A Bio-Inspired Algorithm based on Membrane Computing for Engineering Design problem

Jian-hua Xiao[1*], Yu-fang Huang[2], Zhen Cheng[3*]

[1] Logistics Research Center, Nankai University
Tianjin 300071, China

[2] College of Mathematics, Southwest Jiaotong University
Chengdu 610031, China

[3] College of Computer Science and Technology, Zhejiang University of Technology
Hangzhou 310023, China

Abstract

Membrane computing is an emergent branch of natural computing, which has been extensively used to solve various NP-complete and intractable problems. In this paper, a bio-inspired algorithm based on membrane computing (BIAMC) is proposed to solve the engineering design problem. BIAMC is designed with the framework and rules of a cell-like P systems, and particle swarm optimization with the neighborhood search. Simulation and experimental results demonstrate that the improved algorithm is valid and outperforms other evolutionary algorithms for engineering design problems.

Keywords: *Engineering Design Problem; Membrane Computing; Particle Swarm Optimization; Neighborhood Search*

1. Introduction

Membrane computing (P systems) was initiated by Paun [1] in 1998, which is a class of new computing model abstracted from the structure and functioning of living cells, as well as from the interactions of living cells in tissues or higher order biological structures. In recent years, many variant of membrane computing models have developed rapidly, and also have turned out that membrane computing has significant potential to be applied to various computationally hard problems in feasible time, such as PSPACE-complete problem [2], 0-1 knapsack problem [3], maximum clique problem [4], Hamilton path problem [5], tripartite matching problem [6].

Inspired from framework and function of living cells, membrane algorithm was firstly proposed by Nishida [7, 8], and used solving the traveling salesman problem. In those membrane algorithms, the nest membrane structure was used together with ideas from genetic algorithms. After Nishida, Huang et al. [9] proposed a membrane algorithm combining the nested membrane structure and conventional genetic algorithm to solve multi-objective numerical optimization problems. Leporati et al. [10] developed a polynomial time membrane algorithm that computed approximate solutions to the instances of min storage. The quantum-inspired evolutionary algorithm based on P systems was also developed to solve the knapsack problem [11], the satisfiability problem [12] and the radar emitter signals problem [13]. Zhao et al. [14] used a bio-inspired algorithm based on membrane computing to optimize gasoline blending scheduling. In 2012, Yang et al. [15] developed a P systems based hybrid optimization algorithm to estimate the parameters of FCCU reactor regenerator model. Xiao et al. [16] used the membrane evolutionary algorithm to select the DNA sequences in DNA Computation.

In the paper, a bio-inspired based on membrane computing with neighborhood search strategy is proposed to solve the engineering design problems (EDP). The rest of this paper is organized as follows. In section 2, a hybrid membrane evolutionary algorithm will be proposed. The simulation results and analyses will be given in section 3. Section 4 is the conclusion and further remark.

2. The Bio-inspired Algorithm based on Membrane Computing

2.1 Cell-like P systems

P systems can be classified into the cell-like P systems, the tissue-like P systems and neural-like P systems [17]. In a cell-like P systems, the membrane structure is a hierarchical arrangement of membranes

* Corresponding author (jhxiao@nankai.edu.cn)

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

embedded in the skin membrane. A membrane without any other membranes inside is said to be elementary membrane. Each membrane has a region containing a multiset of objects and a set of evolutionary rules. The multisets of objects evolve in each region and move from a region to a neighboring one by applying the rules in a nondeterministic and maximally parallel way. The membrane structure of a cell-like P systems is shown in Fig. 1.

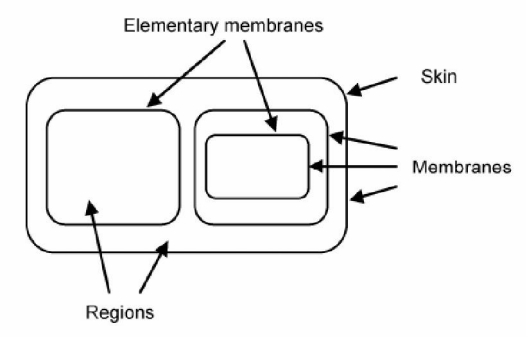

Fig. 1 The membrane structure

The membrane structure of a cell-like P system can be formally defined as follows [16].

$$\Pi = (O, T, u, s_1, \cdots, s_n, R_1, \cdots, R_n, i_0)$$

where:

(i) O is the alphabet of objects;

(ii) T is the output alphabet, $T \subseteq O$;

(iii) μ is a membrane structure consisting of n membranes, and the membranes labeled with $1, 2, \cdots, n$; n is called the degree of the system $\prod$;

(iv) s_i $(1 \le i \le n)$ are strings which represent multisets over O associated with the region $1, 2, \cdots, n$ of μ.

