A Constrained Optimization Evolutionary Algorithm Based on Membrane Computing

Haizhu Chen, Jianyun Lu
Department of Software Engineering
Chongqing College of Electronic Engineering
Chongqing, 401331
{cqchz@cqu.edu.cn, lujianyun2007@yahoo.com.cn}

ABSTRACT: Based on membrane computing (MCCOP for short), a constrained optimization evolutionary algorithm is proposed with the evolutionary operations and strategies designed. In MCCOP, a membrane is associated with a constraint and the tentative solutions evolved according to the rules in the membrane. And it is evaluated by the constraint function value as the fitness. The sub-populations can communicate efficiently during the evolution process by making use of the structure of P systems and the communication mechanism among the membranes. The computational experiments show that MCCOP can converge to optimal or close to optimal solutions efficiently and that MCCOP outperforms or performs similarly to the other techniques referred to in terms of the quality of the resulting solutions.

Categories and Subject Descriptors:
G.1.6 [Optimization] Constrained optimization; I.1.2 Algorithms

General Terms: Computing Algorithms, Membrane Computing, Optimization Algorithms

Keywords: Membrane Computing, P system, Optimization Algorithm, Constrained Optimization

Received: 6 November 2011, Revised 19 December 2011, Accepted 24 December 2011

1. Introduction

Membrane computing (also called P system) is an emerging branch of nature computing since introduced by Gh.Paun in 1998 as a kind of distributed parallel computing model [1]. It is aiming to abstract computing models from the structure and the function of living cells and from their interactions in tissues or higher order biological structures.

Membrane algorithm firstly proposed by Nishida [2] is a new type of approximate algorithm based on membrane computing for optimization problems. Nishida’s algorithm is used to solve traveling salesman problems (TSPs for short) and computer experiments show that the membrane algorithms solve TSPs better than the simulated annealing algorithm. Combining P system and differential evolution operator, literature [3] present a hybrid distributed evolutionary algorithm for continuous optimization problems. Huang designs several membrane algorithms for single-objective and multi-objective optimization problems based on nested structures of P systems [4] [5] [6].

Nevertheless, the problems solved were unconstrained. The bioinspired algorithm based on membrane computing (BIAMC for short) was proposed for solving both unconstrained and constrained problems [7] and the experimental results show that this algorithm can find optimal or close-to-optimal solutions efficiently. However, the efficiency of BIAMC in solving constrained problems is inferior to that of deterministic optimization methods, especially when solving problems with equality constraints.

In this paper, we propose an approximate algorithm based on membrane computing for constrained optimization problems: a membrane associates a constraint and the tentative solutions evolve according to the rules in the membrane and are evaluated by the constraint function value as the fitness; the sub-populations can communicate efficiently during the evolution process by making use of the structure of P systems and the communication mechanism among the membranes. This paper is organized as follows. Section 2 first gives a brief introduction of constrained optimization problems and P systems, and Session 3 discusses MCCOP in detail. Section 4 provides experimental results while conclusions drawn in Section 5.
2. Foundation

2.1 Constrained optimization problems

A constrained optimization problem can be defined as:

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_i(x) \leq 0, \ i = 1, \ldots, m_i \\
& \quad h_j(x) = 0, \ j = m_1 + 1, \ldots, m \quad (m = m_1 + m_2) \\
& \quad x \in X
\end{align*}
\]

(1)

Where, \( x = (x_1, x_2, \ldots, x_n) \) is the vector of solutions, \( m_i \) is the number of inequality constraints and \( m_e \) is the number of equality constraints (in both cases, constraints could be linear or nonlinear). \( X \) is called domain constraint and it includes the value ranges of the variants. If we denote with \( F \) to the feasible region and with \( S \) to the whole search space, then it should be clear that \( F \subseteq S \).

2.2 P Systems

The P systems proposed up to now can be classified as [1]: 1) cell-like P systems abstracting the compartmental structure and function of a living cell and composed by membrane structure, objects and rules; 2) tissue-like P system based on intercellular communication and cooperation between neurons; 3) neural-lick P system introduced inspired by the neurophysiologic behavior of neurons sending electrical impulses (spikes) along axons to other neurons.

