

## 基于正交试验设计的克隆选择函数优化\*

余航<sup>1,2+</sup>, 焦李成<sup>1,2</sup>, 公茂果<sup>1,2</sup>, 杨咚咚<sup>1,2</sup>

<sup>1</sup>(西安电子科技大学 智能信息处理研究所,陕西 西安 710071)

<sup>2</sup>(西安电子科技大学 智能感知与图像理解教育部重点实验室,陕西 西安 710071)

### Clonal Selection Function Optimization Based on Orthogonal Experiment Design

YU Hang<sup>1,2+</sup>, JIAO Li-Cheng<sup>1,2</sup>, GONG Mao-Guo<sup>1,2</sup>, YANG Dong-Dong<sup>1,2</sup>

<sup>1</sup>(Institute of Intelligent Information Processing, Xidian University, Xi'an 710071, China)

<sup>2</sup>(Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education, Xidian University, Xi'an 710071, China)

+ Corresponding author: E-mail: yuhang9551@163.com

**Yu H, Jiao LC, Gong MG, Yang DD. Clonal selection function optimization based on orthogonal experiment design. *Journal of Software*, 2010,21(5):950-967. <http://www.jos.org.cn/1000-9825/3472.htm>**

**Abstract:** This paper presents a clonal selection operation: clonal selection operation based on orthogonal experiment design (CSO-OED). This design is later combined with the typical clonal selection operation and results in two algorithms: CSO+CSO-OED(I) adopting parallel mechanism and CSO+CSO-OED(II) adopting series mechanism. The validation in 9 classical benchmark functions and 6 complex functions has showed that CSO-OED can not only maintain the diversity of population, but also help avoid premature. Implemented in CSO+CSO-OED(I) and CSO+CSO-OED(II), the strategy that separates the local search and global search can not only guarantee the convergence but also improve the accuracy of global solution and the robustness of the algorithm.

**Key words:** artificial immune; evolutionary algorithm; artificial immune; clonal selection algorithm; orthogonal experiment design; function optimization

**摘要:** 将正交试验设计引入到克隆选择操作中,设计出基于正交试验的克隆选择操作(clonal selection operation based on orthogonal experiment design,简称 CSO-OED),并将其加入到典型的克隆选择算法中,设计出并联式的 CSO+CSO-OED(I)算法和串联式的 CSO+CSO-OED(II)算法.将新设计的算法用于 9 个经典的测试函数和 6 个复杂的测试函数进行对比测试,实验结果表明,CSO-OED 能够有效地保持种群的多样性,避免算法不成熟收敛.CSO+CSO-OED(I)和 CSO+CSO-OED(II)将全局搜索和局部搜索分开进行优化,对比实验表明,这种搜索策略不但能够保证算法的收敛性,还能有效地提高搜索解的精度,增强算法的鲁棒性.

**关键词:** 人工智能;进化算法;人工免疫;克隆选择算法;正交试验设计;函数优化

\* Supported by the National Natural Science Foundation of China under Grant Nos.60703107, 60703108 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2009AA12Z210 (国家高技术研究发展计划(863)); the National Basic Research Program of China under Grant No.2006CB705700 (国家重点基础研究发展计划(973)); the Program for New Century Excellent Talents in University of China under Grant No.NCET-08-0811 (新世纪优秀人才支持计划)

Received 2008-05-06; Accepted 2008-09-18; Published online 2009-04-10

中图法分类号: TP18

文献标识码: A

人工免疫系统(artificial immune system,简称 AIS)是受免疫学启发,模拟免疫学功能、原理和模型来解决复杂问题的自适应系统<sup>[1]</sup>。基于 Burnet 等人于 1959 年提出的克隆选择学说(clonal selection theory)的克隆选择算法是人工免疫系统的三大研究分支之一(另两个为负选择算法<sup>[2]</sup>和免疫网络算法<sup>[3]</sup>)。它是近年来的一个研究热点,吸引了大批学者对其进行研究。巴西学者 de Castro 等人提出了克隆选择算法(the clonal selection algorithm,简称 CLONALG)<sup>[4]</sup>。随后,又在 CLONALG 的基础上提出了免疫响应网络(artificial immune network,简称 aiNet)<sup>[5]</sup>,并将其成功应用到数据压缩<sup>[6]</sup>、聚类应用<sup>[7]</sup>、高维函数优化<sup>[5]</sup>等问题中。我国学者焦李成等人将 CLONALG 中的进化选择方式替换为精英选择,提出了新的克隆选择算法,并从免疫记忆、混沌、免疫优势等角度提出了一系列改进的克隆选择算法<sup>[8-12]</sup>。英国学者 Kim, Bentley 等人提出了动态克隆选择算法(dynamic clonal selection algorithm,简称 dynamiCS)<sup>[13]</sup>,并将其应用到计算机网络入侵检测中,取得了很好的效果。Timmis 等人随后又从生物免疫系统连续体细胞高频突变(contiguous somatic hypermutation)中抽象出一种变异算子,并结合 CLONALG 提出了 B-cell 算法<sup>[14]</sup>。意大利学者 Cutello 等人基于 CLONALG 设计了不同的高频变异操作,提出了用于优化的免疫算法 opt-IA<sup>[15]</sup>。这些算法成功地模拟了生物免疫系统的功能,也证明了人工免疫系统可以很好地解决现实中的优化问题。

正交试验设计<sup>[16]</sup>能够均衡地在解空间进行采样,对实验结果进行量化的分析和预测,这些优秀的特性也吸引了算法设计领域的众多专家对其进行研究和借鉴。香港学者 Yiu-Wing Leung 等人于 1999 年首次将正交试验设计应用于优化流媒体组播路由(multimedia multicast routing)的遗传算法(genetic algorithm)中,创新地提出了正交交叉算子(orthogonal crossover),并提出了用于离散变量的正交遗传算法(orthogonal genetic algorithm,简称 OGA)<sup>[17]</sup>。在 OGA 的基础上,Yiu-Wing Leung 随后又提出了适用于连续变量的量化正交遗传算法(orthogonal genetic algorithm with quantization,简称 OGA/Q)<sup>[18]</sup>。无论是适用于离散变量的 OGA,还是适用于连续变量的 OGA/Q 都成功地增强了算法全局搜索能力<sup>[18]</sup>。台湾学者 Shinn-Ying Ho 等人在 OGA 和 OGA/Q 的基础上首次引入了正交设计的直观分析机理,提出了新的交叉算子(orthogonal array crossover,简称 OAX;又称为 intelligent crossover)<sup>[19]</sup>。OAX 的显著特点是在对子代个体进行选择时,不是简单地从正交试验安排的个体中选择,而是运用正交试验设计的分析机制,对正交表中每个因素的最佳水平进行预测,从而组合出理论的最佳个体和次佳个体。台湾学者 Tsai Jinn-Tsong 等人进一步发掘正交试验设计的分析机制,创新性地将对正交试验设计的信噪比(signal-to-noise ratio,简称 SNR)引入遗传算法(genetic algorithm),提出了新算法——Hybrid Taguchi-Genetic Algorithm(简称 HTGA)<sup>[20]</sup>。在 HTGA 中,Tsai Jinn-Tsong 运用 SNR 对结果进行分析,预测出使实验结果最优的各个因素水平,组合成为最佳个体。从对比实验结果来看,HTGA 确实提高了算法的搜索精度和全局搜索能力。

本文的工作是将正交试验设计引入到克隆选择操作中,设计出了适合全局搜索的基于正交试验的克隆选择操作(clonal selection operation based on orthogonal experiment design,简称 CSO-OED)。对比实验表明,该操作能够有效地保持算法种群的多样性,特别是在算法的搜索前期具有非常快的收敛速度,在避免算法不成熟收敛方面也有一定的贡献。本文将 CSO-OED 与典型的克隆选择操作(CSO)加以结合,设计出并联的 CSO+CSO-OED(I)算法和串联式的 CSO+CSO-OED(II)算法。实验结果表明,这种将全局搜索和局部搜索分开的策略不但能够保证算法的收敛性,还能有效地提高搜索解的精度,增强算法的鲁棒性。另外,本文对比了并联式(I)型算法和串联式(II)型算法。实验结果表明,这两种算法都能以较快的收敛速度搜索到全局极值,但相对而言,并联式算法更具优势。

## 1 相关工作

### 1.1 克隆选择算法(clonal selection algorithm)

克隆选择算法的生物学模型来自于 1959 年 Burnet 等人提出的著名的克隆选择学说。该学说认为,当抗原侵

入生物体内时,生物体的免疫系统借助克隆(无性繁殖)使体内抗体激活、分化和增殖,以增加该抗体的数量,这一过程一般会持续几个星期,直至产生出与抗原完全匹配的抗体.这时,免疫系统会使匹配抗体成熟,一部分用于形成消灭抗原的血浆细胞,另一部分则形成长期记忆细胞,用于识别再次入侵的该种抗原.对于第 2 次入侵的抗原,由于体内存在相应抗体,免疫响应会更加迅速和有效,消灭抗原的时间会大为缩短<sup>[21]</sup>.免疫过程中所体现出的学习、记忆、抗体多样性等生物特性,正是人工免疫系统所借鉴的.

克隆选择算法正是借鉴克隆选择机理而形成的一种优化算法,其典型的算法流程<sup>[22,23]</sup>见算法 1.

**算法 1.** 典型的克隆选择算法.

Step 1. 随机生成初始化种群  $P$  和记忆种群  $M$ .

Step 2. 计算种群  $P$  中每个抗体的亲和度值.

Step 3. 选择  $P$  中亲和度最高的  $n$  个抗体进行克隆操作和变异操作.

Step 4. 选择  $P$  中  $j$  个亲和度最高抗体取代  $M$  中  $j$  个亲和度最低的抗体.

Step 5. 将  $P$  中亲和度最低的  $i$  个抗体用随机生成的抗体取代.

Step 6. 条件判断:如果满足终止条件,则停止进化,输出记忆抗体群  $M$ ;否则,跳往 Step 2.

从克隆选择操作的具体步骤可以看出,在克隆选择操作中,没有传统进化算法中的交叉操作,只有变异操作来推动抗体群进化,变异操作实现的是个体局部区域的搜索.在克隆选择算法中,克隆选择机理的实质是每个抗体通过克隆繁殖,在其解空间的邻近区域内进行贪婪的搜索,从而选择出更加优秀的抗体代替父代抗体<sup>[4]</sup>.这一特性特别适合局部搜索,使算法的搜索精确度显著提高,在已知克隆选择算法中,搜索精度高是每个克隆选择算法共有的优点.

