第 28 卷第 1 期
2011 年 1 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 28 No. 1
Jan. 2011

文章编号: 1000−8152(2011)01−0065−08

一种双态免疫微粒群算法

刘朝华[1,2], 张英杰[2], 章 兢[1], 吴建辉[2]

(1. 湖南大学 电气与信息工程学院, 湖南 长沙 410082; 2. 湖南大学 计算机与通信学院, 湖南 长沙 410082)

摘要: 针对基本微粒群算法的缺陷, 提出了一种双态免疫微粒群算法. 把微粒群分为捕食与探索两种状态, 处于捕食状态的精英粒子采用精英学习策略, 指导精英粒子逃离局部极值; 处于探索状态的微粒采用探索策略, 扩大解的搜索空间, 抑制早熟停滞现象. 同时引入免疫系统的克隆选择和受体编辑机制, 增强群体逃离局部极值及多模优化问题全局寻优能力. 实验表明该算法收敛速度快, 求解精度高, 尤其适合高维及多模态优化问题的求解.

关键词: 微粒群; 双态; 精英学习; 人工免疫系统; 多模态函数

中图分类号: TP18　　**文献标识码**: A

A novel binary-state immune particle swarm optimization algorithm

LIU Zhao-hua[1,2], ZHANG Ying-jie[2], ZHANG-Jing[1], WU Jian-hui[2]

(1. College of Electrical and Information engineering, Hunan University, Changsha Hunan 410082, China;
2. School of Computer and Communication, Hunan University, Changsha Hunan 410082, China)

Abstract: Conventional algorithms of particle swarm optimization(PSO) are often trapped in local optima in global optimization. A novel binary-state immune particle swarm optimization algorithm(BIPSO) is proposed. In order to enhance the explorative capacity of the algorithm while avoiding the premature stagnation behavior, the meta-heuristics allow for two behavior states of the particles including Gather State and Explore State during the search. The population is divided into two parts in iterations. Elitist learning strategy is applied to the elitist particle to help the jump out of local optimal regions when the search is identified to be in a gather state. This paper propose a concept of explore strategy to encourage particle in a explore state to escape from the local territory. They exhibit a wide range exploration. Moreover, in order to increase the diversity of the population and improve the search capabilities of PSO algorithm, the mechanism of clonal selection and the mechanism of receptor edition are introduced into this algorithm. Experiments on several benchmarks show that the proposed method is capable of improving the search performance. It is efficient in tackling the high dimensional multimodal optimization problems.

Key words: particle swarm optimization(PSO); binary-state; elitist learning; artificial immune system(AIS); multimodal function optimization

1 引言(Introduction)

粒子群优化算法(particle swarm optimization, PSO)是由Eberhart 博士和Kennedy博士[1]提出的一种新的全局优化进化算法, 它的实现简单且优化性能较好, 广泛的应用于科学与工程领域[2]. 然而, 微粒群算法与其他进化算法类似. 随着问题规模的增大, 在进化后期容易陷入早熟收敛, 限制了微粒群的更广泛应用. 因此, 产生了许多变种的算法. 主要有以下几类改进: 第1类通过改进速度与位置的公式及调整权重参数[3]; 第2类设计出不同类型的拓扑结构与邻域结构提高优化性能[4,5]; 第3类将其他智能搜索算子引入微粒群, 如引入遗传交叉算子[6]、变异算子[7]、局部搜索算子[8], 形成混合优化微粒群算法. 最近学者提出了综合学习微粒群算法(CLPSO)[9], 自适应微粒群算法等[10]. 这些改进在一定程度上改善了算法性能, 然而, 早熟收敛依然是PSO算法的一大难题, 尤其是在复杂高维及多模态优化问题中.

人工免疫系统(artificial immune system, AIS)[11]是模拟生物免疫系统机理的人工智能系统. 其克隆选择机制中存在着克隆、超变异、抗体与抗原特性结合, 免疫记忆产生等过程, 收敛速度快, 多样性好. 其受体编辑机制中允许细胞受体特定的条件下可以发生结构改变, 致使抗原受体不易陷入局部极值.