Our work in this paper is only concerning cell-like P system and Figure 1 illustrates its structure. As suggested by Figure 1, the structure of P system is a hierarchically arranged set of membranes which are usually identified by labels from a given set of labels and contained in a distinguished external membrane (usually called skin). Several membranes can be placed inside skin (they correspond to the membranes present in a cell, around the nucleus, in Golgi apparatus, vesicles, mitochondria, etc.); a membrane without any other membrane inside it is said to be elementary. Each membrane determines a region delimited from above by it and from below by the membranes placed directly inside, if any exists. Clearly, the correspondence membrane-region is one-to-one, so these terms can be used interchangeably.

A cell-like P system with an output set of objects and using transformation and communication rules is formally defined as follows [1]:

\[
\Pi = (O, \mu, w_1, w_2, \ldots, w_m, R_1, R_2, \ldots, R_m, i_o)
\]

(2)

where

(i) \( V \) is an alphabet of objects;
(ii) \( T \subseteq V \) (output alphabet);
(iii) \( \mu \) is a membrane structure, consisting of \( m \) membranes, with the membranes labeled (not necessarily in a one-to-one manner) with elements of a given set \( H \); usually \( H = \{1, 2, \ldots, m\}; m \geq 1 \) (initial degree of \( \Pi \));
(iv) \( w_1, w_2, \ldots, w_m \) are strings over describing the multisets (every symbol in a string representing one copy of the corresponding object) placed in the regions of respectively;
(v) \( R_i, 1 \leq i \leq m \) are sets of rules associated with the regions \( \{1, 2, \ldots, m\} \) of \( \mu \);
(vi) \( i_o \) is a number between 1 and \( m \) which specifies the output membrane of \( \Pi \).

The rules of \( R_i \), \( 1 \leq i \leq m \), have the form \( a \rightarrow v \), where \( a \in v \) and \( v \in (V \times \{\text{here, out, in}\})^* \). The multiset \( v \) consists of pairs \((b, t), b \in V \) and \( t \in \{\text{here, out, in}\} \), where \( \text{here} \) means that \( b \) will stay in the region when the rule is used; \( \text{out} \) shows that \( b \) exits the region and means that \( b \) will be communicated to one of the membranes contained in the current region which is chosen in a non-deterministic way.

A P system, regarded as a model of computation, provides a suitable framework for a distributed parallel computation that develops in steps. In the process of computation, multisets of simple objects are rewritten; the rules associated to regions are employed in a non-deterministic and maximally parallel manner; the rules involving both transformation and communication are responsible for evolving the current objects and transferring them among regions according to some targets; the output region will contain the result of the computation.

3. Details of MCCOP

3.1 Handling Constrains in MCCOP

Handling constrains is one of the most important factors affecting the constrained optimization evolutionary algorithms.

Using the cell-like P system as the framework, MCCOP can be included in the third mechanisms. In MCCOP, each elementary membrane in its structure takes one constraint of the problem each as their fitness function, and the objects in the membrane evolve according to the rules corresponding to the membrane.

MCCOP uses \( H(\bar{x}) = \sum_{j=1}^{m} \text{violate}_j(\bar{x}) \) to valuate the distance between the individual and the border of the feasible region:

\[
\text{violate}_j(\bar{x}) = \sum_{t \in \text{boundaries}} |\text{in}_{\mu(t)}(\bar{x})| - |\text{out}_{\mu(t)}(\bar{x})|
\]

where \( |\text{in}_{\mu(t)}(\bar{x})| \) and \( |\text{out}_{\mu(t)}(\bar{x})| \) are the number of objects coming to and going out of the region \( \mu(t) \) respectively.
\[
\text{ violate}_j (\vec{x}) = \begin{cases} 
\max \{0, g_j (\vec{x})\}, & 1 \leq j \leq g \\
|h_j (\vec{x})|, & g + 1 \leq j \leq m 
\end{cases} 
\] (3)

where, \( g_j (\vec{x}) \), \( h_j (\vec{x}) \) are the non-equal and equal constraints respectively, \( \text{ violate}_j (\vec{x}) \) is the penalty function of the \( j \)-th constrain for individual \( \vec{x} \), \( H(\vec{x}) \) describes the degree against the constrains.

