Information Sciences 276 (2014) 219–241

Contents lists available at ScienceDirect

Information Sciences

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

An evolutionary membrane algorithm for global numerical optimization problems

Min Han [a,*], Chuang Liu [a], Jun Xing [b]

[a] Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian, Liaoning 116024, China
[b] School of the Information Science and Engineering, Dalian Ploytechnic University, Dalian, Liaoning 116034, China

ARTICLE INFO

Article history:
Received 20 May 2012
Received in revised form 31 October 2013
Accepted 9 February 2014
Available online 20 February 2014

Keywords:
Evolutionary membrane algorithm
Membrane system
Cellular automata
Molecular computing
Global numeric optimization

ABSTRACT

Nature-inspired algorithms for optimization are significant topics in the areas of computational intelligence. The contribution of this paper is to present a new heuristic intelligent evolutionary algorithm based on membrane systems to solve the global numerical optimization problems. The proposed algorithm employs the fundamental ingredients of membrane systems, including multisets, reaction rules and membrane structure. In addition, the proposed algorithm incorporates information of the adjacent symbol-objects, to guide the evolution toward the global optimum, efficiently. More specifically, symbol-objects are evolved by the cellular automata model which invokes the rewrite rules to exchange the information of the adjacent symbol-objects. Moreover, sharing information in the skin membrane is implemented, which accelerates the speed of the proposed algorithm to find the global optimal solution. In the extensive experimental study, the effectiveness of the proposed algorithm is demonstrated with the benchmark global numeric optimization problems. The experimental results indicate that the proposed method is a competitive optimizer in comparison with the four state-of-the-art evolutionary algorithms.

1. Introduction

With the rapid development of scientific research and engineering practice, more and more problems can be abstracted as the global numeric optimization problems, which have obtained enormous attention [11]. These problems have the following characteristics, e.g. multi-mode, non-continuous, noisy solution spaces, etc. It is very difficult to use the deterministic mathematical tools such as a gradient method to solve these problems. However, many evolutionary algorithms inspired by natural computing may provide some effective solutions for such complex global numeric optimization problems. Therefore, many researchers all over the world in the evolutionary community have designed various versions of bio-inspired evolutionary approaches to solve such optimization problems [19,41,67,36,9,13,5]. To the best of our knowledge, various kinds of renowned meta-heuristic algorithms have been introduced, such as Genetic Algorithm (GA) [22,12,35,59,1], Artificial Immune Algorithms [15,3,52], Particle Swarm Optimization (PSO) [29,19,31,14,58], Differential Evolution (DE) [53,49,67,6,24,21], Memetic Algorithms [38,56,55], and Human Evolutionary Model [37]. Although these evolutionary algorithms can solve many kinds of optimization problems, their performances are still unsatisfactory in two aspects of both the global search ability and the convergence speed.

* Corresponding author.
E-mail address: minhan@dlut.edu.cn (M. Han).

http://dx.doi.org/10.1016/j.ins.2014.02.057
0020-0255/© 2014 Elsevier Inc. All rights reserved.

As a new branch of natural computing, membrane computing is an abstracted distribute and parallel computing model [44,45,47,48]. Inspired by the structure and functioning of a biological living cell, membrane computing is first proposed by Păun from the European Academy of Sciences in 1998. The computing model investigated in the biological cell membrane is called membrane system (also known as P system) [47]. Membrane systems mainly focuses on the various computational features of the membranes such as transferring chemical substances between membranes or chemical reaction in the region of the membrane, instead of modeling the biological membranes. Strictly speaking, a number of principles are abstracted underlying the functioning of biological membranes, and this abstraction is used as the working mechanism of the computing model. Based on the above-mentioned context, an integral membrane system includes the nested membrane structure, multisets, and reaction rules. Multiset, which consists of a collection of symbol-objects, is placed in the compartments defined by the membrane structure, and it is evolved by executing the reaction rules in a non-deterministic and maximally parallel manner. The membranes except the skin membrane can be dissolved and divided by invoking the corresponding reaction rules. Because the structure of membrane systems provides an enhanced parallelism, membrane systems can solve intractable problems in a polynomial time. In addition, membrane systems have the same computing power with a Turing universal computing model [42,43]. Many studies on the applications of membrane systems have been reported so far in the fields of computer science, mathematics, biology, artificial intelligence, ecology, abstract chemistry, and even linguistics, etc. [48,46,7].

Recently, the research on the applications of membrane systems have received increasing interest in the aspect of optimization, because membrane system can solve many complex optimization problems in polynomial time. In 2004, Nishida [40] proposed a novel membrane algorithm to solve the traveling salesman problem. Huang et al. [26] presented an optimization algorithm inspired by membrane systems to solve the optimization problems with large feasible spaces and many parameters. Zhao and Wang [68] proposed a bio-inspired algorithm based on membrane computing (BIAMC) for solving both unconstrained and constrained optimization problems. Zhang et al. [65] proposed a hybrid algorithm based on the quantum-inspired evolutionary approach and membrane systems to solve a well-known combinatorial optimization for knapsack problem. Zhang et al. [63] proposed a membrane-inspired approximate algorithm for traveling salesman problems. In [25], Huang et al. presented a dynamic multi-objective optimization algorithm, which was inspired by membrane systems. Simulation results verified the effectiveness of the algorithm. Liu et al. [32] presented a novel algorithm based on the membrane systems theory for solving multi-objective optimization problems. Zhang et al. [66] proposed a multi-objective membrane algorithm for knapsack problems. Liu et al. [33] proposed an evolutionary membrane algorithm for solving global numeric optimization problems. In [64], a hybrid approach which combined Differential Evolution algorithms and Tissue P Systems appropriately was utilized for solving a class of constrained manufacturing parameter optimization problems.

From the above-mentioned research results, we found that the applications of membrane systems in optimization are very few, particular in the global numeric optimization. Therefore, the evolutionary algorithms inspired by membrane systems are worthy of further study. The contribution of the present paper is to propose a novel global efficient evolutionary algorithm based on the membrane system, named as evolutionary membrane algorithm (EMA), to solve the global numerical optimization problems. In EMA, a symbol-object represents a candidate solution of the optimization problem at the intersection of all search dimensions, and then a multiset, which consists of several symbol-objects, represents a candidate solution set. Inspired by the basic features and structure of the biological membrane, some mechanisms of EMA simulates the behavior of the molecules (multisets) in the region of the membrane. EMA not only incorporates some reaction rules from membrane systems, but also makes judiciously use of the adjacent knowledge of symbol-objects. EMA is capable of achieving a number of optimal solutions in a finite period with the limited computational resources. It can improve the quality of the candidate solutions, particularly in balancing between exploration and exploitation. Moreover, EMA differs from the existing state-of-the-art evolutionary algorithms in the following aspects:

First, EMA incorporates some concepts and working mechanisms of membrane systems, particularly in the structure of membrane systems. The membrane structure, which is different from the existing state-of-the-art evolutionary algorithms, is easy to implement for the distributed and parallel computing. It can help to accelerate the execution efficiency of EMA. In the simulations, EMA is implemented using the serial mode instead of the parallel one as the experimental conditions are limited.

Second, the key point for evolutionary algorithms solving the optimization problems is how to balance exploration and exploitation during the whole search process [57], that is to say, the solving performance of the evolutionary algorithms depends on the two aspects of the global search ability and the diversity. The solving performance of evolutionary algorithms could be enhanced by properly balancing these two characteristics. In the proposed EMA, some reaction rules are employed to improve the balance relationship. These rules are the abstract representations of both the chemical reactions and evolving reactions inspired by the biological changing processes in the biological cell. More specifically, the reaction rules include the rewrite rules, division rule, dissolve rule, and communication rule. They may help EMA to maintain the diversity of candidate solutions so that EMA can escape from the local minima.

Third, the chemical substances in the biological cell is randomly swimming, and it is very difficult to simulate their swimming process. The cellular automata model can simulate many complex systems in any environment [17]. The random swimming process for evolutionary algorithms is beneficial to traverse the solution space. Therefore, the cellular automata model is incorporated into the proposed algorithm in order to implement the swimming process. The cellular automata model can perform adjacent knowledge acquisition during the search process. The information of the adjacent symbol-objects is useful to improve the ability of exploitation.

The remaining sections of this paper will be organized as follows. Section 2 gives a short literature review about evolutionary algorithms inspired by natural computing. In Section 3, the concept and the mechanism of membrane systems are briefly described. In Section 4, the proposed EMA is outlined and its implementation steps are explained in details. In Section 5, the proposed algorithm are evaluated on the different benchmark test problems. In addition, it is compared with four state-of-the-art evolutionary algorithms. Section 5 also includes a sensitivity statistical analysis of the proposed EMA. Finally, Section 6 summarizes the concluding remarks and future work of this paper.

2. Related work

The natural computing models are abstracted and simulated by modeling various biological phenomena and working mechanisms from the body or system of nature life [10]. Integrated the knowledge of biology, computer and mathematics, some nature-inspired models are built to solve some complex problems. Due to their powerful solving capability, they have been widely used in many areas, especially in the area of optimization.

To the best of our knowledge, many scholars have constructed various interesting evolutionary algorithms inspired by the distinctive biological processes and phenomena. As seen in Fig. 1, they are abstracted and simulated on different levels of life, such as from biological molecules to cells, then to immune system, and finally to population. Inspired by the mechanisms of the biological models in Fig. 1, the corresponding natural computing model can be abstracted, such as Swarm Intelligent, Artificial Immune and Membrane Computing. They can be employed to design evolutionary algorithms for solving the global numeric continuous optimization problems. The short description of the models will be discussed as follows.

Swarm Intelligent (SI) is an important branch of natural computing, and it is the most active research field on evolutionary algorithms in recent years. SI borrows the idea of the collective behavior of animal population [4]. Through observation and study of the biological population, various models have been proposed to solve the optimization problems. Some of them can be simply described as follows. Particle Swarm Optimization (PSO), which is inspired by the feeding process of birds, is guided to move toward the possible direction through the transmission of information among particles [29,60,50]. Artificial Bee Colony (ABC) borrows the idea of the intelligent behavior of honey bees where each honey bee makes the different activities according to its division of labor [28,2,62]. Human Evolutionary Model (HEM) inspired by the activities of human is proposed as a novel computational method to solve the optimization problems [37].

Artificial immune system (AIS) is proposed which inspired by mechanisms, characteristics and principles of the immune system. Moreover, AIS encompasses any system or computational tool that extracts ideas and metaphors from the biological immune system. It is a highly evolved, parallel and distributed adaptive system. An increasing number of its application models have been developed to solve the optimization problems [23,3]. AIS in the optimization mainly focuses on negative selection [18], artificial immune network [27] and clonal selection [16].

As a new branch of nature computing, membrane computing, which is inspired by the function and structure of the living biological cells, is proposed by Păun as an abstracted parallel and distributed computing model. More specifically, the membrane system with an enhanced parallelism is able to trade space for time, that is to say, it can solve intractable problems in a feasible time due to making use of an exponential space. Most membrane systems are computationally universal, they equal to Turing machines in computing power. In a membrane system, multisets are placed in the compartments defined by the membrane structure, the symbol-objects are evolved by executing the reaction rules in a maximally parallel and non-deterministic manner. At present, there are three classes of membrane systems: cell-like membrane systems, tissue-like membrane systems and neural-like membrane systems [48]. In this paper, the cell-like membrane systems are employed to design the proposed algorithm to solve the global numeric optimization problems. The specific descriptions of the cell-like membrane systems will be discussed in the following section.

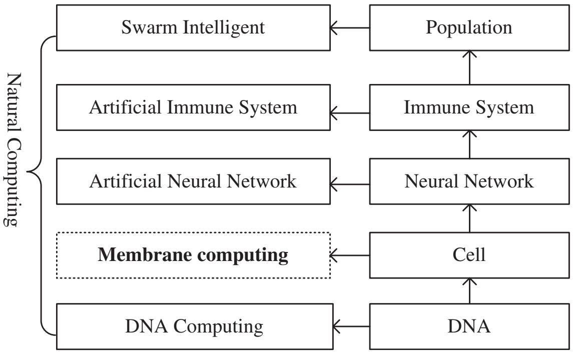


Fig. 1. Hierarchical model of biological simulation of computing systems.

3. Membrane systems

This section will discuss the basic notions and formalizations of the cell-like membrane systems. The cell-like membrane system is a computation model which is inspired by the structure and functioning of a biological living cell [48,46]. The fundamental ingredients of a cell-like membrane system include membrane structure, reaction rules and multisets. The membrane structure is a hierarchically arranged set of the membranes, as shown in Fig. 2. Strictly speaking, the structure has an arrangement compartments delimited by the membranes, which is very suitable to implement the distributed and parallel computing. The reaction rules may process multiset of symbol-objects placed in the compartments of the membrane structure. To further understand a cell-like membrane system, its basic structure with degree n is simply described in Eq. (1).

$$\Pi = (V, T, \mu, w_1, \ldots, w_n, R) \tag{1}$$

(a) V is the alphabet. Its element is named as an object. An object is the abstract representation of atomic, molecular or the other chemical substances. The object may be represented by symbol or string, also named as symbol-object.
(b) $T \subseteq V$, where T is the output alphabet.
(c) μ is a membrane structure with degree n.
(d) $w_i \in V^*,\ 1 \leqslant i \leqslant n,\ w_i$ represents the multiset in the ith region of the membrane structure μ.
(e) R represents the reaction rules in the region of membranes.

