Available online at www.sciencedirect.com

ScienceDirect

Journal of Bionic Engineering 8 (2011) 181–190

A Multi-Agent Immune Network Algorithm and Its Application to Murphree Efficiency Determination for the Distillation Column

Xuhua Shi[1,2], Feng Qian[1]

1. *State Key Laboratory of Chemical Engineering, East China University of Science and Technology, Shanghai* 200237, *P. R. China*
2. *College of Information Science and Engineering, Ningbo University, Ningbo* 315211, *P. R. China*

Abstract

Artificial Immune Network (aiNet) algorithms have become popular for global optimization in many modern industrial applications. However, high-dimensional systems using such models suffer from a potential premature convergence problem. In the existing aiNet algorithms, the premature convergence problem can be avoided by implementing various clonal selection methods, such as immune suppression and mutation approaches, both for single population and multi-population cases. This paper presents a new Multi-Agent Artificial Immune Network (Ma-aiNet) algorithm, which combines immune mechanics and multiagent technology, to overcome the premature convergence problem in high-dimensional systems and to efficiently use the agent ability of sensing and acting on the environment. Ma-aiNet integrates global and local search algorithms. The performance of the proposed method is evaluated using 10 benchmark problems, and the results are compared with other well-known intelligent algorithms. The study demonstrates that Ma-aiNet outperforms other algorithms tested. Ma-aiNet is also used to determine the Murphree efficiency of a distillation column with satisfactory results.

Keywords: bio-inspired optimization, multi-agent immune network, distillation column, Murphree efficiency

doi: 10.1016/S1672-6529(11)60024-3

1 Introduction

Bio-inspired optimization is becoming increasingly popular in many areas of engineering and science. Genetic algorithms with high efficiency and robustness in global searching have great advantages especially in optimizing multimodal functions. However, the drift in evolutionary search leads to premature convergence or populations vulnerable to a single extreme convergence[1]. While analyzing the heredity excursion factor, Yu and Guo[2] pointed out that, on the premise of not increasing the mutation probability, the key to obtaining a global optimum solution is to maintain species diversity and to implement an efficient local search within species. To suppress the genetic drift, algorithms such as fitness sharing algorithm[3], deterministic crowding algorithm[4], and species retaining algorithm[5], have been proposed to improve the population diversity. Most of the algorithms are based on research on niche technologies or their improved technologies. Although these algorithms are better in achieving an optimum solution, they have their own weaknesses, such as poor adaptability and weak local search capability.

The biological immune system draws strength from natural processes for maintaining and boosting diversity. Generally, immune response reflects how antibodies learn antigenic structure patterns and eliminate antigens. When an antigen invades the body for the first time, the immune system produces antibodies in response to eliminate it. Some antibody cells with high fitness will differentiate into memory cells, which will remain in the immune system for a long period of time. In this way, they can promote a faster and more effective response to the same antigenic challenge. The immune network theory is based on the premise that an antibody and other free antibodies can recognize each other. This mutual recognition process will result in an activation or suppression of specific kinds of antibodies. Moreover, the connected antibodies form a network. These capabilities

Corresponding author: Feng Qian
E-mail: fqian@ecust.edu.cn

endow the immune system with an intrinsic dynamic behavior through the diversity of antibodies. Inspired by the biological immune system, the artificial immune system, with strong adaptability, diversity, learning and memory, can produce a variety of antibodies and solve the population diversity issues of the basic genetic algorithm[6]. Therefore, an artificial immune system can be used in designing an appropriate strategy to solve multimodal optimization problems. de Castro[6] proposed the immune clonal selection algorithm (ClonAlg) based on the biological mechanism of clonal selection, which has a high parallel and global convergence. However, it only clones and mutates the antibody groups with high affinity, which results in poor population diversity and a limited capability to deal with high-dimensional multimodal function optimization problems[7].

De Castro and Von Zuben proposed an aiNet algorithm based on principles, such as clonal selection and high-frequency variation, to deal with data analysis and clustering tasks[8]. In a subsequent work, de Castro and Timmis developed a modified version of aiNet, Artificial Immune Network for Optimization (Opt-aiNet), for multimodal optimization problems[9]. Later Gomes *et al.* proposed a copt-aiNet algorithm,[10] as the extension of Opt-aiNet for combinatorial optimization problems. Another algorithm, Artificial Immune Network for Dynamic Optimization[11] (dopt-aiNet), is an improved and expanded version of Opt-aiNet to optimize time-varied functions. Given its complexity, only a small part of the immune mechanism model has been used in the studies mentioned above.

The design of novel artificial immune algorithms to find optimal high-dimensional multi-modal solutions is still a relatively new area of research. In this paper, we propose a Ma-aiNet algorithm, which combines the characteristics of Multi-Agent Genetic Algorithm (MAGA) and Opt-aiNet immune network algorithm.

The remaining parts of the paper are organized as follows. Section **2** describes the Ma-aiNet based on MAGA and Opt-aiNet. Section **3** contains 10 high-dimensional benchmark studies for Ma-aiNet algorithm. Section **4** presents some experimental results in the application of Murphree efficiency determination for distillation column. Concluding remarks are given in Section **5**.