但是,进化压力迫使每个个体尽可能地搜索到局部最优值,因此,克隆选择算法面临的主要问题是保持其抗体群的多样性<sup>[24]</sup>.在进化的过程中,每个个体始终趋向于局部的最优解,一旦寻找到局部最优解,则很难通过变异算子跳出该解,也正因为这个原因,研究学者纷纷在克隆选择算法中加入保持多样性的算子,如交叉算子、高频变异算子等<sup>[9,14,15]</sup>.

## 1.2 正交试验设计(orthogonal experiment design)

正交试验设计是多因素的优化实验设计方法,也称为正交设计,一般是从实验的全部样本点中挑选出部分有代表性的样本点进行实验,利用这些代表点所做的实验能够反映出每个因素各个水平对实验结果的影响.由于这些代表点具有正交性,因此称这组实验为正交试验.挑选正交的样本点,安排正交试验的过程,称为正交试验设计.

正交试验一般用正交表(orthogonal array)来安排实验,表 1 为 4 因素(factor)、3 水平(level)的正交表,记为  $L_M(Q^N)=L_9(3^4)$ ,其中  $L$  代表正交表, $M$  表示要做  $M$  次实验; $Q^N$  表示有  $N$  个因素,每个因素有  $Q$  个水平.表中每一纵列代表同一因素的不同水平;每一横行代表要运行的一次实验,实验完成后,将实验结果( $y_i$ )写在右侧.

**Table 1** Orthogonal array of  $L_9(3^4)$

表 1  $L_9(3^4)$ 的正交表

No.	Factor				Result $y_i$
	A	B	C	D	
1	1	1	1	1	$y_1$
2	1	2	2	2	$y_2$
3	1	3	3	3	$y_3$
4	2	1	2	3	$y_4$
5	2	2	3	1	$y_5$
6	2	3	1	2	$y_6$
7	3	1	3	2	$y_7$
8	3	2	1	3	$y_8$
9	3	3	2	1	$y_9$

利用正交表来安排实验,其优点是明显的:(1) 减少实验次数.对于上述实验,如果要进行全面实验共需要  $3^4=81$  次实验,而按照正交表的安排只需要 9 次实验——只需要部分实验即可.(2) 样本点分布的均衡性.在正交

表的每一列中,不同数字出现的次数相等,且都为  $M/Q$  次;将任意两列中同一行的两个数字看成有序数对,则每种数对出现的次数相等,都为  $M/Q^2$  次.正交试验设计中样本点的均衡性可以通过图 1 所示的  $L_4(2^3)$  的空间模型来进一步说明.图中每个轴线代表一个因素,正方体的 8 个顶点代表了全面实验的 8 个实验点,用正交表确定的 4 个样本点(黑点所示)均衡散布其中.具体来说,1) 正方体每个面 4 个顶点中恰有 2 个点是样本点;2) 每条棱上 2 个顶点中恰有 1 个是样本点;3) 分别沿轴线方向投影,在映射面上样本点恰好完全遍布其中.这些特点适应于一般情况.

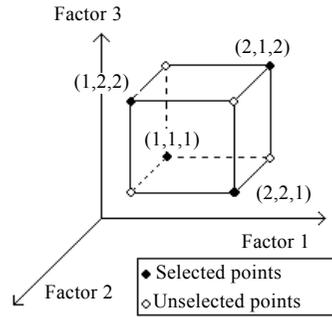


Fig.1 Space model of  $L_4(2^3)$

图 1  $L_4(2^3)$  的空间模型

利用正交表来安排实验的另一个显著优点是可以独立地量化评价同一个因素的各个水平,从而对未进行实验的因素组合进行预测.有如下定义:

$$S_{j,k} = \sum_{i=1}^M y_i \times F_i \tag{1}$$

其中,  $S_{j,k}$  表示因素  $j$  的水平  $k$  对实验的主要影响,  $y_i$  表示输出结果.当第  $i$  个实验中因素  $j$  的水平等于  $k$  时,  $F_i=1$ ; 否则,  $F_i=0$ .例如,  $S_{A,1}=y_1+y_2+y_3, S_{A,2}=y_4+y_5+y_6, S_{A,3}=y_7+y_8+y_9$ .观察表 1 可以注意到,当因素  $A$  中同一水平的输出结果相加时,因素  $B$  的水平在 3 个相加结果中恰好都存在,因此,  $S_{A,1}, S_{A,2}$  和  $S_{A,3}$  比较时,可以忽略因素  $B$  的影响,同理也可以忽略  $C$  和  $D$  的影响.这样,通过比较  $S_{A,1}, S_{A,2}$  和  $S_{A,3}$  就可以判断出因素  $A$  中 3 个水平对实验的不同影响.分别计算  $S_{j,k}$  就可以得到每个因素各个水平对于实验的影响,从而可以预测出最佳的因素水平组合搭配方式.通过计算同一因素各水平主要影响的标准方差  $R_{ss}$ ,还可以判断出各个因素对实验结果影响的剧烈程度.

正交试验设计既保证了样本点分布的均衡性,也能在整个实验过程中独立地评价每个因素的各个水平,而且能够估计各个因素对实验结果影响的剧烈程度.这些特性对于增加算法的搜索效率、减少函数的评价次数、判断不同变量对函数优化的影响,必定会有十分有益的作用.文献[18]给出的一般正交表的算法构造方法产生正交表为  $L_M((Q)^N)=[o_{i,j}]_{M \times N}$ ,其中,  $M, Q, N$  满足  $M=Q \times (N-1)+1$ .

## 2 基于正交试验的克隆选择操作

本节将正交试验设计引入到智能算法中,设计出有益于全局搜索的基于正交试验设计的克隆选择操作.该操作可以分为两个子操作:基因库(genetic library)和正交克隆操作,下面将对其进行详述.

### 2.1 基因库

从生物学的角度来讲,遗传是通过细胞核中的染色体来实现的,每条染色体是由数量不等的具有遗传信息的基因(gene)组成的.染色体上每一个基因座(loca)代表一个独立的显性性状(如眼睛的颜色、耳垂的有无等),而在基因座上的基因就是决定相应显性形状的信息(如眼睛是褐色的、有耳垂等).因此,依次给全部基因座选择不同基因就决定了一个子代个体的全部性状.基因库就是这些不同基因座上全部基因的总和.每个物种基因库的丰富程度决定了该物种多样性的程度,一个物种所包含的基因越丰富,这个物种适应环境的能力就越强<sup>[25]</sup>.在遗

传进化的过程中,基因库是保持种群多样性的重要途径.

受以上生物基因库机制的启发,本文设计基于正交试验的基因库.类似于生物学概念,基因库是由个体所有基因座上基因组成的集合.在函数优化问题中,抗体(antibody)对应优化函数的一个解,基因座对应于优化函数中的变量,基因对应于变量有意义的取值,基因库对应于所有变量具有代表性的取值的矩阵,它是由解空间各个变量有限个可对算法搜索起到促进作用的值组成的矩阵.

### 2.1.1 基因库的构造(construction of genetic library)

不失一般性,本文考虑如下优化问题:

$$\begin{cases} \min & f(\mathbf{X}) \\ \text{s.t.} & \mathbf{L} \leq \mathbf{X} \leq \mathbf{U} \end{cases} \quad (2)$$

其中,  $\mathbf{X}=(x_1, x_2, \dots, x_N) \in \mathbf{R}^N$  为  $N$  维变量,  $f(\mathbf{X})$  为优化函数,  $\mathbf{L}=(l_1, l_2, \dots, l_N) \in \mathbf{R}^N$  为  $N$  维变量的下界,  $\mathbf{U}=(u_1, u_2, \dots, u_N) \in \mathbf{R}^N$  为  $N$  维变量的上界,  $x_i \in [l_i, u_i]$ . 一个  $\mathbf{X}$  就对应一个抗体,  $1, 2, \dots, N$  表示第  $1, 2, \dots, N$  个基因座,  $x_i$  表示第  $i$  个基因座上的基因值. 若每个基因座上具有  $G$  个代表性的值, 则基因库就是一个矩阵  $\mathbf{M}^{gene} = [m_{i,j}^{gene}]_{G \times N}$ . 为了对基因库中的基因进行定量评估, 本文提出了基因评估矩阵  $\mathbf{M}^{fitness} = [m_{i,j}^{fitness}]_{G \times N}$ .  $\mathbf{M}^{fitness}$  与  $\mathbf{M}^{gene}$  中的值具有一一对应的关系, 下面对其进行确定.

#### A. 解空间量化(quantization of solution space)

一般意义下, 优化问题的变量参数均是连续变量, 不能像生物学中的基因那样由一定的氨基酸组成<sup>[21]</sup>, 因此要借鉴生物学中的基因库, 首先要做的就是对解空间的量化操作. 量化操作由式(3)完成.

$$x_{i,j} = l_i + (j-1) \times \frac{u_i - l_i}{Q-1}, \quad i=1, 2, \dots, Q \quad (3)$$

这样, 每维变量被均匀地量化为  $Q$  份. 量化后的解空间组成  $Q \times N$  维矩阵  $\mathbf{X}^{quant} = [x_{i,j}]_{Q \times N}$ , 由该矩阵中的值可以确定  $Q^N$  个不同的抗体.

#### B. 构造基因库(construct genetic library)

将基因座看作正交试验设计中的因素, 将基因座上的基因看作正交试验设计中的水平, 构造正交表:  $\mathbf{L}_M((Q^{ini})^N) = [o_{i,j}]_{M \times N}$ , 从而得到初始抗体群:  $\mathbf{A}^{ini}(0) = [a_{i,j}^{ini}(0)]_{M \times N} = [x_{(o_{i,j}+1),j}]_{M \times N}$ , 将  $\mathbf{A}^{ini}(0)$  代入优化函数计算初始群体适应度值  $\mathbf{F}^{ini}(0) = [f^{ini}(0)]_{M \times 1}$ , 从而形成伴随矩阵  $[\mathbf{A}^{ini}(0) | \mathbf{F}^{ini}(0)]_{M \times (N+1)}$ .

按照式(1), 分别计算每个基因座上各个基因的适应度值  $S_{j,k}$ , 由此可得评估矩阵  $\mathbf{S}^{fitness} = [S_{j,k}]_{Q^{ini} \times N}$ . 由  $S_{j,k}$  的计算过程可知,  $S_{j,k}$  与第  $j$  个基因座上的第  $k$  个基因具有一一对应的关系. 在  $\mathbf{S}^{fitness}$  的每个列向量上, 分别选取  $G$  个最佳值 (本文中优化函数取最小值, 因此最佳值是那些适应度值最小的值), 组成基因评估矩阵  $\mathbf{M}^{fitness} = [m_{i,j}^{fitness}]_{G \times N}$ . 选择与  $m_{i,j}^{fitness}$  对应的基因  $x_{i,j}^{gene}$  组成基因库矩阵  $\mathbf{M}^{gene} = [x_{i,j}^{gene}]_{G \times N}$ .