The concepts in Fig. 2 are explained in details as follows. The external membrane is usually called the skin membrane which separates the internal membranes from the external environment. The internal membranes include the membranes and the elementary membranes. Each membrane determines a compartment, also called a region. If the region of a membrane does not have any other membranes, the membrane is called an elementary membrane. Multisets and reaction rules are associated with the regions of the membranes. The symbol-objects are evolved by executing the reaction rules.

3.1. Symbol-objects and multiset

In membrane systems, a symbol-object is an abstract representation of an element in biological living cells, e.g. molecule, ion or protein, etc. The symbol-object can be transformed into another symbol-object by executing the rewrite rule. A multiset consists of some symbol-objects, and it can pass through one membrane to another membrane. Each region may contain the appointed multiset. It should be noted that the multiset is an information unit or data structure used in membrane systems.

3.2. Reacting rules

Reacting rules are abstracted from either the chemical reactions or the biological processes in the compartments of a biological living cell. In a cell-like membrane system, the abstracted reaction rules include rewrite rules, division rule, dissolution rule, and communication rule. More specifically, rewrite rules may evolve the multiset in the region of membranes. The multiset can be transported from one membrane to the other membranes by executing the communication rule, and the membrane structure can be adjusted by invoking both division rule and dissolution rule. The details of reaction rules will be discussed as follows.

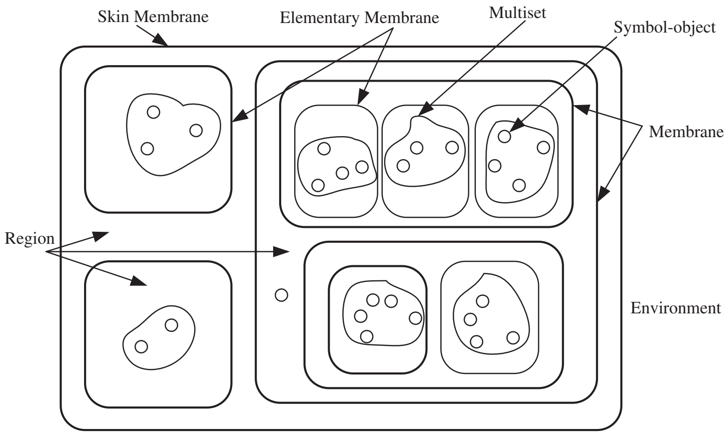


Fig. 2. The basic structure of the membrane systems.

- Rewrite rules may evolve the multiset by exchanging the information of symbol-objects in a non-deterministic manner. For example, a rewrite rules is shown as $[_n a \rightarrow b]_n$, where n represents the label number of the membrane; a is a multiset before the change while b is a new multiset produced by invoking the rewrite rules;
- Division rule may create some membranes which are as the basic computing units. For example, a division rules is shown as $[_0]_0 \rightarrow [_0[_1]_1[_2 \cdots [_n \cdots]_n]_2]_0$, where $[_0]_0$ denotes a skin membrane with label 0. The membranes with label $1, 2, \ldots, n$ are produced by invoking this rule.
- Dissolution rule enables a membrane to be dissolved so that the multisets can be released from the current membrane to the outer layer one. However, the skin membrane cannot be dissolved during the whole evolving process.
- Communication rule may send the multiset from a region to another region. For example, $a[_n]_n \rightarrow [_n a]_n$, where n represents the label of the membrane. The multiset a can be sent into the region of the membrane with the label n after executing the communication rule.

3.3. Calculation process

A cell-like membrane system is a computing model which is abstracted from the structure and functioning of the biological living cell. More specifically, the reactions of chemical substances is happened in their compartmental structure of the biological living cell, and the chemical substances can be passed between the membranes. Based on above two phenomenon, the calculation process of membrane system is designed as follows: the multiset of symbol-objects can be evolved according to the given rewrite rules in a non-deterministic and maximally parallel manner, and the multiset may be transitioned between the membranes. According to executing different reaction rules, a membrane system implements the above-mentioned operators. The calculation process is terminated till all reaction rules are executed in the region of membrane systems. The multiset in the appointed region of the membrane is as the computational result. The skin membrane is chosen as the output results in this paper.

4. Evolutionary membrane algorithm

As discussed in the previous section, the main objective of this paper is to devise a novel intelligent computational model, which is inspired by some basic features of the biological membranes, to solve the global numeric optimization problems. In the following section, EMA is outlined and its implementation steps are explained in details.

The analysis of the implementation processes for some state-of-the-art evolutionary algorithms is useful to design a high-performance EMA. Generally speaking, the state-of-the-art evolutionary algorithms need to generate some random sample points in a feasible region. And then, the attractive regional area is determined according to the objective function values of those sample points. After that the evolutionary operations are invoked to develop candidate solutions in the attractive area. Finally, after the end conditions of the algorithms were reached, all operators of the algorithms are terminated. For example, these operations in Genetic Algorithm [30,51] correspond to selection, crossover and mutation operations. In Particle Swarm Optimization [29,39], the updating operations of the speed and the position are used to evolve the particles of the swarm. Similarly, in the proposed EMA, the multiset in the membrane is employed as the set of candidate solutions of the optimization problems. And then, inspired by the chemical reaction happening in the biological cell, some reaction rules are designed to evolve the multisets. The basic implementation of EMA is shown in Fig. 3. Its specific steps are described as follows.

4.1. Initialization

The parameters of a membrane system need to be first initialized, e.g. membrane structure, the number of symbol-objects, the number of membranes and the reaction rules. In the proposed EMA, the membrane structure is simplified, which consists of a skin membrane containing several elementary membranes. The symbol-object represents a candidate solution of the optimization problems, while the multiset denotes a set of the candidate solutions. The initialization details of symbol-objects will be discussed in Section 4.2.1. The reaction rules include rewrite rules, division rule, dissolution rule, communication rule and partition rule, they are designed in Section 4.2.2.

4.2. Skin membrane

When a skin membrane is established, the symbol-objects and reaction rules will be configured. At first, the symbol-objects will be initialized under satisfying the constraint conditions of the optimization problems in Section 4.2.1. Later, the reaction rules, which are inspired by the function of the biological cell, will be defined in Section 4.2.2.

4.2.1. The initialization of symbol-objects

In order to solve the optimization problems, the symbol-objects need to be encoded. In other words, each symbol-object represents a candidate solution of the optimization problem in a D-dimensional real parameter space. In the inner region of the skin membrane, symbol-objects can be initialized under satisfying upper and lower boundary constraints of the optimization problems.

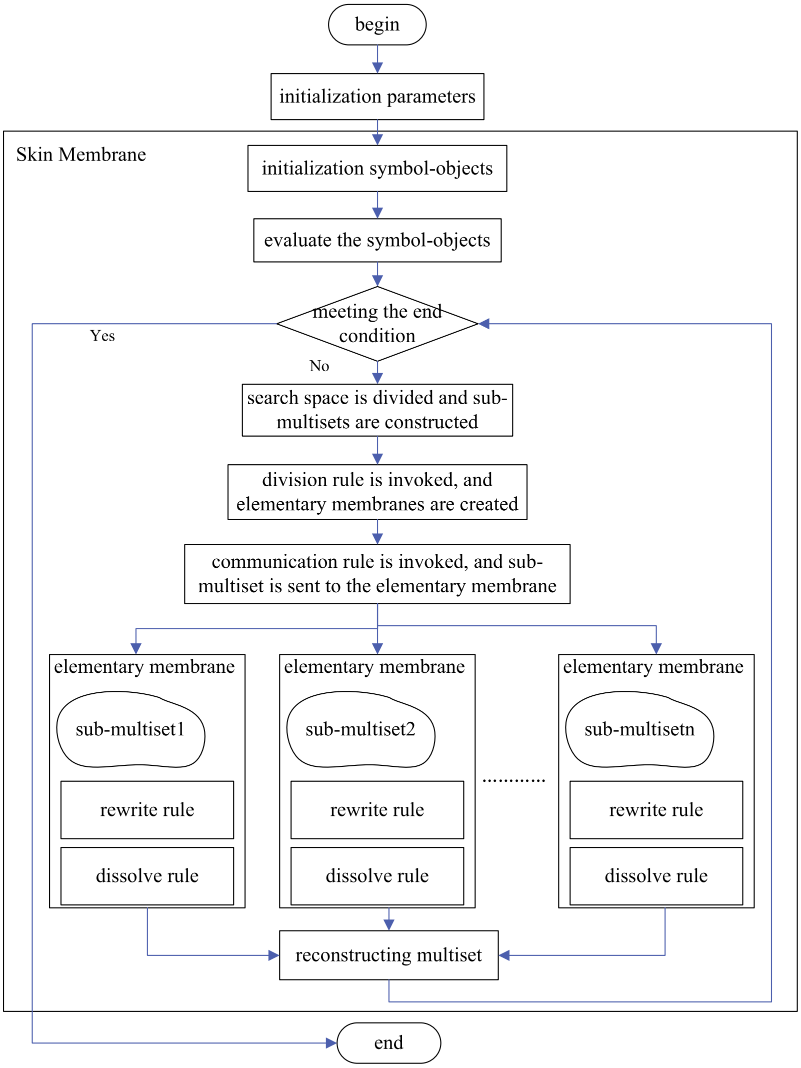

begin
initialization parameters
Skin Membrane
initialization symbol-objects
evaluate the symbol-objects
meeting the end condition
Yes
No
search space is divided and sub-multisets are constructed
division rule is invoked, and elementary membranes are created
communication rule is invoked, and sub-multiset is sent to the elementary membrane
elementary membrane
sub-multiset1
rewrite rule
dissolve rule
elementary membrane
sub-multiset2
rewrite rule
dissolve rule
....
elementary membrane
sub-multisetn
rewrite rule
dissolve rule
reconstructing multiset
end

Fig. 3. The schematic diagram of evolutionary membrane algorithm

Because each symbol-object in the region of the membrane is randomly swimming, an effective mechanism is needed to describe its initialization position. Chaotic system can describe the dynamic behavior relying on the initial conditions and unstable periodic motion within a specific range, and it has some characteristics including the pseudo-randomness and the periodicity [8]. Therefore, chaotic system is introduced to the proposed EMA as the effective mechanism to initialize the symbol-objects in the region of the membranes. Finally, in the proposed EMA, Sinusoidal Iterator is employed to initialize a D-dimensional symbol-object. Its form is described as follows.

(a) The chaotic variable is set to $y_i^0 = 0.6 + 0.1 * r$, where r represents a random function between 0 and 1.
(b) According to the iterative equation of Sinusoidal Iterator based on a chaotic system in Eq. (2), the next generation of chaotic variables y_i^k is calculated as y_i^{k+1}, where $a = 2.3$;

$$y_i^{k+1} = a * (y_i^k)^2 * \sin(\pi * y_i^k) \tag{2}$$

(c) After k was accumulated to 20, the chaotic variable y_i^{k+1} is converted to a symbol-object x_i according to Eq. (3):

$$x_i = x_i^{min} + y_i^{k+1} * (x_i^{max} - x_i^{min}) \tag{3}$$

(d) The initialization of the symbol-object is terminated till $i = i + 1$ is greater than D dimension. Otherwise, this process continues to be executed.
(e) The fitness value of the symbol-object is calculated according to the objective function of the optimization problems.

4.2.2. The definition of reaction rules

Reaction rules are inspired by the biochemical reactions taking place in a internal biological living cell. It is necessary that eight reaction rules are constructed to simulate the biochemical process happened in a region (compartment) of the membrane structure. The reaction rules consist of four rewrite rules and four function rules. To be more precise, four rewrite rules are designed to rewrite the information of symbol-objects. They can help EMA to find the global optimal solutions by balancing exploration and exploitation during the search process. In addition, four functional rules include partition rule, division rule, communication rule and dissolution rule. The detail descriptions of eight reaction rules are discussed as follows.

(a) Rewrite rule 1. This rule may make the current symbol-object to swim toward the global optimum direction using the collected global information. It can operate the present symbol-object with large steps for the global exploration. Its form is described by Eq. (4).

$$x_i' = \begin{cases} x_i^g + (x_i^c - x_i^w) * r_1, & \text{if } r_2 > 0.5, \\ x_i^g + (x_i^l - x_i^c) * r_1, & \text{otherwise}, \end{cases} \tag{4}$$

where x_i^c represents the ith dimension of the symbol-object randomly chosen in the elementary membrane. x_i' is a new position of the ith dimension of the current x_i^c. x_i^g is the ith dimension of the symbol-object with the best fitness from the skin membrane. x_i^w is the ith dimension of the symbol-object with the worse fitness in the elementary membrane. x_i^l is the ith dimension of the symbol-object with the local best fitness in the elementary membrane. r_1 and r_2 are two random numbers with the uniform distribution between 0 and 1, respectively.