2 Ma-aiNet algorithm

2.1 The basics of Opt-aiNet algorithm

The mathematical model for an optimization problem can be expressed as

$$\min_{\boldsymbol{x}\in \boldsymbol{S}\subset \boldsymbol{R}^q} f(\boldsymbol{x}), \tag{1}$$

$$\text{Subject to } \begin{cases} g_i(\boldsymbol{x}) < 0 & i = 1, 2, \cdots, m \\ h_j(\boldsymbol{x}) = 0 & j = 1, 2, \cdots, n, \end{cases}$$

where $\boldsymbol{S} = \prod_{i=1}^{q} < \alpha_{i-}, \alpha_i^- >$ is the domain; α_{i-} and α_i^- correspond to the lower and upper limits of solutions, respectively; $\boldsymbol{x} = (x_1, x_2, \ldots, x_q)\in\boldsymbol{S}$; q is the dimension of the system; $f(\boldsymbol{x})$ is the objective function; and $g_i(\boldsymbol{x})$, $h_j(\boldsymbol{x})$, ($i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$) are the constraints. For any $\boldsymbol{x} \in \boldsymbol{S}$, if $f(x^*) \le f(\boldsymbol{x})$, then $x^* \in \boldsymbol{S}$ is the global optimum solution.

In the Opt-aiNet algorithm[9], $f(\boldsymbol{x})$ is regarded as an invasive antigen, and the probable solutions are regarded as antibodies. The evolution of the antibody cluster depends on the affinity between the antibody and the antigen. The better their match is, the smaller the degree of antibody variability and vice versa. Meanwhile, the antibody cluster is a set of immune networks. If two adjacent antibodies are too close or their affinity is smaller than the network inhibition threshold value, the cells will be implemented with network compression and will have a relatively high affinity as memory cells. The convergence can be reached by iteration. By calculating the affinity of asexual reproduction and high-frequency mutation in each iteration, the memory antibody cluster for the iteration could be obtained, thus allowing for the reduction of similar antibodies by inhibiting the network. This process is repeated until the termination conditions are satisfied. Finally, the highest affinity memory antibody will be the optimal solution of the objective function. The state transformation of antibody population is denoted in Fig. 1.

For the Opt-aiNet algorithm, its characteristics of dynamic variation of population size, local and global search, and ability to maintain any number of optima fully meet the requirements of global optimization search for multimodal functions. However, the time needed for a very large number of objective function evaluations during an Opt-aiNet run would be prohibitive in high-dimensional application systems.

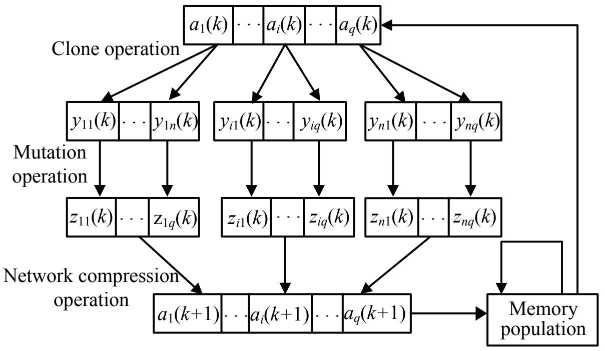

$a_1(k)$ · · · $a_i(k)$ · · · $a_q(k)$
Clone operation
$y_{11}(k)$ · · · $y_{1n}(k)$
$y_{i1}(k)$ · · · $y_{iq}(k)$
$y_{n1}(k)$ · · · $y_{nq}(k)$
Mutation operation
$z_{11}(k)$ · · · $z_{1q}(k)$
$z_{i1}(k)$ · · · $z_{iq}(k)$
$z_{n1}(k)$ · · · $z_{nq}(k)$
Network compression operation
$a_1(k+1)$ · · · $a_i(k+1)$ · · · $a_q(k+1)$
Memory population

Fig. 1 State transformation of antibody population for Opt-aiNet.

2.2 The basics of multi-agent evolution algorithm

A multi-agent system[12] is a set of computable agents with intelligent goals and capabilities. Various studies have shown the outstanding performance of such agents in real-world situations. Ishida[13] introduced a framework for an autonomous multi-agent world for adaptive control of search processes proposing the ε-search and the δ-search, which allow suboptimal solutions and balance the trade-off between exploration and exploitation. Tsui and Liu[14] presented the Extended Data Output (EDO) multi-agent framework for tackling global optimization tasks. Persson *et al.*[15,16] combined agent-based approach and classical optimization techniques in a case study. Zhong *et al.* presented the MAGA[17] and Liu *et al.* proposed a multi-agent evolutionary algorithm for constraint satisfaction problems[18]. The combination of multi-agent system and artificial immune mechanism provided a new idea for the immune algorithm. Ichimura *et al.* proposed a learning method of immune multi-agent neural networks which can automatically classify the training dataset into subclasses[19]. Liu *et al.* combined the characteristics of the multi-agent with the search strategy of immune clonal selection algorithm, and proposed a multi-agent immune clonal selection algorithm that could overcome the most serious drawbacks such as slow convergence rate and "prematurity"[20]. In this paper, we propose a multi-agent aiNet algorithm, which combines the characteristics of Opt-aiNet[9] algorithms and MAGA[17], and apply such combination to Murphree efficiency determination for distillation column.

MAGA[17] is a multi-agent optimization algorithm based on genetics and is suitable for high-dimensional function optimization. It integrates multi-agent systems and genetic algorithms to solve high-dimensional numerical optimization. An agent in MAGA represents a candidate solution of the optimization problem and lives in a lattice-like environment of size $L\times L$, as illustrated in Fig. 2. Each agent is fixed on a grid-point and cannot move, but can communicate and interchange information with surrounding agents. As such, the information can be shared by all agents after a process of diffusion. The goal of each agent is to improve its fitness. Fig. 3 describes an agent and its neighbors. An agent can only interact with other agents in the neighborhood because it only has local sensing. An agent $\boldsymbol{A}_{i,j} = [x_1, x_2, \dots, x_q]$ at location $[i, j]$, as shown in Fig. 3, contains all the q real variables of the objective function. Each agent senses its local environment, i.e., its neighborhood, and finds the best agent in its surrounding. Then the agent competes and exchanges its characteristics with its best neighbor to gain better fitness. In the MAGA[17], neighbor competition, crossover, mutation, and self-learning operators are used by the agents for these purposes. The main steps of the proposed algorithm are as follows:

(1) Create $L\times L$ agents in the lattice-like environment and assign them random values.