分析了基本微粒群算法易陷入局部极值的原因. 结合人工免疫系统及鸟类, 鱼类捕食行为特点, 提出了一种双态免疫微粒群算法, 在搜索过程中依据群体进化因子协调群体勘探与开采, 把微粒群分为“捕食”与“探索”两种状态. 引入免疫系统的克隆选择

收稿日期: 2009−10−09; 收修改稿日期: 2009−12−22.
基金项目: 国家自然科学基金重点资助项目(60634020); 湖南省科技计划重点资助项目(2010GK2022).

和受体编辑机制, 提高群体的多样性分布, 增强群体逃离局部极值能力及多峰极值全局寻优能力.

2 双态免疫微粒群算法(Binary-state immune particle swarm)

2.1 双态微粒群机理(Binary-state immune particle swarm mechanism)

基本粒子群在搜索中, 如果某粒子发现一个当前最优位置, 邻域内其他粒子将迅速向其靠拢. 因此, 容易陷入局部极值, 又没有提供逃离局部解的策略, 解决PSO算法中的停滞现象和较强寻优能力的方法就是在"探索"和"利用"之间寻求一个平衡点. 既要使算法的搜索空间尽可能的大, 同时充分利用有效的历史知识, 从而以更大的概率收敛到全局最优解. 对鱼类、鸟群体捕食行为研究表明, 真实群体智能社会中的群居智能体各司其职是一个有机整体, 从中得到启迪, 在群体进化的过程中, 根据群体进化因子, 可以将具有N只微粒群分成两种不同行为特征子群, 探索群体与捕食群体. 其群体行为可以用简单的自动机模型表示见图1. 其中: S1: 捕食状态(gather state), S2: 探索状态(explore state).

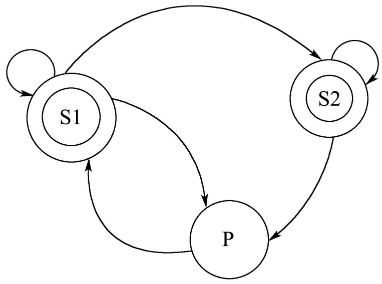


图 1 双态微粒群自动机模型

Fig. 1 A finite state machine of BPSO

由图1可知双态行为表明群体在捕食状态下搜索速度减慢或停止, 群体中一部分微粒将从捕食状态转为探索状态以扩大解空间. 同时群体也可以从探索状态转为捕食状态进行集中优良区域搜索加快搜索速度, 每一个状态都可能收敛到全局最优值. 对于整个微粒群来说, 不会因为遇到局部极值点而停止搜索.

定义 1 捕食状态: 在捕食状态下微粒群行为与传统的微粒群算法一致. 设粒子群体规模为N, 每个粒子在D维搜索空间中, 记粒子$i(i=1,2,\cdots,N)$的当前位置是$x_i=\{x_{i1},x_{i2},\cdots,x_{iD}\}^{\mathrm{T}}$, 飞行的速度为$v_i=\{v_{i1},v_{i2},\cdots,v_{iD}\}^{\mathrm{T}}$每个粒子当前找到的最优解为$P_{\mathrm{id}}$, 种群当前找到的全局最优点为$P_{\mathrm{gd}}$. 为了减少进化迭代过程中粒子离开搜索空间的可能性, 通常限定V_{id}: $V_{\min}\leqslant V_{\mathrm{id}}\leqslant V_{\max}$.

$$V'_{\mathrm{id}}=\omega V_{\mathrm{id}}+c_1r_1(P_{\mathrm{id}}-X_{\mathrm{id}})+c_1r_2(P_{\mathrm{gd}}-X_{\mathrm{id}}), \tag{1}$$

ω为惯性权重. c_1,c_2为调节P_{id}和P_{gd}相对重要性的参数, r_1,r_2为生成介于0和1之间的随机数. 这样, 可以得到粒子的下一位置:

$$X'_{\mathrm{id}}=X_{\mathrm{id}}+V'_{\mathrm{id}}. \tag{2}$$

定义 2 探索状态: 在优化过程中, 微粒群根据群体当前所在的状态, 经过有限次的探寻, 如果没有找到更优的解, 则整个群体已陷入局部极值点. 依据进化因子分散就要强迫一部分微粒群转为探索状态. 群体进化因子定义如下:

$$f=\frac{g\mathrm{best}_i-g\mathrm{best}_{i-1}}{g\mathrm{best}_{i-1}-g\mathrm{best}_{i-2}+\alpha}, \tag{3}$$

