一种基于膜系统的全局智能优化算法

刘 闯[1],韩 敏[1],邢 军[2]
(1.大连理工大学电子信息与电气工程学部,辽宁大连 116023;2.大连工业大学信息科学与工程学院,辽宁大连 116034)

摘 要: 针对全局数值优化问题,本文提出了一种基于膜计算理论的启发式全局优化算法.受细胞内液体分子做无规则运动的启发,该算法构建了液体分子沿任意和某一方向运动的机制,实现了算法全局探索和局部开发的能力.8个benchmark测试优化函数的仿真结果表明,所提算法具有保持解的多样性和跳出局部极值的全局寻优能力.

关键词: 进化膜算法;膜计算;全局优化;细胞自动机
中图分类号: TP18 **文献标识码:** A **文章编号:** 0372-2112 (2013)05-0871-07
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2013.05.007

A Global Intelligent Optimization Algorithm Based on Membrane Systems

LIU Chuang[1], HAN Min[1], XING Jun[2]
(1. *Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian, Liaoning* 116023, *China*;
2. *Faculty of the Information Science and Engineering, Dalian Ploytechnic University, Dalian, Liaoning* 116034, *China*)

Abstract: For global numeric optimization problems, this paper presents a novel heuristic global optimization algorithm based on membrane computing. Inspired by the irregular movement of liquid molecules, two mechanisms are designed to simulate the movement of molecules along an arbitrary direction or a certain direction. These two mechanisms enable the algorithm to implement global exploration and local exploitation. Simulation results for eight benchmark functions indicate that the proposed algorithm can maintain the diversity of solutions, and jump out of local minima.

Key words: evolutionary membrane algorithm; membrane computing; global optimization; cellular automata

1 引言

现实世界中存在着具有单模和多模等特点的优化问题,特别是在科学研究和工程应用等领域.由于进化算法可以有效地求解这类问题,所以国内外学者对其进行了广泛的研究.目前,进化算法的研究包括遗传算法、免疫算法以及群智能等[1~4].虽然这些方法都可以求解不同种类的复杂优化问题,但它们在全局寻优能力和快速收敛问题上难以令人满意.因此,本文尝试设计一种新的进化算法改善其不足.

欧洲科学院院士、罗马尼亚科学家 Păun 受生物细胞功能和结构的启发,提出了作为自然计算的一个新分支——分子级细胞计算理论,即膜计算[5~7].多数膜计算模型被证明具有 Turing 通用性[8,9]和计算有效性[10~12].目前,膜计算理论已被应用于生物系统建模、经济、语言处理、和优化等领域[13].然而,膜计算在优化领域的应用研究虽然引起了学者的关注,但就其现状而言还有很多值得深入研究的方面.文献[14]首次提出了膜内嵌套局部优化算法的膜算法求解 NP 优化问题.文献[15,16]受膜计算理论的启发,提出了可以求解单目标和多目标问题的优化算法.文献[17]提出了量子进化方法与 P 系统相结合的优化算法用于求解 0/1 背包组合的优化问题.文献[18]阐述了对膜优化算法的收敛性和多样性的分析.文献[19]提出了用 MOMC 算法求解多目标优化问题的思想,并在维护解的多样性和提高算法收敛速度方面取得了较好的效果.

本文受液体分子做无规则运动的启发,提出了一种基于膜计算理论的求解全局优化问题的高效进化算法,即进化膜算法(Evolutionary Membrane Algorithm, EMA).为使算法具有跳出局部极值和高效全局寻优能力,本文构建了让液体分子沿任意和某一方向运动的两种机制.考虑到实现这两种机制的复杂性,引入了细胞自动机和混沌搜索加以辅助.细胞自动机可以模拟复杂的运动,因此可以利用其特点描述液体分子运动轨迹,进而有效地遍历解空间.混沌搜索具有伪随机和遍历等特性,利用其特性更新液体分子的位置可以产生更多新位置,进而

收稿日期:2012-05-09;修回日期:2012-12-21
基金项目:国家自然科学基金(No.60674073)

增强候选解的多样性,避免算法陷入局部极值.通过对数值仿真实验的仿真分析,验证了所提 EMA 的有效性.

2 膜计算

膜计算是 Păun 受生物细胞结构和功能启发而提出的一种分布式和并行计算的分子计算理论[5].式(1)列出了度为 n 的膜计算的基本结构.

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

其中,

(1) V 是字母表,其元素被称为字符对象.字符对象是原子、分子或其他化学物质的抽象表示;

(2) $T \subseteq V$,其中 T 是输出字母表;

(3) μ 是包含 n 个膜的膜结构,其中 n 称为系统 Π 的度;

(4) $w_i \in V^*$, w_i 表示膜结构 μ 中的第 i 个区域中包含的多重集, V^* 是由 V 中元素组成的多重集的集合;

(5) R 表示对膜和字符对象作用的规则.