(2) Evaluate the population.

(3) If the stopping criterion has been met, go to step 9; otherwise, continue.

(4) Perform neighborhood competition operator on the agents.

(5) Apply the neighborhood crossover on the agents.

(6) Apply self-learning operator on the agents.

(7) Ensure elitism by replacing the worst current individual with the best individual agent found so far.

(8) Repeat step 2.

(9) Stop.

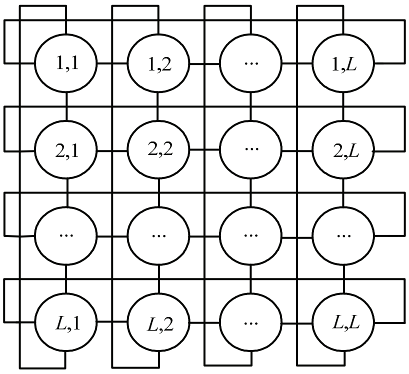


Fig. 2 Multi-agent grid model.

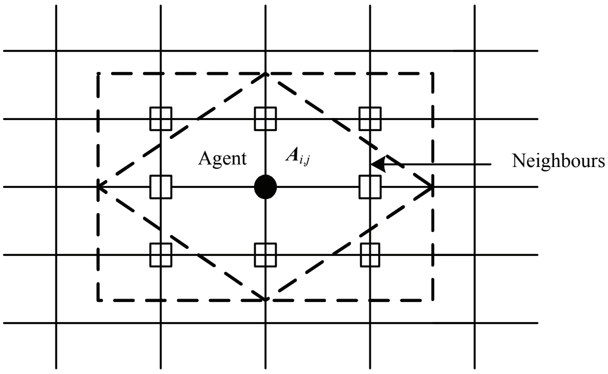


Fig. 3 Agent and its neighbors.

However, even though MAGA can largely deal with high-dimensional multimodal optimization problems, it has limitations. It is based on binary coding evolutionary algorithm, and this will, to some extent, affect the search range and search accuracy in high-dimensional system. During the search process, the agents' scale is always fixed (usually 5×5). Each agent improves its energy only through competition, cooperation, and self-learning. This will partly affect MAGA's convergence speed. Unlike MAGA, the Ma-aiNet, with real number coding, combines the characteristics of MAGA and Opt-aiNet immune network algorithm. Thus, it can automatically adjust agent size during the search process. Instead of inserting random points in the population blindly, Ma-aiNet adopts MAGA's competition, cooperation, and self-learning as global exploration operators. Moreover, it can greatly reduce objective function evaluations, and is more likely to improve search accuracy and speed in high-dimensional space.

2.3 Description of Ma-aiNet

The procedure of Ma-aiNet is described as follows:

(1) Initialization operation

Initial N antibodies are generated randomly in the solution domain, which constitute the initial population A_0. The initial parameters P_m, P_c are generated, denoting the mutation and crossover rates, respectively, from a normal distribution N(0, 1). The maximal number of generations G_m is determined. The current generation is set to $T = 0$.

(2) Clonal reproduction operation

For each antibody $a_i \in \boldsymbol{A}$ (i = 1, 2, … , N) of the population, clone operation duplicates it to the new population $\boldsymbol{A}'$ with the scale of N_s, which is given by

$$N_s = Int\left\lceil N_c \cdot \frac{f(a_i)}{\sum_{i=1}^{N} f(a_i)} \right\rceil \quad s = 1, 2, \cdots N, \tag{2}$$

where N_c is a given integer relating to the clonal scale, which is set to $10N$ here, $Int\lceil x \rceil = min\{t > x | t \in \boldsymbol{R}\}$.

As shown in Eq. (2), the clonal scale of an antibody is related to its affinity. The antibody with a greater affinity has a larger clonal scale.

(3) Mutation operation

To keep the information of parent population, the mutation operation can only be applied to the population after clonal operation. The mutation operation is applied to each unit of every antibody according to the mutation probability P_m. The mutated antibody collection is called $\boldsymbol{M}_T$.

(4) Immune suppression operation

Similar antibodies are removed from $\boldsymbol{M}_T$, whose $s_{i,k} < \sigma_s, s_{i,k} = \left\| M_{n,i} - M_{n,k} \right\|, \forall i, k$; σ_s is the suppressive threshold. After the immune suppression operation, the antibody collection remains $\boldsymbol{R}_T$, and we get a new generation of antibody group $\boldsymbol{A}_T$.

(5) Construction of multi-agent grid

All antibodies $\boldsymbol{A}_T$ are sorted logically. In the multi-agent grid, we assume the given agent to be $\boldsymbol{A}_{m,n} = (a_1, a_2, \ldots , a_q)$. The initialized immune agent grid system is $L^0(T)$. Each antibody is placed at a certain point of the multi-agent grid. To be precise, $\boldsymbol{A}_{1,1}$, … , $\boldsymbol{A}_{1,L}$, $\boldsymbol{A}_{2,1}$, … , $\boldsymbol{A}_{2,L}$, … , $\boldsymbol{A}_{L,1}$, … , $\boldsymbol{A}_{L,L}$ are placed at (1,1), … , (1,L), (2,1), … , (2,L), … , (L,1), … , (L,L), as illustrated in Fig. 2. An agent has energy, which is defined in Eq. (3). The goal of the agents' evolution is to increase their energy.

$$Energy(\boldsymbol{A}_{m,n}) = 1 / f(\boldsymbol{A}_{m,n}), \tag{3}$$

where $f(\,)$ is the objective function.

