文章编号: 1001-0920(2011)01-0059-06

应用于高维优化问题的免疫进化算法

刘星宝[1,2], 蔡自兴[1], 王 勇[1], 彭伟雄[1]

(1. 中南大学 信息科学与工程学院，长沙 410083；2. 湖南商学院 现代教育技术中心，长沙 410205)

摘 要: 针对免疫算法在全局优化过程中多样性不足的问题, 提出一种新型的免疫进化算法. 随机克隆扩张和多受体随机编辑算子是该算法的主要特色, 同时引入改进的超变异算子加强个体的学习能力; 提出一种新的算法性能评价准则, 以比较不同算法在全局优化中的表现. 实验环节中, 首先确定了克隆扩张比; 然后将免疫进化算法与快速克隆算法和Opt-IMMALG算法进行比较. 结果表明, 免疫进化算法明显优于另外两种算法.

关键词: 人工免疫系统；全局优化问题；性能评价准则

中图分类号: TP18 **文献标识码:** A

Novel immune evolutionary algorithm for global optimization

LIU Xing-bao[1,2], *CAI Zi-xing*[1], *WANG Yong*[1], *PENG Wei-xiong*[1]

(1. School of Information Science and Engineering，Central South University，Changsha 410083，China；2. Center of Modern Education Technology，Hu'nan University of Business，Changsha 410205，China．Correspondent：LIU Xing-bao，E-mail：liuxb0608@gmail.com)

Abstract: In order to increase the diversity of immune algorithm when solving global optimization problems, a novel immune evolutionary algorithm(IEA) is proposed. The main characteristics of IEA are clonal expansion and multiple-parent random receptor editor operators. In addition, a modified hypermutation operator is introduced to improve the learning ability of individuals. Particularly, a novel performance evaluation criterion is constructed, by which the performance of different algorithms can be compared easily. In the experimental study, the ratio of clonal expansion is determined, and the IEA is compared with fast clonal algorithm(FCA) and Opt-IMMALG. The results show that IEA is significantly better than FCA and Opt-IMMALG.

Key words: artificial immune systems；global optimization；performance evaluation criterion

1 引 言

克隆选择学说认为, 抗体是存在于细胞表面的内在免疫产物, 可与抗原发生选择性的反应, 该反应可激发抗体的一系列免疫过程, 并表现出记忆、学习、自适应和抗体多样性等特征[1]. 受此启发, 文献[2]提出了B细胞网络模型, 这是最初的人工免疫系统模型. 此后, [3]与[4]同时将克隆选择原理引入到优化领域中, 提出了克隆选择算法. 克隆选择算法又称为免疫算法, 包含抗体初始化、抗体克隆扩张、克隆超变异和抗体更新等几个阶段. 当抗体不满足给定的条件时, 克隆、变异和更新几个阶段重复执行, 这点同其他的仿生优化算法类似. 克隆选择算法中, 克隆和超变异是区别于其他群体算法的重要算子.

在免疫算法中, 根据扩张机制的不同, 克隆扩张分为静态和动态两种. 在静态克隆扩张中, 每个抗体都产生相同的、固定数目的克隆副本[5-7], 因此能够将自身携带的信息完整地传递到下一代, 保持了群体的多样性信息, 同时也延缓了算法的收敛速度. 在动态克隆扩张中, 抗体的克隆数目与进化代数、抗体-抗体亲和度及抗体-抗原亲和度等因素相关[8-13], 致使优秀个体能产生更多的副本, 这些副本通过超变异后操作完成对母体邻域的精细搜索. 另一方面, 普通个体因克隆副本的数目较少, 使算法对其邻域的搜索比较粗糙, 被更新的频率不高. 所以在动态克隆扩张中, 从某种意义上而言, 优秀个体的邻域搜索过程驱动了整个群体的进化, 在加速算法收敛的同时, 也使得算法

收稿日期: 2009-10-30；**修回日期:** 2010-01-04.

基金项目: 国家自然科学基金项目(60373000, 90820302, 60805027)；国家博士点基金项目(200805330005)；湖南省院士基金项目(02152).

作者简介: 刘星宝(1977–), 男, 博士生, 从事人工免疫系统、进化计算等研究；蔡自兴(1938–), 男, 教授, 博士生导师, 从事人工智能、智能控制等研究.