(v) R_i $(1 \le i \le n)$ are the evolution rules over O^*, R_i is associated with region i of μ, and it is of the following forms.

(a) $[_i s_1 \to s_2]_i$, where $i \in \{1, 2, \cdots, n\}$, and $s_1, s_2 \in O^*$.

(Evolution rules: a rule of this type works on a string objects by the local search algorithm or various evolutionary operator, and the new strings object are created in region i.)

(b) $s_1[\ ]_i \to [\ _i s_2]_i$, where $i \in \{1, 2, \cdots, n\}$, and $s_1, s_2 \in O^*$.

(Send-in communication rules; a string object is send in the region i.)

(c) $[_i s_1]_i \to [\ ]_i s_2$, where $i \in \{1, 2, \cdots, n\}$, and $s_1, s_2 \in O^*$.

(Send-out communication rules; a string object is sent out of the region i.)

(vi) i_0 is the output membrane.

P systems, regarded as a model of computation, also is called as membrane algorithm, which is composed of a series of computing steps between configurations. Each computation starts from the initial configuration, and halts when there are no more rules applicable in any region. In the computing process, the system will go from one configuration to a new one by applying the rules associated to regions in a non-deterministic and maximally parallel manner. The result of the computation is obtained in region i_0. The basic pseudocode of the membrane evolutionary algorithm is shown in Fig. 2. For more details about the cell-like P systems, please refer to [17].

```
Begin
    Initialize the membrane structure and parameters;
    gen=0;
    While (Not termination condition) do
        Execute the evolution rules in all elementary membranes;
        Calculate the fitness by the fitness function;
        Execute the communication rules;
        Record the current best solution;
        gen=gen+1;
    end while
End begin
```

Fig. 2 The pseudocode of the membrane algorithm

2.2 The basic idea of PSO

Particle swarm optimization (PSO) is an effective optimization method that belongs to the category of swarm intelligence methods, originally developed by Kennedy and Eberhart [18]. Since the simple concept, easy implementation and effectiveness, PSO has become popular in evolutionary optimization community. In recently years, various heuristic algorithms have been developed to solve hard benchmark and real-world optimization problems, such as the traveling salesman problem (TSP) [19], the production-planning problem [20] and the economic load dispatch [21].

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

In PSO, each particle in the swarm is attracted by its previous best particle (*pbest*) and the global best particle (*gbest*), and is moved toward the optimal point by adding a velocity with its position. For a search problem in N-dimensional space, the velocity $v_{ij}(t)$ and position $x_{ij}(t)$ of the j-th dimension of the i-th particle are updated as follows in t-th generation.

$$v_{ij}(t+1) = v_{ij}(t) + c_1 \times rand(\,) \times (pbest_{ij}(t) - x_{ij}(t)) + c_2 \times Rand(\,) \times (gbest_{j}(t) - x_{ij}(t)) \quad (1)$$

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

where $pbest_{ij}(t)$ represents the best location in the search space ever visited by particle i and $gbest_{j}(t)$ is the best location discovered so far; c_1 and c_2 are the acceleration constants; $rand(\,)$ and $Rand(\,)$ are the uniform random value in the range [0, 1]. The flow chart of the basic particle swarm optimization is shown in Fig. 3.