$g\mathrm{best}_i$为第i代种群最优适应度值, $\alpha\in(0,1)$为平滑系数, 防止分母为0. 式(3)表明, 当$f\equiv 0$时进化停止; 当$0<f<1$时, 表示种群进化速度减慢; 当$f>1$时进化加速. 当$f<1$时以$(1-f)$的概率进行探索. 当进行探索时, 保留性能最好的25%左右的微粒在捕食状态, 其余的微粒转为探索状态处于探索状态时的微粒将重新随机初始化分布在搜索空间.

$$X'_{\mathrm{kd}}=X_{\mathrm{kd}}^{\min}+\mathrm{rand}()*(X_{\mathrm{kd}}^{\max}-X_{\mathrm{kd}}^{\min}), \tag{4}$$

其速度位置更新公式如下:

$$V'_{\mathrm{kd}}=\omega\,\mathrm{sgn}\,r V_{\mathrm{id}}+c_1r_1(P_{\mathrm{kd}}-X_{\mathrm{kd}})+c_1r_2(P_{\mathrm{gd}}-X_{\mathrm{kd}})+c_3u(0,1)(X_{\mathrm{kd}}-h_{\mathrm{pgd}}), \tag{5}$$

$$X'_{\mathrm{kd}}=X_{\mathrm{kd}}+V'_{\mathrm{kd}}, \tag{6}$$

$$\mathrm{sgn}=\begin{cases}1, & r\geqslant 0.01,\\ -1, & r<0.01.\end{cases} \tag{7}$$

c_3为(0, 1)之间的随机数, r为随机数, $\mathrm{sgn}\,r$为符号函数, 为了寻求更优目标, 微粒可能改变飞行方向[12], $u(0,1)$为高斯分布函数. h_{pgd}为当前探索种群中最优粒子在第d维的位置.

2.2 精英学习机制(Elitist learning)

在微粒群算法中, 精英粒子就是群体领导者. 一旦精英微粒陷入局部点, 整个群体很容易陷入局部点, 精英学习机制[10]可以协助精英微粒跳出局部极值点. 对捕食状态中的部分精英微粒进行精英学习, 该机制能正确导向其他微粒的飞行, 加速收敛. 利用高斯分布两翼概率特性, 易产生一个远离原点的随机数, 有可能更快地跳出局部极小的区域, 见下式:

$$p'_{\mathrm{kd}}=p_{\mathrm{kd}}+\eta*(X_{\mathrm{kd}}^{\max}-X_{\mathrm{kd}}^{\min})\mathrm{Gauss}(\mu,\sigma^2), \tag{8}$$

$$\sigma=\sigma_{\max}-(\sigma_{\max}-\sigma_{\min})\frac{t}{T}. \tag{9}$$

η为一个限幅常数, t是群体当前进化代数, T为总的进化代数, σ为学习率线性递减.

2.3 人工免疫算法(Artificial immune)

在多模态(多峰函数)问题优化中, 微粒群易表现出一个缺陷, 随着迭代的进行, 群体易集中收敛于一点, 失去多样性, 很难同时找到全局的多个最优极值

点. 为提高多模态(多峰)问题全局优化能力, 引入免疫系统中的克隆选择机制和受体编辑机制.

2.3.1 克隆选择(Clonal selection)

克隆操作步骤如下:

Step 1 各个粒子的个体极值$\{P_{1\mathrm{d}}, P_{2\mathrm{d}}, \cdots, P_{N\mathrm{d}}\}$生成一个临时克隆种群. 将临时克隆种群每一粒子视为抗体, 克隆规模与亲和度成正比, 克隆倍数N_c见式(10):

$$N_\mathrm{c}=\sum_{i=1}^{N}\mathrm{round}(\frac{\beta N}{i}+b), \tag{10}$$

其中: N为种群规模, $\beta\in(0,1)$. 为了保证每个抗体都有一定克隆数量, 故加上了常量$b\geqslant 1$的整数. 经过克隆扩增生成新群体Sub.