因多样性不足易于陷入局部最优解. 抗体的超变异算子是克隆选择算法的另一个研究热点. 该算子对克隆副本实施高频度变异, 从而搜索母体邻域、增加多样性信息. 变异概率和变异方式是超变异算子研究的两个方面, 基于亲和度的变异是目前广泛采用的方法, 如Cutello等人[9,14]提出的反比例超变异. 在反比例变异中, 抗体发生变异的概率和其亲和度值呈反比, 亲和度越差的抗体发生变异的概率越大. 高斯变异[15-17]、柯西变异[10]和线性变异[9]是常采用的变异方式, 以上几种变异方式使得信息从父代传递到子代即纵向传播, 而抗体-抗体间的横向信息交流很少.

为了解决以上问题, 本文提出了免疫进化算法, 其特点主要体现在:

1) 根据自然免疫系统随机性提出随机克隆扩张策略, 在该策略下, 抗体的克隆规模只与随机数、群体大小相关, 使得所有个体的邻域都能被充分搜索;

2) 改进超变异算子中的信息流动方式, 使新个体在继承母体信息的同时, 向其他个体学习, 加快算法的收敛速度;

3) 提出多受体随机编辑机制, 将多个受体信息泛化产生后代, 以丰富种群的多样性信息、提高算法的全局搜索能力. 此外, 本文提出一种新的算法性能评价准则, 可以比较不同算法在全局优化中的表现.

2 免疫进化算法

免疫进化算法的计算框架如下:

Step 1: 初始化其规模为 SN 的群体 $\boldsymbol{P}$, 评估个体的适应值;

Step 2: 执行 α- 随机克隆扩张以产生克隆体;

Step 3: 对克隆体执行rand/2超变异产生新的候选个体, 若新个体优于父辈则替换父辈, 成为新的进化个体, 从而完成对 $\boldsymbol{P}$ 的第1次更新;

Step 4: 对 $\boldsymbol{P}$ 实施多受体随机编辑操作生成群体 **EditP**;

Step 5: 从 $\boldsymbol{P}$ 与 **EditP** 中选择适应值最好的SN个体, 进入下一个进化过程;

Step 6: 若不满足停机条件, 则重复执行Step 2∼Step 5.

在该计算框架中, α- 随机克隆扩张、rand/2超变异和多受体随机编辑构成了算法的核心, 以下具体分析这3个算子.

2.1 α- 随机克隆算子

设抗体群为 $\boldsymbol{P}=\{x_1,x_2,\cdots,x_{\rm NP}\}, x_i=\{x_{i1}, x_{i2},\cdots,x_{in}\}$, $i=1,2,\cdots,$NP, 则 α- 随机克隆算子表示为对于任意的 $x_i\in\boldsymbol{P}$, 有

$$(x_{i1},x_{i2},\cdots,x_{in})\rightarrow\begin{bmatrix} x_{i1} & x_{i2} & \cdots & x_{in}\\ x_{i1} & x_{i2} & \cdots & x_{in}\\ \vdots & \vdots & \ddots & \vdots\\ x_{i1} & x_{i2} & \cdots & x_{in}\end{bmatrix}_{k_i n}. \quad (1)$$

式(1)中矩阵行数 k_i 由随机函数Rand和参数 α 决定, 群体的克隆规模为

$$\text{Clone\_Size}=\text{Int}(\text{NP}*\alpha*\text{Rand}(\text{NP},1)). \quad (2)$$

其中: Rand(NP, 1) 生成 $[0,1]$ 内随机分布的NP个随机数, Int(x) 为不大于 x 的整数; α 为克隆扩张比, 控制克隆群体的规模.

α- 随机克隆中群体的克隆规模由随机函数Rand决定, 其随机性特性使得所有个体均有机会生成克隆后代, 执行对其邻域的搜索.