(6) Neighbor-competition operation

For $\forall \boldsymbol{A}_{m,n}, \boldsymbol{A}_{m,n} \in \boldsymbol{L}^0(T)$, we define its neighbors $Loc.\boldsymbol{A}_{m,n} = \left\{ \boldsymbol{A}_{m-,n}, \boldsymbol{A}_{m,n-}, \boldsymbol{A}_{m+,n}, \boldsymbol{A}_{m,n+} \right\}$, where

$$m- = \begin{cases} m-1, m \neq 1 \\ L_S, m = 1 \end{cases}, \quad n- = \begin{cases} n-1, n \neq 1 \\ L_S, n = 1 \end{cases},$$

$$m+ = \begin{cases} m+1, m \neq L_S \\ 1, m = L_S \end{cases}, \quad n+ = \begin{cases} n+1, n \neq L_S \\ 1, n = L_S \end{cases},$$

If $\forall \boldsymbol{A}_{m',n'}, \boldsymbol{A}_{m',n'} \in Loc.\boldsymbol{A}_{mn}$, $\boldsymbol{A}_{m',n'} = (a_1, a_2, ..., a_q)$ has more energy than $A_{m,n}$, it can survive in the grid; otherwise, it will die and leave space for new individuals $\boldsymbol{\beta}_{m,n} = (b_1, b_2, \ldots, b_q)$. The neighborhood competition operation can be described as:

If $Energy(\boldsymbol{A}_{m',n'}) \geq Energy(\boldsymbol{A}_{m,n})$,
then $\boldsymbol{A}_{m,n} \leftarrow \boldsymbol{A}_{m',n'}$; or $\boldsymbol{A}_{m,n} \leftarrow \boldsymbol{\beta}_{m,n}$.

$$b_k = \begin{cases} \alpha_{k-}, & a_k + r(-1,1)\cdot(a_k - \alpha_{k-}) < \alpha_{k-} \\ \alpha_k^{-}, & a_k + r(-1,1)\cdot(a_k - \alpha_{k-}) > \alpha_k^{-} \\ a_k + r(-1,1)\cdot(a_k - \alpha_{k-}), & \text{else} \end{cases},$$

$$k = 1, 2, \ldots, q, \tag{4}$$

where $r(-1,1)$ is a random number; and α_{k-} and α_k^{-} correspond to the lower and upper limits, respectively, of the solutions. After the Neighborhood Competition Operation, the antibody network is $\boldsymbol{L}^1(T)$.

(7) Agent crossover operation

Assume $\boldsymbol{A}_{mn,\max}$ has the maximum energy among its neighbors in the current generation according to the probability P_c. The agent crossover operation is applied on $\boldsymbol{A}_{m,n}$ and $\boldsymbol{A}_{mn,\max}$ to realize the collaboration among different agents. After this operation, the antibody network is $\boldsymbol{L}^2(T)$.

(8) Agent self-study operation

The agents can self-study to improve their competition and adaptation ability. After the Crossover Operation, the agent is picked out with the highest energy within the whole agent grid $\boldsymbol{L}^2(T)$, i.e., ***Best***, where ***Best*** $= (a_1^*, a_2^*, \ldots, a_q^*)$. According to Eq. (5), we generate a new multi-agent grid $s\boldsymbol{L}^2(T)$ with a smaller scale $sL\times sL$, $sA_{i,j} \in s\boldsymbol{L}^2(T)$, $i, j = 1, 2, \ldots, sL$, $sA_{i,j}$ is defined by

$$sA_{i,j} = \begin{cases} \boldsymbol{A}_{i,j}, & i=1, j=1 \\ \boldsymbol{New}_{i,j}, & \text{else} \end{cases}, \tag{5}$$

where $\boldsymbol{New}_{ij}$ is generated by mutation operation on a random single gene unit of ***Best*** according to the probability P_m. Then, the best agent s***Best*** on $s\boldsymbol{L}^2(T)$ will be generated after applying neighbor-competition operation step (6) and agent crossover operation step (7) repeatedly.

If *Energy* (***sBest***)> *Energy* (***Best***), then ***sBest*** replaces ***Best***.

Ma-aiNet stops if *Energy* (***Best***) does not change any more or the number of iterations reaches the preset value. Otherwise, $T = T+1$, and step (2) is repeated.

3 Benchmark studies

3.1 Benchmark functions

To evaluate the performance of Ma-aiNet, 10 high-dimensional benchmark functions[21] are used in our experimental studies. The benchmark functions and their optimal solutions are given in Table 1. Functions f_1–f_7 are multimodal functions where the number of local optima increases exponentially with the problem dimension. Functions f_8–f_{10} are unimodal functions. The initialization range and dimensions of these 10 benchmarks are also listed in Table 1.

In this section, the performance of Ma-aiNet is studied using the 10 benchmark problems, and the obtained results are compared with those of the OGA/Q algorithm, as proposed by Leung and Wang[21], the MAGA presented by Zhong *et al.*[17], and the Opt-aiNet algorithm[9]. OGA/Q[21] is an advanced genetic algorithm with excellent optimization on high-dimensional functions. It uses orthogonal design to generate original population and crossover offspring.

3.2 Experimental results

The parameters selected for each algorithm should achieve the best search results in terms of search speed and accuracy. In all the experiments, the population sizes for OGA/Q and MAGA are 100 and 25, respectively, while those for Opt-aiNet and Ma-aiNet are 20, respectively. The settings of other parameters for these algorithms are presented in Table 2.