Step 2 对群体Sub中的每个个体实施高频变异, 其方法为基于变尺度的邻域内变异. 受自然生物进化思想启发[13], 算法中在进化初期以一定的变异概率将采用较大的变异尺度以保持种群的多样性, 而在进化后期变异尺度逐渐缩小以提高局部微调能力. 其变异算子如下:

$$\begin{aligned}P_{\mathrm{id}}^{\mathrm{new}}=&p_{\mathrm{id}}+[rd>p_\mathrm{m}]*\eta*P_{\mathrm{id}}*U(0,1)-\\&[rd\leqslant p_\mathrm{m}]*\eta*P_{\mathrm{id}}*U(0,1),\end{aligned} \tag{11}$$

$$[rd\leqslant p_\mathrm{m}]=\begin{cases}1, rd\leqslant p_\mathrm{m},\\0, \text{其他}.\end{cases} \tag{12}$$

rd为随机数, $P_\mathrm{m}=0.5$.

$$\eta(t)=1-r^{[1-\frac{t}{T}]^b}, \tag{13}$$

t是群体当前进化代数, T为总进化代数, b为正常数一般取值为2, $r\in(0,1)$, $U(0,1)$为(0,1)之间的均匀随机数. 从式(14)可看出, 在进化初期, 对于较小的r值时, $\eta(t)\approx 1$, 此时的变异空间大; 而在进化后期, t接近T时$\eta(t)\approx 0$, 在小范围空间内进行局部搜索.

Step 3 免疫选择操作, 从克隆变异后的个体中选择亲和度最优的个体进入下一代通过局部择优实现了种群的压缩, 同时保证了抗体群中的最优解不会变差.

2.3.2 受体编辑(Receptor editing)

受体编辑机制[14]是指T细胞和B细胞受体在特定的条件下结构发生改变, 从而发生亲和力的变化, 受体编辑进一步丰富了抗原受体的多样性, 在免疫双态微粒群算法系统中, 每隔一定的代数就要鉴别出不活跃的、细胞受体, 对它们中的25%～30%进行受体编辑.

$$aff_{\mathrm{avg}}=\frac{1}{n}\sum_{i=1}^{n}aff_i, \tag{14}$$

$$\sigma^2=\frac{1}{n}\sum_{i=1}^{n}(aff_i-aff_{\mathrm{avg}})^2. \tag{15}$$

式(15)中aff_{avg}为群体的平均亲和度, σ^2为群体是适应度的方差, 反映了群体的收敛程度, 当适应度方差为趋于零的一个很小的阈值时, 此时算法得到的最优解还未达到全局最优解, 则群体已陷入局部最优解. 对于满足不等式(16)的抗原受体i进行受体编辑:

$$\frac{aff_i-aff_{\min}}{aff_{\max}-aff_{\min}+\varepsilon}<\theta. \tag{16}$$

式(16)中为$\varepsilon\in(0,1)$的之间的平滑系数, 以保证分母不为0, θ为阈值. 对于不活跃的、细胞受体, 利用非线性Logistic序列进行随机又有规律的位置大漂移. 具体如下式:

$$X'_{\mathrm{id}}=X_{\mathrm{id}}+\frac{X_{\mathrm{id}}^{\max}-X_{\mathrm{id}}^{\min}}{m}U_{r+1}, \tag{17}$$

$X_{\mathrm{id}}^{\max}$, $X_{\mathrm{id}}^{\min}$为位置的上界、下界, m为正常数, 视具体问题而定. U_{r+1}为混沌序Logistic, 其中U_{r+1}为混沌序列Logistic[13]映射如下:

$$\begin{aligned}&U_{r+1}=\mu U_r(1-U_r), r=0,1,2,\cdots,\\&0<U_0<1,\end{aligned} \tag{18}$$

μ为系统的状态控制参量, 已证明[13], 当$\mu=4$, 初始值$U_0\notin$(0.25, 0.5, 0.75,1)时, 上式表示的系统完全处于混沌状态, U_{r+1}在(0,1)范围内遍历.