如图 1 所示,膜结构是由一系列分层排列的膜构成.膜的内部又称区域(regions)包含了膜、多重集(multi-set)和规则.最外层的膜是系统的表层膜(skin membrane).表层膜将环境与其内部分开,它的内部包含了非基本膜(membrane)和基本膜(elementary membrane).非基本膜的区域可以嵌套任何非基本膜或基本膜,而基本膜的区域则不再包含任何膜.

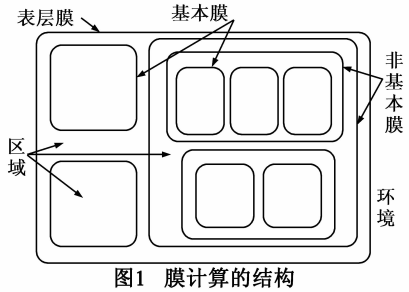

图1 膜计算的结构

3 进化膜算法的提出

本文的主要目的是设计一个基于膜计算理论的用于求解优化问题的全局启发式并行搜索演化算法.一个完整的膜计算系统由多重集、反应规则和膜结构这三部分组成.当使用进化膜算法求解优化问题时,多重集中的字符对象表示优化问题的一个可行解;反应规则不仅描述了字符对象的演化规则,还包括了对膜操作的规则;膜结构定义了算法执行的整体架构.

3.1 多重集及其初始化

在膜计算中,字符对象是原子或液体分子等化学物质的抽象化表示,而多重集是由字符对象组合成的集合.在所提 EMA 中,字符对象表示优化问题的解,多重集表示优化问题的解集.在满足约束条件下,字符对象按十进制进行编码.

考虑到字符对象(液体分子)分布具有随机的特点,而混沌系统可以通过确定的方程生成具有随机性的状态,因此,本文采用 Sinusoidal Iterator 混沌系统[20],对 D 维的字符对象初始化.具体描述如下:

步骤 1 设置混沌变量初始值 $y_0 = 0.7$;

步骤 2 根据 Sinusoidal Iterator 混沌系统的迭代式(2),本文计算了混沌变量 y_k 的下一代混沌变量 y_{k+1}.其中 $a = 2.3$;

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

步骤 3 通过式(3)将混沌变量 y_{k+1} 转换为字符对象 x_{k+1}.其中, $x_k^{\max}$ 表示字符对象第 k 维的取值上界,而 $x_k^{\min}$ 表示字符对象第 k 维的取值下界;

$$x_{k+1} = x_k^{\min} + y_{k+1} * (x_k^{\max} - x_k^{\min}) \tag{3}$$

步骤 4 如果维数 $k = k + 1$ 大于 D,结束初始化字符对象操作,否则执行步骤 2.

步骤 5 计算该字符对象的适应度值.

3.2 反应规则

在所提 EMA 中,反应规则由演化规则和操作规则两部分组成.演化规则模拟了字符对象(液体分子)的游动过程;操作规则实现了基本膜创建、搜索空间的划分、膜间的通信和膜的溶解操作.

3.2.1 演化规则

演化规则 1 实现了当前字符对象对二个活跃相邻字符对象综合作用后形成新的字符对象.这个规则增强了算法在已知解空间开发的能力,具体形式如式(4)所示.

$$\begin{aligned} x^{\mathrm{nbSet}_1} &= x^{\mathrm{nbSet}_1} + (x^{\mathrm{bestMember}} - x^{\mathrm{nbSet}_2}) * \alpha \\ x^{\mathrm{nbSet}_2} &= x^{\mathrm{nbSet}_2} + (x^{\mathrm{bestMember}} - x^{\mathrm{nbSet}_1}) * \alpha \end{aligned} \tag{4}$$

其中, x^{nbSet_i} 是当前字符对象相邻状态为 1 的第 i 个字符对象; $x^{\mathrm{bestMember}}$ 是全局最好字符对象;通过仿真实验发现,当 α 为[0,2.5]之间的随机数时,能较好的维护解的多样性.

演化规则 2 实现了当前字符对象对三个活跃相邻字符对象综合作用后形成的新字符对象.这个规则增强了算法在已知解空间开发的能力,具体形式如式(5)所示.

$$\begin{aligned} x^{\mathrm{nbSet}_1} &= x^{\mathrm{nbSet}_1} + (2 * x^{\mathrm{bestMember}} - x^{\mathrm{nbSet}_2} - x^{\mathrm{nbSet}_3}) * \alpha \\ x^{\mathrm{nbSet}_2} &= x^{\mathrm{nbSet}_2} + (2 * x^{\mathrm{bestMember}} - x^{\mathrm{nbSet}_1} - x^{\mathrm{nbSet}_3}) * \alpha \\ x^{\mathrm{nbSet}_3} &= x^{\mathrm{nbSet}_3} + (2 * x^{\mathrm{bestMember}} - x^{\mathrm{nbSet}_1} - x^{\mathrm{nbSet}_2}) * \alpha \end{aligned} \tag{5}$$

其中, x^{nbSet_i} 是当前字符对象相邻状态为 1 的第 i 个字符对象; $x^{\mathrm{bestMember}}$ 是全局最好字符对象;通过仿真实验发现,当 α 为[0,2.5]之间的随机数时,能较好的维护解的多样性.

