.58E+01
	nCSIA	1.18E+01	1.62E+01	3.30E+00

Table 13. Results of function 6 with increasing dimension

f6	Algo.	Best	Mean	STD
30	pPSO	0	4.53	6.53
	ImPSO	0	3.33E-02	3.33E-02
	nCSIA	0	0	0
60	pPSO	3.00E+01	5.94E+01	6.14E+02
	ImPSO	4.10E+01	7.23E+01	3.40E+02
	nCSIA	1.00E+00	4.47E+00	3.98E+00
90	pPSO	2.12E+02	4.07E+02	1.61E+04
	ImPSO	1.20E+02	2.12E+02	1.74E+03
	nCSIA	1.10E+01	2.17E+01	4.05E+01
120	pPSO	6.64E+02	2.29E+03	8.29E+05
	ImPSO	2.58E+02	4.40E+02	1.13E+04
	nCSIA	2.90E+01	5.70E+01	2.37E+02

Table 14. Results of function 10 with increasing dimension

f10	Algo.	Best	Mean	STD
30	pPSO	3.12E-07	1.21E-02	1.27E-02
	ImPSO	2.02E-14	9.36E-03	8.20E-05
	nCSIA	2.04E-08	6.23E-03	1.23E-04
60	pPSO	5.98E-01	2.68E+00	6.94E+00
	ImPSO	1.96E-01	3.53E-01	8.82E-03
	nCSIA	1.15E-04	4.51E-03	3.71E-05
90	pPSO	9.91E+00	3.98E+01	3.87E+02
	ImPSO	2.46E-01	6.81E-01	1.70E-02
	nCSIA	6.21E-03	1.68E-02	4.13E-05
120	pPSO	6.68E+01	1.55E+02	1.90E+03
	ImPSO	6.31E-01	8.38E-01	9.26E-03
	nCSIA	4.22E-02	7.72E-02	2.98E-04

Table 15. Results of function 11 with increasing dimension

f11	Algo.	Best	Mean	STD
30	pPSO	3.84E+00	9.78E+00	2.04E+01
	ImPSO	2.22E-06	3.07E+00	1.37E+01
	nCSIA	1.61E-11	2.83E+00	5.44E+00
60	pPSO	8.57E+00	1.46E+01	1.50E+01
	ImPSO	1.48E+01	2.81E+01	7.25E+01
	nCSIA	9.15E-01	4.02E+00	3.27E+00
90	pPSO	8.52E+00	1.45E+01	9.46E+00
	ImPSO	3.45E+01	8.52E+01	8.28E+03
	nCSIA	3.11E+00	5.00E+00	1.43E+00
120	pPSO	1.22E+01	2.20E+01	1.38E+02
	ImPSO	4.90E+01	4.16E+02	2.48E+05
	nCSIA	3.38E+00	5.79E+00	1.68E+00

Table 16. Results of function 12 with increasing dimension

f12	Algo.	Best	Mean	STD
30	pPSO	8.05E-07	1.29E-02	5.27E-04
	ImPSO	7.71E-13	7.32E-04	7.77E-06
	nCSIA	3.81E-11	3.06E-10	9.19E-20
60	pPSO	7.35E-01	2.87E+01	5.36E+02
	ImPSO	8.22E+01	1.19E+02	7.09E+02
	nCSIA	1.31E-07	1.74E-01	3.24E-01
90	pPSO	6.57E+01	1.19E+02	4.82E+02
	ImPSO	1.49E+02	5.77E+02	5.11E+05
	nCSIA	7.41E-04	8.79E-02	8.88E-02
120	pPSO	1.75E+02	1.10E+03	1.08E+07
	ImPSO	2.91E+02	4.86E+03	1.21E+07
	nCSIA	3.74E-01	2.28E+01	5.25E+02

Acknowledgements

This paper is supported by the National Natural Science Foundation of China (No. 61070207) and the Fundamental Research Funds for the Central Universities (BUPT2009RC0701) of China