这样就完成了基因库矩阵  $\mathbf{M}^{gene}$  和基因评估矩阵  $\mathbf{M}^{fitness}$  的构造. 从构造过程可以看出, 基因库中的基因是在整个解空间中挑选能够促进算法搜索的各个基因座上的优秀值, 是算法搜索空间中各变量参数最佳基因的组合, 因此, 基因库中的基因具有全局性的特点. 基因评估矩阵  $\mathbf{M}^{fitness}$  是对基因库矩阵  $\mathbf{M}^{gene}$  中每个基因优劣的量化评价, 与基因库矩阵  $\mathbf{M}^{gene}$  具有一一对应的关系, 是算法在搜索过程中选取基因的判断依据.

### 2.1.2 基因库的应用(application of genetic library)

基因库中的基因是由全局范围内挑选优秀的基因组成的, 因此基因库的主要作用是保持群体的多样性, 避免算法陷入局部优值. 具体操作如下:

A. 最佳抗体预测(anticipation for best antibody). 根据基因评估矩阵  $\mathbf{M}^{fitness}$ , 从基因库矩阵  $\mathbf{M}^{gene}$  中选择每个基因座最优秀的基因组成预测最佳抗体  $\mathbf{A}^{anti} = [a_i^{anti}]_{1 \times N}$ . 预测最佳抗体  $\mathbf{A}^{anti}$  是潜在的最优抗体, 它可能是经过正交表已安排的基因组合, 也可能是正交表未安排的基因组合. 通过这种正交试验的分析机制, 不仅可以得到均衡分布于解空间的抗体群, 而且可以预测出除了抗体群之外的潜在优秀抗体, 这就有效地减少了算法的函数评价次数, 加速了算法收敛速度.

- B. 正交初始化(orthogonal initialization).根据基因评估矩阵  $M^{fitness}$ ,应用 MonteCarlo 赌轮<sup>[26]</sup>从  $M^{gene}$  中给每个基因座各选择  $M$  个基因,组成初始种群  $A(0)=[a_{i,j}]_{M \times N}$ .应用 MonteCarlo 赌轮的策略是出于让优秀的基因可以被多次应用于抗体的考虑.
- C. 正交免疫基因操作(orthogonal immune genetic operation).在群体进化的每一代中,运用基因库来指导变异操作,增强抗体群的多样性.具体做法是,在变异阶段,依照变异概率  $P$  对克隆增殖后的抗体进行按位的变异,凡是被选择变异的基因,从基因库对应的基因座随机地选择基因将其代替.
- D. 正交凋亡操作(orthogonal apoptosis operation).区别于死亡,生物学中的凋亡是一个主动的过程,是生物新陈代谢的必需<sup>[27]</sup>.凋亡操作就是借鉴生物学中的凋亡机制设计而成的,一般的凋亡操作是用随机生成新抗体的方法替代凋亡的抗体,而正交凋亡操作是用基因库中新产生的抗体来代替凋亡抗体,其产生方法与由基因库矩阵  $M^{gene}$  产生初始种群的方法相同.

基因库代替了部分传统上需要随机产生数据的方法,对比实验表明,它增强了算法在全局范围内的寻优能力,有效地增强了算法的多样性,能够避免算法的不成熟收敛.

## 2.2 正交克隆操作(orthogonal clonal operation)

在典型的克隆操作中,每个父代抗体通过无性繁殖复制为相同的多个子代抗体,然后通过变异操作将这些子代抗体散布于父代抗体的临近区域中.子代抗体具有随机分布特性,算法不能保证抗体在解空间内均匀分布,搜索效率很低,搜索容易陷入局部最优,搜索精度不高.为了克服这一缺点,本文将正交试验设计引入到克隆选择算法中,提出了正交克隆操作.正交克隆操作与传统方法最大的区别就是应用正交试验设计中实验点均衡分布的特性,让每个克隆的子代抗体在以父代抗体为中心的局部空间中均衡分布.这样做可以使算法的搜索效率得到很大的提高.假设对抗体  $A=[a_{i,j}]_{1 \times N}$  实施正交克隆操作,则具体步骤如下:

### 2.2.1 确定搜索空间(decide the search space)

对于不同的抗体,正交克隆操作首先确定搜索空间.当对抗体  $[a_{i,j}]_{1 \times N}$  进行操作时,其正交搜索空间是以  $[a_{i,j}]_{1 \times N}$  为中心的一个  $N$  维空间,各个边长为上、下界限的  $\frac{1}{\omega \times Q}$ ,即

$$\left\{ X \mid \bigcap_{j=1}^N \left( \left( a_{i,j} - \frac{u_j - l_j}{2 \times \omega \times Q} \right) \leq x_j \leq \left( a_{i,j} + \frac{u_j - l_j}{2 \times \omega \times Q} \right) \right) \right\} \quad (4)$$

其中,  $X=(x_1, x_2, \dots, x_N)$ ,  $\omega$  是自适应系数,控制着正交搜索空间的大小,也影响着基于正交克隆操作的搜索精度.考虑到基因库基因之间的距离为上、下界限的  $1/Q$ ,本文以进化代数  $gen$  的负指数形式给出  $\omega=c^{-gen}$ ,  $c \in (1, 1.5)$ ,这样,随着进化代数的增加,基于正交克隆操作的搜索精度也在提高.对于位于解空间边界的抗体,搜索的空间会有部分超出解空间,此时将舍去对解空间以外区域的搜索.

### 2.2.2 确定正交表的因素(choose the factors of orthogonal array)

正交克隆增殖操作就是根据正交试验设计,在已经确定的搜索空间里均衡地产生子代抗体的操作.在应用正交试验设计构造正交抗体之前,首先需要确定参加正交试验的各个因素,针对因素的不同划分,本文提出两种策略来确定正交试验的因素,即分裂法和组合法.

分裂法是在所有基因座中,按照一定的策略将所有基因座分裂为两部分:(1) 参加正交试验的基因座;(2) 不参加正交试验的基因座.将参加正交试验的每一个基因座作为参加正交试验的因素,安排正交表,而将未被选中的基因座仅仅通过复制操作,然后与正交表进行按位合并即可.每组基因位于其分裂前相应的基因座位置,从而组成克隆增殖后的子代抗体.分裂法的实质是在搜索空间内确定一个低维子空间进行搜索,而对高维空间的补空间不进行搜索.它实现了对不同基因座进行更具针对性的搜索(对结果影响剧烈的基因座被挑中的概率较大),在低维子空间内,基因的分布具有均衡性,从而增加了群体的多样性,提高了算法的搜索效率.这适合于优化函数的参数数量较多、各个参数比较独立的情况.

组合法是将多个基因座进行组合,作为一个统一的因素来参加正交试验,每一个因素含有一个或者多个基因座,参加正交试验时,同一因素代表的所有基因座同增、同减,作为整体,按照相同的比例进行量化操作.组合法

是将搜索空间通过变量的合并化一进行降维,缩减为低维空间,适合于优化函数的不同变量相互影响的情况.

不失一般性,设正交试验的因素数目为  $V^{clonal}$ ,水平数为  $Q^{clonal}$ ,共有  $M^{clonal}$  种排列组合,对抗体  $A=(a_1, a_2, \dots, a_N)$  进行正交克隆增殖操作.分裂法根据极差  $R_{st}$  应用 MonteCarlo 赌轮<sup>[26]</sup>,选择出  $V^{clonal}$  个不同的基因座  $Loca=[loca_1, loca_2, \dots, loca_{V^{clonal}}]$ ,这样做保证了对抗原反应剧烈的基因座会以更高的概率被选中.组合法是对抗体  $A$  的基因座进行组合,划分为  $V^{clonal}$  个部分,分别为  $[a_{i,1}, a_{i,2}, \dots, a_{i,part(1)}], [a_{i,part(1)+1}, a_{i,part(1)+2}, \dots, a_{i,part(2)}], \dots, [a_{i,part(V^{clonal}-1)+1}, a_{i,part(V^{clonal}-1)+2}, \dots, a_{i,part(V^{clonal})}]$ ,将每个部分看作一个整体,作为一个因素参加正交试验.虽然每个部分有多个基因座,但统一根据正交表的同一列来取值,从而构造出正交克隆抗体群.

### 2.2.3 正交克隆增殖操作(orthogonal clonal propagation operation)

正交克隆增殖操作是根据确定的搜索空间和正交表的因素设计正交表,产生子代抗体的操作.由于正交表的因素分别按照分裂法和组合法确定,下面分别按照两种方法给出正交克隆增殖操作.

A. 分裂法.首先结合选出的基因座  $Loca=[loca_1, loca_2, \dots, loca_{V^{clonal}}]$ ,按照公式(3),对公式(4)确定的搜索空间进行量化,得到:

$$x_{i,j}^{quant} = a_{i,j} - \frac{u_j - l_j}{2 \times \omega \times Q} + (i-1) \times \frac{u_j - l_j}{\omega \times Q \times (Q^{clonal} - 1)} \quad (5)$$

其中,  $i=1, 2, \dots, Q^{clonal}$ ,  $j=loca_1, loca_2, \dots, loca_{V^{clonal}}$ .构造正交表  $L_{M^{clonal}}((Q^{clonal})^{V^{clonal}})=[o_{i,j}^{clonal}]_{M^{clonal} \times V^{clonal}}$ ,从而得到子代抗体的正交部分基因:  $A^{orth}(k)=[a_{i,j}^{orth}(k)]_{M^{clonal} \times V^{clonal}}=[x_{(o_{i,j}^{clonal}+1),j}^{quant}]_{M^{clonal} \times V^{clonal}}$ ,将未被选中的其他  $N-V^{clonal}$  位基因座  $A^{remain}=[a_1^{remain}, a_2^{remain}, \dots, a_{N-V^{clonal}}^{remain}]$  仅进行复制:  $[A^{remain}, A^{remain}, \dots, A^{remain}]_{M^{clonal} \times (N-V^{clonal})}^T$ ,其中,  $[A]^T$  表示对行向量的转制.将  $A^{orth}(k)$  和  $A^{remain}$  进行合并,组成克隆增殖后的子代抗体  $A^{offspring}(k)$ ,在合并过程中,每个基因返回到对应基因座所在列的位置.