演化规则 3 描述了基本膜内字符对象受到相邻对象的相互作用,此规则可以保持所提 EMA 求得候选解

的多样性,具体形式如式(6)所示.

$$x_{i,j}=\begin{cases}x_j^{\text{localStrObj}}, & r<1/D\\ x_{i,j}, & r\geqslant 1/D\end{cases} \tag{6}$$

其中,$x_{i,j}$表示当前多重集中第 i 个字符对象的第 j 维. $x_j^{\text{localStrObj}}$ 是多重集中局部最好的字符对象的第 j 维;D 表示优化问题的维数;$r=\text{rand}$ 是[0,1]的随机数.

演化规则 4 设计了字符对象状态更新操作,它的具体形式如式(7)所示:

$$x^{\text{state}}=\begin{cases}1, & \text{if}\sum_{i=1}^{8}x^{i,\text{state}}=2,3\\ 0, & \text{其他}\end{cases} \tag{7}$$

其中,x^{state}表示当前字符对象状态,$x^{i,\text{state}}$表示其相邻的第 i 个字符对象状态的累加和.

演化规则 5 模拟了当前字符对象在某一个方向受到的撞击作用强致使字符对象向其方向运动这一过程.本文使用 Tent Map 混沌系统(见式(8))模拟了一个 D 维字符对象碰撞过程.

$$x_{n+1}=\begin{cases}x_n/0.7, & x_n<0.7\\ \dfrac{1}{0.3}*x_n(1-x_n), & \text{否则}\end{cases} \tag{8}$$

其中,n 表示迭代的代数. $X_n\in(0,1)$是第 n 个混沌变量.演化规则 5 的具体描述如下:

步骤 1　选择多重集中的字符对象 x_k^i,使用式(9)将 x_k^i 从决策空间映射到混沌空间的混沌变量 y_k^i.

$$y_k^i=\frac{x_k^i-x_k^{\min}}{x_k^{\max}-x_k^{\min}} \tag{9}$$

其中,$x_k^{\min}$ 表示字符对象第 i 维取值的下界,而 $x_k^{\max}$ 表示字符对象第 i 维取值的上界.

步骤 2　混沌变量 y_k^i 使用式(10)产生下一代混沌变量 y_{k+1}^i.

$$y_{k+1}^i=\begin{cases}y_k^i/0.7, & y_k^i<0.7\\ \dfrac{1}{0.3}*y_k^i(1-y_k^i), & \text{否则}\end{cases} \tag{10}$$

步骤 3　将混沌变量 y_{k+1}^i转换为字符对象 x_{k+1}^i,计算公式如式(11)所示.

$$x_{k+1}^i=x_k^{\min}+y_{k+1}^i*(x_k^{\max}-x_k^{\min}) \tag{11}$$

步骤 4　如果 $k=k+1$ 大于维数 D,混沌搜索操作结束,否则算法将继续步骤 1.

步骤 5　比较混沌搜索前和搜索后字符对象 x_i^k 和 x_i^{k+1} 的适应度值,并选择适应度最优的字符对象.

3.2.2　操作规则

在表层膜的区域,创建基本膜规则可以产生若干具有求解优化问题能力的基本膜,规则的形式如式(12)所示:

$$[\,]_0\longrightarrow[[\,]_1,[\,]_2,\cdots,[\,]_n]_0 \tag{12}$$

其中,n 是基本膜个数;$[\,]_0$ 表示表层膜;而$[\,]_i$ 表示第 i 个基本膜.

为使每个基本膜的区域都有与之对应的多重集,本文提出了划分搜索空间的规则.此规则可以对优化问题的搜索空间进行划分加快算法的搜索效率和提高解的多样性.该规则的具体实现思路如下:先通过目标函数评估多重集中字符对象的适应度值;然后根据适应度值大小,对多重集中字符对象进行排序;并将排序后的多重集划分成等大小的子多重集.具体形式见式(13).

$$\begin{aligned}&W'=\text{sort}(W);\\ &W'=\{w_1,w_2,\cdots,w_m\}\\ &w_i=W'((i-1)*n+1:n:i*n)\\ &n=\text{sizeof}(W)/m\end{aligned} \tag{13}$$

其中,$1\leqslant i\leqslant m$,m 是基本膜个数;W 是在表层膜中的多重集,sizeof(W)表示多重集中字符对象个数;而 W' 是按照多重集中字符对象的适应度排序后的多重集;w_i 是划分后的第 i 个基本膜的子多重集,$W'((i-1)*n+1:n:i*n)$表示从排序后的多重集中的第$(i-1)*n+1$ 个字符对象位置起顺序取 n 个字符对象.