进化群体大小NP, 克隆扩张比 α 和随机函数Rand共同决定了整个群体的克隆规模. 在本文的计算平台Matlab 2008 a中, Rand函数生成数值的均值为0.5, 在实验部分 $\text{NP}=15$, $\alpha=0.5$, 因此随机克隆扩张产生的全部克隆体的数目为

$$\|\text{Int}(\text{NP}*\alpha*\text{Rand}(\text{NP},1))\|=56. \quad (3)$$

2.2 rand/2 超变异算子

设 $x_i^j=(x_{i1}^j,x_{i2}^j,\cdots,x_{in}^j)$ 是 x_i 的第 j 个克隆体, x_{uh} 与 x_{vh} 分别是随机个体 x_u 和 x_v 的第 h 维标量, 则 x_i^j 的第 h 维rand/2超变异算子 ξ 定义为

$$\xi(u,v,h)=x_{ih}^j+\delta(x_{uh}-x_{vh}), \quad (4)$$

其中 δ 是 $[-1,1]$ 之间的随机数, 且在 x_i^j 的每一维变异时, 重新随机选择 u 和 v. 在免疫算法中, 由于克隆体和母体重合, 每个克隆体均要经历概率为1的超变异.

Rand/2超变异算子实现了不同个体间的相互学习, 其特点主要体现在: 1) rand/2超变异算子使用两个非己个体信息, 实现了不同个体间的横向学习, 增加了新个体的多样性信息; 2) rand/2超变异算子的作用对象是个体的每一维分量, 且不同分量使用的个体随机选择产生, 更好地体现了自然免疫系统的随机特性; 3) 在该变异算子中没有需要调整的参数, 具有较强的鲁棒性.

2.3 受体编辑

为了提高算法跳出局部极值点的能力, 本文提出一种新的随机编辑机制. 设 x_1,x_2,x_3 是参与编辑的受体, 编辑过程为

$$v=\alpha_1x_1+\alpha_2x_2+\alpha_3x_3. \quad (5)$$

其中: $\alpha_i(i=1,2,3)$ 是均匀分布、相互独立、取值在 $[0,1]$ 之间的随机数; $x_i(i=1,2,3)$ 是从群体中随机选择的个体.

多受体随机编辑算子的特点主要是: 1) 体现了克

隆选择原理的随机性; 2) 在整个算子中没有需要调整的参数, 提高了算法的鲁棒性; 3) 后代包含 3 个父体的信息, 提高了群体的多样性.

以取 3 个点 $x_1=(8,9), x_2=(9,6), x_3=(1,0)$ 为例, 其生成后代的可能分布如图 1 所示. "*"表示 3 个母体, "·"表示随机编辑算子生成的后代.

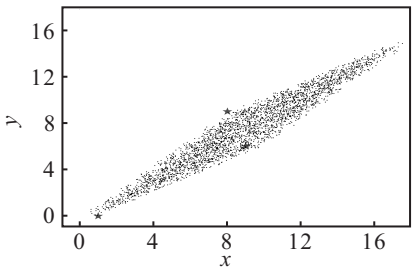

图 1　受体编辑算子生成的后代分布

3　算法性能评价准则

为了便于比较同一算法在不同参数下的性能, 比较不同算法之间的寻优能力, 提出一种新的算法性能评价准则, 定义为

$$\mathrm{PEv}=\varpi\sqrt{\frac{\sum_{i=1}^{k}(m_i-t_i)^2}{k}}+(1-\varpi)\sqrt{\frac{\sum_{i=1}^{k}(s_i)^2}{k}}. \quad (6)$$

其中: $\varpi\in[0,1]$ 表示该项的权重; m_i, s_i 和 t_i 分别表示第 i 个函数的均值、方差和理论最优值; k 表示测试函数的数目.

参数 ϖ 决定了均值和方差的权重分配: 当 ϖ 趋近于 0 时, 评估准则中含有均值的第 1 项对评估函数的影响减小, 相应的含有方差的第 2 项权重增加, 即算法方差在性能评价中所占权重增加; 反之, 当 ϖ 趋近于 1 时, 算法的均值所占权重增加, 因此 ϖ 调节算法稳定性与求解精度. 以下实验同等程度地重视均值和方差, 令 $\varpi=0.50$, 使得均值与方差有相同的权重. 在应用该准则评估算法性能时, 分为两个过程进行:

1) 当算法在多次独立优化第 i 个函数后, 若每次都能找到或接近理论全局最优解 t_i, 则最优结果的均值 m_i 与 t_i 的平方差 $(m_i-t_i)^2$ 趋近于 0, 同时均方差 s_i 趋近于 0;