$$A^{offspring}(k)=[a_{i,j}^{offspring}]_{M^{clonal} \times N} = \begin{cases} a_{i,j}^{orth}, & j \in Loca \\ a_{i,j}^{remain}, & j \notin Loca \end{cases} \quad (6)$$

B. 组合法.开始与分裂法相同,对整个搜索空间进行量化和构造正交表.然后,根据抗体  $A$  的  $V^{clonal}$  个部分  $[a_{i,1}, a_{i,2}, \dots, a_{i,part(1)}], [a_{i,part(1)+1}, a_{i,part(1)+2}, \dots, a_{i,part(2)}], \dots, [a_{i,part(V^{clonal}-1)+1}, a_{i,part(V^{clonal}-1)+2}, \dots, a_{i,part(V^{clonal})}]$ ,确定克隆增殖的子代抗体,每个部分的不同基因座具有一致的水平,同增、同减,统一根据正交表的同一列来取值,从而构造出正交克隆抗体群.对第  $m$  个部分  $[a_{i,part(m-1)+1}, a_{i,part(m-1)+2}, \dots, a_{i,part(m)}]$ ,由正交表构造的水平如下:

$$\begin{cases} [a_{i,j}^{orth}(k)] = [x_{(o_{i,m}^{clonal}+1),j}^{quant}] \\ j = part(m-1)+1, part(m-1)+2, \dots, part(m) \end{cases} \quad (7)$$

将各个基因座上产生的子代抗体的基因组合在一起,就组成克隆增殖后的子代抗体  $A^{offspring}(k)$ .

## 2.3 CSO-OED

将基因库和正交克隆操作按照一定的方式进行联合,本文提出基于正交试验的克隆选择操作(clonal selection operation based on orthogonal experiment design,简称 CSO-OED),该操作的流程见算法 2.

算法 2. 基于正交实验的克隆选择操作(CSO-OED).

Step 1. 构造基因库:基因库矩阵  $M^{gene}$  和基因评估矩阵  $M^{fitness}$ .

Step 2. 正交初始化:应用 MonteCarlo 赌轮,从基因库中给每个基因座各选择  $M-1$  个基因,加上预测抗体  $A^{anti}$ ,组成  $M$  个抗体.

Step 3. 正交克隆增殖操作:分别对每个抗体进行正交克隆操作.

Step 4. 正交免疫基因操作:采用基因库指导变异操作.

Step 5. 克隆选择:计算变异后子代抗体的适应度值,选择最优秀的抗体与父代抗体进行比较,选择其中优秀者作为新的子代抗体.

Step 6. 正交凋亡操作:采用基因库指导凋亡操作.

Step 7. 条件判断:满足条件,算法结束搜索;不满足条件,跳转到 Step 3 继续搜索.

### 3 算法描述

克隆选择算法具有强有力的局部搜索能力,搜索解的精度很高,但在保持群体多样性方面有志欠缺,容易出现不成熟收敛,而正交试验设计能够保证解在解空间的均匀散布性.根据这一特性,本文将 CSO-OED 与典型的克隆选择算法相结合,设计出了局部搜索+全局搜索的两种算法:

- 并联式的克隆选择操作+基于正交试验的克隆选择操作(CSO+CSO-OED(I)).
- 串联式的克隆选择操作+基于正交试验的克隆选择操作(CSO+CSO-OED(II)).

#### 3.1 CSO+CSO-OED(I)

CSO+CSO-OED(I)的流程如图 2 所示.

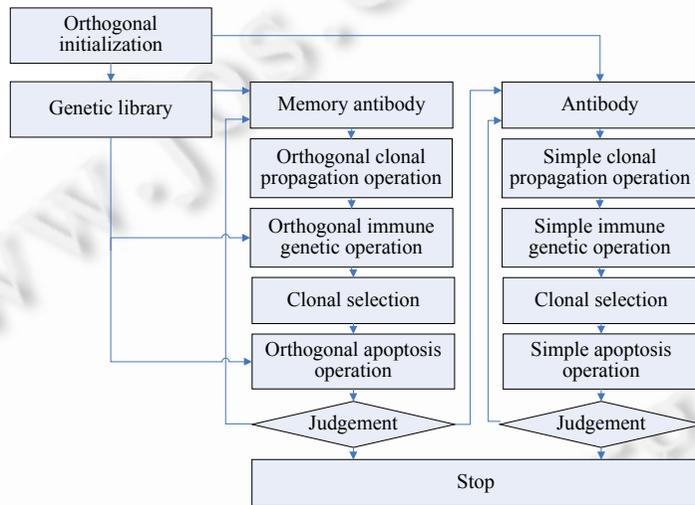


Fig.2 Flow chart of CSO+CSO-OED(I)

图 2 CSO+CSO-OED(I)流程图

可以看出,本算法的最大特色是,算法在搜索过程中使用局部搜索和全局搜索分开的并联机制,应用偏重于局部搜索的 CSO 进行局部搜索,应用偏重于全局搜索的 CSO-OED 进行全局搜索,当进化完一代后,局部搜索与全局搜索互换交流搜索信息.为了加以区别,将基于正交试验的克隆选择操作应用于记忆抗体的进化,而将典型克隆选择操作应用于一般抗体群的进化.两者的区别如下:

- 在初始化(initialization)过程中,记忆抗体群(memory antibody)的产生是由基因库产生的;抗体群由正交初始化(orthogonal initialization)群体  $A^{mi}(0)$  中最优秀的抗体直接组成.
- 记忆抗体群应用正交克隆增殖操作(orthogonal clonal propagation operation);抗体应用一般的克隆增殖操作(simple clonal propagation operation).
- 记忆抗体群应用基因库指导的正交免疫基因操作(orthogonal immune genetic operation)和正交凋亡操作(orthogonal apoptosis operation);抗体群应用一般的基于按位变异的免疫基因操作操作(simple immune genetic operation)和凋亡操作(simple apoptosis operation).
- 每一代进化结束后,记忆抗体将其最优秀的抗体传递给抗体群,而抗体群不将抗体信息传递给记忆抗体群.之所以只将记忆抗体群中的最优抗体传递给抗体群,是由于记忆抗体群偏重于全局搜索,而抗体群偏重于局部搜索,具有更高的精度.

### 3.2 CSO+CSO-OED(II)

CSO+CSO-OED(II)的流程如图 3 所示.

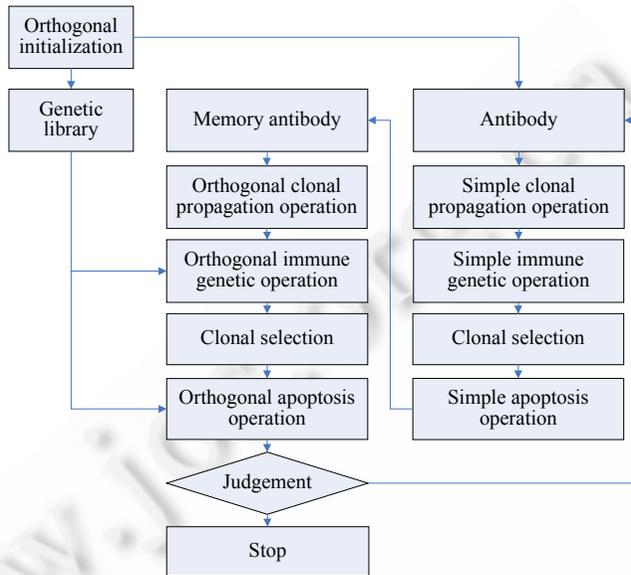


Fig.3 Flow chart of CSO+CSO-OED(II)

图 3 CSO+CSO-OED(II)流程图

可以看出,与 CSO+CSO-OED(I)相比,CSO+CSO-OED(II)是将局部搜索和全局搜索串联在一起,CSO 与 CSO-OED 相互轮流交替进行搜索.首先进行的是抗体群的进化,其初始值来源于正交初始化群体  $A^{mi}(0)$ ,待其进化一代结束后,将抗体群直接全部传递给记忆抗体群进行进化,待记忆抗体群进化完成后,进行条件判断,如果满足结束条件,则停止算法搜索;否则,记忆抗体群将其值传递给抗体群进行下一代的进化,周而复始,直至满足终止条件.

## 4 算法有效性验证

### 4.1 对经典函数进行优化搜索

为了验证算法的性能,本文应用 9 个被广泛采用的经典测试函数(见附录)对 CSO+CSO-OED(I), CSO+CSO-OED(II), OGA/Q<sup>[18]</sup>和标准粒子群算法(PSO)<sup>[28]</sup>进行对比测试.为了使每种算法充分搜索,设定终止条件是函数评价次数为 20 000 次,每种算法独立运行 30 次.统计结果见表 2.

在表 2 中,最优值、最劣值分别是指算法 30 次搜索到的最优的、中间的和最差的全局最优值.从以上对 9 个经典测试函数的测试数据可以看出,CSO+CSO-OED(I)和 CSO+CSO-OED(II)均有能力搜索到更加精确的最优值,好于 OGA/Q 和 PSO.从 30 次搜索中各种算法搜索的最劣值可以看出,CSO+CSO-OED(I)拥有 5 个最优,4 个次优;CSO+CSO-OED(II)拥有 5 个最优,2 个次优.相比之下,OGA/Q 和 PSO 的表现就差了很多,特别是对  $f_2, f_3, f_6, f_7$  和  $f_9$ , OGA/Q 和 PSO 明显陷入了不成熟收敛,其最劣值的数量级远远高于 CSO+CSO-OED(I)和 CSO+CSO-OED(II)搜索的值,这说明 CSO+CSO-OED(I)和 CSO+CSO-OED(II)具有更高的成功率和更强的鲁棒性.从 30 次搜索中各种算法搜索的中间值可以看出,除了在  $f_4$  的中间值 CSO+CSO-OED(II)略差于 PSO(同一个数量级比较)以外,CSO+CSO-OED(I)和 CSO+CSO-OED(II)均好于 OGA/Q 和 PSO.而在标准方差一栏中,CSO+CSO-OED(I)和 CSO+CSO-OED(II)算法优秀的鲁棒性和搜索最优解精确性再次得到证明:除了在  $f_5$  的搜索上 CSO+CSO-OED(II)差于 PSO 以外,在其他函数上,CSO+CSO-OED(I)和 CSO+CSO-OED(II)明显好于

OGA/Q 和 PSO.在一些函数中,如  $f_1, f_2, f_3, f_5, f_6, f_7, f_8$  和  $f_9$ ,本文提出的算法搜索最优解标准方差比 OGA/Q 和 PSO 至少要好 12 个数量级.