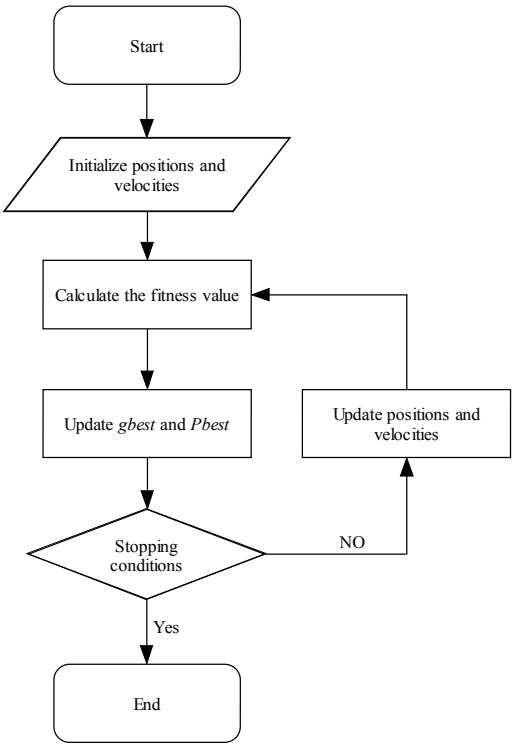

Fig.3 The flow chart of the particle swarm optimization

In PSO, proper selection of c_1 and c_2 is crucial to improve the search ability during the optimization process. However, it is difficult to obtain the optimal values because the different optimization problems have the different values. In [22], Krohling and Coelho implemented the Gaussian probability distribution to generate the accelerating coefficients of PSO, and got good performance. In the paper, the velocity equation of PSO is modified as follows.

$$v_{ij}(t+1) = |randn(\,)| \times (pbest_{ij}(t) - x_{ij}(t)) + |Randn(\,)| \times (gbest_{j}(t) - x_{ij}(t)) \quad (3)$$

where $|randn(\,)|$ and $|Randn(\,)|$ are positive random numbers generated using $abs(N(0,1))$.

3.3 The bio-inspired algorithm based on computing for EDP

In this subsection, the bio-inspired algorithm based on membrane computing will be proposed by using the concepts and mechanism of both P systems and PSO with the neighborhood search. In this algorithm, a P systems-like framework is introduced to arrange objects and evolution rules, and two neighborhood searches are employed to enhance the ability of exploration and exploitation. The procedure of the hybrid membrane evolutionary algorithm based on PSO is described as follows.

Step 1: Initialize membrane structure and $X(t)$, $V(t)$. Specify one level membrane structure $[_0\,[_1\,]_1, [_2\,]_2, \cdots, [_n\,]_n\,]_0$ which composed of a skin membrane denoted by 0 and n regions inside the skin membrane; randomly generate m individuals in each elementary membrane. Multisets are initialized as follows:

$$\begin{aligned} s_0 &= \lambda \\ s_1 &= b_{1,1} b_{1,2} b_{1,3} \cdots b_{1,m} \\ s_2 &= b_{2,1} b_{2,2} b_{2,3} \cdots b_{2,m} \\ s_3 &= b_{3,1} b_{3,2} b_{3,3} \cdots b_{3,m} \\ &\cdots \cdots \\ s_n &= b_{n,1} b_{n,2} b_{n,3} \cdots b_{n,m} \end{aligned}$$

where m is the population size of each elementary membrane, and $b_{i,j}$, $1 \le i \le n$, $1 \le j \le m$ is an individual.

Step 2: Evolution rules in each of the region 1 to n are implemented. The particle swarm optimization (PSO) based on Gaussian distribution will be performed in each elementary membrane simultaneously.

Step 3: Implement the send-out communication rules, the strings are sent to skin membrane from each elementary membrane;

Step 4: To improve the disadvantage of the premature convergence problem, the local and global

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

neighborhood searches are implemented in the skin membrane to improve the ability of exploration and exploitation. The equations of local neighborhood search are defined as follows [23].

$$LX_i = r_1 \cdot X_i + r_2 \cdot pbest_i + r_3(X_c - X_d) \quad (4)$$

$$LV_i = V_i \quad (5)$$

where X_i is the position vector of the i-th particle, $pbest_i$ is the previous best particle of P_i; X_c and X_d are the position vectors of two random particles in the k-neighborhood radius of P_i, $c, d \in [i-k, i+k] \wedge c \neq d \neq i$; r_1, r_2 and r_3 are three uniform random numbers within (0,1), and $r_1 + r_2 + r_3 = 1$.

The equations of global search are shown as follows [23].

$$GX_i = r_4 \cdot X_i + r_5 \cdot gbest + r_6 \cdot (X_e - X_f) \quad (6)$$

$$GV_i = V_i \quad (7)$$

where $gbest$ is the global best particle, X_e and X_f are the position vectors of two random particles chosen from the entire swarm, $e, f \in [1, N], e \neq f \neq i$; r_4, r_5 and r_6 are three uniform random numbers in [0, 1], and $r_4 + r_5 + r_6 = 1$.