In the tests using the above benchmark functions, the stopping criterion for OGA/Q is that no elite solution is found in the latest 50 generations after the implementation of 1000 generations, as described in Ref. [21]. MAGA stops after running 150 generations[17]. To maintain the consistency of parameters and make the comparison meaningful, the stopping criterion for Ma-aiNet is 150 generations, while Opt-aiNet has the same stopping criterion as OGA/Q.

Table 3 lists the statistical results of 50 independent runs using OGA/Q[21], MAGA[17], Opt-aiNet[9], and Ma-aiNet algorithms. The results of OGA/Q and MAGA runs are taken from Refs. [21] and [17], respectively. All test functions can be categorized into three classes by carefully examining the simulation results in Table 3:

Table 1 Parameter settings for the benchmark functions

Test functions	*x* domain	Dimension	Optimum
Schwefelha function: $f_1(x)=\sum_{i=1}^{n}(-x_i\sin\sqrt{\lvert x_i\rvert})$	[−500,500]	30	−12569.5
Rastrigin function: $f_2(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	[−5.12,5.12]	30	0
Ackley function: $f_3(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))+20+e$	[−32,32]	30	0
Griewank function: $f_4(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(x_i/\sqrt{i})+1$	[−600,600]	30	0
Penalty function1: $f_5(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, x_i>a\\ 0,-a\le x_i\le a; y_i=1+\frac{1}{4}(x_i+1)\\ k(-x_i-a)^m, x_i<-a\end{cases}$	[−50,50]	30	0
Penalty function2: $f_6(x)=\frac{1}{10}\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	[−50,50]	30	0
Global function: $f_7(x)=\sum_{i=1}^{n}x_i^2$	[−100,100]	30	0
Schwefell function 1: $f_8(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	[−10,10]	30	0
Schwefell function 2: $f_9(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	[−100,100]	30	0
Schwefell function 3: $f_{10}(x)=\max\{\lvert x_i\rvert,1\le i\le n\}$	[−100,100]	30	0

Table 2 Parameter settings

	OGA/Q	MAGA	Opt-aiNet	Ma-aiNet
P_c	0.60	0.10	−	0.10
P_m	0.01	0.10	0.01	0.10
S_c	−	−	4	−
		Ls=5,sLs=3	f_r=0.5,σ_s=0.5	σ_s=0.5 Ls=5,sLs=3

P_c: crossover rate; P_m: mutation rate; S_c: clone scale; f_r: feedback ratio; σ_s: suppression threshold.

Table 3 Performance comparison of four algorithm

Test function	Mean function value (standard deviation)				Mean number of function evaluations			
	OGA/Q	MAGA	Opt-aiNet	Ma-aiNet	OGA/Q	MAGA	Opt-aiNet	Ma-aiNet
f_1	−12569.4537 (6.447×10^{-4})	−12569.4866 (7.121×10^{-12})	−12569.4932 (5.327×10^{-12})	−12569.4963 (4.872×10^{-12})	302 166	10 866	286 166	11 257
f_2	0 (0)	0 (0)	5.584×10^{-11} (3.482×10^{-9})	0 (0)	224 710	11 427	209 427	13 307
f_3	4.440×10^{-16} (3.989×10^{-17})	4.440×10^{-16} (0)	7.694×10^{-16} (3.851×10^{-9})	0 (0)	112 421	9 656	107 556	10 016
f_4	0 (0)	0 (0)	6.857×10^{-17} (4.752×10^{-10})	0 (0)	134 000	9 777	127 420	10 372
f_5	6.019×10^{-6} (1.159×10^{-6})	1.142×10^{-18} (4.390×10^{-18})	8.125×10^{-16} (8.891×10^{-9})	1.124×10^{-18} (3.994×10^{-18})	134 556	10 545	128 286	11 312
f_6	1.869×10^{-4} (2.615×10^{-5})	1.142×10^{-18} (4.196×10^{-17})	1.793×10^{-10} (7.991×10^{-7})	8.104×10^{-19} (3.108×10^{-17})	134 143	11 269	129 286	12 479
f_7	0 (0)	0 (0)	0 (0)	0 (0)	112 559	9 502	108 689	10 319
f_8	0 (0)	0 (0)	0 (0)	0 (0)	112 612	9 591	108 662	10 395
f_9	0 (0)	0 (0)	0 (0)	0 (0)	112 576	9 479	107 948	10 378
f_{10}	0 (0)	0 (0)	0 (0)	0 (0)	112 893	9 603	109 996	10 493

(1) Functions f_8–f_{10} are unimodal functions; all four algorithms get the best results.

(2) Functions f_2, f_4, and f_7 are multimodal functions. The global optimal solution exists in the orthogonal array-based initial populations generated using OGA/Q method[22]. Therefore, OGA/Q can obtain the optimal solution with zero standard deviation. Meanwhile, the solution quality of Opt-aiNet is lower than that of OGA/Q, MAGA, and Ma-aiNet.

(3) Functions f_1, f_3, f_5 and f_6 are multimodal functions. The global optimal solution does not exist in the orthogonal array-based initial populations generated using the OGA/Q method. Hence, OGA/Q derives poor results. Meanwhile, Ma-aiNet performs much better than OGA/Q, Opt-aiNet, and MAGA.

However, from the point of view of function evaluations, the average number of evaluations used in Ma-aiNet is about 11,000, similar to that used in MAGA, while Opt-aiNet uses 15,000 to 20,000, and OGA/Q uses 20,000 to 30,000. These results suggest that Ma-aiNet works better in high-dimensional function optimization than OGA/Q, MAGA, and Opt-aiNet.