References

1. L. N. de Castro, and F. J. von Zuben, Artificial Immune Systems：Part I-Basic Theory and Applications, Technical Report TR-DCA 01/99, 1999.
2. L. N. de Castro, and F. J. von Zuben, Artificial Immune Systems： Part II-A Survey of Applications, Technical Report DCA-RT 02/00, 2000.
3. X. Shen, X.Z. Gao, R.F. Bie, Artificial Immune Networks: Models and Applications, International Journal of Computational Intelligence System, 1(2):168-176, 2008.
4. E. Hart and J. Timmis. Application areas of AIS: The past, the present and the future, Applied Soft Computing, 8(1):191-201, 2008.
5. L. N. de Castro, and F. J. von Zuben, Learning and Optimization using the Clonal Selection Principle, IEEE Trans. Evolutionary Computation, 6(3):239-251, 2002.
6. L. Li, H.-Q. Li, S.-L. Xie and X.-Y. Li, Immune Particle Swarm Optimization Algorithms Based on Clone Selection, Computer Science, 35(10):253-255, 2008.
7. M.G. Gong, L.C. Jiao and L.N. Zhang. Baldwinian Learning in Clonal Selection Algorithm for Optimization. Information Sciences, 180(8):1218-1236, 2010.
8. L. N. De Castro and J. Timmis, Artificial Immune Systems: A New Computational Intelligence Approach. Springer-Verlag, Berlin, 2002.
9. J. Kennedy and R. C. Eberhart, Particle swarm optimization, Proc. IEEE Int. Conf. Neural Netw., Perth, Australia, 1995, vol. 4, pp1942–1948.
10. X.L. Kou, S.Y. Liu, J.K. Zhang, and W. Zheng. Co-evolutionary particle swarm optimization to solve constrained optimization problems. Computers & Mathematics with Applications, 57(11-12):1776-1784, 2009.
11. S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, OPSO: Orthogonal particle swarm optimization and its application to task assignment problems, IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 38(2):288–298, 2008.
12. Y. Marinakis, M. Marinaki, and G. Dounias. A hybrid particle swarm optimization algorithm for the vehicle routing problem, Engineering Applications of Artificial Intelligence, 23(4):463-472, 2010.
13. L. Zhao, F. Qian, and Y.P. Yang et al. Automatically extracting T–S fuzzy models using cooperative random learning particle swarm optimization, Applied Soft Computing, 10(3):938-944, 2010.
14. Y. Wang, B. Li, T. Weise et al. Self-adaptive learning based particle swarm optimization, Information Sciences, (in Press).
15. X.C. Zhao. A perturbed particle swarm algorithm for numerical optimization, Applied Soft Computing, 10(1):119-124, 2010.
16. D. Dubois, H. Prade. Possibility Theory: An Approach to