**Table 2** Experimental results on 9 classical benchmark functions

表 2 9 个经典函数的实验结果

	The best reaching value			
	CSO+CSO-OED(I)	CSO+CSO-OED(II)	OGA/Q	PSO
$f_1$	-1.256948e+004	-1.256948e+004	-1.256942e+004	-1.095082e+004
$f_2$	0	0	0	2.188908e+001
$f_3$	8.881784e-016	8.881784e-016	1.000467e-001	4.954562e-005
$f_4$	0	0	0	5.738470e-007
$f_5$	1.570544e-032	1.570544e-032	8.866527e-004	1.472335e-009
$f_6$	1.349783e-032	1.349783e-032	3.741506e-003	1.272269e-007
$f_7$	0	8.236092e-084	6.285746e-000	2.679471e-008
$f_8$	1.951784e-006	1.945260e-004	1.471729e-004	6.356012e-003
$f_9$	0	0	0	4.440892e-014
	The worst reaching value			
	CSO+CSO-OED(I)	CSO+CSO-OED(II)	OGA/Q	PSO
$f_1$	-1.256948e+004	-1.256948e+004	-1.250436e+004	-9.272909e+003
$f_2$	0	0	2.123545e-000	5.173780e+001
$f_3$	7.993605e-015	8.881784e-016	5.088814e-000	2.117243e-003
$f_4$	9.857284e-003	1.299903e-001	4.535828e-000	1.026111e-001
$f_5$	3.509917e-021	7.505341e-001	4.552832e+005	1.072067e-001
$f_6$	2.277165e-020	1.349783e-032	1.500286e+006	1.098788e-002
$f_7$	7.174648e-043	4.278249e-047	5.332908e+002	1.983468e-005
$f_8$	1.830208e-004	1.213231e-002	7.380042e-004	1.888119e-002
$f_9$	2.746727e-025	2.442697e-024	7.646252e-001	4.984912e-007
	Median reaching value			
	CSO+CSO-OED(I)	CSO+CSO-OED(II)	OGA/Q	PSO
$f_1$	-1.256948e+004	-1.256948e+004	-1.255268e+004	-9.805903e+003
$f_2$	0	0	5.835239e-001	3.084370e+001
$f_3$	8.881784e-016	8.881784e-016	1.213292e-000	4.181567e-004
$f_4$	0	1.354669e-002	1.258491e-000	1.231953e-002
$f_5$	1.570544e-032	1.570544e-032	1.212092e-001	1.2108421e-008
$f_6$	2.462633e-024	1.349783e-032	4.934904e-000	2.464084e-006
$f_7$	4.668282e-051	8.814972e-051	5.435002e+001	1.040477e-006
$f_8$	6.683415e-005	2.477782e-003	2.638256e-002	1.170649e-002
$f_9$	5.973397e-026	1.074998e-025	3.980195e-001	1.643156e-008
	Standard deviation			
	CSO+CSO-OED(I)	CSO+CSO-OED(II)	OGA/Q	PSO
$f_1$	4.866568e-012	4.721264e-012	1.714526e+001	3.941987e+002
$f_2$	0	0	7.323107e-001	8.404605e-000
$f_3$	1.588821e-015	0	1.385042e-000	6.098531e-004
$f_4$	2.685081e-003	2.928765e-002	1.229672e-000	2.925425e-002
$f_5$	7.835399e-022	2.323148e-001	1.017832e+005	2.397213e-002
$f_6$	5.061555e-021	2.808011e-048	3.603391e+005	2.452591e-003
$f_7$	1.604300e-043	9.538100e-048	1.239749e+002	6.542510e-006
$f_8$	4.370936e-005	3.945353e-003	2.095701e-002	3.970775e-003
$f_9$	7.119982e-026	7.130794e-025	2.583917e-001	1.304965e-007

各个函数的 30 次优化结果箱图如图 4 所示.其中,横轴上 I 代表 CSO+CSO-OED(I),II 代表 CSO+CSO-OED(II),O 代表 OGA/Q,P 代表 PSO.从图 4 的 9 个箱图中可以看出,CSO+CSO-OED(I)和 CSO+CSO-OED(II)在对这 9 个经典优化函数的搜索中,无论在搜索的成功率上还是在搜索精度上均好于 OGA/Q 和 PSO.

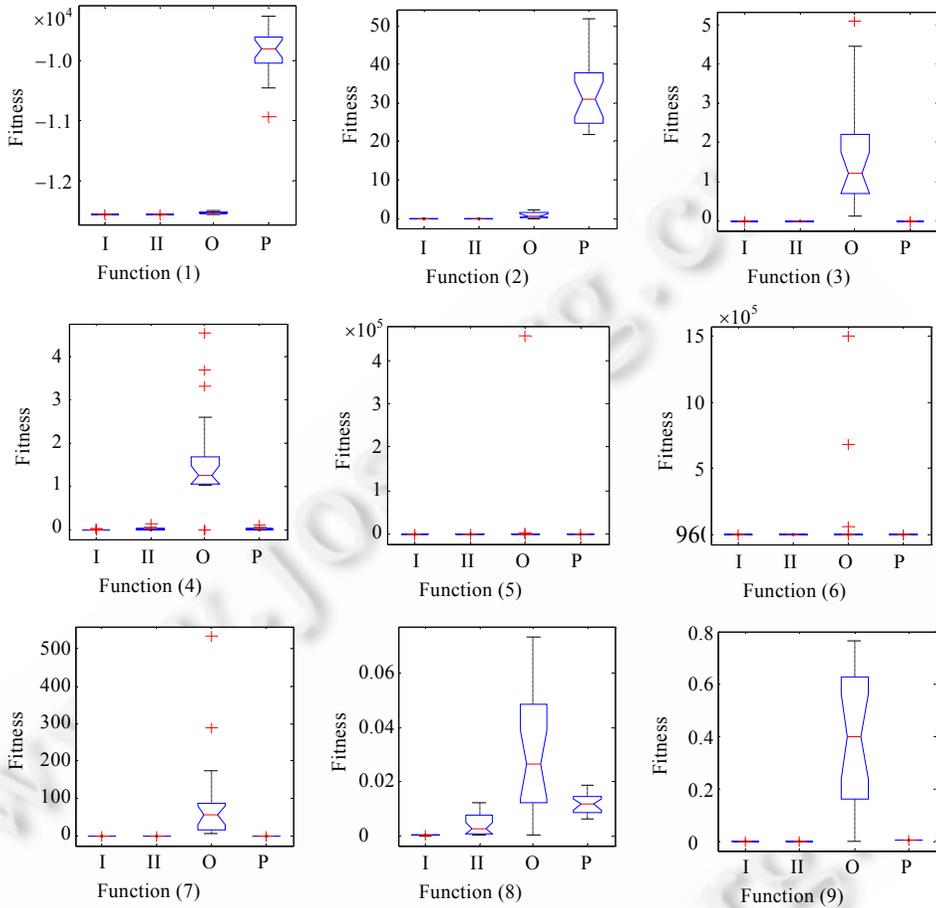


Fig.4 Box plots of the results on 9 classical benchmark functions

图 4 9 个经典函数对比测试结果箱图

图 5 给出算法对这 9 个函数搜索的在线平均图:各种算法对函数进行独立优化 30 次,然后取每个迭代搜索到最优值 30 次的平均,然后画出各种算法的在线搜索曲线.

由于 PSO 无论在收敛速度上还是在精确度上都明显劣于其他 3 种算法,存在多个数量级的差异,因此,为了便于观察,本文在这里只画出 CSO+CSO-OED(I),CSO+CSO-OED(II)和 OGA/Q 的群体进化函数,如图 5 所示.其中,实线代表 CSO+CSO-OED(I),点线代表 CSO+CSO-OED(II),点划线代表 OGA/Q.

从图 5 中可以看出,本文提出的 CSO+CSO-OED(I)和 CSO+CSO-OED(II)在收敛速度上均优于 OGA/Q,特别是在搜索到最优值附近时,本文提出的算法能够继续向最优值进行搜索,提高搜索解的精度;而 OGA/Q 则陷入停滞状态,无法搜索到更加精确的解.

从对以上 9 个经典函数优化的统计结果来看,CSO+CSO-OED(I)和 CSO+CSO-OED(II)均好于 OGA/Q 和 PSO.本文提出的这两种算法,相对而言,CSO+CSO-OED(I)更加优秀一些,但并不明显.因此,本文在后面对 6 个复杂函数的优化搜索中,将进一步对 CSO+CSO-OED(I)和 CSO+CSO-OED(II)进行深入比较和研究.

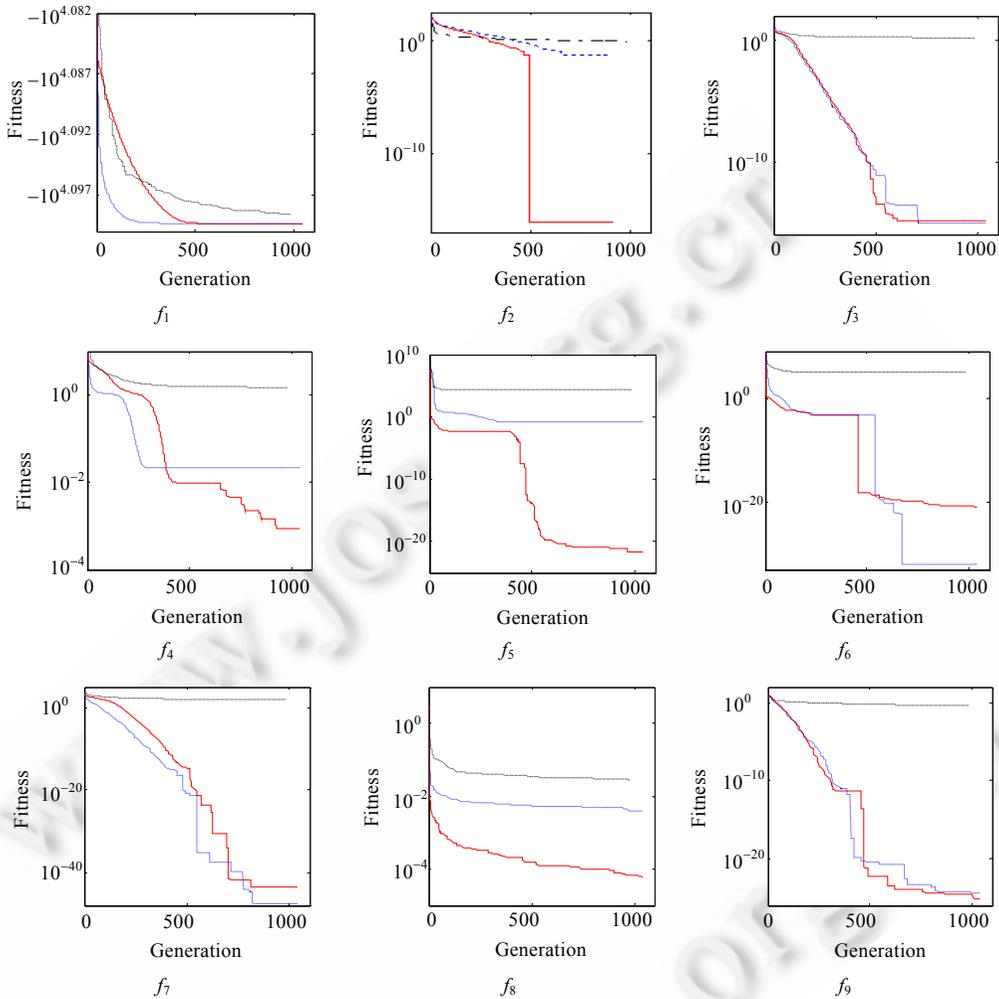


Fig.5 Mean values versus generations over 30 independent runs on 9 classical benchmark functions  
图 5 对 9 个经典函数独立测试 30 次的平均值迭代曲线