3.3 Population diversity analysis

Population diversity is an important evaluation parameter for evolution algorithms. The study of Ref. [23] introduced Phenotypical Diversity (*PDM*) and Genotypical Diversity (*GDM*) evaluation parameters of the population in the search process. *PDM* and *GDM* are defined as[23]:

$$PDM = aff_{\text{avg}} / aff_{\max}, \tag{6}$$

$$GDM = (\overline{E} - E_{\min}) / (E_{\max} - E_{\min}), \tag{7}$$

where aff_{avg} and $aff_{\max}$ represent the average and maximum fitness values of the population in the current generation, respectively; $\overline{E}$ is the average Euclidean distance between all individuals and the best individuals of the population in the current generation; $E_{\max}$ and $E_{\min}$ represent the maximum and minimum Euclidean distances of the population, respectively; and *PDM* and *GDM* both belong to the interval [0,1]. If *PDM* is near 1, then convergence has been reached; when all the chromosomes are identical, *GDM* is zero. Usually, if $PDM \geq 0.9$ and $GDM < 0.1$, the algorithm tends to converge, and the gene types of the antibody population are nearly homogenous. Similarly, if $0 < PDM \leq 0.9$ and $GDM \geq 0.1$, then the algorithm is in the normal search process.

To demonstrate the results of the combination of multi-agent genetic and immune network algorithms, *PDM* and *GDM* tests of the population for function $f_2(x)$ are shown in Fig. 4. In these tests, the four algorithms iterated for 500 generations. Fig. 4 clearly shows that, compared with the other three algorithms, the *PDM* curve of Ma-aiNet is more stable. This indicates that Ma-aiNet can better keep gene segments in the search process and converge to the optimal point in a stable manner. Furthermore, note that the *GDM* curve of Ma-aiNet fluctuates considerably during the search process, but is always above those of the other three algorithms. This verifies that multi-agent neighbor-competition and agent crossover can recruit new antibodies into the gene base to keep high population diversity during the run.

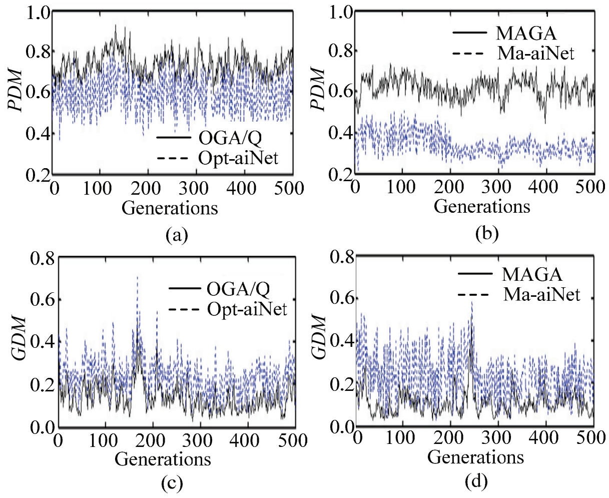


Fig. 4 *PDM* and *GDM* curves of the algorithms during the run. (a) *PDM* curves of OGA/Q and Opt-aiNet; (b) *PDM* curves of MAGA and Ma-aiNet; (c) *GDM* curves of OGA/Q and Opt-aiNet; (d) *GDM* curves of MAGA and Ma-aiNet.

4 A case study

In petrochemical industry, process simulation for distillation procedures is an important work. The key parameter in distillation process simulation is tray efficiency, which cannot be obtained easily. There are some complex factors affecting tray efficiency, such as plate structure, manipulation conditions, and the varying nature of the physical systems in use, which can lead to deviations from theoretical tray models. The key factor in modeling industrial distillation unit is the determination of tray efficiency, which is a high-dimensional optimization problem. In this section, an optimization model to determine the tray efficiency of an industrial distillation unit is established first. Based on this, four algorithms (i.e., OGA/Q, Opt-aiNet, MAGA, and Ma-aiNet) are used to make an omnidirectional search for tray efficiency. By comparing the calculated tray temperature distribution, the tower's top and bottom output distributions with the measured ones, the effectiveness of Ma-aiNet is verified.

Murphree efficiency is one of the expression of tray efficiency which is popularly used. Here is the definition of Murphree efficiency:

$$Eff_{i,j}^{M} = \frac{Y_{i,j} - Y_{i,j+1}}{K_{x,y}X_{i,j} - Y_{i,j+1}}, \tag{8}$$

where Eff^M is the Murphree efficiency, K is the equilibrium constant, X is the liquid phase mole fraction, Y is the vapor phase mole fraction, i and j represent the ith component and the jth tray, respectively.

The actual temperature distribution of the trays can indirectly characterize the separation effect of the distillation columns. Hence, a model where the objective function[24] is the minimum of F is proposed as

$$F = min\sum_{j}(\frac{T_j - \hat{T}_j}{T_j})^2, \tag{9}$$

where j is the tray number with a temperature measuring point, T is the measured tray temperature (°C), and $\hat{T}$ is the calculated tray temperature (°C). Typically, the distillation unit has more than 50 plates. The global optimal solution for high-dimensional and highly nonlinear problems can be searched by adjusting the tray efficiency to get the minimum F of Eq. (9). The flowchart of the Ma-aiNet algorithm to determine tray efficiency is shown in Fig. 5.