During the optimization, \( \text{better}(\vec{x}_1, \vec{x}_2) \) is used in MCCOP to compare the individuals:

\[
\text{better}(\vec{x}_1, \vec{x}_2) = \begin{cases} 
\text{true, violate}_j (\vec{x}_1) < \text{ violate}_j (\vec{x}_2) \\
\text{false, violate}_j (\vec{x}_1) < \text{ violate}_j (\vec{x}_2) \\
\text{true, violate}_j (\vec{x}_1) < \text{ violate}_j (\vec{x}_2) \land f(\vec{x}_1) < f(\vec{x}_2) \\
\text{true, violate}_j (\vec{x}_1) < \text{ violate}_j (\vec{x}_2) \land f(\vec{x}_1) > f(\vec{x}_2) 
\end{cases} 
\] (4)

When \( \text{better}(\vec{x}_1, \vec{x}_2) = \text{true} \), \( \vec{x}_1 \) is better than \( \vec{x}_2 \), otherwise, is better than. From formula (4), we know that:

1. When the degree against constrains is different, the degree need to be compare and the lower one is better.
2. When the degree against the constrains is the same, the fitness of the objective function is needed to be compared and the smaller one is better.

### 3.2 Handling Objects and Structure

A vector of solutions is represented as an object in the cell-like P system corresponding to MCCOP, and is coded as real vector.

```
  1
o  |
|  2
  |
```

Figure 2. The one level membrane structure

In MCCOP, the membrane structure consisted of a skin membrane and elementary membranes inside as shown in Figure 2. Comparing to the multi-nested membrane structure used in literatures [2] [4] [5] [6], such structure is simplified and the control strategies will be realized easily.

### 3.3 Rules

Each membrane has its own rule set. Different rules affect its associate objects or symbol strings. In MCCOP, there are two types of rules: evolution rule, and transmission rule. These rules are similar to or simplified from the rewriting rules and uniopt rules in a transition P system [1].

1. Evolution rules
2) Crossover rules

The objects in the elementary membranes and the skin will evolve according the modes as follows respectively:

Object \( \vec{x} \) in the \( i \)-th elementary membrane (\( i \in 1, ...., m \)) will evolve to \( \vec{x}' \) by

\[
\vec{x}' = \vec{x} - \lambda_1 \frac{h_i (\vec{x} + \tau_1)}{\tau_1} \\
\vec{x}' = \vec{x} - \lambda_1 \frac{g_i (\vec{x} + \tau_1)}{\tau_1}, \; \lambda_1 \sim N(0, 1), \; \tau_1 = 0.0001
\] (5)

Object \( \vec{x} \) in the skin will evolve to \( \vec{x}' \) by

\[
\vec{x}' = \vec{x} - \lambda_2 \frac{f(\vec{x} + \tau_2)}{\tau_2}, \; \lambda_2 \sim N(0, 1), \; \tau_2 = 0.0001
\] (6)

2) Crossover rules

The objects in the \( i \)-th elementary membrane (\( i \in 1, ...., m \)) will crossover with the ones sent by the skin. Assume that we have \( \vec{x}_1 \) in the elementary membrane and \( \vec{g}_i \) sent by the skin then they crossover in the elementary membrane and produce new object \( \vec{x}'' \):

\[
\vec{x}'' = \vec{x}_1 \vec{g}_i\vec{x}_2 ... \vec{g}_n \quad (i = 1, 2, ..., n)
\] (7)

where, \( CR \) is the probability for crossover operator and \( i \) is the random integer among \([i=1, 2, ..., n]\).

2. Transmission rules

\[
x_{1,i}, x_{2,i} \rightarrow \vec{x}', \quad \vec{x}' = \begin{cases} 
x_{1,i} x_{1,2} ... x_{1,n} \quad \text{or} \quad i = i_{\text{rand}} \leq CR \\
x_{1,i} x_{2,2} ... x_{2,n} \quad \text{or} \quad i = i_{\text{rand}} \\
x_{1,i} x_{2,i} ... x_n \quad \text{Otherwise} \end{cases}
\] (8)

The elementary membrane uses the transmission rules to send several of its own best objects into the skin. These objects will be sent into each membrane randomly after been optimized in the skin and the objects in each membrane will continue evolving. The rules are:

\[
[\vec{x}], \rightarrow [\vec{x}], \vec{x} \quad (1 \leq i \leq m)
\]

(9)

Formula (9) means that the \( i \)-th elementary membrane copies its better object \( \vec{x} \) and send the copied one into the skin while formula (10) means that the skin send object into the \( i \)-th elementary membrane and the object replace one of the worse objects in the objective membrane.