Step 5: Calculate the fitness of each string object by fitness function, and save the current best strings;

Step 6: Implement the send-in communication rules between the skin membrane and each elementary membrane simultaneously. The detail description is as follows.

(i) First, the best strings and $m-1$ strings with the worst fitness are sent to the elementary membrane 1;

(ii) Subsequently, in the remaining strings, the current best strings and $m-1$ strings with the worst fitness are sent to the elementary membrane 2;

(iii) The above process is executed constantly until the strings from the skin membrane back to each region;

Step 7: If the stopping condition is met, then output the results; otherwise, return to step 2.

4. Experimental Results

In this section, we will carry out numerical simulation based on several well-known engineering design problems to test the effectiveness and efficiency of the proposed algorithms.

4.1 Parameters Setting

In our experiment, the bio-inspired algorithm based on membrane computing for the engineering design problems is executed with Matlab 7.0. The parameters of the algorithm used are shown in Table 1. For each engineering design problem, 30 independent runs are carried out.

Table 1: Parameters used in the proposed algorithm

Parameters	Value	Meaning
$iter_{\max}$	500	The maximum number of iterations
N_s	8	The swarm size of each elementary membrane
N_m	20	The number of the elementary membrane
N_R	30	The run times independently for each test function
ε	0.0001	Tolerated equality constraint violation
k	5	The neighborhood radius

4.2 Welded beam design problem (Example 1)

The welded beam design problem is taken from [24], which is designed for minimum cost subject to constraints on shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c), end deflection of the beam (δ), and side constraints. There are four design variables ($h(x_1)$, $l(x_2)$, $t(x_3)$ and $b(x_4)$) as shown in Fig. 4.

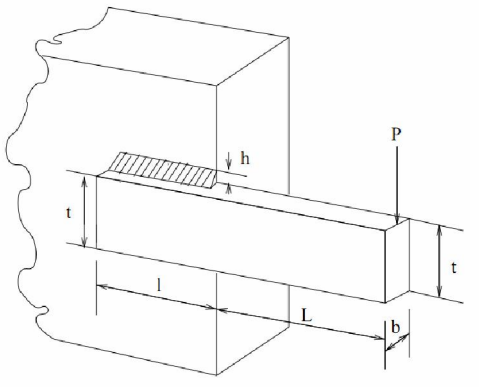

Fig.4 The welded beam design problem (Example 1)

The problem can be mathematically formulated as follows [25].

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

Minimize:

$$f(X) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \quad (8)$$

Subject to:

$$g_1(X) = \tau(X) - 13600 \le 0 \quad (9)$$

$$g_2(X) = \sigma(X) - 30000 \le 0 \quad (10)$$

$$g_3(X) = x_1 - x_4 \le 0 \quad (11)$$

$$g_4(X) = 0.10471x^2_1 + 0.04811x_3x_4(14 + x_2) - 5 \le 0 \quad (12)$$

$$g_5(X) = 0.125 - x_1 \le 0 \quad (13)$$

$$g_6(X) = \delta(X) - 0.25 \le 0 \quad (14)$$

$$g_7(X) = 6000 - P_c(X) \le 0 \quad (15)$$

where:

$$\tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2} \quad (16)$$

$$\tau' = \frac{6000}{\sqrt{2}x_1x_2} \quad (17)$$

$$\tau'' = \frac{MR}{J} \quad (18)$$

$$M = 6000(14 + \frac{x_2}{2}) \quad (19)$$

$$R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2} \quad (20)$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + (\frac{x_1 + x_3}{2})^2\right]\right\} \quad (21)$$

$$\sigma(X) = \frac{504000}{x_4x_3^2} \quad (22)$$

$$\delta(X) = \frac{2.1952}{x_3^3x_4} \quad (23)$$

$$P_c(X) = 64746.022(1 - 0.0282346x_3)x_3x_4^3 \quad (24)$$

Various methods were proposed to solve this problem, such as co-evolutionary particle swarm optimization (CPSO) [25], GA-based co-evolution model [26], and the culture algorithm (CA) [27]. In the paper, the variable regions are defined as follows: $0.1 \le x_1, x_4 \le 2$, $0.1 \le x_2, x_3 \le 10$. The comparison of the experiment results for the welded beam problem shown in Table 2, where the highlighted boldface is the better results. Table 3 also presents the statistical results.