After collecting 100 groups of typical steady-state operation data as sample data, the distillation unit model with 60 trays is corrected, with the tray temperature test points at 10. We also use Eq. (9) as the objective function, and determine the Murphree efficiency using OGA/Q, Opt-aiNet, MAGA, and Ma-aiNet algorithms. To make the comparison meaningful, the function evaluations of the four algorithms is fixed to 10,000. Fig. 6 shows the fitting results of the four algorithms for the tray temperature distribution. In OGA/Q and Opt-aiNet, the average absolute deviation between the calculated and measured temperatures is 1.41 °C and 1.03 °C, respectively, and the average relative deviation is 0.73 and 0.61 for each. In MAGA, the average absolute deviation is 0.82 °C and the relative average deviation is 0.35. Meanwhile, Ma-aiNet has an average absolute deviation of 0.71 °C, with an average relative deviation of only 0.29. When the tray's Murphree efficiency is determined using Ma-aiNet, the calculated temperatures are

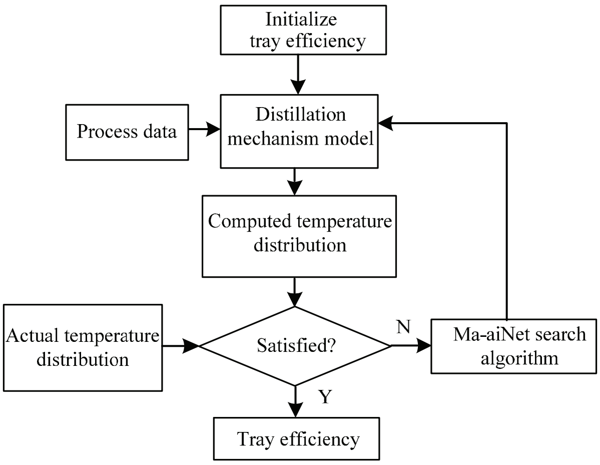
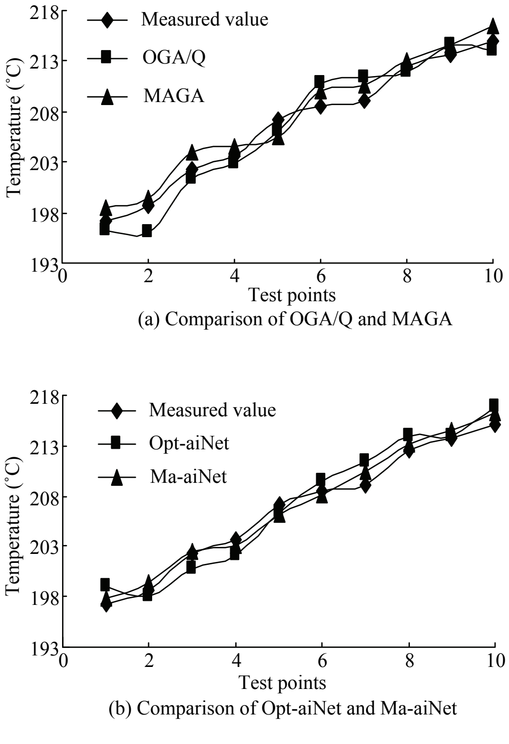

Initialize tray efficiency
Process data
Distillation mechanism model
Computed temperature distribution
Actual temperature distribution
Satisfied?
N
Ma-aiNet search algorithm
Y
Tray efficiency

Fig. 5 The flowchart of Ma-aiNet for the determination of tray efficiency.



(a) Comparison of OGA/Q and MAGA

(b) Comparison of Opt-aiNet and Ma-aiNet

Fig. 6 The comparison between the calculated tray temperature and the measured value.

consistent with the measured temperatures. This suggests that the model describes the separation effect of the column very well. On this basis, we establish the distillation unit model and predict the amount of production from the tower's top and bottom. Table 4 displays the calculated and measured values of the main products under the steady-state operation. Average absolute errors in the evaluation of the main products from the tower's top and bottom are 2.841 and 3.318 kmol/hr for OGA/Q, 2.221 and 1.882 kmol/hr for MAGA, 2.671 and 2.682 kmol/hr for Opt-aiNet, and 1.570 and 1.685 kmol/hr for Ma-aiNet, respectively. At the same time, average relative errors are about 0.96% for OGA/Q, 0.62% for MAGA, 0.69% for Opt-aiNet, and 0.47% for Ma-aiNet. Ma-aiNet clearly outperformed the other three algorithms significantly.

Table 4 The comparison between the calculated and measured values for the products (kmol/hr)

Distillation unit		Measured value	OGA/Q	MAGA	Opt-aiNet	MA-aiNet
Tower top products	C7H8	25.7	22.4	23.4	22.7	23.9
	C8H10-3	611.1	613.2	613.5	611.3	611.8
	C8H10-2	1393.8	1391.3	1395.4	1391.3	1395.3
	C8H10-1	527.9	529.7	529.3	529.7	529.2
	C8H10-4	237.5	236.2	239.9	235.5	238.7
	C8H16-1	190.4	187.8	191.6	193.1	192.2
Tower Bottom products	C9H12-5	48.6	45.3	46.5	45.7	47.3
	C9H12-7	56.3	59.5	58.0	59.2	58.7
	C9H12-8	21.2	18.2	21.9	19.9	22.7
	C9H12-3	15.1	14.3	12.4	14.5	13.4
	C10H14-7	3.2	5.1	2.9	4.9	3.0

5 Conclusions

In this paper, we described a new multi-agent algorithm, Ma-aiNet, based on the immune network strategy. In a multi-agent system, competition and cooperation between neighboring agents exist, and the agents have self-study ability. We combined these characteristics with the powerful global searching ability of an immune network algorithm to solve high-dimensional optimization problems. The operations performed by the Ma-aiNet algorithm include clonal reproduction, mutation, immune suppression, neighborhood competition, agent crossover, and agent self-study. In benchmark function experiments, Ma-aiNet was tested on 10 benchmark functions with 30 dimensions, and the results were compared with those of MAGA, Opt-aiNet, and OGA/Q algorithms. The experiments indicated that Ma-aiNet outperforms the other three algorithms in terms of search accuracy. The computational cost of Ma-aiNet is obviously less than that of OGA/Q and Opt-aiNet, and is close to that of MAGA. Both *PDM* and *GDM* population diversity tests show that Ma-aiNet can keep good gene segments in the search process and converge at the optimal point in a stable manner. Moreover, Multi-Agent neighbor-competition and agent crossover introduced by Ma-aiNet can recruit new antibodies into the gene base to keep high population diversity during the run. Thus, the algorithm was subsequently used to estimate the Murphree efficiency of the multi-component distillation column. The results marked a noticeable improvement over previously reported solutions.