(b) Rewrite rule 2. The second rule may produce a new symbol-object by exchanging with the information of two adjacent symbol-objects with state 1. If the new symbol-object takes the best characteristics from the adjacent symbol-objects, the new symbol-object may be better than the old one. The rule makes the algorithm to converge within a promising area. Its form is described by Eq. (5).

$$x_i' = r * x_i^{n_1} + (1 - r) * x_i^{n_2}, \tag{5}$$

where x_i' is the ith dimension of the new symbol-object. $x_i^{n_1}$ and $x_i^{n_2}$ are two adjacent symbol-objects of the current x_i^c, and their states are of 1. r is a random number with uniform distribution between 0 and 1.

(c) Rewrite rule 3. The third rule can operate the symbol-object with the median step for the local search. The rule makes the information of the adjacent symbol-object to be fully shared. The rule can maintain the diversity of the candidate solutions in the search space. It is described by Eq. (6).

$$x_i' = \begin{cases} 0.8 * \frac{1}{n}\sum_{j=1}^{n} x_i^{n_j} + 0.2 * (x_i^l - x_i^c), & \text{if } r < 0.2 \\ x_i^c, & \text{otherwise}, \end{cases} \tag{6}$$

where x_i^c represents the ith dimension of the symbol-object randomly chosen in the elementary membrane. x_i' is a new position of the ith dimension of the current x_i^c. $x_i^{n_j}$ is the ith dimension of the jth adjacent symbol-object of x_i^c. x_i^l is the ith dimension of the symbol-object with the best fitness in the elementary membrane. r is a random number with uniform distribution between 0 and 1.

(d) Rewrite rule 4. The rule is used to exchange the information between the current symbol-object and the symbol-object with the global best fitness from the skin membrane. It is analogous to the reproduction and biological crossover. The diversity of the approximate optimal solutions is improved, and the exploration capability is enhanced in the search space. It is described by Eq. (7).

$$x_i' = \begin{cases} x_i^c, & \text{if } 0 \leqslant r \leqslant 0.9, \\ x_i^g, & \text{otherwise}, \end{cases} \tag{7}$$

where x_i^c represents the ith dimension of the symbol-object randomly chosen in the elementary membrane. x_i' is a new position of the ith dimension of the current x_i^c. x_i^g is the ith dimension of the symbol-object with the best fitness from the skin membrane. r is a random number with uniform distribution between 0 and 1.

(e) Partition rule. The rule is employed to improve the search efficiency of the proposed algorithm. The rule is implemented by dividing the search space of the optimization problems as follows. First, symbol-objects in multiset are sorted according to their fitness values. Second, the sorted symbol-objects are divided into some sub-multisets with the same size so that each membrane has its own sub-multiset. Its implementation is described by Eq. (8).

$$\begin{aligned} &W' = sort(W) \\ &W' = \{w_1, w_2, \ldots, w_m\} \\ &w_i = W'((i-1) * n + 1 : n : i * n) \\ &n = |W|/m \end{aligned} \tag{8}$$

where $1 \leqslant i \leqslant m,\ m$ is the number of the membranes. W denotes the multiset in the region of the skin membrane. $|W|$ represents the number of the symbol-objects in the multiset. W' represents the sorted multiset according to the fitness of the symbol-objects. w_i is the multiset assigned in the ith elementary membrane. $W'((i-1) * n+1 : n : i * n)$ is a sub-multiset which n symbol-objects in the sorted multiset are taken from the starting position $(i-1) * n+1$.

(f) Division rule. Inspired by the process of cell division, division rule is designed to change the structure of the membrane system. More specifically, the elementary membranes, which are the basic evolutionary unit, are created by invoking division rule in the region of the skin membrane. Its specific implementation is described by Eq. (9).

$$[\,]_0 \xrightarrow{division\_rule} [[\,]_1, [\,]_2, \ldots, [\,]_m]_0, \tag{9}$$

where $[\,]_0$ indicates the skin membrane, and $[\,]_i$ is the ith elementary membrane, m is the total number of the elementary membranes.

(g) Communication rule. The rule implements a process in which the multiset (chemical substances) in the skin membrane can be sent into the region of the elementary membranes. In EMA, the multiset of the skin membrane may be sent to the corresponding elementary membranes by invoking communication rule. This rule makes the proposed algorithm to search for the approximate optimal solutions in an appointed search space of the optimization problem. Its detail form is described by Eq. (10).

$$[w_1, w_2, \ldots, w_m]_0 \xrightarrow{communication\_rule} [[w_1]_1, [w_2]_2, \ldots, [w_m]_m]_0 \tag{10}$$

where m is the total number of the elementary membranes; $[\,]_0$ indicates the skin membrane; while $[\,]_i$ is the ith elementary membrane; w_i is the ith sub-multiset based on the divided multiset.

(h) Dissolution rule. When all evolutionary operations had been executed in the region of the elementary membrane, dissolution rule is invoked to release the multiset (the optimal approximate solutions) from the elementary membrane into the skin membrane. In the region of the skin membrane, the information is shared on symbol-objects from the different membranes. And the rule can help EMA to find the global approximate optimum solutions quickly. Its specific form is given in Eq. (11).

$$[[w_1]_1, [w_2]_2, \ldots, [w_m]_m]_0 \xrightarrow{dissolve\_rule} [w_1, w_2, \ldots, w_m]_0 \tag{11}$$

where m is the total number of the elementary membranes; $[\,]_0$ indicates the skin membrane; while $[\,]_i$ is the ith elementary membrane; w_i is the ith sub-multiset based on the divided multiset.

4.3. The elementary membrane

The elementary membrane is seen as a basic evolutionary unit of the proposed algorithm. The evolutionary mechanisms are implemented by the simulation of both the collisions and the random motion of the molecules (multiset) in the region of the membrane. However, the simulation on molecules (multiset) is very complex. The cellular automata model is a discrete dynamical system and powerful tool to simulate the evolution of both the macro-structure and the micro-structure. At the micro-level, the cellular automata model can simulate the complex phenomenon by updating the simple states of the cells. In this paper, the cellular automata model is introduced to simulate the process, which can help the proposed algorithm to find the global optimal symbol-object using the adjacent information of the symbol-object.

Cellular automata is a mathematical model of the dynamical complex systems, which consists of a large number of simple components named as cells. Because the cells have strong learning capabilities by executing the evolutionary rules, they may simulate the complicated behavioral patterns. At a specific moment, the distributed state of cells is independent, discrete and limited in the multi-dimensional space, which is evolved according to the certain local rules. The state of the current cell is updated based on its adjacent states by invoking the evolutionary rules. And they may simulate the evolutionary process of the complex systems. In the proposed EMA, it is worth mentioning that a cell in the cellular automata model is equivalent to a symbol-object in the membrane systems. Fig. 4 describes the movement of the molecular simulated by Cellular automata. Its detail forms are described as follows.

First, the multiset in the elementary membrane is mapped to a two-dimensional solutions space grid in Fig. 4. The rows in the grid equal $\lceil\sqrt{|w_i|}\rceil$ (the ceiling of the square root of the size of w_i). At the same time, a corresponding state grid is generated, and each state is randomly set to 0 or 1. Where the state 0 represents an active cell, while the state 0 denotes an inactive cell.

Second, a symbol-object is randomly chosen from the above-mentioned solutions space grid. And then, its adjacent symbol-objects can be found from the grid. Its corresponding state and its adjacent state are also found from the state space grid in Fig. 4. When the state value of the current symbol-object equals 1, some rules will be executed as follows.

If the sum of its adjacent state value is 1, then the rewrite rule 1 is executed.
If the sum of its adjacent state value is 2, then the rewrite rule 2 is executed.
If the sum of its adjacent state value is 3, then the rewrite rule 3 is executed.
If the sum of its adjacent state value is 4, then the rewrite rule 4 is executed.

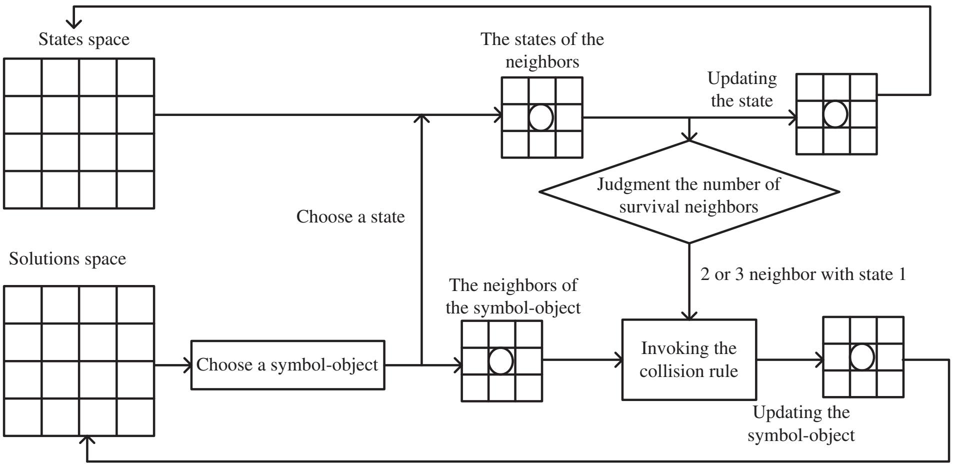

States space
Solutions space
Choose a symbol-object
Choose a state
The states of the neighbors
Updating the state
Judgment the number of survival neighbors
2 or 3 neighbor with state 1
The neighbors of the symbol-object
Invoking the collision rule
Updating the symbol-object

Fig. 4. The movement of molecular simulated by Cellular automata.

Finally, the rule of the changing state is invoked for updating the state of the current symbol-object. The specific form is described in Eq. (12).

$$x_c^s = \begin{cases} 1, & \text{if } 2 \leqslant \sum_{i=1}^{8} x_i^s \leqslant 3, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

where x_c^s denotes the state of the current symbol-object chosen randomly from the solutions space grid. x_i^s is the ith adjacent state of the current symbol-object.

The state change of the cell is illustrated as follows. As observed in Fig. 5, the states of the cells are updated in the state space. The solid circle represents an active cell with the state value of 1, while the hollow circle denotes an inactive cell with the state value of 0. A circle with the direction arrow is a cell randomly chosen from the solutions space grid. When the sum of its adjacent state is two or three, the state of the current cell is modified to be 1; Otherwise, its state is set to 0.

The above-mentioned steps are repeated till the specific termination criterion is reached. When the proposed algorithm is terminated, the multiset in the skin membrane represents the desired global optimal approximate solutions to solve the optimization problems. The pseudo-code of the proposed EMA is summarized in Fig. 6.

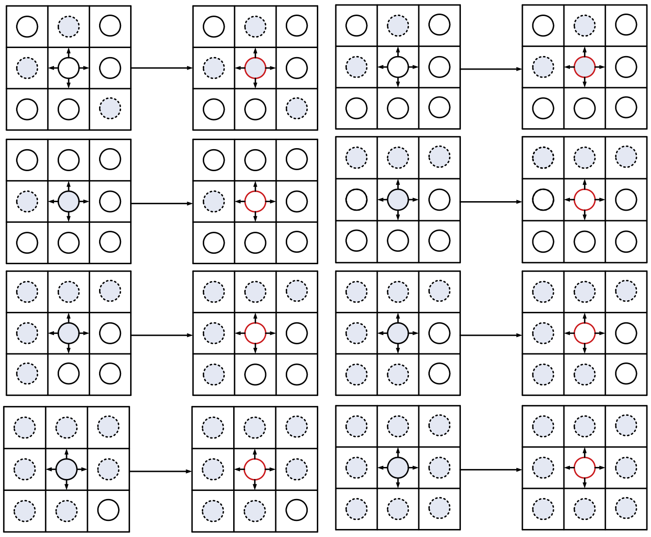
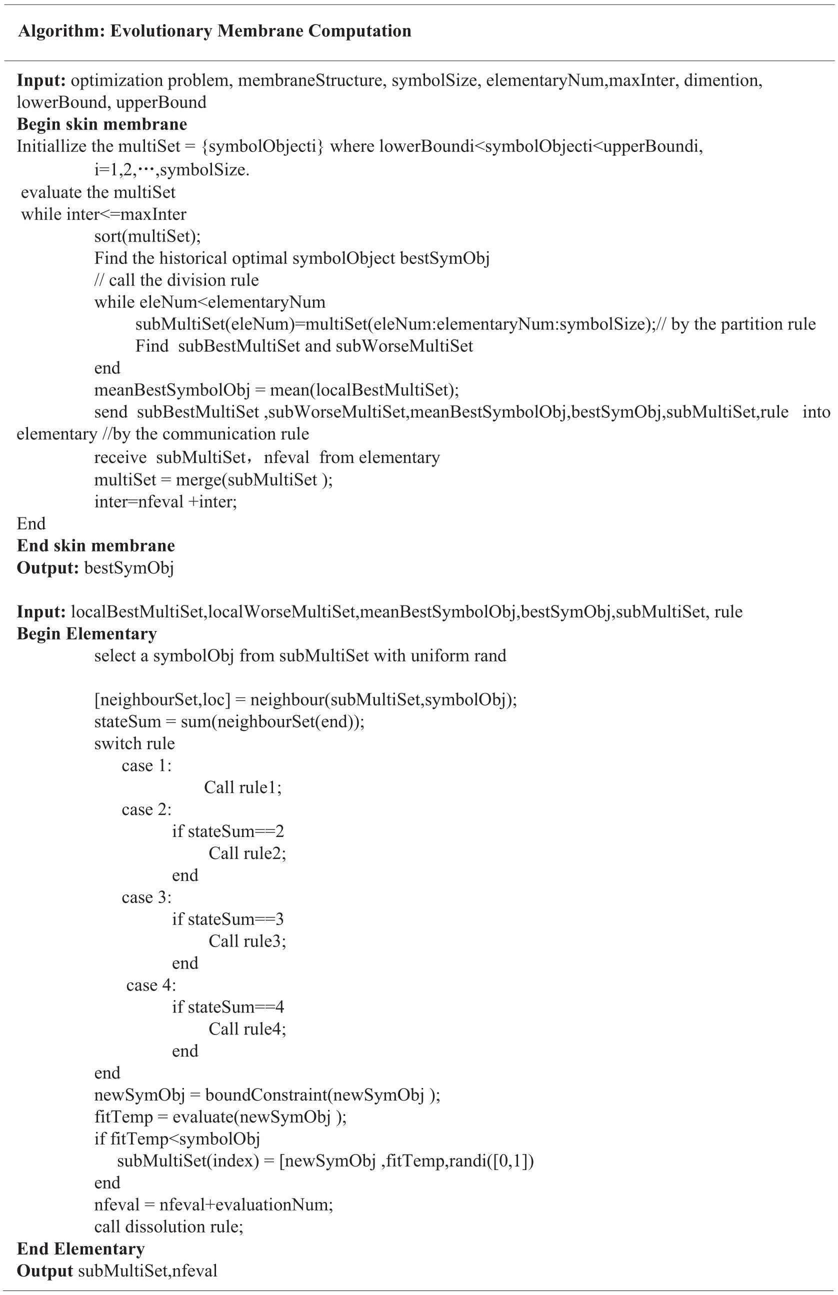