2) 当算法在多次独立优化 k 个函数后, 若每次都能分别找到或接近理论全局最优解 $t_1,t_2,\cdots,t_k$, 即

$$\sum_{i=1}^{k}(m_i-t_i)^2\to 0\bigwedge\sum_{i=1}^{k}(s_i)^2\to 0, \quad (7)$$

则有

$$\varpi\sqrt{\frac{\sum_{i=1}^{k}(m_i-t_i)^2}{k}}+(1-\varpi)\sqrt{\frac{\sum_{i=1}^{k}(s_i)^2}{k}}\to 0. \quad (8)$$

在空间结构上 k 个测试函数的均值与方差分别构成了 k 维向量 $\boldsymbol{M}=(m_1,m_2,\cdots,m_k)$, $\boldsymbol{S}=(s_1,s_2,\cdots,s_k)$. 同样, k 个测试函数的理论最优值也构成最优值向量 $\boldsymbol{T}=(t_1,t_2,\cdots,t_k)$. 图 2 给出了均值向量 $\boldsymbol{M}$ 和方差向量 $\boldsymbol{S}$ 的模拟收敛路径.

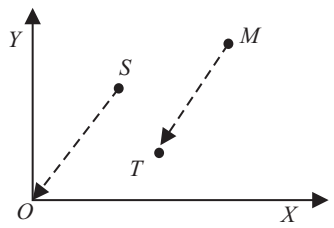

图 2　均值和向量的收敛路径

定理 1　应用理想算法 $\hat{A}$ 搜索 k 个函数的函数最小值, 在多次独立运行后有 $\mathrm{PEv}(\hat{A})=0$.

证明　假设存在理想算法 $\hat{A}$ 在每次独立运行时均能找到第 i 个测试函数的最优值, 即在单次运行过程中, 随着进化代数的增加, 实验最优值逼近函数的理论最优值. 当进化到足够的代数后, $\hat{A}$ 找到 t_i. 于是, 在对该测试函数独立运行若干次后得到 $m_i=t_i$ 和 $s_i=0$. 那么对所有测试函数独立运行若干次后, 有 k 维向量 $\boldsymbol{M}$ 和 $\boldsymbol{T}$ 重合, 即

$$(m_1,m_2,\cdots,m_k)=(t_1,t_2,\cdots,t_k). \quad (9)$$

因为每次独立运行都找到了问题的最优解, 有

$$(s_1,s_2,\cdots,s_k)=\bar{0}, \quad (10)$$

所以得到算法 $\hat{A}$ 的性能评价值为 0. □

可以看出, 在优化测试函数时, 若算法表现出稳定、求解精度高的良好性能, 则该算法的评价值趋近于 0, 即算法的性能和评价值成反比例关系. 得到如下推论.

推论 1　在应用 A 和 B 两种算法优化 k 个测试函数时, 若算法 A 的表现好于算法 B, 则 $\mathrm{PEv}A<\mathrm{PEv}B$.

4　实验与分析

为了验证算法性能, 本文采用 23 个测试函数[21] 作为测试样例. 实验采用 Intel Pentium 双核 (1.6 GHz, 1.6 GHz) 和 1 G 内存的硬件环境, 计算平台是 Matlab 2008 a.

4.1　克隆扩张比实验

为了选择适当的克隆扩张比, 实验测试了不同值对算法性能的影响. 程序独立运行 20 次, 每次的评价次数均为 10^5 次, 结果如表 1 所示. 表 1 表明:

1) 单峰高维测试函数 $F_1\sim F_7$: 在 α 较小时, 克隆操作产生较少的个体, 因此在一代进化过程中使用的评估次数减少, 致使进化代数增加, 从而提高算法