通信规则是实现发送表层膜的多重集到基本膜区域的一个过程.在表层膜区域中,当执行划分搜索空间规则后,调用通信规则,可以依次地发送表层膜中的子多重集到不同的基本膜.这使得基本膜可以在子多重集对应的搜索空间内搜索近似最优解,具体形式如式(14)所示:

$$[w_1,w_2,\cdots,w_m]_0\rightarrow[[w_1]_1,[w_2]_2,\cdots,[w_m]_m]_0 \tag{14}$$

其中,m 是基本膜个数;$[\,]_0$ 表示表层膜,$[\,]_i$ 表示第 i 个基本膜;w_i 为根据 W 划分的第 i 个子多重集.

当基本膜内的规则执行结束后,调用溶解规则,溶解当前膜,直到所有的膜都被溶解,才进行下一步操作.溶解规则可以使基本膜中的多重集(近似最优解集)释放到表层膜中,进而使得来自不同基本膜的多重集被有效共享,它有利于加速算法找到全局最优近似解.具体形式如式(15)所示:

$$[[w_1]_1,[w_2]_2,\cdots,[w_m]_m]_0\rightarrow[w_1,w_2,\cdots,w_m]_0 \tag{15}$$

其中,m 是基本膜个数,$[\,]_0$ 表示表层膜,$[\,]_i$ 表示第 i 个基础膜,w_i 是第 i 个子多重集.

3.3　膜结构

从前文的图 1 中可以看出,膜计算有着由一个表层膜包含多个嵌套膜组成的多层结构.在所提 EMA 中,本文将膜计算的结构简化成了只有一个表层膜包含多个基本膜的二层特殊结构.它可以表示为 $[_0[_1]_1,[_2]_2,\cdots,[_m]_m]_0$,其中,0 表示表层膜的标号,$1\cdots m$ 表示基本膜的标号,m 是基本膜的最大个数.

3.3.1　表层膜

表层膜是膜计算的最外层膜.它实现了来自不同

基本膜中字符对象的信息交换,建立于膜计算参数的初始化操作结束之后.在表层膜内部区域,如果字符对象未被初始化,则需要先初始化字符对象,然后将这些字符对象组成多重集.初始化过程见 3.1 节.

3.3.2 基本膜

每个基本膜的区域都包含着与之对应的多重集和反应规则.在基本膜的区域,通过细胞自动机调用进化规则更新多重集中的字符对象来实现字符对象间的碰撞模拟.

由于单一的进化规则很难模拟字符对象间碰撞后沿任意方向运动的过程,因此,需要某种机制调用多个进化规则共同模拟这一复杂过程.细胞自动机可以模拟从简单的状态到复杂的现象,所以引入细胞自动机调用进化规则模拟这一复杂的碰撞过程.细胞自动机调用规则的具体流程如图 2 所示.值得一提的是细胞自动机中的细胞与算法中的字符对象(液体分子)是一一对应的.

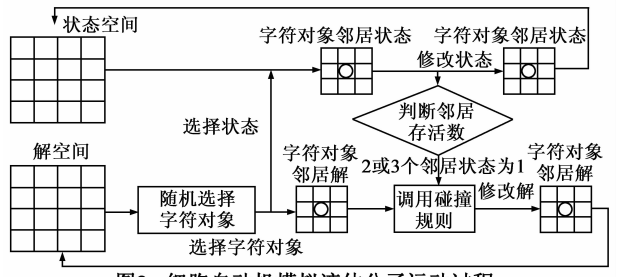

图2 细胞自动机模拟液体分子运动过程

为使基本膜中子多重集内字符对象都能填入图 2 中的解空间网格,则需要先创建解空间网格,网格的行数等于 $\left\lfloor \sqrt{\text{size}(w_i)} \right\rfloor$(子多重集大小开平方的下取整),而网格的列数等于子多重集中字符对象个数除以其行数的下取整.例如,多重集的大小为 100,则网格的行数等于 $\left\lfloor \sqrt{100} \right\rfloor = 10$,而列数等于 $\left\lfloor 100/10 \right\rfloor = 10$,从而形成了 10×10 的网格.而后将字符对象依次地填入到解空间网格;同时生成与解空间网格一一对应的状态网格,每个网格的状态值分别随机取为 0 或者 1.其中,0 表示活跃的字符对象,而 1 表示非活跃的字符对象.接下来,需要从解空间网格中随机选取一个字符对象并查找其相邻字符对象,然后查找与之对应的状态和其相邻状态:当前字符对象状态为 1,相邻状态的累加和为 2 时,则执行演化规则 1,见式(4);当前字符对象状态为 1,其相邻状态的累加和为 3 时,则执行演化规则 2,见式(5);否则,则执行演化规则 3,见式(6).如果需要更新字符对象状态,则执行规则 4,见式(7).