Table 2: Comparison of the best solution for the welded beam design problem

Design Variables	Our Algorithm	Deb [28] (1991)	Coello [29] (2000)	Coello [26] (2002)	CA [27] (2012)
x_1	0.205675	0.248900	0.208800	0.205986	0.202369
x_3	3.470993	6.173000	3.420500	3.471328	3.544214
x_3	9.040587	8.178900	8.997500	9.020224	9.048210
x_4	0.205728	0.253300	0.210000	0.206480	0.205723
$g_1(X)$	-2.640579	-5758.603777	-0.337812	-0.074092	-12.839796
$g_2(X)$	-26.062354	-255.576901	-353.902604	-0.266227	-1.247467
$g_3(X)$	-0.000053	-0.004400	-0.001200	-0.000495	-0.001498
$g_4(X)$	-3.432268	-2.982866	-3.411865	-3.430043	-3.429347
$g_5(X)$	-0.080675	-0.123900	-0.083800	-0.080986	-0.079381
$g_6(X)$	-0.235559	-0.234160	-0.235649	-0.235514	-0.235536
$g_7(X)$	-1.588096	-4465.270928	-363.232384	-58.666440	-11.681355
$f(X)$	**1.725507**	2.433116	1.748309	1.728226	1.728024

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

Table 3: Statistical results of different methods for the welded beam design problem

Optimization Method	*Best*	*Mean*	*Worst*	*Std.*
Our Algorithm	**1.725507**	**1.726594**	**1.728121**	**7.246e-04**
CA (2012)	1.728024	N/A	N/A	N/A
CPSO (2007)	1.728024	1.748831	1.782143	0.012926
Coello (2002)	1.728226	1.792654	1.993408	0.074713
Coello (2000)	1.748309	1.771973	1.78535	0.011220
Deb (1991)	2.433116	N/A	N/A	N/A

From Table 2, it can be seen that the best feasible solution found by our algorithm is better than other evolutionary algorithms. From Table 3, it is also clear that the proposed algorithm performs better than other methods according to all statistical values for the welded beam design problem. Furthermore, the proposed algorithm provides smaller standard deviation in 30 independent runs.

4.3 Pressure vessel design problem (Example 2)

The objective of the pressure vessel design problem is to minimize the total cost, including the cost of the material, forming and welding. Four variables (thickness of the shell $T_s(x_1)$, thickness of the head $T_h(x_2)$, inner radius $R(x_3)$ and length of cylindrical section of the vessel $L(x_4)$, not including the head) are shown in Fig. 5.

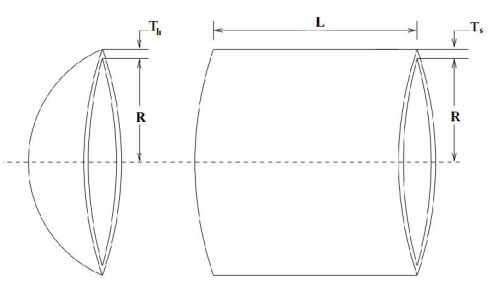

Fig.5 The pressure vessel design problem (Example 2)

The problem can be mathematically formulated as follows [25]:

Minimize:

$$f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (25)$$

Subject to:

$$g_1(X) = -x_1 + 0.0193x_3 \le 0 \quad (26)$$

$$g_2(X) = -x_2 + 0.00954x_3 \le 0 \quad (27)$$

$$g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \quad (28)$$

$$g_4(X) = x_4 - 240 \le 0 \quad (29)$$

In recent years, various approaches are used to optimize the pressure vessel design problem, such as feasibility-based co-evolutionary PSO [29], genetic adaptive search [30], and quantum-behaved PSO [31]. In the paper, the variable regions are defined as follows: $1 \le x_1, x_4 \le 99$, $10 \le x_2, x_3 \le 200$. Table 4 and Table 5 present the comparison and statistical results of the experiment, respectively.

