Evolutionary Computation Based on Membrane Computing

Liang Huang, Il Hong Suh
Intelligence and Communications for Robots Laboratory,College of Information and Communications, Hanyang University
e-mail : {hl,ilsuh}@incorl.hangyang.ac.kr

Abstract

Membrane computing is extended as an optimization algorithm. The algorithm inherits the evolutionary idea from evolutionary computing and the framework from membrane computing. Tested by two benchmark functions, the excellent performance of the new algorithm is described by the comparison with other algorithms.

1. Introduction

Many intractable problems are difficult to be solved by conventional evolutionary algorithms. Therefore, an evolutionary computation based on membrane computing (ECMC) is investigated. As a theoretic frame of computing devices, membrane computing is difficult to solve the optimization problems directly [1]. In ECMC, the frame of membrane computing is explored and the idea of evolutionary computing is inherited.

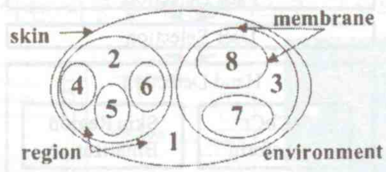
2. ECMC algorithm

The mechanism of the ECMC is illustrated by the following example. The test functions with different difficulty are shown below [2]:

$$f_1(x) = \min(\sum_{i=1}^{50} x_i^2) \ ; \ x_i \in [-5.12, 5.12] \quad (1)$$

$$f_2(x) = \min(\sum_{i=1}^{50} (x_i^2 - 10\cos(2\pi x_i)) + 10) \ ; \ x_i \in [-5.12, 5.12] \quad (2)$$

In principle, the structures of membrane computing are flexible and unlimited. In these optimization cases, the structure as Figure 1 is adopted. There are many strings of the independent variables in each membrane. These strings are feasible solutions which are called as the objects or chromosomes. The algorithm is realized according to these following steps.



[Figure1] Cell-like membrane structure

Step 0: The algorithm structure and objects in membrane is designed and the parameters is given on the initial step. In following experiments, there are 100 objects in the skin m_1 , 25 objects in membrane m_2 and m_3 , and 10 objects in each membrane m_4 to m_8 .

Step 1: The objects in each membrane evolve according to their own associated rules.

Step 1.1: An object S evolve into S' according to the evolution rule as formula (3).

$$\begin{cases} S = x_1 x_2 \cdots x_l \\ S' = y_1 y_2 \cdots y_l \\ y_i = x_i \ or \ x_i + d_i \end{cases} \quad (3)$$

where, l is the length of the strings. x_i, y_i are the variables of a function. d_i is a uniformly distributed random number. The distribution ranges are various in different membranes.

Step 1.2: The objects U,V are recombined as Z,W , where $U = u_1u_2; V = v_1v_2; Z = u_1v_2; W = v_1u_2$ and u_1, u_2 , v_1, v_2 are substrings of the variables. These called splicing rule or crossover rule.

Step 1.3: The selection rule exploits the idea of the heat motion of molecule. The selection rule is simplified as the following formula.

$$\begin{cases} S_{new} = S_i, f_a = f_i; & \text{if } f_i \geq f_a \\ S_{new} = S_i, f_a = f_i; & \text{if } f_i < f_a, P_s > P_i \\ S_{new} = S_a; & \text{if } f_i < f_a, P_s \leq P_i; \end{cases} \tag{4}$$

where, S_i is an arbitrary string in a membrane. S_a is the string which has good fitness at last comparison, S_{new} is the new string which is produced according to the rule for the next generation; f_a is the fitness of S_a which was considered as a good string at the last comparison; f_i is the fitness of the current string S_i; $P_s = 0.2/g \in (0,1)$ is the given probability of the string S_i that is copied to the next generation; g is the generations which the systems has evolved; $P_i(0,1)$ is a random constant for the current string S_i . The rule describes that a string S_i will be selected and copied if its fitness f_i is higher than the fitness f_a of the last copied string S_a. Otherwise it will be copied with a certain probability P_s.

Step 1.4: The communication rule adopts the ideas of biochemistry and the heat motion of molecules. It is expressed by the following formula:

$$P_{communication} = e^{(f_{new} - f_m)/kg} \tag{5}$$

where, f_{new} is the fitness of a string; f_m is the best fitness of the last generation; k is a given constant. The generations is represented by g . The rule means that the string S_{new} will drill through the membrane if $f_{new} \geq f_m$. Otherwise, it will pass the membrane into the outer membrane with the probability $P_{communication}$ and replaced the worst string in the outer membrane.

Step 2: In a serial computer, each membrane evolves in turn. If the halt condition is not satisfied, the procedure jumps to Step 1 for simulating the evolution of another membrane.

After 2,500,000 function evaluations on f_1 and 5,000,000 function evaluations on f_2 , the results of several algorithms is shown in Table 1 [2,3]. The table shows that ECMC converges fast with the highest precision. In fact, the FADE arrives at 258.49 with 5,000,000 function evaluations on the Rastrigin function. However, ECMC only costs 248,000 function evaluations to obtain the same value.

Table 1. Comparison of several algorithms

Functions	CGA	DE	FADE	ECMC
f_1	33.94	39.7	2.35e-10	3.0e-12
f_2	357.15	472.68	258.49	2.378e-7

3. Conclusions

Membrane computing is extended for the optimization. It converges fast with high precision. The excellent performance is confirmed by the simulation experiments compared with other algorithms.

References

1. Gh.Păun. Computing with membranes. *Journal of Computer and System Sciences.* 61(1), 2000:108 143.

2. J.Liu, J.Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Computing* 9(2005): 448-462;

3. Storn R.. On the usage of differential evolution for function optimization. In: *Biennial conference of the North American fuzzy information processing society,* 1996:519-523.