上述内容描述了细胞自动机模拟字符对象碰撞后沿任意方向运动的过程.接下来,本文模拟当前字符对象在某一个方向受到的撞击作用强致使字符对象向其方向运动这一过程.本文引入 Tent Map 混沌系统模拟字符对象下一步可能运动的位置.Tent Map 混沌系统的引入能有效的提高近似最优解的多样性,提升算法在搜索空间内探索的能力.具体实现过程见演化规则 5.

3.4 结束条件

当算法总的适应度评估次数超过给定上限时(上限设置为 10000 乘以问题维数),进化膜算法将停止执行;否则,进化膜算法将继续执行循环体内的操作.

3.5 伪代码

为便于理解所提 EMA,根据上述设计流程,下面给出了所提算法的伪代码.

进化膜算法的伪代码

```
输入 : optimization problem , membraneStructure,maxInter,
       dimention, lowerBound , upperBound
Begin skin membrane //表层膜
Initiallize the multiSet by Sinusoidal Interator .//初始化多重集
 evaluate the multiSet //适应度评估
 while inter <=maxInter //迭代条件
        sort (multiSet);//排序多重集
        Find the historical optimal symbol object as bestMember
         //查找最优字符对象
        // call the divided search space rule //调用创建基本膜规则
        while eleNum <elementaryNum//执行基本膜串行化操作
              create subMultiSet(eleNum);
              Find  a local best string object as localStrObj
     end
     receive  subMultiSet from elementary //接收基本膜的多重集
    multiSet =merge  (subMultiSet);//合并多重集
    nfeval = evaluate the multiSet //适应度评估
     inter =nfeval+inter;
End
End skin membrane
输出: bestMember
输入: subMultiSet , bestMember, rule
Begin Elementary //基本膜
          randomly select a symbolObj from subMultiSet
          //随机从多重集中选择字符对象
         [neighbourSet ,loc ] = neighbour(subMultiSet,symbolObj);
         //查找当前字符对象的相邻对象
         stateSum = sum (neighbourSet (end ));//计算相邻状态
         if symbolObj .state == 1//当前字符对象状态为1
             if stateSum == 2
               execute 演化规则 1
             elseif stateSum == 3
               execute 演化规则 2
           end
             execute 演化规则 3
       end
             execute 演化规则 4
        Tent Map chaotic system  is invoked //演化规则 5
        call dissolution rule ;//调用溶解操作
End Elementary
输出  subMultiSet
```

4　仿真实验

4.1　测试问题及实验设置

为验证所提 EMA 的性能,下面将通过表 1 中的 8 个具有单模和多模等特性的测试函数对算法进行测试.单模函数可以测试算法的收敛性,而多模函数可以测试算法的全局寻优能力.与单模函数相比,多模函数求解更加困难,易陷入局部极值点.仿真实验中使用的测试函数 F1 是简单不可分离的单模函数,F2 到 F8 是具有现实优化问题特点的多模函数.

表 1　测试函数

F1: $\sum_{i=1}^{N} x_i^2$,	$-100\le x_i\le 100$ $F^*(0,\cdots,0)=0$
F2: $20+\exp(1)-20*\exp(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2})$ $-\exp(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i))$,	$-32\le x_i\le 32$ $F^*(0,\cdots,0)=0$
F3: $\sum_{i=1}^{N}\frac{x_i^2}{4000}-\prod_{i=1}^{N}\cos\frac{x_i}{\sqrt{i}}+1$,	$-600\le x_i\le 600$ $F^*(0,\cdots,0)=0$
F4: $10N+\sum_{i=1}^{N}(x_i^2-10\cos(2\pi x_i))$,	$-5\le x_i\le 5$ $F^*(0,\cdots,0)=0$
F5: $-\cos\left(2\pi\sqrt{\sum_{i=1}^{N}x_i^2}\right)+0.1\sqrt{\sum_{i=1}^{N}x_i^2}+1$,	$-100\le x_i\le 100$ $F^*(0,\cdots,0)=0$
F6: $\sum_{j=1}^{N}\sum_{i=1}^{N}\left(\frac{y_{i,j}^2}{4000}-\cos(y_{i,j})+1\right)$ $y_{i,j}=100(x_j-x_i^2)^2+(1-x_i)^2$,	$-100\le x_i\le 100$ $F^*(1,\cdots,1)=0$
F7: $\frac{\pi}{N}\{10\sin^2(\pi y_1)+\sum_{i=1}^{N-1}(y_i-1)^2$ $\times[1+10\sin^2(\pi y_{i+1})]+(y_N-1)^2\}$ $+\sum_{i=1}^{N}u(x_i,10,100,4)$, $y_i=1+\frac{1}{4}(x_i+1)$,	$-50\le x_i\le 50$ $F^*(-1,\cdots,-1)=0$
F8: $0.1\{\sin^2(\pi 3x_1)+\sum_{i=1}^{N-1}(x_i-1)^2$ $\times[1+\sin^2(3\pi x_{i+1})]+(x_N-1)^2\times$ $[1+\sin^2(2\pi x_N)]\}+\sum_{i=1}^{N}u(x_i,5,100,4)$,	$-50\le x_i\le 50$ $F^*(1,\cdots,1)=0$