#### 4.2 对复杂函数进行优化搜索

实际的优化问题往往非常复杂,为了验证算法的有效性,本文采用 6 个复杂的函数<sup>[29]</sup>对该算法作进一步的验证.这些函数非常复杂,是由多个函数经过旋转、平移等操作叠加而成的,最优值的位置具有非对称性和随机性.我们分别将本文提出的算法、OGA/Q 和标准粒子群算法(PSO)应用于这 6 个函数进行对比.考虑到这 6 个函数非常复杂,算法很难搜索到全局最优值,因此,对比实验策略为:选取足够多次函数评价次数(设定为 50 000 次),让每种算法搜索到尽可能好的最优值.各种算法 30 次搜索的统计结果如图 6 所示.其中,横轴上 I 代表 CSO+CSO-OED(I),II 代表 CSO+CSO-OED(II),O 代表 OGA/Q,P 代表 PSO.从这些图中可以看出,尽管除了对  $f_1$  和  $f_2$  以外没有一种算法能够搜索到函数的全局最优值(全局最优值为 0),但仍然可以比较出算法的优劣. CSO+CSO-OED(I)和 CSO+CSO-OED(II)对  $f_1, f_2, f_4, f_5, f_6$  的搜索均好于 OGA/Q 和 PSO,特别是对  $f_1, f_2, f_6$  的搜索.在对  $f_3$  的搜索中,CSO+CSO-OED(I)和 PSO 基本上都好于 CSO+CSO-OED(II)和 OGA/Q.而从最优解的稳定性来看,CSO+CSO-OED(I)的搜索更为出色.本文提出的算法在求解精度上均好于其他算法,且本算法的搜索成功率和稳定性均好于其他算法.

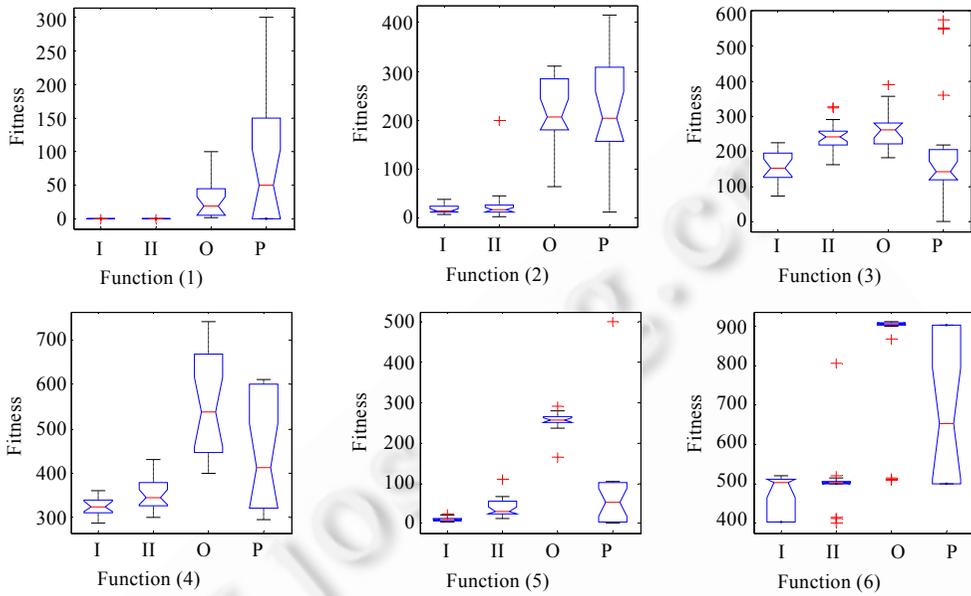


Fig.6 Box plots of the results on 6 complex functions  
图 6 6 个复杂函数对比测试结果箱图

4.3 本文提出的3种算法的比较

为了区别研究局部搜索和全局搜索两种策略对算法的影响,本文设计如下实验:只用局部搜索部分 CSO、只用全局搜索部分 CSO-OED、使用局部+全局搜索的并联形式 CSO+CSO-OED(I)和使用局部+全局搜索的串联形式 CSO+CSO-OED(II)对这 6 个函数进行优化测试,实验中设定终止条件仍为函数评价次数为 50 000 次,实验结果见表 3.

Table 3 Experimental results on 6 complex functions for the tree proposed algorithms

表 3 本文提出的 3 种算法对 6 个复杂函数的实验结果

	The best reaching value				The worst reaching value			
	CSO-OED	CSO	CSO+CSO-OED (I)	CSO+CSO-OED (II)	CSO-OED	CSO	CSO+CSO-OED (I)	CSO+CSO-OED (II)
$f_1$	8.7948E+01	7.2000E-19	1.5329E-19	8.1568E-16	1.2197E+02	2.0000E+02	2.7990E-13	9.8555E-08
$f_2$	8.1004E+01	6.0864E+00	7.5240E+00	2.3214E+00	9.8805E+01	3.3030E+02	1.8133E+01	2.0000E+02
$f_3$	2.9548E+02	1.7413E+02	7.3583E+01	1.6254E+02	4.3257E+02	9.0000E+02	1.4278E+02	3.2741E+02
$f_4$	3.8209E+02	4.4501E+02	2.8828E+02	2.9996E+02	4.5988E+02	9.0000E+02	3.4973E+02	4.3097E+02
$f_5$	6.5566E+01	4.2266E+01	3.8849E+00	1.1397E+01	8.3780E+01	1.1390E+02	8.5873E+00	3.2638E+01
$f_6$	4.3287E+02	9.0000E+02	4.0237E+02	4.0155E+02	5.3116E+02	9.0000E+02	5.0042E+02	5.2161E+02
	Median reaching value				Standard deviation			
	CSO-OED	CSO	CSO+CSO-OED (I)	CSO+CSO-OED (II)	CSO-OED	CSO	CSO+CSO-OED (I)	CSO+CSO-OED (II)
$f_1$	1.0730E+02	3.3000E-10	2.8000E-13	1.8070E-10	1.2586E+01	6.7495E+01	1.5160E-13	1.3778E-07
$f_2$	9.1733E+01	4.3957E+01	1.4660E+01	1.5540E+01	4.9551E+00	1.2405E+02	2.9704E+00	5.7417E+01
$f_3$	3.6198E+02	2.6786E+02	1.5234E+02	2.4024E+02	3.8653E+01	2.0933E+02	2.1790E+01	4.0981E+01
$f_4$	4.2163E+02	5.9904E+02	3.2551E+02	3.4608E+02	2.4732E+01	1.2795E+02	1.9988E+01	3.4046E+01
$f_5$	7.7628E+01	7.3455E+01	8.5193E+00	3.0848E+01	5.6767E+00	2.1897E+01	1.6898E+00	7.3991E+00
$f_6$	4.3753E+02	9.0000E+02	5.0467E+02	5.0201E+02	4.4602E+01	0.0000E+00	2.9931E+01	3.6041E+01

从表 3 可以得出:

1. 最优值.进行 30 次算法测试,各种算法获得的最优值相互比较,CSO+CSO-OED(I)测到 4 个最优值,1 个次优值,1 个第三优值;CSO+CSO-OED(II)搜索到 2 个最优值,3 个次优值,1 个第三优值;CSO 算法搜索到 2 个次优值,2 个第三优值,2 个最差值;CSO-OED 算法测到 2 个第三优值,4 个最差值.这说明:1) 在解的

精确性上,使用局部搜索+全局搜索的策略好于只使用局部搜索和只使用全局搜索的策略;2) 可以注意到,只要局部算法能够搜索到最优值附近,就有能力搜索到比较精确的全局最优值,局部搜索算法在精度上好于全局搜索算法.