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

Table 4: Comparison of the best solution for the welded beam design problem

Design Variables	Our Algorithm	Deb [30] (1997)	Coello [29] (2000)	Coello [26] (2002)	CPSO [25] (2007)	QPSO [31] (2010)
x_1	0.827599	0.937500	0.812500	0.812500	0.8125	0.8125
x_3	0.413794	0.500000	0.437500	0.437500	0.437500	0.4375
x_3	42.703137	48.329000	40.323900	42.097398	42.091266	42.0984
x_4	169.965254	112.679000	200.00000	176.650405	176.746500	176.0984
$g_1(X)$	-0.003429	-0.004750	-0.034324	-0.000020	-0.000139	-8.7999e-07
$g_2(X)$	-0.006406	-0.038941	-0.052847	-0.035891	-0.035949	-3.5881e-02
$g_3(X)$	-3897.842439	-3652.87638	-27.105845	-27886075	-116.382700	-0.2179
$g_4(X)$	-70.034746	-127.321000	-40.00000	-63.345953	-63.253500	-63.3628
$f(X)$	**6029.181059**	6410.3811	6288.7445	6059.9463	6061.0777	6059.7208

Table 5: Statistical results of different methods for the welded beam design problem

Optimization Method	*Best*	*Mean*	*Worst*	*Std.*
Our Algorithm	**6029.1811**	**6136.7807**	**6288.2630**	56.76628
QPSO (2010)	6059.7209	6839.9326	8017.2816	479.2671
CPSO (2007)	6061.0777	6147.1332	6363.8041	86.4545
Coello (2002)	6059.9463	6177.2533	6469.3220	130.9297
Coello (2000)	6288.7445	6293.8432	6308.1495	**7.4133**
Deb (1997)	6410.3811	N/A	N/A	N/A

From Table 4 and Table 5, it can be observed that the proposed algorithm is robust and find solutions which are better than other evolutionary algorithm. Furthermore, BIAMC finds a better "*Best*", "*Mean*" and "*Worst*" results, except for "*Std*" result.

4.4 Tension/compression string problem (Example 3)

In this case, we will consider the design of a tension/compression spring to be designed for minimum weight subject to constraints on minimum deflection, shear stress, surge frequency, limits on the outside diameter, and on design variables [31]. Three variables (mean coil diameter $D(x_1)$, the wire diameter $d(x_2)$, the number of active coils $N(x_3)$) are shown in Fig. 6.

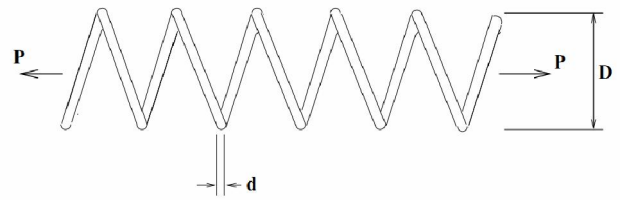

Fig.6 The tension/compression string problem (Example 3)

The problem can be mathematically formulated as follows.

Minimize:

$$f(X) = (x_3 + 2)x_1x_2^2 \tag{30}$$

Subject to:

$$g_1(X) = 1 - \frac{x_1^3 x_3}{71785x_2^4} \le 0 \tag{31}$$

$$g_2(X) = \frac{4x_1^2 - x_1x_2}{12566(x_1x_2^3 - x_2^4)} + \frac{1}{5108x_2^2} - 1 \le 0 \tag{32}$$

$$g_3(X) = 1 - \frac{140.45x_2}{x_1^2x_3} \le 0 \tag{33}$$

$$g_4(X) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \tag{34}$$

where $0.05 \le x_1 \le 2$, $0.25 \le x_2 \le 1.3$, and $2 \le x_3 \le 15$. Table 6 and Table 7 present the comparison and statistical results of the experiment, respectively.