本次实验在 Intel Pentium 双核 2.93GHz,2G 内存的硬件环境上和 windows XP 操作系统下运行,采用 matlab 7.11 工具进行编程.本文选择了两个比较算法 CMAES[21]和 OXDE[22],它们的参数设置请见文献.在实验中,测试函数维数分别为 5、10 和 30 维.适应度最大评估次数等于 10000 * 维数.在每次试验中,比较方法和所提 EMA 分别独立运行 25 次,然后对运行结果进行统计分析,以便进一步比较算法的求解性能.本文设计了 Mean 和 Std 指标.其中,指标 Mean 表示执行 25 次独立实验的平均适应度值,反映了算法求解的集中趋势;指标 Std 表示标准方差,反映了算法的稳定性和鲁棒性.最后,所提 EMA 根据式(16)进行初始化.

$$\Pi=\begin{cases}\{w_{i,j}\ \mid\ 1\le i\le 10,\ \ 1\le j\le 10\},\\ [_0[_1]_1,[_2]_2,\cdots,[_{10}]_{10}]_0,\\ w_1\cdots w_{10},\\ \{w_{i,1},\cdots w_{i,10}\to w_{i,j},\ w_{i,j1}\,w_{i,j2}\to w'_{i,j1}\,w'_{i,j2},\\ w_{i,j}\to w'_{i,j},[]_0\to[[]_1,[]_2,\cdots,[]_{10}]_0,\\ [w_i]_0\to[[w_i]_i]_0\\ [[w_i]_i]_0\to[w_i]_0,1\le i\le 10\}\end{cases}\tag{16}$$

其中,Π 表示膜系统,"$w_{i,j}$"表示第 i 个多重集中的第 j 个字符对象,"$[_0[_1]_1,[_2]_2,\cdots,[_{10}]_{10}]_0$"表示标记为 0 的表层膜中包含了 10 个基本膜,"$w_1,\cdots,w_{10}$"表示由字符对象组合而成的 1 至 10 个多重集,规则"$w_{i,1},\cdots w_{i,10}\to w_{i,j}$"表示选择字符对象操作,规则"$w_{i,j1}\,w_{i,j2}\to w'_{i,j1}\,w'_{i,j2}$"表示细胞自动机进化操作,规则"$w_{i,j}\to w'_{i,j}$"表示混沌搜索操作,规则"$[]_0\to[[]_1,[]_2,\cdots,[]_{10}]_0$"表示创建基本膜的操作,规则$[w_i]_0\to[[w_i]_i]_0$表示通信规则,规则"$[[w_i]_i]_0\to[w_i]_0$"表示基本膜的溶解操作.

4.2　分析及讨论

由表 2 可知,对于 5 维 benchmark 问题,除了函数 F1 和 F7,所提 EMA 均求得了较好的结果.在函数 F1 和 F7 上,所提 EMA 求得的结果虽然没有 CMAES 方法的精度高,但它的结果是稳定的.而与 OXDE 相比,所提 EMA 优势明显.

表 3 的实验结果表明,本文提出的方法 EMA 在函数 F2,F3,F4,F5,F6 和 F8 上均取得了较好的结果.在函数 F1 和 F7 上,尽管 OXDE 和 CMAES 分别求得了最优的结果,但所提 EMA 的结果精度还是可以接受的.

从表 4 实验结果可以看出,在函数 F3,F4,F6,F7 和 F8 上,所提 EMA 求得了较好的结果.但在函数 F1,F2 和 F5 上,OXDE 取得了较好的结果.

表 2 在 5 维 benchmark 问题上的测试结果比较

	指标	CMAES[21]	OXDE[22]	EMA
F1	Mean	8.82e-84	2.93e+01	1.98e-20
	Std	1.31e-83	5.19e+01	3.26e-20
F2	Mean	1.85e+01	3.06e+00	1.62e-10
	Std	3.91e+00	3.48e+00	2.36e-10
F3	Mean	7.18e+00	1.10e+00	5.63e-03
	Std	2.29e+01	1.89e+00	5.81e-03
F4	Mean	4.01e+01	4.56e+00	4.42e-02
	Std	1.75e+01	1.75e+01	1.99e-01
F5	Mean	1.30e+01	9.79e-01	9.98e-02
	Std	2.30e+00	7.80e-01	7.78e-07
F6	Mean	6.28e+00	2.40e+11	1.56e-01
	Std	3.58e+00	1.16e+12	3.82e-01
F7	Mean	2.46e-31	3.27e+03	4.70e-21
	Std	9.54e-32	1.63e+04	1.53e-20
F8	Mean	2.19e-03	2.24e+03	8.46e-20
	Std	4.48e-03	9.12e+03	3.54e-19