3 算法BIPSO流程及性能分析(Flow of algorithm and analysis)

3.1 算法BIPSO流程(Algorithm flow)

Step 1 随机初始化m个微粒的位置与速度, 及相关参数初始化.

Step 2 评价各微粒的初始适应值, 保存初始最好位置, 及初始最优适应值.

While(不满足退出条件)do //退出条件为找到相应的全局最优值或达到设定的截止代数.

Step 3 for 对微粒群中每只粒子do.

根据式(1)(2)更新粒子的位置与速度, 计算各粒子适应度值.

计算群体进化因子f.

IF (($f<1$) & 适应值没达到全局最优)

{ if($u(0,1)<1-f$) then

将粒子群的适应值排序, 性能较好的$m*r_1$按捕食状态行动, 对捕食状态中的部分优秀粒子进行精英学习;

其余$m*r_2$只粒子按探索状态行动, 按照公式(5),(6)更新粒子的位置与速度.

}

End IF

如果各粒子适应度优于相应粒子历史最优适应度, 相应粒子p_{id}更新.

如果最优粒子适应度优于历史最优粒子适应度, $p_{\rm gd}$更新.

End for

Step 4 各个粒子的个体极值$P_{\rm id}$生成一个临时的克隆种群, 进行克隆选择操作.

Step 5 对趋于凋亡微粒进行受体编辑.

End while

Step 6 输出结果, 算法运行结束.

3.2 算法性能及分析(Performance analysis)

BIPSO算法在优化初期, 群体搜索能力较强, 都聚集在捕食状态下集中寻优. 随着迭代的进行, 寻优速度减慢, 群体分为两种状态, 让一部分微粒转为探索状态, 解空间进一步扩大有利于寻找优良解区域. 对处于捕食状态的精英粒子进行有指导性的精英学习, 优秀粒子能正确导向其他微粒的飞行, 信息通过邻域传播, 进而加速群体收敛速度. 基于个体极值的克隆选择机制, 扩大抗体的分布性, 进一步提高全局搜索能力, 有利于多模态(多峰)优化问题求解. 随着进化的进行, 群体微粒之间的距离逐渐聚集, 种群失去多样性. 这时受体编辑机制非线性Logistic序列进行随机又有规律的位置大漂移, 在进化整个过程中保持了种群微粒多样性, 微粒有充足的动量在解空间中搜索, 提高了微粒的全局搜索能力.

综上所述, BIPSO算法有着广阔的解空间, 高效的精细搜索策略, 超强的逃离局部极值机制及多样性保持机制, 因此算法具有收敛速度快和求解精度高的特点.

3.3 收敛性分析(Convergent analysis)

假设 1 1) 问题P的定义域Ω为$\mathbb{R}^n$的有界闭区域; 2) 目标函数$f(x)$是区域上的连续函数.

定理 1 设$x(k)$是由BIPSO算法产生的种群序列, 其中$x^*(k)\in X(k)$为第k代种群中的最优个体, 即$x^*(k)=\operatorname{argmin} f(x_i(k))$, 如果问题P中的目标函数和可行域满足假设1, 则有$\mathrm{P}\{\lim\limits_{k\to\infty} f(x^*(k))=f^*\}=1$, 即种群序列以概率为1收敛与全局最优解.

证 1) 处于捕食状态的微粒前后两代的位移变化量变为:

$$\Delta x_i(t)=\omega V_{\rm id}(t)+c_1\cdot{\rm rand}\cdot(P_{\rm id}(t)-X_{\rm id}(t))+c_2\cdot{\rm rand}\cdot(P_{\rm gd}(t)-X_{\rm id}(t)),$$

$$\phi_0=\omega V_{\rm id}(t),\phi_1=c_1\cdot{\rm rand}\cdot(P_{\rm id}(t)-X_{\rm id}(t)),$$

$$\phi_2=c_2\cdot{\rm rand}\cdot(P_{\rm gd}(t)-X_{\rm id}(t)),$$

$$\Delta x_i(t)=\phi_0+\phi_1+\phi_2,r_1,r_2\sim(0,\sigma_t).$$

2) 处于探索状态的微粒种群前后两代的位移变化量变为:

$$\Delta x_k(t)=\omega\cdot{\rm sgn}\,r\cdot V_{\rm id}(t)+c_1\cdot{\rm rand}\cdot(P_{\rm kd}(t)-X_{\rm kd}(t))+c_2\cdot{\rm rand}\cdot(P_{\rm gd}(t)-X_{\rm kd}(t))+c_3\cdot u(0,1)\cdot(X_{\rm kd}(t)-h_{\rm pgd}),$$