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

Table 6: Comparison of the best solution for the tension/compression string problem

Design Variables	Our Algorithm	Coello [29] (2000)	Coello [26] (2002)	QPSO [31] (2010)	CA [27] (2012)
x_1	0.051988	0.051480	0.051989	0.051515	0.015728
x_3	0.363936	0.351661	0.363965	0.352529	0.357644
x_3	10.892225	11.632201	10.890522	11.538862	11.244543
$g_1(X)$	-0.000021	-0.002080	-0.000013	-4.8341e-5	-0.00845
$g_2(X)$	-0.000024	-0.000110	-0.000021	-3.5774e-5	-1.2600e-5
$g_3(X)$	-4.061229	-4.026318	-4.061338	-4.0455	-4.051300
$g_4(X)$	-0.722717	-4.02318	-0.722698	-0.73064	-0.727090
$f(X)$	0.012681	0.0127048	0.0126810	**0.012665**	0.0126747

Table 7: Statistical results of different methods for the tension/compression string problem

Optimization Method	*Best*	*Mean*	*Worst*	*Std.*
Our Algorithm	0.012681	**0.012687**	**0.012694**	**2.9393e-06**
CA (2012)	0.0126747	N/A	N/A	N/A
QPSO (2010)	**0.012669**	0.013854	0.018127	0.001341
CPSO (2007)	0.0126747	0.012730	0.012924	5.1985e-05
Coello (2002)	0.0126810	0.0127420	0.012973	5.900e-05
Coello (2000)	0.0127048	0.012769	0.012822	3.939e-05

From Table 6 and Table 7, it can be seen that the best feasible solution found by QPSO is better than the best solutions found by our algorithm. Nevertheless, our proposed algorithm performs better than other evolutionary algorithms for "Mean" and "Worst" results. Moreover, the standard deviation of our algorithm is very small in 30 independent runs.

5. Conclusion

In this paper, a bio-inspired algorithm based on membrane computing was presented to solve the constrained engineering design optimization problems. The method proposed combined the neighborhood search strategy in to particle swarm optimization to improve the exploration and exploitation the ability of the membrane algorithm. By comparing with other evolutionary algorithms, our algorithm can get the good solutions for some well-known engineering design problems.

However, we have some further work to do. The dynamic membrane evolutionary algorithm based on the divide rule of P systems will be considered. We will apply the improved algorithm to solve other optimization hard problems and the real engineering design problems.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 60903105, 70673046, 61100055 and 61202204), and the Fundamental Research Funds for the Central Universities (Grant No. NKZXB1110).

References

[1] G. H. Paun. "Computing with Membranes". Technical Report. Finland: Turku Center for Computer Science, 1998

[2] A. Alhazov, C. Martin-Vide, L. Q. Pan. "Solving a PSPACE-complete problem by recognizing P systems with restricted active membranes". Fundamenta Informaticae, 2003, 58: 67-77

[3] L. Q. Pan, C. Martin-Vide. "Solving multidimensional 0-1 knapsack problem by P systems with input and active

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.