Fig. 5. The current state changed by Cellular automata.



```
Algorithm: Evolutionary Membrane Computation

Input: optimization problem, membraneStructure, symbolSize, elementaryNum,maxInter, dimention,
lowerBound, upperBound
Begin skin membrane
Initiallize the multiSet = {symbolObjecti} where lowerBoundi<symbolObjecti<upperBoundi,
          i=1,2,⋯,symbolSize.
 evaluate the multiSet
 while inter<=maxInter
          sort(multiSet);
          Find the historical optimal symbolObject bestSymObj
          // call the division rule
          while eleNum<elementaryNum
               subMultiSet(eleNum)=multiSet(eleNum:elementaryNum:symbolSize);// by the partition rule
               Find  subBestMultiSet and subWorseMultiSet
          end
          meanBestSymbolObj = mean(localBestMultiSet);
          send  subBestMultiSet ,subWorseMultiSet,meanBestSymbolObj,bestSymObj,subMultiSet,rule   into
elementary //by the communication rule
          receive  subMultiSet， nfeval  from elementary
          multiSet = merge(subMultiSet );
          inter=nfeval +inter;
End
End skin membrane
Output: bestSymObj

Input: localBestMultiSet,localWorseMultiSet,meanBestSymbolObj,bestSymObj,subMultiSet, rule
Begin Elementary
          select a symbolObj from subMultiSet with uniform rand

          [neighbourSet,loc] = neighbour(subMultiSet,symbolObj);
          stateSum = sum(neighbourSet(end));
          switch rule
             case 1:
                        Call rule1;
             case 2:
                   if stateSum==2
                        Call rule2;
                   end
             case 3:
                   if stateSum==3
                        Call rule3;
                   end
              case 4:
                   if stateSum==4
                        Call rule4;
                   end
          end
          newSymObj = boundConstraint(newSymObj );
          fitTemp = evaluate(newSymObj );
          if fitTemp<symbolObj
             subMultiSet(index) = [newSymObj ,fitTemp,randi([0,1])
          end
          nfeval = nfeval+evaluationNum;
          call dissolution rule;
End Elementary
Output subMultiSet,nfeval
```

Fig. 6. Pseudo-code of EMA.

5. Experimental study

The main goal of this paper is to present a cell-inspired evolutionary algorithm to solve global numeric optimization problems. To evaluate its effectiveness and efficiency, this section shows the simulation studies of the proposed EMA in solving global numerical optimization problems. Moreover, the performance of the proposed EMA is compared with some state-of-the-art evolutionary algorithms by using the numerous benchmark functions. Finally, the superiority of the proposed algorithm is shown by the analysis of the experimental data.

To perform a comprehensive evaluation, the presentations of the experimental results are divided into two subsections. First, the performance of the proposed algorithm is analyzed with the different parameters in Section 5.2. Second, an overall performance of the proposed EMA in comparison with the four state-of-the-art evolutionary algorithms is provided in Section 5.3.

5.1. Test functions

To provide a comprehensive comparison and highlight the different aspects of the proposed EMA, the presentation of the simulation experiment will be divided into the following subsections. In Section 5.2, 10 well-known comprehensive benchmark functions, which are briefly described in Table 1, are employed to analyze the performance of EMA with different parameters. These functions are used to conduct the analysis of the parameters of EMA, which have different characteristics, e.g. unimodality, multi-modality, ill-condition, noise, etc. In addition, 25 test instances proposed in the CEC2005 special sessions on real-parameter optimization are employed to verify the performance of the proposed EMA in comparison with the state-of-the-art algorithms in the following Section 5.3. These benchmark test problems are very popular in the global numeric optimization, which can be divided into the following four classes: unimodal problems from f_1 to f_5, basic multimodal functions from f_6 to f_{12}, expanded multimodal problems from f_{13} to f_{14}, and hybrid composition functions implementing a combination of several well-known benchmark functions from f_{15} to f_{25} with a large number of local minima. A thorough detailed description of these benchmark functions can be found in [54].

5.2. Experiment 1

Two parameters, including the number of the elementary membranes and the number of the symbol-objects, affect the solving performance of the proposed EMA. In this study, it is found that EMAs with the different setting parameters can produce different results. The parameters of EMA are discussed in the following experiment.

5.2.1. Experimental setup

All simulations are run on a Pentium dual-core computer with 2.93-GHz and 2-GB RAM. The software MATLAB 7.11 is employed to develop the program of EMA. The dimensional size of all test functions is set to 30 on this experiment. The default parameters of EMA are set, that is the number of the elementary membranes set to 25, and the number of the symbol-objects set to 200. When one of the two parameters is analyzed, another parameter is set to the default value. To reduce the randomness of the algorithm, each test function is executed independently 25 times. The terminated condition of the proposed EMA is configured as the maximum 300,000 fitness evaluations.

5.2.2. Comparison among EMAs with different parameters

The performance of the proposed algorithm is analyzed with different numbers of the elementary membranes. The elementary membrane represents a computing unit where the symbol-objects can be evolved by executing rewrite rules. In the simulation, the proposed EMA is executed independently 25 times on each benchmark test function with 30-dimensional optimization problems in Table 1 [61]. The number of the symbol-objects is regularly set to 200. Moreover, the number of the elementary membranes is set to 5, 10, 20, 25 and 50, respectively. The results obtained by five EMAs using a different number of elementary membranes are tabulated in Table 2.

From Table 2, we can draw a conclusion that EMA with 25 elementary membranes has the best results on Ackley, Griewank, Rastrigin and Schwefel, and the proposed EMA with 50 elementary membranes has the optimal results on Sphere, Whitely and Penalized2. When the number of elementary membranes is set to 20, the algorithm gets two good results on Rosenbrock and Salomon. The proposed algorithm with 10 elementary membranes has only one good result on Penalized1. The experimental results indicate that the number of elementary membranes relates to the ability of solving the global nu-

Table 1
Benchmark functions used in this experiment, including a short description of their characteristics.

Functions	Formula
Sphere	$\sum_{i=1}^{N} x_i^2, -100 \leqslant x_i \leqslant 100$
Rosenbrock	$\sum_{i=1}^{N-1}(100(x_{i+1}-x_i^2)^2+(1-x_i)^2), -30 \leqslant x_i \leqslant 30$
Ackley	$20+\exp(1)-20 * \exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}\right)-\exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right),\ -32 \leqslant x_i \leqslant 32$
Griewank	$\sum_{i=1}^{N}\frac{x_i^2}{4000}-\prod_{i=1}^{N}\cos\frac{x_i}{\sqrt{i}}+1,\ -600 \leqslant x_i \leqslant 600$
Rastrigin	$10N+\sum_{i=1}^{N}(x_i^2-10\cos(2\pi x_i)),\ -5 \leqslant x_i \leqslant 5$
Schwefel	$418.9829N-\sum_{i=1}^{N}\left(x_i \sin(\sqrt{\lvert x_i \rvert})\right),\ -500 \leqslant x_i \leqslant 500$
Salomon	$-\cos\left(2\pi\sqrt{\sum_{i=1}^{N}x_i^2}\right)+0.1\sqrt{\sum_{i=1}^{N}x_i^2}+1,\ -100 \leqslant x_i \leqslant 100$
Whitely	$\sum_{j=1}^{N}\sum_{i=1}^{N}\left(\frac{y_{i,j}^2}{4000}-\cos(y_{i,j})+1\right),\ -100 \leqslant x_i \leqslant 100$ $y_{i,j}=100(x_j-x_i^2)^2+(1-x_i)^2$
Penalized1	$\frac{\pi}{N}\left\{10\sin^2(\pi y_1)+\sum_{i=1}^{N-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_N-1)^2\right\}+\sum_{i=1}^{N}u(x_i,10,100,4),\ y_i=1+\frac{1}{4}(x_i+1),\ -50 \leqslant x_i \leqslant 50$
Penalized2	$0.1\left\{\sin^2(\pi 3x_1)+\sum_{i=1}^{N-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_N-1)^2[1+\sin^2(2\pi x_N)]\right\}+\sum_{i=1}^{N}u(x_i,5,100,4), -50 \leqslant x_i \leqslant 50$

Table 2
Experimental results obtained by five EMAs using different numbers of the elementary membranes.

Function	5	10	20	25	50
Sphere	3.63e−40	1.05e−44	4.26e−48	1.95e−51	**8.24e−54**
	1.61e−39	3.12e−44	1.96e−47	4.30e−51	**2.75e−53**
Rosenbrock	3.19e+00	1.33 e+00	**8.34e−01**	1.28e+00	2.11e+00
	4.72e+00	2.55e+00	**1.60e+00**	1.89e+00	2.06e+00
Ackley	1.17e+00	9.17e−01	8.09e−01	**6.42e−01**	7.07e−01
	9.71e−01	8.68e−01	9.79e−01	**8.96e−01**	8.35e−01
Griewank	2.54e−02	2.21e−02	2.17e−02	**1.73e−02**	1.24e−02
	2.84e−02	1.89e−02	2.57e−02	**1.89e−02**	1.69e−02
Rastrigin	3.05e+01	2.80e+01	2.41e+01	**2.06e+01**	2.52e+01
	7.81e+00	8.02e+00	6.59e+00	**7.15e+00**	9.67e+00
Schwefel	2.24e+03	2.29e+03	2.14e+03	**2.06e+03**	2.12e+03
	4.60e+02	3.55e+02	3.79e+02	**4.65e+02**	5.76e+02
Salomon	5.99e−01	4.91e−01	**4.43e−01**	4.67e−01	4.87e−01
	1.32e−01	7.02e−02	**9.60e−02**	7.48e−02	8.32e−02
Whitely	2.63e+02	2.34e+02	2.58e+02	2.57e+02	**2.53e+02**
	1.27e+02	7.36e+01	1.19e+02	9.17e+01	**1.28e+02**
Penalized1	9.96e−02	**3.73e−02**	1.32e−01	1.16e−01	1.62e−01
	2.21e−01	**1.07e−01**	2.42e−01	1.72e−01	4.67e−01
Penalized2	3.28e−01	2.58e−01	5.48e−02	5.23e−03	**2.19e−03**
	8.18e−01	8.54e−01	1.92e−01	6.33e−03	**4.48e−03**

The bold values represent good resutls obtained by optimization methods.

Table 3
Experimental results obtained by four EMAs using different numbers of symbol-objects.