表 3 在 10 维 benchmark 问题上的测试结果比较

	指标	CMAES[21]	OXDE[22]	EMA
F1	Mean	5.99e-39	9.55e-171	9.92e-20
	Std	3.20e-39	0	2.12e-19
F2	Mean	1.94e+01	7.01e-01	9.60e-02
	Std	3.48e-01	1.66e+00	3.32e-01
F3	Mean	7.78e-03	5.92e-02	3.85e-03
	Std	8.48e-03	4.93e-02	6.95e-03
F4	Mean	7.27e+01	4.17e+00	9.13e-01
	Std	2.66e+01	2.94e+00	8.41e-01
F5	Mean	1.68e+01	4.55e-01	1.48e-01
	Std	5.40e+00	6.62e-01	5.03e-02
F6	Mean	3.14e+01	1.63e+01	1.13e-01
	Std	1.24e+01	1.38e+01	5.68e-01
F7	Mean	2.61e-31	6.21e-02	4.99e-20
	Std	6.74e-32	3.10e-01	1.64e-19
F8	Mean	1.75e-03	4.43e-01	7.65e-20
	Std	4.11e-03	9.59e-01	1.97e-19

由表 2、表 3 和表 4 可知,所提 EMA 在单模测试函数 F1 上均取得了较好的结果,说明其具有较好的寻优速度和精度.随着问题维度的增加,算法 CMAES 较难找到全局最优解;而随着问题维数降低,算法 OXDE 寻找最优解的性能也随之下降.然而,对所提 EMA 而言,在低维的 benchmark 问题上,它在收敛精度和鲁棒性方面有较好的结果;随着问题维数的增加,虽然所提 EMA 在多模函数 F2 和 F5 上没有取得最优的结果,但在所有实验中其结果是稳定的.仿真数据表明无论是在低维函数优化还是高维函数优化中所提 EMA 都能够并行地遍历搜索空间内潜在的优化解,有效地搜索解空间,进而使它能跳出局部极值,保持候选解的多样性.这些结论说明了基于膜计算的所提算法求解不同特点的测试优化问题均表现出了较好的求解性能,从而证明了所提 EMA 在收敛速度慢和易陷入局部极值点的优化问题中求解是有效的.

表 4 在 30 维 benchmark 问题上的测试结果比较

	指标	CMAES[21]	OXDE[22]	EMA
F1	Mean	6.41e-29	6.37e-59	3.03e-17
	Std	1.52e-29	1.39e-58	2.16e-17
F2	Mean	1.95e+01	4.62e-02	9.98e-02
	Std	1.03e-01	2.31e-01	3.43e-01
F3	Mean	8.87e-04	2.85e-03	8.72e-11
	Std	2.45e-03	6.18e-03	4.35e-10
F4	Mean	2.31e+02	9.31e+00	1.52e+00
	Std	4.73e+01	3.29e+00	1.96e+00
F5	Mean	3.05e+01	2.03e-01	7.06e-01
	Std	6.34e+00	4.54e-02	4.54e-02
F6	Mean	4.16e+02	3.33e+02	4.75e+01
	Std	6.99e+01	5.49e+01	6.75e+01
F7	Mean	4.14e-03	2.48e-02	3.40e-19
	Std	2.07e-02	8.61e-02	3.79e-19
F8	Mean	4.39e-04	1.75e-03	7.97e-18
	Std	2.19e-03	4.11e-03	6.74e-18

5 结论

本文提出了一种基于膜计算理论的求解优化问题的全局优化方法,即进化膜算法.所提算法引入了细胞自动机和混沌搜索模拟液体分子做无规则运动的两个机制,很大程度地保持了候选解的多样性,避免进化膜算法陷入局部极值,增强了算法的全局寻优能力.针对不同维的单模和多模测试函数,使用所提算法与 CMAES 和 OXDE 算法进行了比较,结果表明所提 EMA 具有较好地收敛速度和保持候选解的多样性等优势.因此,进化膜算法在优化相关领域中有着良好的应用前景.

本文认为基于膜计算的进化膜算法有待更深入的研究,后续将从膜结构、反应规则、空间划分策略和膜之间的协同关系的设计等方向深入研究.

参考文献

[1] S A Hofmeyr, S Forrest. Architecture for an artificial immune system[J]. Evolutionary Computation, 2000, 8(4): 443 - 473.

[2] 郭一楠,刘丹丹,程健,王辉. 自适应混合变异文化算法[J]. 电子学报,2011,39(8):1973 - 1918.
Guo Yi-nan, Liu Dan-dan, Cheng Jian, Wang Hui. Adaptive cultural algorithm adopting mixed mutation[J]. Acta Electronica Sinica, 2011, 39(8): 1973 - 1918. (in Chinese)

[3] K H Han, J H Kim. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(6): 580 - 593.