表 1 不同的扩张比对算法优化性能的影响, 设定群体规模 $SN = 15$

F	均 值(方 差)				
	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1.5$	$\alpha = 3.5$	$\alpha = 10$
F_1	0.00(0.00)	0.00(0.00)	2.38e-212(0.00)	2.12e-117(6.72e-117)	2.29e-55(1.97e-55)
F_2	0.00(0.00)	2.07e-228(0.00)	1.23e-107(3.82e-107)	6.16e-50(1.16e-59)	2.59e-29(2.61e-29)
F_3	0.00(0.00)	2.45e-272(0.00)	8.63e-132(1.85e-131)	2.57e-76(7.76e-76)	1.28e-36(3.96e-36)
F_4	0.00(0.00)	2.09e-203(0.00)	1.92-98(6.88e-99)	1.81e-56(9.46e-57)	1.33e-36(7.11e-27)
F_5	0.00(0.00)	0.00(0.00)	2.25e-214(0.00)	4.21e-120(9.25e-120)	4.87e-57(1.25e-56)
F_6	0.00(0.00)	0.00(0.00)	8.29e-5(4.51e-5)	0.00(0.00)	0.00(0.00)
F_7	1.88e-5(1.59e-5)	4.59e-5(3.52e-5)	4.59e-5(3.52e-5)	1.60e-4(9.32e-5)	1.62e-4(6.07e-5)
F_8	-10435.2(1275.6)	-12569.47(2.23e-12)	-12006.4(593.97)	-12133.6(428.05)	-11660.5(1508.4)
F_9	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
F_{10}	8.88e-16(0.00)	8.88e-16(0.00)	8.88e-16(0.00)	8.88e-16(0.00)	8.88e-16(0.00)
F_{11}	0.00 (0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
F_{12}	0.9463(0.2206)	2.40e-17(1.21e-16)	2.73e-22(7.50e-22)	1.25e-14(3.86e-15)	7.99e-8(6.31e-8)
F_{13}	2.9943(0.0022)	6.39e-9(4.52e-8)	1.79e-19(3.61e-19)	0.0024(0.0075)	0.0011(0.0035)
F_{14}	6.2796(4.6448)	0.998 (0.00)	0.998 (1.05e-16)	1.4927(1.25)	0.998(1.81e-16)
F_{15}	0.0288(0.0089)	0.0044(2.28e-6)	0.0044(1.63e-5)	0.0044(3.21e-6)	0.0044(8.49e-10)
F_{16}	-1.0253(0.0148)	-1.01316(2.24e-16)	-1.01316(7.40e-17)	-1.0316(7.40e-17)	-1.0316(0.00)
F_{17}	0.4127(0.0467)	0.3979(2.37e-8)	0.3979(0)	0.3979(0.00)	0.3979(2.24e-15)
F_{18}	3.0000(1.90e-6)	3.00(3.46e-15)	3.00(1.48e-16)	3.0000(2.09e-16)	3.0000(7.40-16)
F_{19}	-3.8567(0.0072)	-3.8628(3.14e-15)	-3.8628(9.36e-16)	-3.8628(9.36e-16)	-3.86(9.36e-16)
F_{20}	-2.9725(0.3696)	-3.3244(1.71e-3)	-3.3268(0.00)	-3.3022(0.0519)	-3.3268(0.00)
F_{21}	-10.1531(1.97e-5)	-10.1532(8.97e-15)	-10.1532(0.00)	-10.1532(0)	-10.1532(0.00)
F_{22}	-10.4028(4.57e-5)	-10.4029(8.97e-15)	-10.4029(1.87e-15)	-10.4029(1.87e-15)	-10.4029(1.87e-15)
F_{23}	-10.5363(2.52e-5)	-10.5365(8.87e-15)	-10.5365(6.70e-9)	-10.5365(1.64e-15)	-10.5365(1.45e-15)
PEv	**355.5085**	**0.0022**	**121.4845**	**90.0732**	**252.0314**

求解的精度. 另一方面, 当α变大时, 克隆扩张产生更多的个体, 增加了评价次数, 相应地减少了进化代数, 致使求解精度变差.

2) 复杂多峰函数 $F_8 \sim F_{13}$: 算法在不同的α值下仍然找到了 F_9 和 F_{11} 的最优值, 同时 F_{10} 的结果也很稳定. 对于 F_8, F_{12} 与 F_{13} 而言, 当$\alpha = 0.5$时实验结果最好, 且 F_8 找到了最优解.

3) 低维、含有少量局部极值点的测试函数 $F_{14} \sim F_{23}$: 实验结果分为两种情况, 在$\alpha = 0$时, 免疫进化算法仅找到5个函数的最优值且不稳定; 在α取其他值时, 优化精度与稳定性显著提高.

为了从整体上评价不同α值对算法优化性能的影响, 表1的最后一行给出了不同α下的算法性能评价值. 在$\alpha = 0.5$时性能评价值远远小于其他情况; 而在$\alpha = 0$时性能评价值最大. 表明免疫进化算法需要克隆扩张操作, 并结合超变异操作产生新的个体, 以完成邻域过程、增加进化群体的多样性信息. 另一方面, α取值不能太大, 否则产生过多的副本, 耗费宝贵的评价次数. 综合以上实验结果, 以下实验均采用 $\alpha = 0.5$.