Function	50	100	200	500
Sphere	**7.11e−103**	4.02e−79	1.95e−51	4.68e−42
	2.71e−102	1.15e−78	4.30e−51	1.58e−41
Rosenbrock	**1.11e+00**	1.43e+00	1.28e+00	1.66e+00
	1.82e+00	1.95e+00	1.89e+00	2.34e+00
Ackley	3.42e+00	1.68e+00	**6.42e−01**	4.10e+00
	1.00e+00	9.46e−01	**8.96e−01**	7.02e−01
Griewank	7.57e−02	1.86e−02	**1.73e−02**	1.86e−02
	1.67e−01	1.59e−02	**1.89e−02**	2.54e−02
Rastrigin	3.75e+01	2.88e+01	**2.06e+01**	2.26e+01
	1.27e+01	7.90e+00	**7.15e+00**	5.78e+00
Schwefel	3.00e+03	2.79e+03	**2.07e+03**	2.12e+03
	5.70e+02	4.76e+02	**4.65e+02**	3.49e+02
Salomon	1.07e+00	6.23e−01	4.67e−01	**4.07e−01**
	2.61e−01	1.53e−01	7.48e−02	**8.12e−02**
Whitely	3.97e+02	2.99e+02	2.57e+02	**2.29e+02**
	1.82e+02	1.28e+02	9.17e+01	**9.26e+01**
Penalized1	2.40e−01	2.61e−01	1.16e−01	**9.12e−02**
	3.83e−01	5.11e−01	1.72e−01	**1.56e−01**
Penalized2	1.98e−01	2.81e−01	**5.23e−03**	5.71e−03
	7.31e−01	9.24e−01	**6.33e−03**	9.57e−03

The bold values represent good resutls obtained by optimization methods.

meric optimization problems. Therefore, when the proposed algorithm has a suitable number of elementary membranes, it can find good approximate global optimal solutions for the numeric optimization problems.

Next, the performance of EMA is analyzed with different numbers of symbol-objects. The proposed EMA is executed independently 25 times on each benchmark test function. The number of elementary membranes is set to 25. The number of symbol-objects is set to 50, 100, 200 and 500, respectively. The statistical results obtained by four EMAs using different numbers of symbol-objects are shown in Table 3.

From Table 3, EMA with 200 symbol-objects outperforms the other EMAs with different numbers of symbol-objects on Ackley, Griewank, Rastrigin, Schwefel, and Penalized2. When the number of the symbol-objects is set to 500, the proposed algorithm has the best results on Salomon, Whitely, and Penalized1. When the number of the symbol-objects is set to 50,

Table 4
Experimental results of CLPSO, GL-25, EPSDE, SaDE and EMA on 30-dimensional functions.

	CLPSO [31] Mean(Std) Rank	GL-25 [20] Mean(Std) Rank	EPSDE [34] Mean(Std) Rank	SaDE [49] Mean(Std) Rank	EMA Mean(Std) Rank
F1	0 (0) =	5.60e−27 (1.76e−26) +	0 (0) =	0 (0) =	**0** **(0)** /
F2	8.40e+02 (1.90e+002) +	4.04e+01 (6.28e+01) +	**4.23e−26** **(4.07e−26)** –	8.26e−06 (1.65e−05 +	2.74e−17 (6.90e−17) /
F3	1.42e+07 (4.19e+006) +	2.19e+06 (1.08e+06) +	8.74e+05 (3.28e+06) =	4.27e+05 (2.08e+05) +	**1.89e+05** **(1.03e+05)** /
F4	6.99e+03 (1.73e+003) +	9.07e+02 (4.25e+02) +	3.49e+02 (2.23e+03) +	1.77e+02 (2.67e+02) +	**3.96e+01** **(9.78e+02)** /
F5	3.86e+03 (4.35e+002) =	2.51e+03 (1.96e+02) +	1.40e+03 (7.12e+02) =	3.25e+03 (5.90e+02) +	**1.40e+03** **(5.97e+02)** /
F6	4.16e+00 (3.48e+00) +	2.15e+01 (1.17e+00) =	**6.38e−01** **(1.49e+00)** –	5.31e+01 (3.25e+01) +	1.75e+00 (2.01e+00) /
F7	4.51e−01 (8.47e−02) +	2.78e−02 (3.62e−02) =	1.77e−02 (1.34e−02) =	1.57e−02 (1.38e−02) +	**1.29e−02** **(2.88e−02)** /
F8	2.09e+01 (4.41e−02) =	2.09e+01 (5.94e−02) =	2.09e+01 (5.81e−02) =	2.09e+01 (4.95e−02) =	**2.01e+01** **(3.18e−01)** /
F9	0 (0) =	2.45e+01 (7.35e+00) +	0 (0) =	2.39e−01 (4.33e−01) +	**0** **(0)** /
F10	1.04e+02 (1.53e+01) +	1.42e+02 (6.45e+01) +	5.36e+01 (3.03e+01) +	4.72e+01 (1.01e+01) +	**1.27e+01** **(1.19e+01)** /
F11	2.60e+01 (1.63e+00) +	3.27e+01 (7.79e+00) +	3.56e+01 (3.88e+00) +	1.65e+01 (2.42e+00) +	**1.05e+01** **(9.38e−01)** /
F12	1.79e+04 (5.24e+003) +	6.53e+04 (4.69e+04) +	3.58e+04 (7.05e+03) +	**3.02e+03** **(2.33e+03)** –	8.21e+03 (3.71e+03) /
F13	**2.06e+00** **(2.15e−01)** =	6.23e+00 (4.88e+00) +	1.94e+00 (1.46e−01) =	3.94e+00 (2.81e−01) =	2.25e+00 (1.89e−01) /
F14	1.28e+01 (2.48e−01) +	1.31e+01 (1.84e−01) +	1.35e+01 (2.09e−01) +	1.26e+01 (2.83e−01) +	**1.21e+01** **(1.74e−01)** /
F15	5.77e+01 (2.76e+01) =	3.04e+02 (1.99e+01) +	2.12e+02 (1.98e+01) +	3.76e+02 (7.83e+01) +	**2.65e+01** **(1.44e+01)** /
F16	1.74e+02 (2.82e+01) +	1.32e+02 (7.60e+01) +	1.22e+02 (9.19e+01) +	8.57e+01 (6.94e+01) +	**1.65e+01** **(3.48e+01)** /
F17	2.46e+02 (4.81e+01) =	1.61e+02 (6.80e+01) –	1.69e+02 (1.02e+02) –	**7.83e+01** **(3.76e+01)** –	2.43e+02 (3.40e+01) /
F18	9.13e+02 (1.42e+00) +	9.07e+02 (1.48e+00) =	8.20e+02 (3.35e+00) =	8.68e+02 (6.23e+01) +	**8.13e+02** **(1.98e+00)** /
F19	9.14e+02 (1.45e+00) +	9.06e+02 (1.24e+00) +	8.21e+02 (3.35e+00) =	8.74e+02 (6.22e+01) =	**8.12e+02** **(1.58e+01)** /
F20	9.14e+02 (3.62e+00) +	9.07e+02 (1.35e+00) +	8.22e+02 (4.17e+00) +	8.78e+02 (6.03e+01) =	**8.09e+02** **(2.00e+01)** /
F21	5.00e+02 (3.39e−13) =	5.00e+02 (4.83e−13) =	8.33e+02 (1.00e+02) +	5.52e+02 (1.82e+02) =	**5.00e+02** **(2.31e−013)** /
F22	9.72e+02 (1.20e+01) =	9.28e+02 (7.04e+01) =	**5.07e+02** **(7.26e+00)** –	9.36e+02 (1.83e+01) =	9.84e+02 (1.51e+01) /

(*continued on next page*)

Table 4 (*continued*)

	CLPSO [31] Mean(Std) Rank	GL-25 [20] Mean(Std) Rank	EPSDE [34] Mean(Std) Rank	SaDE [49] Mean(Std) Rank	EMA Mean(Std) Rank
F23	**5.34e+02** **(2.19e−04)** −	5.34e+02 (4.66e−04) =	8.58e+02 (6.82e+01) +	5.34e+02 (3.57e−03) =	5.34e+02 (5.00e−04) /
F24	2.00e+02 (1.49e−12) +	2.00e+02 (5.52e−11) +	2.13e+02 (1.52e+00) +	2.00e+02 (6.20e−13) +	**2.00e+02** **(5.22e−13)** /
F25	**2.00e+02** **(1.96e+00)** =	2.17e+02 (1.36e−01) +	2.13e+02 (2.55e+00) +	2.14e+02 (2.00e+00) +	2.31e+02 (3.03e+00) /
+/=/−	14/10/1	17/7/1	12/9/4	15/8/2	−

The bold values represent good resutls obtained by optimization methods.

EMA has two good results on Sphere and Rosenbrock, respectively. Therefore, the simulation results indicate that the number of the symbol-objects has some influence on the solving performance of EMA to find the global approximate optimal solutions.

Overall, as observed in Tables 2 and 3, the experimental results show that the proposed EMA can obtain some good results on 10 well-known benchmark numeric optimization functions. Even though the proposed EMA with the different parameters has different results, it is effective to solve this kind of the benchmark functions because the proposed algorithm can improve the relationship of both exploration and exploitation during the solving process. The above-mentioned results indicate that the suitable parameters will help the proposed EMA to find the global optimal solutions of the optimization problems. Moreover, the effectiveness of setting EMA's parameters directly impact on its performance to solve the optimization problems.

5.3. Experiment 2

In this subsection, the proposed EMA is compared with four state-of-the-art evolutionary algorithms on 30 and 50 dimensional versions of the 25 benchmark functions in CEC2005. More specifically, the simulation settings are given in Section 5.3.1. The solution error measure, defined as $f(x') - f(x^*)$, is employed to evaluate the performance of the proposed EMA, where x^* is the global optimum of the benchmark function and x' is the best solution attained by the algorithms in the experiment. The compared algorithms and the proposed EMA are run independently 30 times, respectively, in order to obtain an estimate of the mean solution error (Mean) and its standard deviation (Std). In addition, to evaluate the difference performance of the two evolutionary algorithms, a two-sided Wilcoxon rank sum test is employed at the 5% significance level. The purpose of the hypothesis testing is to judge whether the results obtained by the proposed algorithm differ in the four competitors' or not. Depending on the experimental data, the null hypothesis will or will not be rejected. When the null hypothesis is rejected at the 5% significance level, the algorithm exhibits the superior performance. "+" is used to denote that the proposed EMA exhibits the exceptional performance. On the other hand, a failure to reject the null hypothesis at the 5% significance level indicates that the algorithm exhibits the inferior performance. "−" is used to denote the inferior performance. "=" represents the performance which is not obviously different between the two algorithms. At the bottom of Tables 4 and 5, the total number of the aforementioned statistical significant cases (+/=/−) are given for each pair.

5.3.1. Experimental setup

The parameter settings of all compared algorithms are inherited from the referenced papers [31,20,34,49]. The compared algorithms are simply described as follows.

(1) Test Algorithm 1—CLPSO [31]: it is a variant of particle swarm optimizers (PSO), called the compressive learning particle swarm optimizer. In CLPSO, a novel learning strategy is incorporated into the standard PSO, that all other particles' historical best information is used to update a particle's velocity. This strategy enables the diversity of the swarm to be preserved to avoid the premature convergence.

(2) Test Algorithm 2—GL-25 [20]: it is an improved Genetic Algorithm by implementing three processes to enhance the parent-centric crossover operators.

(3) Test Algorithm 3—EPSDE [34]: it is an ensemble of mutation strategies and control parameters with Differential Evolution. A pool of distinct mutation strategies along with a pool of values for each control parameter coexists throughout the evolution process and competes to produce offspring.

(4) Test Algorithm 4—SaDE [49]: it is a new self-adaptive differential evolution algorithm, in which both trial vector generation strategies and their associated control parameter values are gradually self-adapted by learning from their previous experiences in the generating promising solutions.

Finally, the proposed algorithm is initialized according to Eq. (13).

$$\prod = \begin{cases} \{w_{i,j} \mid 1 \leqslant i \leqslant 8, 1 \leqslant j \leqslant 25\}, \\ [_0[_1]_1, [_2]_2, \ldots, [_{25}]_{25}]_0, \\ \left\{ W \rightarrow \bigcup_{i}^{25} w_i, w_{i,j} \rightarrow w'_{i,j}, \right. \\ w_{i,j} \rightarrow [w_i]_j, [w_i]_j \rightarrow w_{i,j}, \\ \left. [\,]_0 \rightarrow [_0[_1]_1, [_2]_2, \ldots, [_{25}]_{25}]_0, \right\} \end{cases} \tag{13}$$

where $\prod$ denotes a membrane system; $[_0[_1]_1, [_2]_2, \ldots, [_{25}]_{25}]_0$ represents a membrane structure where the skin membrane contains twenty-five elementary membranes. $w_{i,j}$ indicates the jth symbol-object in the ith multiset. $W \rightarrow \bigcup_{i}^{25} w_i$ represents a partition rule. w_i means the ith divided multiset, which consists of eight symbol-objects. The rule of $w_{i,j} \rightarrow w'_{i,j}$ is the rewrite rules. The rule of $w_{i,j} \rightarrow [w_i]_j$ is a communication rule. The rule of $[w_i]_j \rightarrow w_{i,j}$ denotes a dissolution operator. $[\,]_0 \rightarrow [_0[_1]_1, [_2]_2, \ldots, [_{25}]_{25}]_0$ represents a division rule.