[4] 张雪霞,陈维荣,戴朝华. 带局部搜索的动态多群体自适应差分进化算法及函数优化[J]. 电子学报,2010,38(8):1825 - 1830.

Zhang Xue-xia, Chen Wei-rong, Dai Chao-hua. Dynamic multi-group self-adaptive differential evolution algorithm with local search for function optimization[J]. Acta Electronica Sinica, 2010, 38(8): 1825 – 1830. (in Chinese)

[5] G Păun. Computing with membranes[J]. Journal of Computer and System Sciences, 2000, 61(1): 108 – 143.

[6] 张葛祥,潘林强. 自然计算的新分支——膜计算[J]. 计算机学报, 2010, 33(2): 208 – 214.
Zhang Gexiang, Pan Linqiang. A survey of membrane computing as a new branch of natural computing[J]. Chinese Journal of Computers, 2010, 33(2): 208 – 214. (in Chinese)

[7] 康琦,安静,汪镭,吴启迪. 自然计算的研究综述[J]. 电子学报, 2012, 40(3): 548 – 558.
Kang Qi, An Jing, Wang Lei, Wu Qi-di. Nature-inspired computation: A survey[J]. Acta Electronica Sinica, 2012, 40(3): 548 – 558. (in Chinese)

[8] L Pan, G Păun. Spiking neural P systems: An improved normal form[J]. Theoretical Computer Science, 2010, 411(6): 906 – 918.

[9] L Pan, X Zeng, X Zhang. Time-free spiking neural P systems [J]. Neural Computation, 2011, 23(5): 1320 – 1342.

[10] L Pan, M J Pérez-Jiménez. Computational complexity of tissue-like P systems[J]. Journal of Complexity, 2010, 26(3): 296 – 315.

[11] T-O Ishdorj, A Leporati, L Pan, X Zeng, X Zhang. Deterministic solutions to QSAT and Q3SAT by spiking neural P systems with pre-computed resources[J]. Theoretical Computer Science, 2010, 411(25): 2345 – 2358.

[12] L Pan, G Păun, M Pérez-Jiménez. Spiking neural P systems with neuron division and budding[J]. Science China Information Sciences, 2011, 54(8): 1596 – 1607.

[13] G Ciobanu, M J Pérez-Jiménez, G Păun. Applications of Membrane Computing [M]. Natural Computing Series Springer, 2006.

[14] T Y Nishida. An approximate algorithm for NP-complete optimization problems exploiting P systems[A]. Proceedings of Brainstorming Workshop on Uncertainty in Membrane Computing[C]. Palma de Majorca, 2004. 185 – 192.

[15] Liang Huang, Ning Wang, Jin-Hui Zhao. Multiobjective optimization for controller design[J]. Acta Automatica Sinica, 2008, 34(4): 472 – 477.

[16] Liang Huang, Ning Wang. An optimization algorithm inspired by membrane computing[A]. In Advances in Natural Computation[C]. Springer Berlin Heidelberg, 2006, 4222. 49 – 52.

[17] Gexiang Zhang, M Gheorghe, Chao-Zhong Wu. A quantum-inspired evolutionary algorithm based on P systems for knapsack problem[J]. Fundamenta Informaticae, 2008, 87(1): 93 – 116.

[18] Gexiang Zhang, Chunxiu Liu, M Gheorghe. Diversity and convergence analysis of membrane algorithms[A]. Proceedings of the 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)[C]. IEEE Press, 2010. 596 – 603.

[19] C Liu, M Han, X Wang. A multi-objective evolutionary algorithm based on membrane systems[A]. 2011 Fourth International Workshop on Advanced Computational Intelligence (IWACI)[C]. IEEE Press, 2011. 103 – 109.

[20] R Caponetto, L Fortuna, S Fazzino, M G Xibilia. Chaotic sequences to improve the performance of evolutionary algorithms [J]. IEEE Transactions on Evolutionary Computation, 2003, 7(3): 289 – 304.

[21] N Hansen, A Ostermeier. Completely derandomized self-adaptation in evolution strategies[J]. Evolutionary computation, 2001, 9(2): 159 – 195.

[22] Y Wang, Z Cai, Q Zhang. Enhancing the search ability of differential evolution through orthogonal crossover[J]. Information Sciences, 2011, 185(1): 153 – 177.

作者简介

刘　闯　男, 1984 年 5 月出生, 辽宁沈阳人, 大连理工大学博士研究生. 从事自然计算、智能技术及优化算法及复杂工业系统建模的研究. E-mail: chuang.liu@mail.dlut.edu.cn

韩　敏　女, 1959 年 8 月出生, 吉林延吉人, 教授、博士、博士生导师, 大连理工大学模糊信息处理与机器智能研究所副所长. 从事神经网络理论及其应用, 复杂系统建模及自适应控制、智能技术及优化算法的研究.
E-mail: minhan@dlut.edu.cn