Acknowledgments

This work was supported by the National Science Fund for Distinguished Young Scholars (No.60625302), National Natural Science Foundation of China (2009CB320603), Shanghai Key Technologies R&D Program(10JC1403500), Changjiang Scholars and Innovative Research Team in University(IRT0721), the 111 Project(B08021), Shanghai Leading Academic Discipline Project(B504) and Zhejiang Natural Science Fund (Y1090548).

References

[1] Mahfoud S W. Genetic drift in sharing methods. *Proceedings of the* 1*st IEEE Conference on Evolutionary Computation*, Piscataway, USA, 1994.

[2] Yu S Y, Guo G Q. Genetic drift analysis of selection. *Journal of Computer Research and Development*, 2004, **41**, 346–351. (in Chinese)

[3] Goldberg D E, Richardson J. Genetic algorithms with sharing for multimodal function optimization. *Proceedings of the* 2*nd International Conference on Genetic Algorithms*, Hillsdale, USA, 1987, 41–49.

[4] Mahfoud S W. Crowding and preselection revisited. In Manner R and Manderick B (eds), *Parallel Problem Solving from Nature*, Elsevier, Amsterdam, Netherland, 1992, 27–36.

[5] Li J P, Balazs M E, Parks G T. A species conserving genetic algorithm for multimodal function optimization. *Journal of Evolutionary Computation*, 2002, **10**, 207–234.

[6] de Castro L N, Von Zuben F J. Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 2002, **6**, 239–251.

[7] Yıldız A R. A novel hybrid immune algorithm for global

optimization in design and manufacturing. *Robotics and Computer-Integrated Manufacturing*, 2009, **25**, 261–270.

[8] de Castro L N, Von Zuben F J. aiNet: An artificial immune network for data analysis. In Abbas H, Sarker R, Newton C (eds), *Data Mining*: *A Heuristic Approach*, Idea Publishing Group, Hershey, USA, 2001, l–37.

[9] de Castro L N, Timmis J. An artificial immune network for multimodal function optimization. *Proceedings of IEEE Congress on Evolutionary Computation*, Hawaii, USA, 2002, 699–704.

[10] Gomes L de C T, de Sousa J S, Bezerra G B, de Castro L N, Von Zuben F J. Copt-aiNet and the Gene Ordering Problem. *Proceedings of WOB*, Rio de Janeiro, Brazil, 2003, 28–37.

[11] de Franca F O,Von Zuben F J, de Castro L N. An artificial immune network for multimodal function optimization on dynamic environments. *Proceedings of the* 2005 *Conference on Genetic and Evolutionary Computation*, Washington, USA, 2005, 289–296.

[12] Singh M P. Multi-Agent System: A Theoretical Framework for Intention, Know-How and Communication, Springer-Verlag, Berlin, Germany, 1994.

[13] Ishida T. Real-time search for autonomous agents and multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 1998, **1**, 139–167.

[14] Tsui K C, Liu J M. Multiagent diffusion and distributed optimization. *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, Melbourne, Australia, 2003, 169–176.

[15] Persson J A, Davidsson P. Combining agent-based approaches and classical optimization techniques. *Proceedings of the Third European Workshop on Multi-Agent Systems*, Brussels, Belgium, 2005, 260–269.

[16] Davidsson P, Persson J A, Holmgren J. On the Integration of agent-based and mathematical optimization techniques. *Lecture Notes in Computer Science*, 2007, **4496**, 1–10.

[17] Zhong W C, Liu J, Xue M Z, Jiao L C. A multi-agent genetic algorithm for global numerical optimization. *IEEE Transactions on Systems*, *Man and Cybernetics* (*Part B*), 2004, **34**, 1128–1141.

[18] Liu J, Zhong W C, Jiao L C. A multiagent evolutionary algorithm for constraint satisfaction problems. *IEEE Transactions on System*, *Man*, *and Cybernetics* (*Part B*), 2006, **36**, 54–73.

[19] Ichimura T, Oeda S, Suka M, Yoshida K. A learning method of immune multi-agent neural networks. *Neural Computing & Applications*, 2005, **14**, 132–148.

[20] Liu F, Liu Y, Chen X, Wang J S. Multi-agent immune clonal selection algorithm based multicast routing. *Lecture Notes in Computer Science*, 2006, **4222**, 319–327.

[21] Leung Y W, Wang Y P. An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 2001, **5**, 41–53.

[22] Ho S Y, Shu L S, Chen J H. Intelligent evolutionary algorithms for large parameter optimization problems. *IEEE Transactions on Evolutionary Computation*, 2004, **8**, 522–540.

[23] Herrera F, Lozano M. Adaptation of genetic algorithm parameters based on fuzzy logic controllers. In Herrera F, Verdegay J L (eds), *Genetic Algorithms and Soft Computing*, Physica-Verlag, Heidelberg, Germany, 1996, 95–125.

[24] Yan X F, Yu J, Li F, Qian F. Simulation of atmospheric-vacuum distillation unit based on evolution algorithm. *Acta petrolei sinica* (*Petroleum Processing Section*), 2006, **22**, 41–48. (in Chinese)