设:

$$\phi_0=\omega\cdot{\rm sgn}\,r\cdot V_{\rm id}(t),$$
$$\phi_1=c_1\cdot{\rm rand}\cdot(P_{\rm kd}(t)-X_{\rm kd}(t)),$$
$$\phi_2=c_2\cdot{\rm rand}\cdot(P_{\rm gd}(t)-X_{\rm kd}(t)),$$
$$\phi_3=c_3\cdot u(0,1)\cdot(X_{\rm kd}(t)-h_{\rm pgd}),$$
$$\Delta x_k(t)=\phi_0+\phi_1\cdot r_1+\phi_2\cdot r_2+\psi\cdot\phi_3\cdot r_3,$$
$$r_1,r_2,r_3\sim(0,\sigma_t).$$

其中if (explore) then $\psi=1$ else $\psi=0$. 显然有: $\Delta x_i(t)\sim N(0,\sigma_t)$.

由于克隆变异选择操作之后获得的最优解至少比操作之前的要好及精英学习采用最优选择策略. 因此, 最优种群仍然不会退化. 由文献[15,16]可知, 定理1成立(详细证明参见文献[15,16]).

4 仿真实验及分析(Experiments)

实验仿真环境为Windows XP系统, AMD处理器, 1.90 GHz 448 MB内存, 仿真软件MATLAB 7.0. 为了验证本文提出的双态免疫微粒群算法(BIPSO)对函数优化时的收敛速度, 全局优化能力, 多峰寻优能力及超高维等求解性能, 引入个基准优化问题, 进行分析, 将该算法与其他算法变种进行了对比这个基准函数具有不同的特点, 可以充分考察新型算法对不同类型问题的优化性能. (MATLAB精度值1e−308 (MATLAB当小于这个值时显示0))文中测试函数来自文献[10].

测试函数:

$$f_1(x)=\sum_{i=1}^{D}x_i^2,-100\leqslant x_i\leqslant 100,$$

$$f_2(x)=\sum_{i=1}^{D}|x_i|+\prod_{i=1}^{D}|x_i|,-10\leqslant x_i\leqslant 10,$$

$$f_3(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2,-100\leqslant x_j\leqslant 100,$$

$$f_4(x)=\sum_{i=1}^{D}(100(x_(i+1)-x_i^2)^2+(x_i-1)^2),$$

$$f_5(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2,-100\leqslant x_i\leqslant 100,$$

$$f_6(x)=\sum_{i=1}^{D}ix_i^4+{\rm random}(0,1),-1.28\leqslant x_i\leqslant 1.28,$$

$$f_7(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{x_i}),-500\leqslant x_i\leqslant 500,$$

$$f_8(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10],-500\leqslant x_i\leqslant 500,$$

$$f_9(x) = \sum_{i=1}^{D}[y_i^2 - 10\cos(2\pi y_i) + 10].$$

这里:

$$y_i = \begin{cases} x_i, & |x_i| < 0.5, \\ \mathrm{round}(\frac{2x_i}{2}), & |x_i| \geqslant 0.5, \end{cases} -5.12 \leqslant x_i \leqslant 5.12,$$

$$f_{10}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}) - \exp(-0.2\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e, -32 \leqslant x_i \leqslant 32,$$

$$f_{11}(x) = \frac{1}{400}\sum_{i=1}^{D}x_i^2 - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1, \quad -600 \leqslant x_i \leqslant 600,$$

$$f_{12}(x) = \frac{\pi}{D}\{10\sin(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_i + 1)] + (y_D - 1)^2\} + \sum_{i=1}^{D}u(x_i, 10, 100, 4).$$

其中:

$$y_i = 1 + \frac{1}{4}(x_i + 1),$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & |x_i| \leqslant a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases} \quad -50 \leqslant x_i \leqslant 50.$$

实验参数设置: 对于单模态函数($f_1 \sim f_6$)微粒个数为60, 多模态函数($f_7 \sim f_{14}$)微粒个数为100, 加速因子$c_1 = c_2 = 1.49445$[9], 惯性权重[5], $w \in (0.9, 0.4)$克隆变异初始概率为0.7, 低维的函数基本迭代上限为600代, 高维函数基本迭代上限为1000代.