5.3.2. Comparison with the four state-of-the-art evolutionary algorithms

In the experiments, the proposed EMA is compared with the four above-mentioned state-of-the-art evolutionary algorithms on 30 dimensional versions of the 25 benchmark functions. The same parameter settings are configured for these four algorithms as in their original literatures. The number of the fitness evaluation in all methods is set as $D \times 10^4 = 300,000$. The experimental results on 30 dimensional benchmark functions are obtained from 30 independent runs with 300,000 fitness evaluations (FES) in Table 4. The best entries in Table 4 are marked in boldface.

In order to compare the solving performance of all methods, the statistical results are calculated and tabulated in Table 4. As observed in Table 4, the proposed EMA obtained good results in some benchmark test functions. The analysis and discussions of the experimental results are given in the following section.

(1) On unimodal function F1–F5, the proposed EMA clearly performs better than competitors on four test functions (F1, from F3 to F5) except F2. EPSDE offers the best performance on F2, which achieves better results than CLPSO, GL-25 and SaDE. SaDE supplies better results than CLPSO, GL-25 and EPSDE on F3. In Table 4, we can draw a conclusion that the outstanding performance of the proposed EMA is attributed to its distinctive membrane structure and the reaction rules. Therefore, EMA has the good exploitation ability in the terms of solving the unimodal functions.

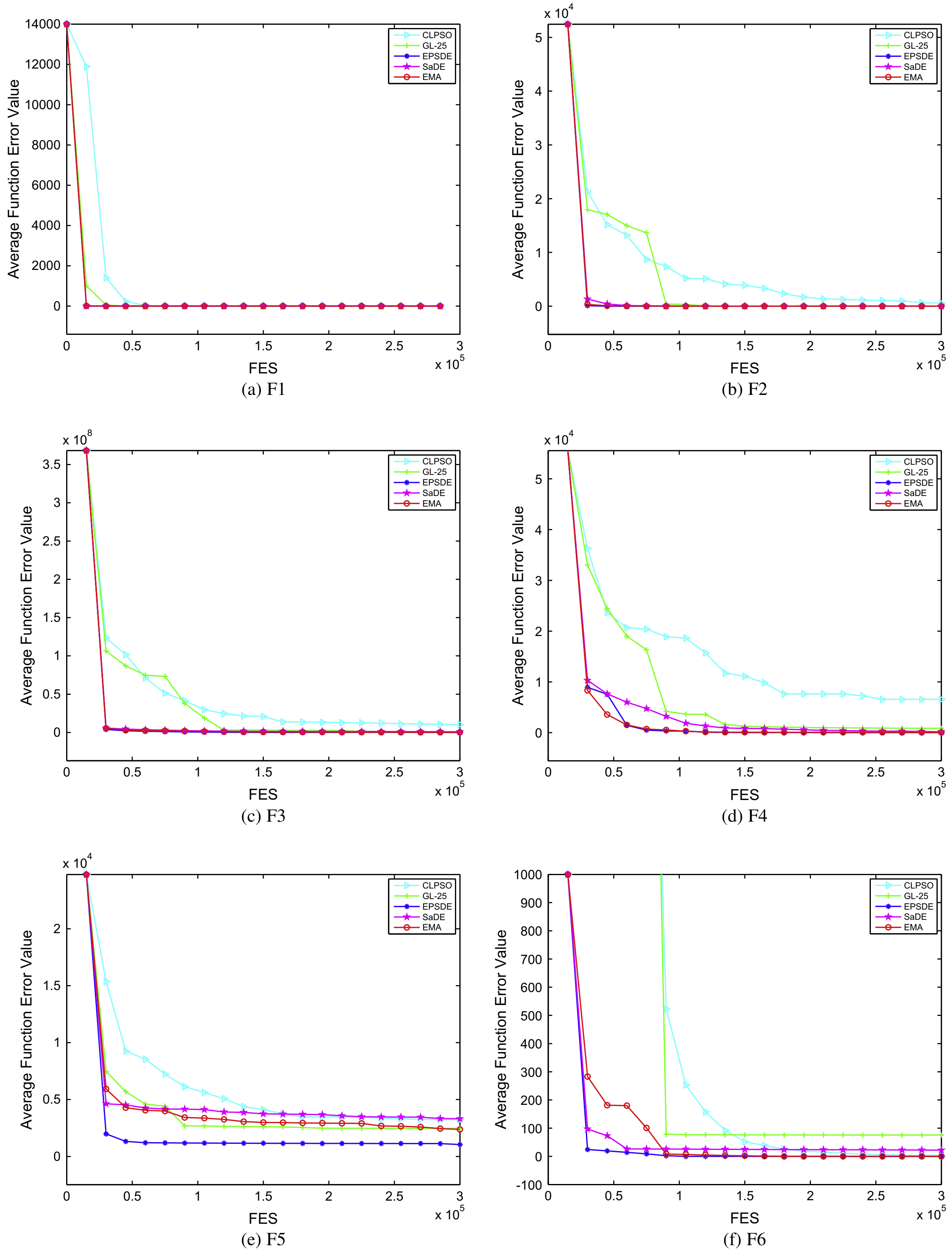
(2) On basic multimodal functions F6–F12, simulation results show that the proposed EMA is not worse than other experimental algorithms on five benchmark test functions (from F7 to F11). SaDE outperforms all the other algorithms on F12. This is attributed to its self-adaptive strategies having both trial vector generation strategies and different control parameters setting. EPSDE can find some good candidate solutions on F6. The proposed algorithm has good results on the multimodal functions with many local optimums, which indicates the proposed algorithm has the pleasurable exploration ability.

(3) On expanded multimodal functions (F13–F14), CLPSO has the best results on F13 which is shifted expanded griewank's plus rosenbrock's function. EMA exhibits a slight good performance in comparison with all the other state-of-the-art algorithms on F14. Therefore, the proposed algorithm has a small advantage to deal with expanded multimodal functions.

(4) On hybrid Composition Functions (F15–F25). Because there are a huge number of local minima from F15 to F25, they are very challenging problems in terms of the solving algorithms. EMA performs either similar or significantly better than the other experimental algorithms on F15, F16, from F18 to F21. SaDE has the best results on F17 and F24, which is attributed to the algorithm's self-adaptive strategies with both trial vector generation strategies and different control parameters setting. EMA is worse than EPSDE on F22. This is attributed to that EPSDE employs an ensemble of mutation strategies and control parameters to maintain the diversity. CLPSO outperforms those algorithms on F23 and F25.

Because the convergence speed of evolutionary algorithms is a very important factor to evaluate their performance for solving the optimization problems, the convergence performance of the proposed algorithm is compared with the four state-of-the-art algorithms over 24 benchmark functions except F7. In Fig. 7, the convergence graphs of the different 30 dimensional benchmark functions are shown to evaluate the convergence performance of all algorithms. In Fig. 7, the proposed EMA has a good convergence performance in comparison with the four algorithms on unimodal functions, basic multimodal functions, and expanded multi-modal functions and hybrid composition functions.

To further investigate the solving ability of the proposed algorithms in the high-dimensional problems, the proposed EMA is evaluated on the 50-dimensional version of the set of benchmark functions in CEC2005. Table 5 summarizes the



(a) F1

(b) F2

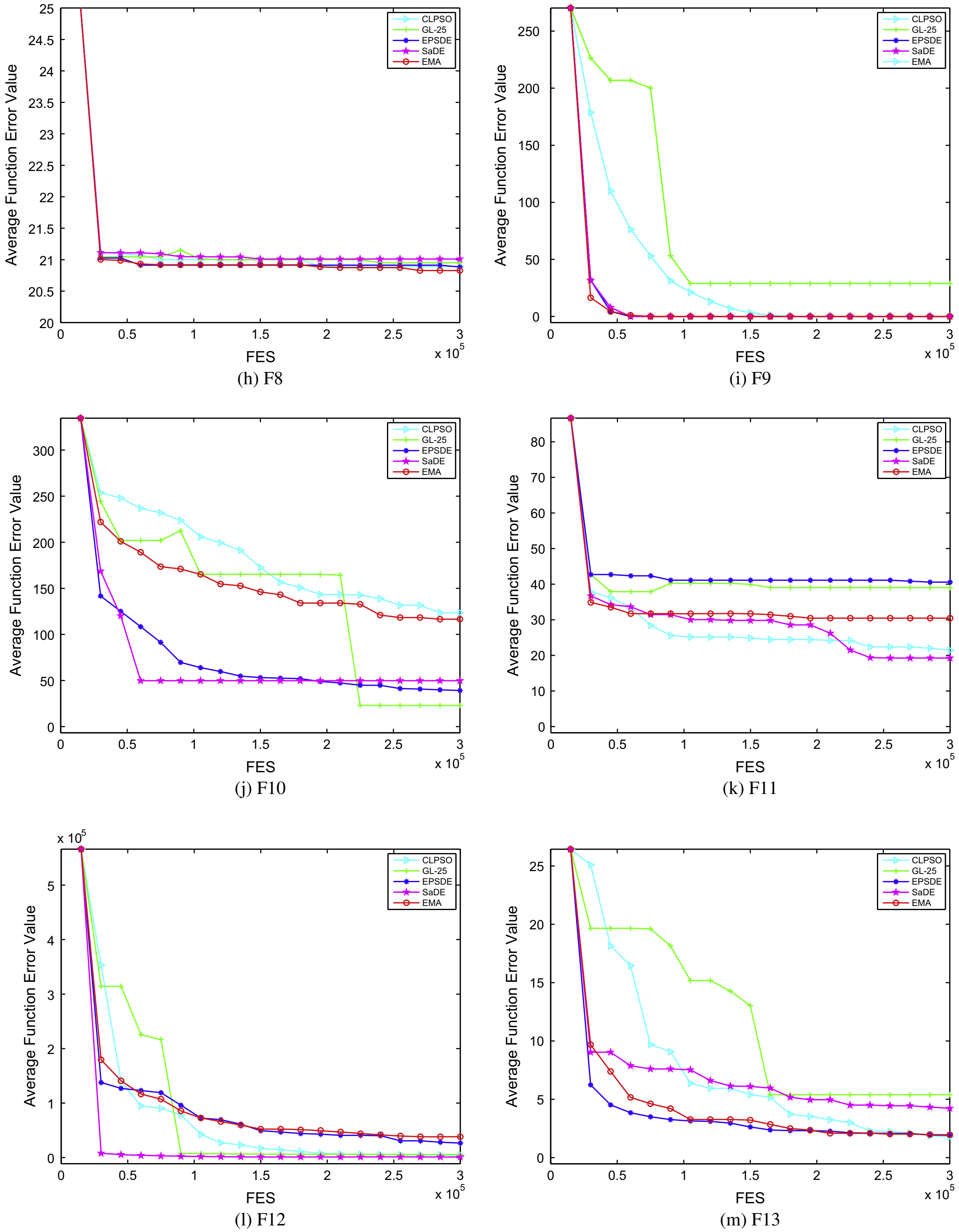
(c) F3

(d) F4

(e) F5

(f) F6

Fig. 7. (a–f), (h–m) Convergence performance of the algorithms in 30-dimensional benchmark functions. (n–y) Convergence performance of the algorithms in 30-dimensional benchmark test functions.



(h) F8

(i) F9

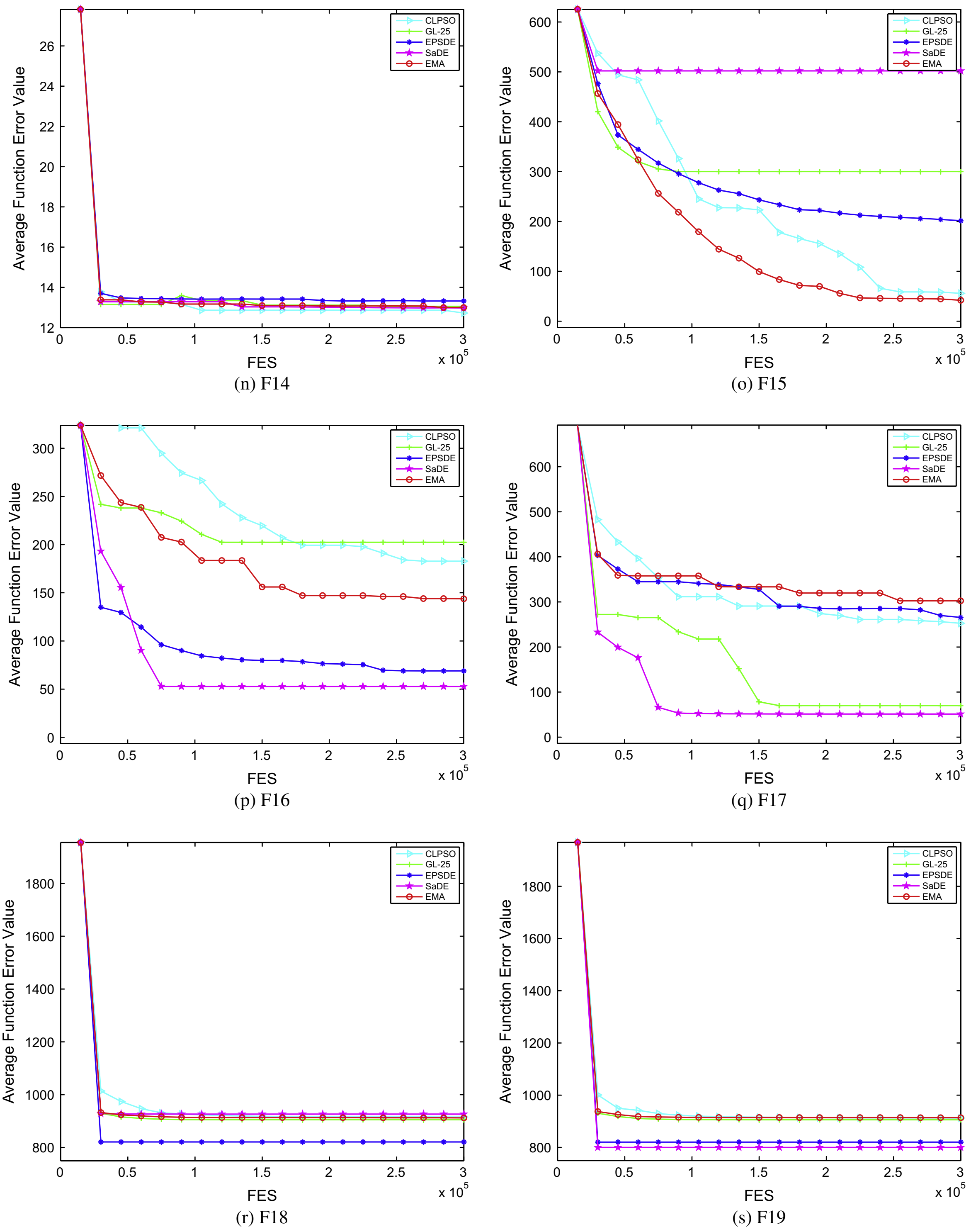
(j) F10

(k) F11

(l) F12

(m) F13

Fig. 7 (*continued*)