### 3.4 Rules

The overall algorithm looks as follows:

Algorithm, MCCOP

Input: objective function and its constrains \( g_i \) or \( h_i (1 \leq i \leq m) \)
Procedure:
step1: set the parameters including Maxgen (the maximum evolution generations), CellSize (the number of the objects in membranes 1,...,m), TranNum (the number of the objects).

step2: initial the objects in each membrane randomly, and the objects in the $i$-th elementary membrane are evaluated by the penalty function violate; no object is in the skin at the initial stage.

step3: the objects in the $i$-th membrane (ie 1,...,m) evolve according to the evolution rules introduced before.

step4: in the $i$-th elementary membrane, new objects produced in step3 are evaluated by, and all the objects in the same membrane are ranked by the function introduced before.

step5: the $i$-th elementary membrane sends the best $TransNum$ objects into the skin and keeps their copies in its own region, so the skin receives $m*TransNum$ objects.

step6: the objects evolve according to the rules in the skin and the new objects will be evaluated (namely, $f(\vec{z})$ and $f(\vec{g})$) are computed for any object($\vec{x}$). Then the optimum feasible solution $f(\vec{g})$ will be found ($\vec{z}$ satisfies $H(\vec{z}) = 0$), and if $f(\vec{g}) < global$ we will have global=$f(\vec{g})$.

step7: the skin send the $m*TransNum$ objects to the elementary membranes 1,...,m randomly. In the objective membrane, the objects in $S_1$ (the set of the objects kept in the membrane originally) and the ones in $S_2$ (the set of the objects sent into the objective membrane by the skin) will crossover and produce new objects which compose a new set $S_3$. The function better() is used to rank the objects in the set $S(S = S_1 \cup S_2 \cup S_3)$ and the worse ones will be eliminated to keep the size of the set of the objects in the membrane to be |$S$|.

step8: $gen \leq gen + 1$, if $gen \leq Maxgen$, then turn step3; otherwise, turn step9.

step9: output the optimum $global$ and the corresponding solution vector.

Similar to the standard membrane computing, MCCOP can be tested on several parallel computing devices. But for convenient, the evolution of each membrane is processed in a computer serially in this paper.

4. Experiments and results

To evaluate the performance of the proposed algorithm we used the 4 test functions described in [8]. The test functions chosen contain characteristics that are representative of what can be considered “difficult” global optimization problems for a $G_1$ constrained optimization evolutionary algorithm.

$G_1$ Minimize:

$$f(\vec{x}) = 3x_1 + 0.000001x_1^2 + 2x_2 + (0.000002/3)x_2^2$$

Subject to:

$$g_1(\vec{x}) = -x_4 + x_5 - 0.55 \leq 0$$
$$g_2(\vec{x}) = -x_4 - x_5 - 0.55 \leq 0$$
$$h_1(\vec{x}) = 1000 \sin(-x_4 - 0.25) + 1000 \sin(-x_4 - 0.25) + 984.8 - x_5 = 0$$
$$h_2(\vec{x}) = 1000 \sin(x_4 - 0.25) + 1000 \sin(x_4 - x_5 - 0.25) - 1294.8 = 0$$

$0 \leq x_4 \leq 1200, 0 \leq x_5 \leq 1200, -0.55 \leq x_1 \leq 0.55, -0.55 \leq x_4 \leq 0.55$