4.1　BIPSO参数研究(Parameters research)

为了分析探索微粒与捕食微粒的比例对算法性能的影响($r_2 : r_1$), 选取了实例f_2(Schwefel's)函数(30维, 理论最优值$f_{\min} = 0$)和f_{12}(generalized penalized)函数(30维, 理论最优值$f_{\min} = 0$)进行测试. 表2、表3中best, worst, mean, stdDev, T表示测试获得的最优值、最差值、平均值、标准方差、求解时间. 从表1、表2可以得出当$r_2 : r_1 = 0.75 : 0.25$时, 即探索群体与捕食群体比例为0.75 : 0.25时算法的性能较好, 最优值, 最差值均值, 以及标准方差均优于其他比例. 之所以探索群体的数量为捕食群体数量的3倍左右, 是因为需探索的解空间比局部精细搜索解的空间大得多. 当搜索空间变大时, $r_2 : r_1$的比例有所调整. $r_2 : r_1$比例起到了协调算法探索与利用的作用. 当$r_2 : r_1$过大时(0.8 : 0.2), 算法的利用能力不足而探索能力过剩; 当$r_2 : r_1$过小时, 算法的利用能力过剩而探索能力不足.

表 1　参数$r_2 : r_1$对BIPSO算法的影响(f_2(30次测试))

Table 1　Performance affection of BIPSO with proportional $r_2 : r_1$ on f_2(30 tests)

$r_2 : r_1$	best	worst	mean	stdDev	T/s
0.2:0.8	0	9.04e−41	1.11e−41	2.71e−41	2.3
0.3:0.7	0	1.87e−41	6.24e−43	3.36e−42	0.9
0.5:0.5	0	7.39e−39	2.46e−40	1.32e−39	1.3
0.6:0.4	0	4.68e−58	1.56e−59	8.40e−59	1.0
0.7:0.3	0	0	0	0	0.2
0.75:0.25	**0**	**0**	**0**	**0**	**0.1**
0.8:0.2	0	0	0	0	0.2
0.85:0.15	0	2.21e−30	2.21e−31	6.6e−31	1.3

表 2　参数$r_2 : r_1$对BIPSO算法的影响(f_{12}(30次测试))

Table 2　Performance affection of BIPSO with proportional $r_2 : r_1$ on f_{12}(30 tests)

$r_2 : r_1$	best	worst	mean	stdDev	T/s
0.2:0.8	1.57e−32	8.03e−10	8.73e−11	2.3e−10	33.4
0.3:0.7	1.57e−32	1.89e−10	1.89e−11	5.68e−11	32.7
0.5:0.5	1.57e−32	2.06e−9	2.06e−10	6.1e−10	33.2
0.6:0.4	1.57e−32	1.24e−7	1.33e−8	3.72e−8	33.6
0.7:0.3	1.57e−32	6.49e−9	1.95e−9	2.2e−9	34.4
0.75:0.25	**1.57e−32**	**1.57e−32**	**1.57e−32**	**2.74e−48**	**34.5**
0.8:0.2	5.69e−20	1.28e−12	1.98e−13	2.90e−13	34.0
0.85:0.15	1.57e−32	5.6e−27	5.69e−28	1.70e−27	34.3

4.2　对比实验研究(Comparisons)