(n) F14

(o) F15

(p) F16

(q) F17

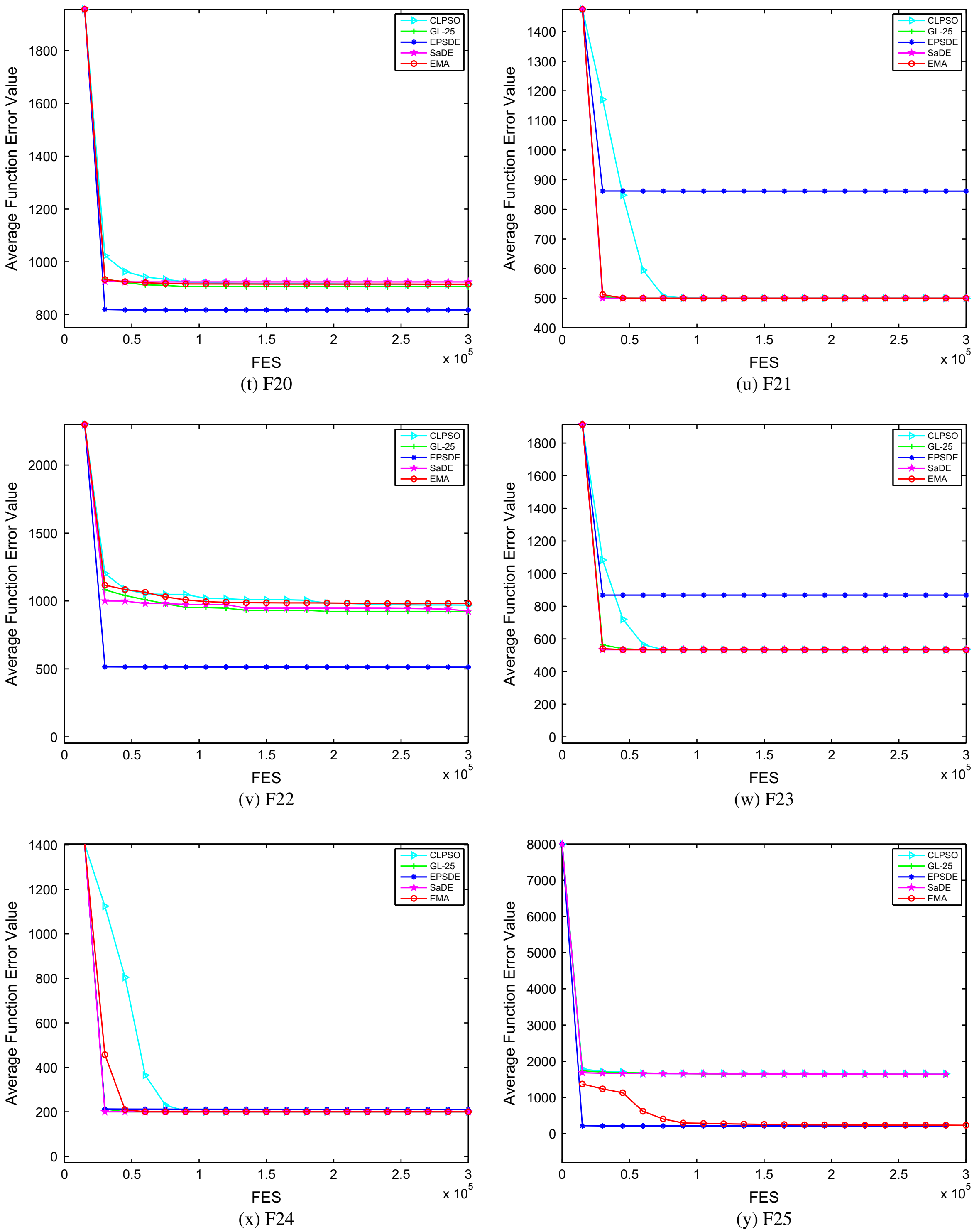
(r) F18

(s) F19

Fig. 7 (*continued*)

experimental averaged results over 30 independent runs with 500,000 FES on 25 benchmark functions. The best entries in Table 5 are marked in boldface.

From Table 5, the proposed EMA presents the best performance on F1, F3-F5, F7-F11, F15, F16, F19 and F23. CLPSO supplies the best performance on F1, F9, F13, F14 and F21. GL-25 offers the best performance on F12. EPSDE shows the best



(t) F20

(u) F21

(v) F22

(w) F23

(x) F24

(y) F25

Fig. 7 (*continued*)

performance on F1, F2, F6, F9, F18, F20 and F22. SaDE provides the best performance on F1, F9, F17, F24 and F25. Therefore, the experimental study indicates that EMA performs good results over 13 benchmark test functions on unimodal functions, multimodal functions, expanded multimodal functions and hybrid composition functions. The outstanding performance of EMA is due to its distinctive membrane structures and reaction rules. The structure of EMA is different from the comparison

Table 5
Experimental results of CLPSO, GL-25, EPSDE, SaDE and EMA on 50-dimensional functions.

	CLPSO [31] Mean(Std) Rank	GL-25 [20] Mean(Std) Rank	EPSDE [34] Mean(Std) Rank	SaDE [49] Mean(Std) Rank	EMA Mean(Std) Rank
F1	0 (0) =	2.11e−21 (1.01e−20) +	0 (0) =	0 (0) +	**0** **(0)** /
F2	9.77e+3 (2.29e+3) +	1.37e+4 (7.59e+2) +	**2.08e−23** **(4.23e−23)** −	9.20e−2 (6.09e−2) =	4.83e−12 (3.82e−10) /
F3	4.66e+7 (7.36e+6) +	5.95e+6 (2.39e+6) +	2.32e+7 (1.67 e+7) +	7.80e+5 (2.50e+5) =	**2.91e+5** **(5.89e+5)** /
F4	3.20e+4 (6.61e+3) +	8.85e+3 (2.13e+3) +	3.81e+3 (3.40e+3) +	3.15e+3 (1.70e+3) +	**8.18e+2** **(2.32e+3)** /
F5	9.57e+3 (4.82e+2) =	5.55e+3 (5.80e+2) +	4.81e+3 (1.10 e+3) =	5.68e+3 (5.94e+2) +	**4.21e+3** **(8.12e+2)** /
F6	4.48e+0 (1.92e+0) +	5.06e+1 (2.20e+1) +	**7.97e−1** **(1.78 e+0)** −	9.81e+1 (4.19e+1) +	2.12e+00 (2.98e+0) /
F7	6.19e+3 (1.11e−11) +	2.11e+1 (3.94e−2) +	9.85e−1 (1.05 e−1) +	6.19e+3 (4.54e−13) +	**7.01e−1** **(1.31e−2)** /
F8	2.12e+1 (4.17e−2) =	5.60e+1 (1.11e+1) =	2.11e+1 (4.44 e−2) =	2.11e+1 (2.66e−2) =	**2.10e+1** **(5.18e−1)** /
F9	0 (0) =	2.13e+2 (1.50e+2) +	0 (0) =	0 (0) +	**0** **(0)** /
F10	2.17e+2 (4.17e+1) +	6.36e+1 (1.16e+1) =	1.60e+2 (1.62 e+1) =	9.41e+1 (9.81e+0) +	**1.58e+1** **(2.61e+1)** /
F11	4.90e+1 (2.49e+0) =	4.32e+4 (1.75e+4) +	7.36e+1 (6.82 e−2) +	4.13e+1 (1.23e+1) =	**2.13e+1** **(8.67e+0)** /
F12	7.37e+4 (2.22e+4) =	**1.40e+4** **(1.03e+4)** −	3.38e+5 (2.17 e+2) +	2.65e+4 (1.47e+4) =	4.32e+4 (3.17e+3) /
F13	**4.04e+0** **(3.48e−1)** −	2.26e+1 (2.46e−1) +	6.28e+0 (4.59 e−1) =	1.11e+1 (6.27e−1) +	8.15e+0 (1.98e−1) /
F14	**2.25e+1** **(2.54e−1)** −	3.68e+2 (7.44e+1) =	2.35e+1 (2.73 e−1) +	2.35e+1 (1.38e−1) +	4.21e+1 (3.94e−1) /
F15	1.35e+2 (7.22e+1) =	1.72e+2 (9.11e+1) =	3.02e+2 (7.79 e+1) +	3.91e+2 (8.82e+1) +	**1.15e+2** **(2.14e+1)** /
F16	2.34e+2 (3.32e+1) +	1.46e+2 (9.94e+1) =	1.97e+2 (1.25 e+2) +	1.32e+2 (1.50e+2) =	**1.01e+2** **(4.21e+2)** /
F17	3.08e+2 (6.47e+1) −	2.38e+2 (7.24e+1) −	2.09e+2 (3.53 e+1) −	**9.06e+1** **(6.07e+1)** −	8.12e+2 (5.12e+2) /
F18	9.48e+2 (6.82e+0) =	9.19e+2 (2.50e+1) =	**8.68e+2** **(5.06 e+1)** −	9.58e+2 (7.50e+1) =	9.12e+2 (4.23e+1) /
F19	9.43e+2 (4.36e+0) +	9.24e+2 (2.45e+0) +	8.46e+2 (2.22 e+0) +	9.58e+2 (1.29e+1) +	**8.33e+2** **(3.11e+1)** /

Table 5 (*continued*)

	CLPSO [31] Mean(Std) Rank	GL-25 [20] Mean(Std) Rank	EPSDE [34] Mean(Std) Rank	SaDE [49] Mean(Std) Rank	EMA Mean(Std) Rank
F20	9.48e+2 (5.94e+0) +	9.18e+2 (2.48e+1) +	**8.43e+2** **(2.83e+0)** **=**	9.67e+2 (5.48e+0) +	8.91e+2 (6.72e+1) /
F21	**5.00e+2** **(5.45e−10)** −	6.02e+2 (4.83 + 1) +	7.30e+2 (2.34 e+0) +	6.37e+2 (3.06e+2) +	5.23e+2 (3.21e+0) /
F22	9.92e+2 (7.56e+0) =	9.62e+2 (6.18e+1) =	**5.20e+2** **(3.17 e+0)** −	9.75e+2 (8.97e+0) =	9.97e+2 (4.21e+1) /
F23	5.39e+2 (2.57e−5) =	5.63e+2 (4.68e−3) +	7.34e+2 (7.61 e+0) +	6.69e+2 (2.92e+2) +	**5.38e+2** **(5.23e−4)** /
F24	4.13e+2 (3.91e+2) +	4.13e+2 (5.32e+2) +	4.12e+2 (4.07 e+2) +	**2.00e+2** **(1.79e−12)** =	2.01e+2 (1.23e−11) /
F25	1.70e+3 (8.57e+0) +	2.01e+2 (1.28e+0) +	2.36e+2 (9.85e+0) =	**1.69e+2** **(6.04e+0)** −	2.82e+2 (8.23e+0) /
+/=/−	11/10/4	17/6/2	12/8/5	14/9/2	−

The bold values represent good resutls obtained by optimization methods.

algorithms, and it can enhance the ability of the algorithm to maintain the diversity of solutions by exploring the unknown search area. The symbol-objects are evolved by invoking the rewrite rules, and the adjacent knowledge of the symbol-objects is utilized, and the sharing information in the skin membrane is implemented among the symbol-objects from the different elementary membrane. In addition, the reaction rules improve the relationship of both the exploitation and exploration during the search process on these test functions. The experimental results can clearly show that the proposed algorithm is better than the four state-of-the-art competition algorithms for some benchmark test functions. These experimental results indicate that the proposed algorithm is effective to solve the global numeric optimization problems.