4.2 与其他免疫算法的比较

将IEA (immune evolutionary algorithm) 与FCA (fast clonal algorithm)[10]和Opt-IMMALG(optimization immune algorithm)[22]进行对比, 测试3种优化算法的表现. 实验中每个测试函数独立运行20次, 评价次数限定为10^5. 为了公平比较, FCA和Opt-IMMALG的实验结果采用文献[10]给出的数据. 3种算法的实验结果如表2所示.

对于单峰函数 $F_1 \sim F_7$ 而言, 3种算法均取得了不错的优化效果. FCA找到4个函数的最优值, 且方差均为0.00; Opt-IMMALG找到了5个函数的最优值, 方差也为0.00; 同样免疫进化算法找到了3个函数的最优值, 方差同样都是0.00. 但是前两种算法对函数 F_5 的优化效果不理想, 探测到的实验最优结果均值(方差)分别为2.74 (2.49) 和16.29 (13.96), IEA的优化结果为0.00(0.00). 另外IEA对于函数 F_2, F_3 与 F_4 的优化精度均小于10^{-200}, 且方差均为0. 表明IEA对于单模函数同样具有很强的探测能力.

对于高维、多峰函数 $F_8 \sim F_{13}$ 而言, IEA和FCA均找到了 F_8, F_9 和 F_{11} 的最优解, 但前者对于 F_{10} 和 F_{12} 的优化精度比后者高, 对于 F_{13} 的优化效果没有后者好. Opt-IMMALG找到了函数 F_{10} 和 F_{11} 的最优解, 对于函数 F_{12} 与 F_{13} 的优化能力比IEA和FCA好, 但对于 F_9 的优化结果仅为0.6, 比其他两种算法差很多.

$F_{14} \sim F_{23}$ 是含有少量局部极值点的低维函数. 在对于这10个函数的优化实验中, FCA找到1个函数最优值, Opt-IMMALG找到3个函数最优值, 而IEA

表 2　IEA 与其他免疫算法的优化性能比较

F	均　值(方　差)			
	FCA	Opt-IMMALG	IEA	optima
F_1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
F_2	0.00(0.00)	0.00(0.00)	2.07e-228(0.00)	0.00(0.00)
F_3	4.53e-7(1.52e-7)	0.00(0.00)	2.45e-272(0.00)	0.00(0.00)
F_4	0.00(0.00)	0.00(0.00)	2.09e-203(0.00)	0.00(0.00)
F_5	2.74(2.49)	16.29(13.96)	0.00(0.00)	0.00(0.00)
F_6	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
F_7	5.95e-6(3.07e-6)	1.99e-5(2.35e-5)	4.59e-5(3.52e-5)	0.00(0.00)
F_8	-12 569.47(0.001 6)	-12 535.15(62.81)	-12 569.47(2.23e-12)	-12 569.5(0.00)
F_9	0.00(0.00)	0.60(4.18)	0.00(0.00)	0.00(0.00)
F_{10}	1.56e-7(3.12e-7)	0.00(0.00)	8.88e-16(0.00)	0.00(0.00)
F_{11}	0.00 (0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
F_{12}	7.94e-11(4.25e-12)	1.77e-21(8.77e-24)	2.40e-17(1.21e-16)	0.00(0.00)
F_{13}	2.65e-14(8.17e-14)	1.68e-21(8.77e-24)	6.39e-9(4.52e-8)	0.00(0.00)
F_{14}	1.04(3.65e-2)	0.998(1.1e-3)	0.998 (0.00)	0.998(0.00)
F_{15}	3.17e-4(8.23e-6)	3.2e-4(2.67e-5)	0.004 4(2.28e-6)	0.000 307 5(0.00)
F_{16}	-1.031 6(8.36e-6)	-1.013(2.21e-2)	-1.013(2.24e-16)	-1.031 63(0.00)
F_{17}	0.401(1.16e-3)	0.42(3.21e-2)	0.398(2.37e-8)	0.398(0.00)
F_{18}	3.012 9(2.15e-4)	5.84(3.74)	3.00(3.46e-15)	3.00(0.00)
F_{19}	-3.762 8(1.29e-5)	-3.72(7.84e-3)	-3.862 8(3.14e-15)	-3.86(0.00)
F_{20}	-3.311 9(6.5e-6)	-3.292(3.09e-2)	-3.324 4(1.7e-3)	-3.32(0.00)
F_{21}	-9.924 4(0.045 2)	-10.153 2(1.03e-7)	-10.153 2(8.97e-15)	-10.153 2(0.00)
F_{22}	-9.943 8(0.038 4)	-10.402 9(1.03e-5)	-10.402 9(8.97e-15)	-10.402 9(0.00)
F_{23}	-9.962 2(0.050 3)	-10.536 5(1.05e-3)	-10.536 5(8.87e-15)	-10.536 4(0.00)
PEv	**0.556 7**	**10.708 7**	**0.002 2**	**0.00**