2. 最劣值.进行 30 次算法测试,各种算法获得的最劣值相互比较,CSO+CSO-OED(I)测到全部 6 个最优值;CSO+CSO-OED(II)搜索到 5 个次优值,1 个第三优值;CSO 算法测到 6 个最差值;CSO-OED 算法测到 1 个次优值,5 次第三优值.这说明:1) 在成功跳出局部优值的能力方面,使用局部搜索+全局搜索的策略好于只使用局部搜索和只使用全局搜索的策略,特别是并联的局部搜索+全局搜索策略更加优良.这主要是串联的联合搜索策略局部搜索和全局搜索间隔执行,每次执行完全局搜索后即向局部搜索交接数据,进行局部搜索,这样就破坏了全局搜索的连贯性,降低了全局搜索算法中抗体群的多样性;2) 单独使用局部搜索或使用全局搜索,都不能成功地避免不成熟收敛问题,但相比而言,全局搜索能够比较有效地跳出局部优值点,更容易避免不成熟收敛,而局部算法更容易陷入函数编造者精心设计的局部最优值(100,200,300,400,500,600,700,800,900)之中.
3. 中间值.进行 30 次算法测试,各种算法获得的中间值相互比较,CSO+CSO-OED(I)测到 5 个最优值,1 个第三优值;CSO+CSO-OED(II)搜索到 6 个次优值;CSO 算法测到 3 个第三优值,3 个最差值;CSO-OED 算法搜索到 1 个全局最优值,1 个第三优值,4 个最差值.这说明:1) 在解的稳定性方面,使用局部搜索+全局搜索的策略好于只使用局部搜索和只使用全局搜索的策略;2) 在进行算法有效性验证的过程中我们注意到, $f_6$  对于任何一种算法都是一个“灾难”,它有很多局部优值点,各种算法很难从中跳出来.从以上的数据也可以注意到,各种算法对函数 6 的搜索最不成功.而相对而言,全局搜索算法对函数 6 的搜索比较成功,甚至好于局部搜索+全局搜索的组合算法.这也说明全局搜索在保持群体多样性和避免不成熟收敛方面具有更加优秀的能力.
4. 标准方差.各种算法获得的标准方差相互比较,CSO+CSO-OED(I)得到 5 个最好值,1 个次好值;CSO+CSO-OED(II)得到 1 个次好值,5 个第三好值;局部搜索算法得到 1 个最好值,5 个差值;全局搜索算法测到 4 个次好值,1 个第三好值,1 个差值.注意到,在对  $f_6$  的测试过程中,局部搜索算法的最优值、最劣值和中间值均为 900,这说明局部搜索算法在对函数  $f_6$  最优值进行搜索的过程中,均陷入了局部极值(900),而且从实际的搜索曲线观察,局部搜索算法在搜索的前 5 代即可搜索到 900,但直至最终的 400 多代也不能取得任何进步.因此,局部搜索算法对  $f_6$  搜索得到的标准方差为 0 就不足为奇.这不能反映出局部搜索算法对函数  $f_6$  的搜索能力更强,应加以忽略.因此,从标准方差这一栏可以看出,CSO+CSO-OED(I)>CSO+CSO-OED(II)>CSO-OED 算法>CSO 算法.这一结果进一步表明全局搜索算法具有很强的全局搜索能力,局部搜索算法则更容易陷入局部极值.而使用局部搜索+全局搜索的策略综合性能最好,成功率最高,搜索解最稳定.无论在避免不成熟收敛上,还是在搜索精度上,均表现最好.

将 30 次得到的结果画出箱图,如图 7 所示.

由图 7 可以看出,在对  $f_2, f_3, f_4, f_5$  的优化搜索中,第 3 列和第 4 列的值得明显好于第 1 列和第 2 列的值,这说明局部搜索+全局搜索的效果好于只使用全局搜索和只使用局部搜索得到的效果.在对  $f_1$  的搜索中,第 3 列和第 4 列与第 2 列的值近乎相同,而第 1 列的值比第 2 列和第 3 列高出一个数量级,这说明,当算法搜索到全局最优值附近时,局部搜索的效果要好于全局搜索的效果.在对  $f_6$  的搜索中,第 3 列和第 4 列与第 1 列的效果几乎相同,但距离真正的全局最优值(0)还存在差距,甚至第 1 列的平均值要好于第 3 列和第 4 列的平均值,而第 2 列的值得明显陷入了局部最优值,这说明,在避免不成熟收敛的能力上,CSO-OED 搜索算法好于 CSO 搜索算法.

对 6 个函数分别只用局部搜索部分 CSO 和只用全局搜索部分 CSO-OED 进行优化,其搜索的在线迭代曲线如图 8 所示.其中,实线代表 CSO-OED 搜索,点划线代表 CSO 搜索.由图中可以看出,在对  $f_2, f_3, f_4, f_5, f_6$  的优化中,CSO-OED 搜索到的全局最优值的均值均小于 CSO,特别是从对  $f_4, f_6$  的搜索可以看出,CSO 完全陷入了局部优解,无法成功跳出.相比之下,CSO-OED 能够更加成功地跳出局部优解.从对  $f_1$  的搜索可以看出,在都能搜索到全局最优解附近时,局部搜索算法能够搜索到更加精确的全局最优解.这也进一步说明,本文提出的 CSO-OED

确实能够提高算法的全局搜索能力,能够有效地避免算法不成熟收敛.

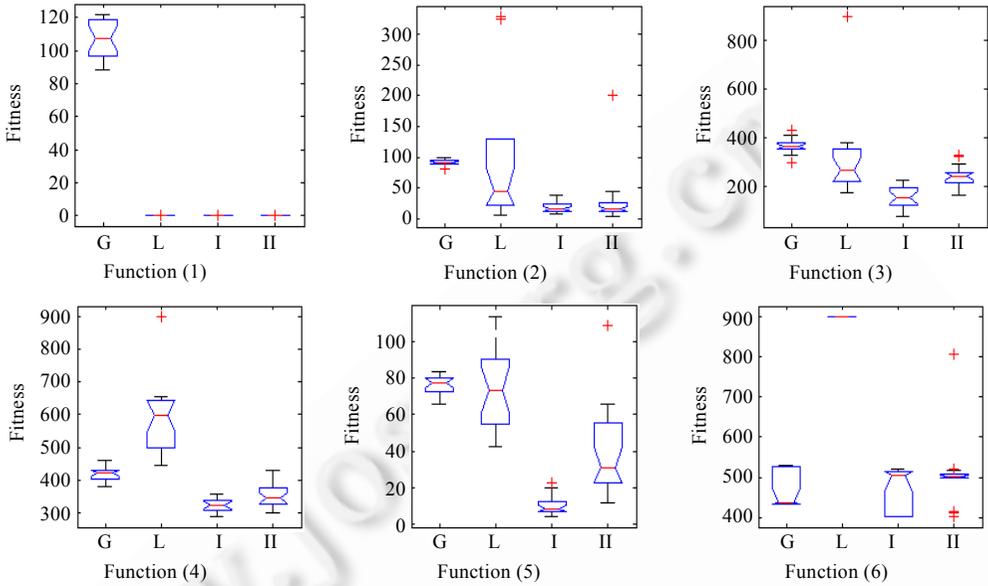


Fig.7 Box plots of the experimental results on 6 complex functions for the three proposed algorithms

图7 本文提出的3种算法对6个复杂函数实验结果的箱图

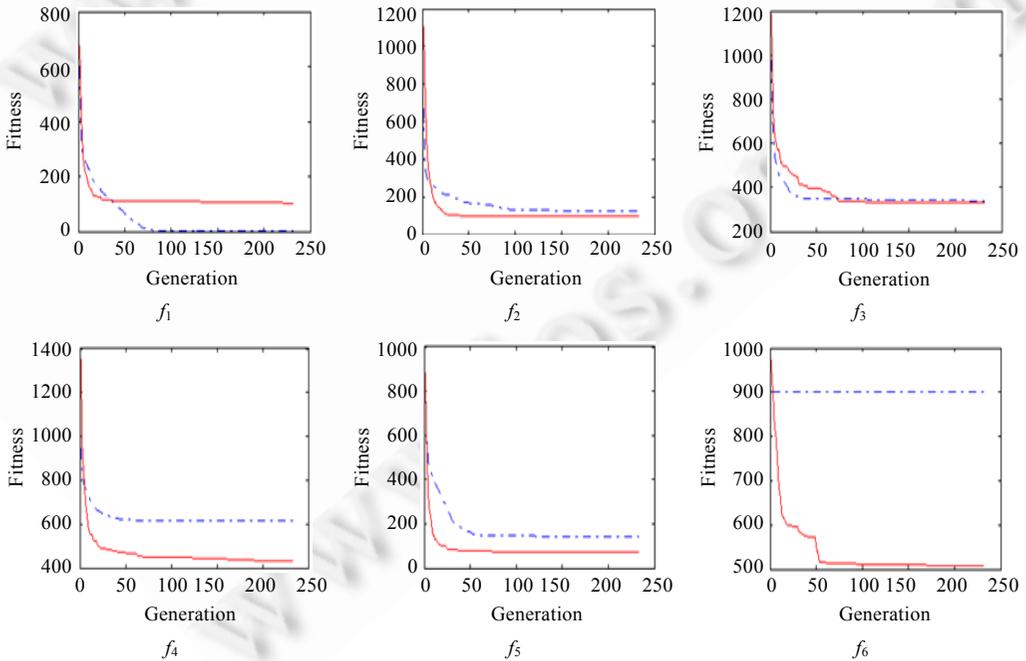


Fig.8 Mean values versus generations over 30 independent runs on 6 complex functions for CSO and CSO-OED

图8 CSO与CSO-OED对6个复杂函数独立测试30次的平均值迭代曲线

另外,记录 CSO+CSO-OED(I)和 CSO+CSO-OED(II)对  $f_6$  进行 30 次优化的迭代曲线,如图 9 所示.实线代表 CSO-OED 部分每代的搜索迭代曲线,点划线代表 CSO 部分每代的搜索迭代曲线.从该图中可以看出,在局部优

值(如图 9(a)所示的 900,550,500、图 9(b)所示的 740,700,600,550)处,局部搜索算法容易陷入停滞状态,而全局搜索算法则能够比较容易地跳出局部最优值点;在搜索的前期,实线大多数时间是在点划线的下方,这说明在前期搜索中,全局搜索更加容易跳出局部最优值,搜索到新的最优值;在搜索后期,全局搜索出现停滞状态,而局部搜索则能够搜索到更加精确的最优值,局部搜索的局部搜索能力明显强于全局搜索,局部搜索可以保证算法搜索到更加精确的解.全局搜索部分确实起到了避免不成熟收敛的作用,局部搜索部分确实起到了提高解的准确性的作用.