5.4. Discussions

In this work, based on the membrane computing theory, a new inspired-cell algorithm is proposed to solve the global numerical optimization problems. The proposed algorithm inherits some important ingredients, including the membrane structure, reaction rules and multiset, from membrane systems. The performance and the solving accuracy of the proposed algorithm have been tested on 25 benchmark functions. By observing the experimental data, EMA is suitable to solve global numeric optimization problems. Based on the experimental results and analysis, we can draw some conclusions as follows.

First, several reaction rules, inspired by chemical reactions in the biological cells, are employed to enhance explorative and exploitative abilities during the whole search process. These reaction rules, which are the abstract representation of either chemical reactions or evolving reactions, have made EMA to maintain the diversity of the candidate solutions to escape from the local optimums. By improving the relationship of both exploration and exploitation, the proposed algorithm has a good performance to solve the global numeric optimization problems.

Second, the cellular automata model is introduced in the proposed algorithm because it can simulate the complex systems in any environment. Moreover, the characteristics of the cellular automata model have made the proposed algorithm to perform the adjacent knowledge acquisition in the search process. It is possible to guide the evolution toward a global optimum direction by incorporating the information from the adjacent symbol-objects.

Finally, the proposed EMA is very easy for implementation with a cell-structure. The experimental results show that the proposed EMA is effective to solve global numeric optimization problems. Moreover, most of the strategies can be easily incorporated into the proposed EMA, which can improve its solving performance.

6. Conclusions

Inspired by some basic features and structure of the biological membranes, EMA is devised as a new intelligence evolutionary optimization algorithm to solve the global numeric optimization problems. EMA not only incorporates some reaction rules from membrane systems, but also makes judiciously use of the adjacent knowledge of the symbol-objects. These factors can help EMA to effectively solve the global numeric optimization problems.

A thorough experimentation indicates that EMA can provide the reasonable results in many benchmark test functions. 10 well-known benchmark functions in the first experiment were employed to analyze the performance of EMA with the different parameters. The second experimental study was carried out on 25 global numeric optimization problems used in CEC2005 special session on the real-parameter optimization. Moreover, EMA was compared with the four state-of-the-art evolutionary algorithms. The experimental results indicate that its overall performance is better than the four competitors'. The appealing performance of EMA is attributed to three ingredients from membrane systems and the adjacent knowledge acquisition from the cellular automata model. In summary, the proposed model performs well on most of the global numeric optimization problems.

In our further work, EMA will be applied to combinatorial optimization problems, system identification, multi-objective optimization problems, etc.

Acknowledgments

This work was supported by the Project (61074096) of the National Nature Science Foundation of China. We are very grateful to the editor and the anonymous referees for their constructive comments and helpful suggestions to improve the quality of this paper.

References

[1] O.A. Abdul-Rahman, M. Munetomo, K. Akama, An adaptive parameter binary-real coded genetic algorithm for constraint optimization problems: performance analysis and estimation of optimal control parameters, Inform. Sci. 233 (2013) 54–86.
[2] B. Akay, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Inform. Sci. 192 (2012) 120–142.
[3] V. Aragn, S. Esquivel, C. Coello Coello, A t-cell algorithm for solving dynamic optimization problems, Inform. Sci. 181 (2011) 3614–3637.
[4] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, vol. 1, Oxford University Press, USA, 1999.
[5] I. Boussad, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Inform. Sci. 237 (2013) 82–117.
[6] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (2006) 646–657.
[7] C. Buiu, C. Vasile, O. Arsene, Development of membrane controllers for mobile robots, Inform. Sci. 187 (2012) 33–51.
[8] R. Caponetto, L. Fortuna, S. Fazzino, M. Xibilia, Chaotic sequences to improve the performance of evolutionary algorithms, IEEE Trans. Evol. Comput. 7 (2003) 289–304.
[9] O. Castillo, P. Melin, Optimization of type-2 fuzzy systems based on bio-inspired methods: a concise review, Inform. Sci. 205 (2012) 1–19.
[10] L.N. de Castro, Fundamentals of natural computing: an overview, Phys. Life Rev. 4 (2007) 1–36.
[11] E. Chong, S. Zak, An Introduction to Optimization, Wiley-Interscience, 2004.
[12] I. Ciornei, E. Kyriakides, Hybrid ant colony-genetic algorithm (GAAPI) for global continuous optimization, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 42 (2012) 234–245.
[13] P. Civicioglu, Artificial cooperative search algorithm for numerical optimization problems, Inform. Sci. 229 (2013) 58–76.
[14] M. Daneshyari, G. Yen, Cultural-based multiobjective particle swarm optimization, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 41 (2011) 553–567.
[15] D. DasGupta, An Overview of Artificial Immune Systems and Their Applications, Springer, 1999.
[16] L.N. De Castro, F.J. Von Zuben, Learning and optimization using the clonal selection principle, IEEE Trans. Evol. Comput. 6 (2002) 239–251.
[17] G. Ermentrout, L. Edelstein-Keshet, Cellular automata approaches to biological modeling, J. Theor. Biol. 160 (1993) 97–133.
[18] S. Forrest, A.S. Perelson, L. Allen, R. Cherukuri, Self-nonself discrimination in a computer, in: IEEE Computer Society Symposium on Research in Security and Privacy, 1994. Proceedings., 1994, IEEE, 1994, pp. 202–212.
[19] H. Gao, W. Xu, A new particle swarm algorithm and its globally convergent modifications, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 41 (2011) 1334–1351.
[20] C. Garcia-Martinez, M. Lozano, F. Herrera, D. Molina, A. Snchez, Global and local real-coded genetic algorithms based on parent-centric crossover operators, Eur. J. Oper. Res. 185 (2008) 1088–1113.
[21] S. Ghosh, S. Das, S. Roy, S. Minhazul Islam, P. Suganthan, A differential covariance matrix adaptation evolutionary algorithm for real parameter optimization, Inform. Sci. 182 (2012) 199–219.
[22] D. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Professional, 1989.
[23] M. Gong, L. Jiao, L. Zhang, Baldwinian learning in clonal selection algorithm for optimization, Inform. Sci. 180 (2010) 1218–1236.
[24] W. Gong, Á Fialho, Z. Cai, H. Li, Adaptive strategy selection in differential evolution for numerical optimization: an empirical study, Inform. Sci. 181 (2011) 5364–5386.
[25] L. Huang, I. Suh, A. Abraham, Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants, Inform. Sci. 181 (2011) 2370–2391.
[26] L. Huang, N. Wang, J. Zhao, Multiobjective optimization for controller design, Acta Autom. Sinica 34 (2008) 472–477.
[27] N.K. Jerne, Towards a network theory of the immune system, in: Annales d'immunologie, vol. 125, pp. 373–389.
[28] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global optimiz. 39 (2007) 459–471.
[29] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, 1995. Proceedings., vol. 4, IEEE, 1995, pp. 1942–1948.
[30] D. Kim, A. Abraham, J. Cho, A hybrid genetic algorithm and bacterial foraging approach for global optimization, Inform. Sci. 177 (2007) 3918–3937.
[31] J. Liang, A. Qin, P. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (2006) 281–295.
[32] C. Liu, M. Han, X. Wang, A multi-objective evolutionary algorithm based on membrane systems, in: Fourth International Workshop on Advanced Computational Intelligence (IWACI), 2011, IEEE, 2011, pp. 103–109.
[33] C. Liu, M. Han, X. Wang, A novel evolutionary membrane algorithm for global numerical optimization, in: Third International Conference on Intelligent Control and Information Processing (ICICIP), 2012, IEEE, 2012, pp. 727–732.
[34] R. Mallipeddi, P. Suganthan, Q. Pan, M. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2011) 1679–1696.
[35] B. McGinley, J. Maher, C. O'Riordan, F. Morgan, Maintaining healthy population diversity using adaptive crossover, mutation, and selection, IEEE Trans. Evol. Comput. 15 (2011) 692–714.
[36] D. Molina, M. Lozano, C. Garcìa-Martìnez, F. Herrera, Memetic algorithms for continuous optimisation based on local search chains, Evol. Comput. 18 (2010) 27–63.
[37] O. Montiel, O. Castillo, P. Melin, A. Daz, R. Seplveda, Human evolutionary model: a new approach to optimization, Inform. Sci. 177 (2007) 2075–2098.
[38] P. Moscato, C. Cotta, A. Mendes, Memetic algorithms, in: New Optimization Techniques in Engineering, Springer, 2004, pp. 53–85.

[39] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Haldar, P. Suganthan, A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization, Inform. Sci. 209 (2012) 16–36.

[40] T. Nishida, An application of p-system: a new algorithm for np-complete optimization problems, in: Proceedings of the 8th World Multi-Conference on Systems, Cybernetics and Informatics, vol. 5, pp. 109–112.

[41] M. de Oca, T. Stutzle, K. Van den Enden, M. Dorigo, Incremental social learning in particle swarms, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 41 (2011) 368–384.

[42] L. Pan, J. Wang, H.J. Hoogeboom, Spiking neural p systems with astrocytes, Neural Comput. 24 (2012) 805–825.

[43] L. Pan, X. Zeng, X. Zhang, Time-free spiking neural p systems, Neural Comput. 23 (2011) 1320–1342.

[44] G. Paun, Computing with membranes, J. Comput. Syst. Sci. 61 (2000) 108–143.

[45] G. Paun, Membrane computing, in: Fundamentals of Computation Theory, Springer, 2003, pp. 177–220.

[46] G. Paun, Introduction to membrane computing, in: G. Ciobanu, G. Paun, M. Prez-Jimnez (Eds.), Natural Computing Series, Springer, Berlin Heidelberg, 2006, pp. 1–42.

[47] G. Paun, G. Rozenberg, A guide to membrane computing, Theor. Comput. Sci. 287 (2002) 73–100.

[48] G. Paun, G. Rozenberg, A. Salomaa, The Oxford Handbook of Membrane Computing, Oxford University Press, Inc., 2010.

[49] A.K. Qin, V. Huang, P. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.

[50] B.Y. Qu, J.J. Liang, P.N. Suganthan, Niching particle swarm optimization with local search for multi-modal optimization, Inform. Sci. 197 (2012) 131–143.

[51] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Inform. Sci. 179 (2009) 2232–2248.

[52] R. Shang, L. Jiao, F. Liu, W. Ma, A novel immune clonal algorithm for mo problems, IEEE Trans. Evol. Comput. 16 (2012) 35–50.

[53] R. Storn, K. Price, Differential evolution-simple and efficient heuristic for global optimization over continuous spaces, J. Global optimiz. 11 (1997) 341–359.

[54] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, 2005.

[55] K. Tang, Y. Mei, X. Yao, Memetic algorithm with extended neighborhood search for capacitated arc routing problems, IEEE Trans. Evol. Comput. 13 (2009) 1151–1166.

[56] M. Urselmann, S. Barkmann, G. Sand, S. Engell, A memetic algorithm for global optimization in chemical process synthesis problems, IEEE Trans. Evol. Comput. 15 (2011) 659–683.

[57] R. Ursem, Diversity-guided evolutionary algorithms, Parall. Probl. Solving Nat. PPSN VII, Lect. Notes Comput. Sci. 2439 (2002) 462–471.

[58] F. Valdez, P. Melin, O. Castillo, An improved evolutionary method with fuzzy logic for combining particle swarm optimization and genetic algorithms, Appl. Soft Comput. 11 (2011) 2625–2632.

[59] F. Valdez, P. Melin, O. Castillo, Toolbox for bio-inspired optimization of mathematical functions, Comput. Appl. Eng. Educ. (2011) 1–12.

[60] H. Wang, I. Moon, S. Yang, D. Wang, A memetic particle swarm optimization algorithm for multimodal optimization problems, Inform. Sci. 197 (2012) 38–52.

[61] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (1999) 82–102.

[62] A.R. Yildiz, Optimization of cutting parameters in multi-pass turning using artificial bee colony-based approach, Inform. Sci. 220 (2013) 399–407.

[63] G. Zhang, J. Cheng, M. Gheorghe, A membrane-inspired approximate algorithm for traveling salesman problems, Romanian J. Inform. Sci. Technol. 14 (2011) 3–19.

[64] G. Zhang, J. Cheng, M. Gheorghe, Q. Meng, A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems, Appl. Soft Comput. 13 (2013) 1528–1542.

[65] G. Zhang, M. Gheorghe, C. Wu, A quantum-inspired evolutionary algorithm based on p systems for knapsack problem, Fundam. Inform. 87 (2008) 93–116.

[66] G. Zhang, Y. Li, M. Gheorghe, A multi-objective membrane algorithm for knapsack problems, in: IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2010, IEEE, 2010, pp. 604–609.

[67] J. Zhang, A. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (2009) 945–958.

[68] J. Zhao, N. Wang, A bio-inspired algorithm based on membrane computing and its application to gasoline blending scheduling, Comput. Chem. Eng. 35 (2011) 272–283.