$G_2$ Minimize:

$$f(\vec{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + (x_3 - 11)^2 + 10x_4^2 + 7x_6^2 + x_4^2 - 4x_6x_4 - 10x_6 - 8x_5$$
$$g_1(\vec{x}) = -127 + 2x_1^2 + 3x_2^2 + x_3 + 4x_4^2 + 5x_6 \leq 0$$
$$g_2(\vec{x}) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0$$
$$g_3(\vec{x}) = -196 + 23x_1 + x_2^2 + 6x_3^2 + 8x_6 \leq 0$$
$$g_4(\vec{x}) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0$$
$$-10 \leq x_i \leq 10 (i = 1,...,7)$$

$G_3$ Minimize:

$$f(\vec{x}) = x_1 + x_2 + x_3$$

Subject to:

$$g_1(\vec{x}) = -1 + 0.0025(x_3 + x_4) \leq 0$$
$$g_2(\vec{x}) = -1 + 0.01(x_5 - x_3) \leq 0$$
$$g_3(\vec{x}) = -x_1x_6 + 833.33252x_6 + 100x_7 - 8333.333 \leq 0$$
$$g_4(\vec{x}) = -x_2x_6 + 1250x_6 + x_4x_6 - 1250x_7 \leq 0$$
$$g_5(\vec{x}) = -x_3x_7 + 125000x_7 + x_5x_7 - 2500x_8 \leq 0$$
$$100 \leq x_4 \leq 10000 \quad 1000 \leq x_5 \leq 10000 (i = 2,3)$$
$$-10 \leq x_i \leq 1000 (i = 4, ..., 8)$$

$G_4$ Minimize:

$$f(\vec{x}) = x_1 + (x_2 - 1)^2$$

Subject to:

$$h(\vec{x}) = x_1 - x_2^2$$
$$-1 \leq x_1 \leq 1, -1 \leq x_2 \leq 1$$

These functions are composed by multivariate function and multi-dimension constrains: the objective functions are linear and nonlinear ones; the constraints are nonlinear equality and inequality (in this case, constraints could be linear or nonlinear). Among the four functions, $G_1$ and $G_4$ test the abilities of handling the constraints for they have equality constrains, $G_2$ and $G_3$ test the efficiency of the algorithm.

For each function, we perfume 20 independent tails, and the parameters are set as follows: $CR=0.6$; $Cellsize=10$ for $G_1$ and $G_2$ and $G_3$; for $G_4$, $Cellsize=50$ since it have only one constraint; the fitness function evaluations (FFE$s$ for short) is set by 180 000. Table 1 gives the statistics results of the MCCOP on the four benchmark functions including the best, mean, worst values and the standard deviations (St.dev. for short). From Table 1, we can see that MCCOP can find the global optimums for $G_2$, $G_3$, $G_4$ ; and find the solution close to the global optimum.
We compare MCCOP with three other algorithms: HCVEGA [8], DMS-PSO [9] and SMES [10]. In HCVEGA the subpopulations evolve according to the singular constraint or the object function respectively and the function to be optimized can be changed in every generation. DMS-PSO is similar to HCVEGA, but the difference is that the function to be optimized is self-adaptively allocated to the subpopulation and the function can be changed periodically. SMES preserves the best unfeasible solution in the next generation with some probability (such as 0.03) to keep the diversity which is important in the case that the global optimum is near to the boundary of the feasible region.

Table 2 summarizes the comparing results for the four algorithms. From Table 2, we can see that MCCOP super to the three other algorithms on the best, mean, worst values and St.dev. for \( G_1, G_2 \) and \( G_3 \). For \( G_4 \), the efficiency of MCCOP is similar to the ones of DMS-PSO and SMES. On the computing cost, the FFEs of MCCOP, HCVEGA and DMS-PSO are 180 000 respectively, the one of SMES is 240 000.

5. Conclusion

In this paper, a constrained optimization evolutionary algorithm based on membrane computing is proposed. In MCCOP, a membrane is associated with a constraint and the tentative solutions evolved according to the rules in the membrane. And it is evaluated by the constraint function value as the fitness. The sub-populations can communicate efficiently during the evolution process by making use of the structure of P systems and the communication mechanism among the membranes. Computer experiments show that MCCOP can converge to optimal or close to optimal solutions efficiently and that MCCOP outperforms or performs similarly to the other techniques referred to in terms of the quality of the resulting solutions. Further researches will be done to find more efficient operators to improve the performance of the proposed algorithm.

References