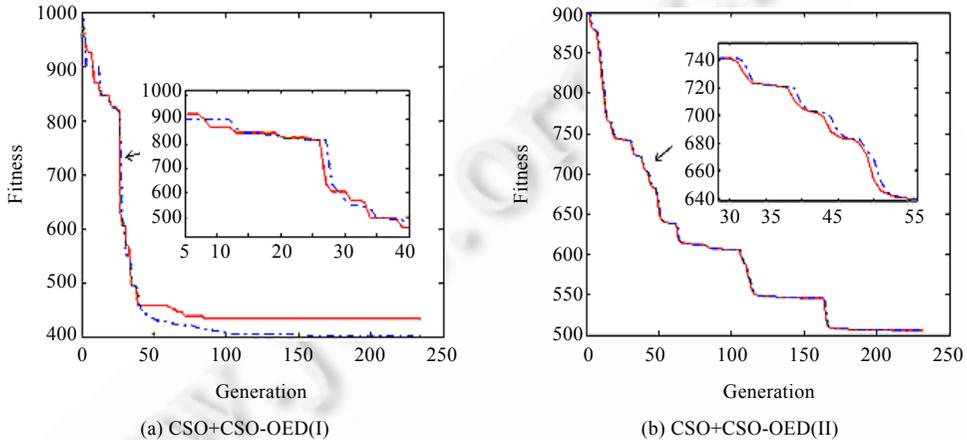


Fig.9 Mean values versus generations over 30 independent runs on  $f_6$  for CSO+CSO-OED(I) and CSO+CSO-OED(II)

图9 CSO+CSO-OED(I)和 CSO+CSO-OED(II)对  $f_6$  独立测试 30 次的平均值迭代曲线

## 5 结论与展望

本文将正交试验设计技术引入到克隆选择操作中,设计出基于正交试验的克隆选择操作,并将其加入到典型的克隆选择算法中,设计出并联的 CSO+CSO-OED(I)算法和串联式的 CSO+CSO-OED(II)算法.大量对比结果表明,CSO+CSO-OED(I)算法和 CSO+CSO-OED(II)算法在函数优化中是有效的,具有搜索精度高、鲁棒性强的优点.这种将全局搜索和局部搜索分开的策略能够有效地提高算法搜索的效率和精度.另外,将适合局部搜索的 CSO 和适合全局搜索的 CSO-OED 进行对比实验,结果表明,本文设计的适合全局搜索的操作是有效的,全局搜索在算法搜索前期收敛速度快于局部搜索,全局搜索具有更强的跳出局部极值的能力.

今后的工作将放在以下几个方面:1) 继续研究提高智能算法的全局搜索和避免早熟的能力,将新的机制引入到免疫算法中;2) 从理论上研究全局搜索的数学模型,从理论的角度证明全局搜索的有效性;3) 将算法用于指导具体问题中.

## References:

- [1] de Castro LN, Timmis J. Artificial Immune Systems: A New Computational Intelligence Approach. Berlin: Springer-Verlag, 2002.
- [2] Forrest S, Perelson AS, Allen L, Cherukuri R. Self-Nonself discrimination in a computer. In: Proc. of the 1994 IEEE Computer Society Symp. on Research in Security and Privacy. Los Alamitos: IEEE Computer Society Press, 1994. 202-212.
- [3] Farmer JD, Packard NH, Perelson AS. The immune system, adaptation, and machine learning. Physica (D), 1986,2(1-3):187-204.
- [4] de Castro LN, von Zuben FJ. Learning and optimization using the clonal selection principle. IEEE Trans. on Evolutionary Computation, 2002,6(3):239-251.
- [5] de Castro LN, von Zuben FJ. aiNet: An artificial immune network for data analysis. In: Abbass HA, Sarker RA, Newton CS, eds. Data Mining: A Heuristic Approach. Hershey: Idea Group Publishing, 2001. 231-259.

- [6] de Castro LN, von Zuben FJ. Immune and neural network models: Theoretical and empirical comparisons. In: Nebel B, ed. Proc. of the 17th Int'l Joint Conf. on Artificial Intelligence (IJCAI 2001). Seattle: Morgan Kaufmann Publishers, 2001. 239–257.
- [7] de Castro LN, von Zuben FJ. An evolutionary immune network for data clustering. In: Carlos HC, Ribeiro, Felipe MG, FranGa, eds. Proc. of the IEEE 6th Brazilian Symp. on Neural Networks. Rio de Janeiro: IEEE Press, 2000. 84–89.
- [8] Du HF, Gong MG, Jiao LC, Liu RC. A novel artificial immune system algorithm for high-dimensional function numerical optimization. *Progress in Natural Science*, 2005,15(5):463–471.
- [9] Du HF, Gong MG, Liu RC, Jiao LC. Adaptive chaos clonal evolutionary programming algorithm. *Science in China: Series F—Information Sciences*, 2005,48(5):579–595.
- [10] Gong MG, Du HF, Jiao LC. Optimal approximation of linear systems by artificial immune response. *Science in China: Series F—Information Sciences*, 2006,49(1):63–79.
- [11] Liu RC, Jiao LC, Du HF. Clonal strategy algorithm based on the immune memory. *Journal of Computer Science and Technology*, 2005,20(5):728–734.
- [12] Gong MG, Jiao LC. A population-based artificial immune system for numerical optimization. *Neurocomputing*, 2008. DOI: 10.1016/j.neucom.2007.12.041
- [13] Kim J, Bentley PJ. Towards an artificial immune system for network intrusion detection: An investigation of dynamic clonal selection. In: Fogel DB, El-Sharkawi MA, Yao X, Greenwood G, Iba H, Marrow P, Shackleton M, eds. Proc. of the 2002 Congress on Evolutionary Computation (CEC 2002). Honolulu: IEEE Press, 2002. 1015–1020.
- [14] Kelsey J, Timmis J. Immune inspired somatic contiguous hypermutation for function optimisation. In: Cantu-Paz E, *et al.*, eds. Proc. of the 2003 Genetic and Evolutionary Computation Conf. (GECCO 2003). Chicago: Springer-Verlag, 2003. 207–218.
- [15] Cutello V, Narzisi G, Nicosia G, Pavone M. Exploring the capability of immune algorithms: A characterization of hypermutation operators. In: Nicosia G, Cutello V, Bentley PJ, Timmis JI, eds. Proc. of the 3rd Int'l Conf. on Artificial Immune Systems (ICARIS 2004). Catania: Springer-Verlag, 2004. 263–276.
- [16] Montgomery DC. Design and Analysis of Experiments. 3rd ed., New York: Wiley, 1991.
- [17] Zhang Q, Leung YW. An orthogonal genetic algorithm for multimedia multicast routing. *IEEE Trans. on Evolutionary Computation*, 1999,3(1):53–62.
- [18] Leung YW, Wang YP. An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Trans. on Evolutionary Computation*, 2001,5(1):41–53.
- [19] Ho SY, Chen JH, Huang MH. Inheritable genetic algorithm for biobjective 0/1 combinatorial optimization problems and its applications. *IEEE Trans. on Systems, Man, and Cybernetics*, 2004,34(1):609–620.
- [20] Tsai JT, Liu TK, Chou JH. Hybrid Taguchi-genetic algorithm for global numerical optimization. *IEEE Trans. on Evolutionary Computation*, 2004,8(4):365–377.
- [21] Chen WF, *et al.* Medical Immunology. 4th ed., Beijing: People's Medical Publishing House, 2006 (in Chinese).
- [22] Timmis J. Artificial immune systems—Today and tomorrow. *Natural Computing*, 2007,6(1):1–18.
- [23] Timmis J, Andrews P, Owens N, Clark E. An interdisciplinary perspective on artificial immune systems. *Evolutionary Intelligence*, 2008,1(1):5–26.
- [24] Garrett S. How do we evaluate artificial immune systems? *Evolution Compute*, 2005,13(2):145–177.
- [25] Krishnamurthy KV, Wrote; Zhang ZW, *et al.*, Trans. Textbook of Biodiversity. Beijing: Chemical Industry Press, 2005 (in Chinese).
- [26] Chen GL, Wang XF, Zhuang ZQ, Wang DS. Genetic algorithm and its application. Beijing: Posts & Telecommunications Press, 1996 (in Chinese).
- [27] Yu H, Gong MG, Jiao LC, Zhang B. Clonal selection algorithm with immunologic regulation for function optimization. In: Hao Y, Liu J, Wang Y, Cheung YM, Yin H, Jiao LC, Ma J, Jiao YC, eds. Proc. of the 2005 Int'l Conf. on Computational Intelligence and Security. Xi'an: Springer-Verlag, 2005. 858–863.
- [28] Kennedy J, Eberhart RC. Particle swarm optimization. In: Proc. of the IEEE Int'l Conf. on Neural Networks. Piscataway: IEEE Press, 1995. 1942–1948.
- [29] Liang JJ, Suganthan PN, Deb K. Novel composition test functions for numerical global optimization. In: Proc. of the 2005 IEEE Int'l Swarm Intelligence Symp. Pasadena: IEEE Computer Society Press, 2005. 68–75.

附中文参考文献:

- [21] 陈慰峰,等.医学免疫学.第4版,北京:人民卫生出版社,2006.
- [25] Krishnamurthy KV,著;张正旺,等,译.生物多样性教程.北京:化学工业出版社,2005.
- [26] 陈国良,王煦法,庄镇泉,王东生.遗传算法及其应用.北京:人民邮电出版社,1996.

附录. 9 个经典的测试函数及其性质

Table 4 Nine classical benchmark functions

表 4 9 个经典的测试函数

Classical benchmark functions	Dimensions	Optimal value
$f_1 = \sum_{i=1}^N (-x_i \sin(\sqrt{ x_i }))$	30	-12 569.48
$f_2 = \sum_{i=1}^N (x_i^2 - 10 \cos(2\pi x_i) + 10)$	30	0
$f_3 = -20 \exp\left(-0.2 \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2}\right) - \exp\left(\frac{1}{N} \sum_{i=1}^N \cos(2\pi x_i)\right) + 20 + \exp(1)$	30	0
$f_4 = \frac{1}{4000} \sum_{i=1}^N x_i^2 - \prod_{i=1}^N \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	0
$f_5 = \frac{\pi}{N} \left\{ 10 \sin^2(\pi y_i) + \sum_{i=1}^{N-1} (y_i - 1)^2 \cdot [1 + 10 \sin^2(\pi y_i + 1)] + (y_N - 1)^2 \right\} + \sum_{i=1}^N u(x_i, 10, 100, 4)$		
where: $y_i = 1 + \frac{1}{4}(x_i + 1)$	30	0
$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & a < x_i \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$		
$f_6 = \frac{1}{10} \left\{ \sin^2(3\pi x_i) + \sum_{i=1}^{N-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_N - 1)^2 [1 + \sin^2(3\pi x_N)] \right\} + \sum_{i=1}^N u(x_i, 5, 100, 4)$	30	0
$f_7 = \sum_{i=1}^N x_i^2$	30	0
$f_8 = \sum_{i=1}^N x_i^4 + \text{random}[0, 1)$	30	0
$f_9 = \sum_{i=1}^N  x_i  + \prod_{i=1}^N  x_i $	30	0



余航(1982—),男,陕西西安人,博士生,助教,主要研究领域为自然计算,数据挖掘,模式分类.



公茂果(1979—),男,博士,副教授,CCF 高级会员,主要研究领域为自然计算,图像处理,数据挖掘.



焦李成(1959—),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为自然计算,数据挖掘,图像处理.



杨咚咚(1982—),男,博士生,主要研究领域为进化计算,机器学习.