则找到 7 个函数的最优值; 而在对其他函数的优化中, 3 种算法都接近了最优值, 但是 IEA 对于 F_{15} 的实验结果不如另外两种算法.

表 2 的最后一列给出了算法性能评价值, FCA, Opt-IMMALG和 IEA的评价值分别为0.556 7, 10.708 7 和 0.002 2. 可见 IEA 在整体上求解精度更高, 独立运行时性能更为稳定. 总体而言, 尽管在某些问题上 IEA 的优化精度不如 FCA 和 Opt-IMMALG 高, 但对于 23 个问题的优化中, IEA 一直表现平稳, 没有大的波动, 表明 IEA 在对全局高维函数进行优化的过程中表现得更为稳定.

5　结　论

本文分析了免疫算法在处理高维全局优化问题时存在多样性不足、个体间信息交流少等问题, 并提出了一种新的解决方案-免疫进化算法. 相对于传统免疫算法, 新算法的创新体现在以下几个方面: 1) 提出新的 α- 随机克隆扩张机制, 以丰富进化群体的多样性信息、加强算法对所有个体邻域的探测能力; 2) 在超变异算子中引入不同个体的信息, 以加强个体间的相互学习, 提高算法跳出局部极值点的能力, 加快算法的收敛速度; 3) 提供一种新型的个体间的信息交流方式, 即无参数的多父体随机编辑算子, 以加强算法的全局探测能力. 最后, 提出一种新的算法性能评价函数, 通过计算均值向量、方差向量与函数最优值向量间的距离, 评价算法在函数过程中的表现. 实验环节中将免疫进化算法与 FCA 和 Opt-IMMALG 进行了比较, 结果表明免疫进化算法在稳定性和求解质量上均表现出色.

下面的工作将进一步研究克隆和进化算法相结合的免疫进化机制, 在加强邻域搜索能力和全局搜索能力的同时, 兼顾搜索精度和广度, 保持两者的动态平衡, 以提高算法的稳定性和求解精度.

参考文献(References)

[1] 漆安慎, 杜蝉英. 免疫的非线性模型[M]. 上海: 上海科技教育出版社, 1999.
(Xi A S, Du C Y. The nonlinear model of immune[M]. Shanghai: Shanghai Science and Technology Education Press, 1999.)

[2] Hunt J E, Cooke D E. An adaptive, distributed learning system based on immune system[C]. IEEE Int Conf on System, Man and Cybernetics. Vancouver: IEEE press, 1995: 2494-2499.

[3] 王磊, 潘进, 焦李成. 免疫规划[J]. 计算机学报, 2000, 23(8): 806-812.
(Wang L, Pan J, Jiao L C. The immune programming[J]. Chinese J of Computers, 2000, 23(8): 806-812.)

[4] De Castro L N, Von Zuben F J. The clonal selection algorithm with engineering application[C]. Proc of the Genetic and Evolutionary Computation Conf on Workshop

on Artificial Immune System and Their Applications. Morgan: Kaufmann Publishers, 2000: 36-37.

[5] Cutello V, Nicosia G. The clonal selection principle for in silico and in vitro computing in recent developments in biologically inspired computing[M]. Hershey: Idea Group Publishing, 2004.

[6] Cutello V, Nicosia G. An immunological approach to combinatorial optimization Problems[C]. Proc of 8th Ibero-American Conf Artificial Intelligence. Seville, 2002: 361-370.

[7] Cutello V, Krasnogor N, Nicosia G, et al. Immune algorithm versus differential evolution: A comparative case study using high dimensional function optimization[C]. ICANNGA 2007. Berlin: Springer-Verlag, 2007: 93-101.