Computerized Processing and Uncertainty, Plenum Press, New York, 1988.
17. Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs (3rd Edition), Springer, 1996.
18. D.B. Fogel, Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, New York: IEEE Press, 1995.
19. X. Yao, Y. Liu and G.M. Lin. Evolutionary Programming Made Faster, IEEE Trans. on Evolutionary Computation, 3(2):82-102, 1999.
20. C.Y. Lee and X. Yao. Evolutionary Programming Using Mutations Based on the Lévy Probability Distribution, IEEE Trans. Evolutionary Computation, 8(1):1-13, 2004.
21. X.C. Zhao, X.-S. Gao and Z.-C. Hu. Evolutionary programming based on non-uniform mutation, Applied Mathematics and Computation, 192(1):1-11, 2007.
22. H.B. Dong, J. He, H.K. Huang, and W. Hou, Evolutionary programming using a mixed mutation strategy, Information Sciences, 177:312-327, 2007.
23. A.O. Tarakanov, D. Dasgupta. A Formal Model of an Artificial Immune System, Biosystems, 55:151-158, 2000.
24. V. Cutello, G. Nicosia, M. Pavone, Exploring the capability of immune algorithms: a characterization of Hypemutation operators, in: Proceedings of Third International Conference on Artificial Immune Systems, LNCS3239, 263–276, 2004.
25. T. Fukuda, K. Mori, M. Tsukiyama. Immune networks using genetic algorithm for adaptive production scheduling, Proceedings of the 15th IFAC World Congress, 57–60, 1993.
26. S. M. Garrett, How do we evaluate artificial immune systems, Evolutionary Computation, 13(2):145–178, 2005.
27. D. Dasgupta and F. Nino, Immunological Computation: Theory and Practice, CRC Press, 2008.
28. R.B. Xiao, P.B. Cao and Y. Liu, Engineering Immune Computing, Beijing: Science Press, 2007.
29. C.A. Laurentys, G. Ronacher, R.M. Palhares, W.M. Caminhas. Design of an Artificial Immune System for fault detection: A Negative Selection Approach, Expert Systems with Applications, 37:5507–5513, 2010.
30. P.K. Harmer, P.D. Williams, G.H. Gunsch and G.B. Lamont. An artificial immune system architecture for computer security applications, IEEE Transactions on Evolutionary Computation, 6 (3):252–280, 2002.
31. M. Swimmer. Using the danger model of immune systems for distributed defense in modern data networks. Computer Networks, 51(5):1315-1333, 2007.
32. I. Aydin, M. Karakose, and E. Akin. Chaotic-based hybrid negative selection algorithm and its applications in fault and anomaly detection, Expert Systems with Applications, 37(7):5285-5294, 2010.
33. J.H. Carter, The immune system as a model for pattern recognition and classification, J. Am. Med. Inf. Assoc., 7 (3):28–41, 2000.
34. B. Chen, C.Z. Zang. Artificial immune pattern recognition for structure damage classification, Computers & Structures, 87(21-22):1394-1407, 2009.
35. Z.Q. Ji, Q.M.J. Wu, An improved artificial immune algorithm with application to multiple sensor systems, Information Fusion, 11:174-182, 2010.
36. H. Kahramanli, N. Allahverdi. Extracting rules for classification problems: AIS based approach, Expert Systems with Applications, 36(7):10494-10502, 2009.
37. K. Trojanowski , S.T. Wierzchon. Immune-based algorithms for dynamic optimization, Information Sciences, 179:1495–1515, 2009.
38. X.Q. Zuo, H.W. Mo, and J.P. Wu. A robust scheduling method based on a multi-objective immune algorithm, Information Sciences, 179:3359-3369, 2009.
39. X. Wang, X.Z. Gao, S. J. Ovaska, A hybrid artificial immune optimization method, International Journal of Computational Intelligence System, 2(3):248 –255, 2009.
40. C.A. Coello Coello and N.C. Cortes, Solving multiobjective optimization problems using an artificial immune system, Genetic Programming and Evolvable Machines, 6:163–190, 2005.
41. F. Freschi and M. Repetto. Multiobjective optimization by a modified artificial immune system algorithm, Proceedings of the 4th International Conference on artificial immune systems (ICARIS 2005), LNCS3627, 248–261, 2005.
42. M.G. Gong, L.C. Jiao and H.F. Du et al. Multiobjective Immune Algorithm with Pareto-optimal Neighbor-based Selection, Evolutionary Computation, 16(2):225-255, 2008.
43. J.Y. Chen, Q.Z. Lin, and Z. Ji. A hybrid immune multiobjective optimization algorithm, European Journal of Operational Research, 204:294–302, 2010.
44. R. Poli, J. Kennedy and T. Blackwell. Particle swarm optimization, Swarm Intelligence, 1:33-57, 2007.
45. Z.-H. Zhan, J. Zhang and Y. Li et al. Adaptive Particle Swarm Optimization, IEEE Trans. Syst., Man, Cybern. B, Cybern., 39(6):1362-1381, 2009.
46. D.S. Liu, K.C. Tan, C.K. Goh and W.K. Ho, A Multiobjective Memetic Algorithm Based on Particle Swarm Optimization, IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics, 37(1):42-50, 2007.