membranes". Journal of Parallel and Distributed Computing, 2005, 65: 1578-1584
[4] M. Garcia-Arnau, D. Manrique, A. Rodriguez-Paton, et al. "A P system and a constructive membrane-inspired DNA algorithm for solving the maximum clique problem". BioSyetems, 2007, 2(5): 1-11
[5] L. Q. Pan, A. Alhazov. "Solving HPP and SAT by P systems with active membrane and separation rules". Acta Inform, 2006, 43: 131-145
[6] Y. Y. Niu, L. Q. Pan, M. J. Perez-Jimenez, et al. "A Tissue P Systems Based Uniform Solution to Tripartite Matching Problem". Fundamenta Informaticae, 2011, 109: 1-10
[7] T. Y. Nishida. "An Application of P-System: A New Algorithm for NP-Complete Optimization Problems". The 8th World Multi-Conference on Systems, Cybernetics and Informatics, 2004, pp. 109-112
[8] T. Y. Nishida. "An approximate algorithm for NP-complete optimization problems exploiting P systems". The Brainstorming Workshop on Uncertainty in Membrane Computing, 2004, pp185-192
[9] L. Huang, X. X. He, N. Wang, et al. "P systems based multi-objective optimization algorithm". Progress in Nature Science, 2007, 17, 458-465
[10] A. Leporati, D. Pagani. "A membrane algorithm for the min storing problem". Proceedings of Membrane Computing, International Workshop, WMC7, Leiden, The Netherlands, 2006, pp397-416
[11] G. X. Zhang, M. Gheorghe, C. Z. Wu. "A quantum-inspired evolutionary algorithm based on P systems for knapsack problem". Fund Inform, 2008, 87: 93-116
[12] G. X. Zhang, C. X. Liu, M. Gheorghe, et al. "Solving satisfiability problems with membrane algorithm". Fourth International Conference on Bio-Inspired Computing, 2009, 1-8
[13] G. X. Zhang, C. X. Liu, H. N. Rong. "Analyzing radar emitter signals with membrane algorithms". Math Comput Model, 2010, 52: 1997-2010
[14] J. H. Zhao, N. Wang. "A bio-inspired algorithm based on membrane computing and its application to gasoline blending scheduling". Computers and Chemical Engineering, Computers and Chemical Engineering, 2011, 35: 272-283
[15] S. P. Yang, N. Wang. "A P systems based hybrid optimization algorithm for parameter estimation of FCCU reactor regenerator model". Journal of Chemical Engineering, 2012, 508-518
[16] G. Paun. "Tracing some open problems in membrane computing". Romanian Journal of Information Science and Technology, 2007, 10: 303-314
[17] J. H. Xiao, X. Y. Zhang, J. Xu. "A membrane evolutionary algorithm for DNA sequence design in DNA computing". Chinese Science Bulletin, 2012, 57(6): 698-706
[18] J. Kennedy, R. C. Eberhart. "Particle Swarm Optimization". The IEEE International Conference on Neural Networks, Perth, Australia, 1995, 1942-1948
[19] K P Wang, L Huang, C G Zhou, et al. "Particle swarm optimization for traveling salesman problem". The 2nd ICMLC, 2003, pp. 1583-1585
[20] Y. Y. Chen, J. T. Lin. "A modified particle swarm optimization for production planning problems in the TFT Array process". Expert Systems with Applications, 36 (2009), 12264-12271
[21] L. D. S. Coelho, V C Mariani. A novel chaotic particle swarm optimization approach using Henon map and implicit filtering local search for economic load dispatch". Chaos, Solitons & Fractals, 2009, 39: 510-518
[22] R. A. Krohling, L. S. Coelho. "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems". IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics, 2006, 36(6): 1407-1416
[23] H. Wang, H. Sun, C. H. Li, et al. "Diversity enhanced particle swarm optimization with neighborhood search". Information Sciences, 2013, 223: 119-135
[24] S. S. Rao. "Engineering Optimization". Wiley, New York, 1996.
[25] Q. He, L. Wang. "An effective co-evolutionary particle swarm optimization for constrained engineering design problems". Engineering Applications of Artificial Intelligence, 2007, 20: 89-99
[26] C. A. C. Coello. "Use of a self-adaptive penalty approach for engineering optimization problem". Computer in Industry, 2000, 4(2): 113-127
[27] X. S. Yan, W. Li, W. Chen, et al. "Cultural algorithm for engineering design problem". International Journal of Computer Science Issues, 2012, 9(6): 53-61
[28] K. Deb. "Optimal design of a welded beam via genetic algorithms". AIAA Journal, 1991, 29(11): 2013-2015
[29] C. A. C. Coello. "Theoretical and numerical constrained-handling techniques used with evolutionary algorithms: a survey of the state of the art". Computer Methods in Applied Mechanics and Engineering, 2002, 191(11-12): 1245-1287
[30] K. Deb. "GeneAS: a robust optimal design technique for mechanical component design". Evolutionary Algorithm in Engineering Applications. Berlin: Springer-Verlag, 1997
[31] L. S. Coelho. "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems". Expert Systems with Application, 2010, 27, pp. 1676-1683

Jian-hua Xiao is received the Ph. D. degree in System Engineering from Huazhong University of Science and Technology, Wuhan, China, in 2008. He is currently a lecture at Nankai University, Tianjin, China. His research interests include combinatorial optimization, Bio-inspired computation and logistics optimization etc.

Yu-fang Huang received the Ph.D. degree from Huazhong University of Science and Technology in 2010. She is now on the Post Doctor research of the Department of Control Science and Engineering at Huazhong University of Science and Technology from 2011. Currently, she is a teacher of the College of Mathematics at Southwest Jiaotong University. Her research interests include combinatorial optimization and molecular computation.

Zhen Cheng received the PH.D degree from Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2010.She is currently a lecturer at Zhejiang University of Technology, Hangzhou, China. Her research interests are combinatorial optimization, Bio-inspired computation.

Copyright (c) 2013 International Journal of Computer Science Issues. All Rights Reserved.