[8] De Castro L N, Von Zuben F J. Learning and optimization using the clonal selection principle[J]. IEEE Trans on Evolutionary Computation, 2002, 6(3): 239-251.

[9] Cutello V, Nicosia G. An immune algorithm for protein structure prediction on lattice models[J]. IEEE Trans on Evolutionary Computation, 2007, 11(1): 101-117.

[10] Nitesh K, Anoop P, Ravi S. et al. Fast clonal algorithm[J]. Engineering Applications of Artificial Intelligence, 2008, 21(1): 106-128.

[11] Ma W P, Jiao L C, Gong M G, et al. Immune clonal selection evolutionary strategy for constrained optimization[C]. Proc of the 9th Pacific Rim Int Conf on Artificial Intelligence. Berlin: Springer-Verlag, 2006: 661-670.

[12] Gong M G, Jiao L C, Du H F, et al. Multi-objective immune algorithm with nondominated neighbor-based selection[J]. Evolutionary Computation, 2008, 16(2): 225-255.

[13] Liu R C, Jiao L C. Immune clonal strategy based on the adaptive mean mutation[C]. LSMS 2007. Berlin: Springer-Verlag, 2007: 108-116.

[14] Cutello V, Nicosia G, Pavone M. Exploring the capability of immune algorithms: A characterization of hypermutation operators[C]. Proc of Int Conf on Artificial Immune Systems. Berlin: Springer, 2004: 263-276.

[15] 杜海峰, 公茂果, 刘若辰, 等. 自适应混沌克隆进化规划算法[J]. 中国科学E辑信息科学, 2005, 35(8): 817-829.
(Du H F, Gong M G, Liu R C, et al. Self-adaptive chaos clonal evolutionary programming[J]. Science in China Ser E(Information Sciences), 2005, 35(8): 817 829.)

[16] 公茂国, 焦李成, 杜海峰, 等. 用于约束优化的人工免疫响应进化策略[J]. 计算机学报, 2007, 30(1): 37-47.
(Gong M G, Jiao L C, Du H F, et al. A novel evolutionary strategy based on artificial immune response for constrained optimizations[J]. Chinese J of Computers, 2007, 30(1): 37-47.)

[17] 刘若辰, 贾建, 赵梦玲, 等. 一种免疫记忆动态克隆策略算法[J]. 控制理论与应用, 2007, 24(5): 777-784.
(Liu R C, Jia J, Zhao M L, et al. An immune memory dynamic clonal strategy algorithm[J]. Control Theory and Applications, 2007, 24(5): 777-784.)

[18] Kim J, Bentley P J. Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with a negative selection operator[C]. Proc of the 2001 Congress on Evolutionary Computation. Seoul: IEEE Computer Society, 2001: 1244-1252.

[19] De Castro L N, Von Zuben F J. Learning and optimization using the clonal selection principle[J]. IEEE Trans on Evolutionary Computation, 2002, 6(3): 239-251.

[20] 刘若辰, 杜海峰, 焦李成. 一种免疫单克隆策略算法[J]. 电子学报, 2004, 32(11): 1880 -1884.
(Liu R C, Du H F, Jiao L C. An immune monoclonal strategy algorithm[J]. Acta Electronica Sinica, 2004, 32(11): 1880-1884.)

[21] Yao X, Liu Y. Evolutionary Programming Made Faster[J]. IEEE Trans on Evolutionary Computation, 1999, 3(2): 82-102.

[22] Cutello V, Nicossia G, Pavone M. Real coded clonal selection algorithm for unconstrained global optimization using a hybrid inversely proportional hypermutation operator[C]. Proc of the 2006 ACM Symposium on Applied Computing. New York: ACM Press, 2006: 950-954.

(上接第58页)

[11] 张化祥, 陆晶. 基于Q学习的适应性进化规划算法[J]. 自动化学报, 2008, 31(7): 819-822.
(Zhang H X, Lu J. An adaptive evolutionary programming algorithm based on Q learning[J]. Acta Automatica Sinica, 2008, 31(7): 819-822.)

[12] 吕艳萍, 李绍滋, 陈水利, 等. 自适应扩散混合变异机制微粒群算法[J]. 软件学报, 2007, 18(11): 2740-2751.
(Lv Y P, Li S Z, Chen S L, et al. Particle swarm optimization based on adaptive diffusion and hybrid mutation[J]. J of Software, 2007, 18(11): 2740-2751.)