为了与表3相关变种PSO算法进行比较, BIPSO测试函数维数同样设为30维, 算法LPSO[3], VPSO[3], (冯·若依曼结构PSO), FIPS(fully informed PSO), HPSO_TVAC[3], DMS-PSO[8](动态多微粒群算法)、CLPSO[9](综合学习微粒群算法)、APSO[10](自适应微粒群算法)该算法根据进化因子通过模糊聚类自适应参数调整微粒群算法并对精英粒子进行学习, 其他7种变种微粒群算法实验数据均来自文献[10]. 为了测试BIPSO算法收敛速度, 限于篇幅文中选取典型函数f_4, f_8, f_{10}, f_{12}测试. 图2中PSO-w为惯性权重线性递减[4], 取迭代次数为600代3种算法收敛速度比较. APSO算法源程序(文献[10]中作者提供), 从图2可以看出BIPSO算法的收敛速度明显优于PSO-w及算法APSO. 从表3中数据可以看出BIPSO性能明显优于其他7种变种PSO, 其原因是双态免疫微粒群算法(BIPSO)精英粒子学习机制和克隆选择机制提供了较好逃离局部极值策略, 探索群体及受体编辑的运用最大程度上保证了群体的多样性.

表 3 BIPSO与其他7种改进PSO方法比较(30次测试)

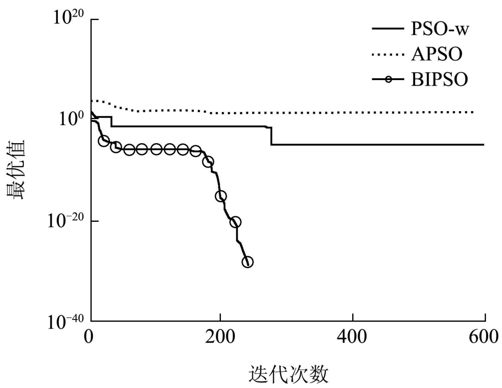
Table 3 Search result comparison among eight PSOs on test functions(30 tests)

函数		LPSO	VPSO	FIPS	HPSO-TVAC	DMS-PSO	CLPSO	APSO	BIPSO
$f_1(0)$	Mean	4.77e−29	5.11e−38	3.21e−30	3.38e−41	3.85e−54	1.89e−19	1.45e−150	**0**
	StdDev	1.13e−28	1.91e−37	3.60e−30	8.50e−41	1.75e−53	1.49e−19	5.73e−150	**0**
$f_2(0)$	Mean	2.03e−20	6.29e−27	1.32e−17	6.9e−23	2.61e−29	1.01e−13	5.15e−84	**0**
	StdDev	2.89e−20	8.68e−27	7.86e−18	6.89e−23	6.6e−29	6.51e−14	1.44e−83	**0**
$f_3(0)$	Mean	18.60	1.44	0.77	2.89e−7	47.5	395	1.0e−10	**0**
	StdDev	30.71	1.55	0.86	2.97e−7	56.4	142	2.13e−10	**0**
$f_4(0)$	Mean	21.8627	37.6469	22.5387	13	32.3	11	2.84	**0**
	StdDev	11.1593	24.9378	0.310182	16.5	24.1	14.5	3.27	**0**
$f_5(0)$	Mean	0	0	0	0	0	0	0	**0**
	StdDev	0	0	0	0	0	0	0	**0**
$f_6(0)$	Mean	1.49e−2	1.08e−2	2.55e−3	5.54e−2	1.1e−2	3.92e−3	4.66e−3	**0**
	StdDev	5.66e−3	3.24e−3	6.25e−4	2.08e−2	3.94e−3	1.14e−3	1.7e−3	**0**
$f_7(-12569.5)$	Mean	−9628.35	−9845.27	−10113.8	−10868.57	−9593.33	−12557.65	**−12569.5**	−12569.49
	StdDev	456.54	588.87	889.58	289	441	36.2	**5.22e−11**	4.67e−6
$f_8(0)$	Mean	34.90	34.09	29.98	2.39	28.1	2.57e−11	5.8e−15	**0**
	StdDev	7.25	8.07	10.92	3.71	6.42	6.64e−11	1.01e−14	**0**
$f_9(0)$	Mean	30.40	21.33	35.91	1.83	32.8	0.167	4.14e−16	**0**
	StdDev	9.23	9.46	9.49	2.65	6.49	0.379	1.45e−15	**0**
$f_{10}(0)$	Mean	1.85e−14	1.4e−14	7.69e−15	2.06e−10	8.52e−15	2.01e−12	1.11e−14	**8.88e−16**
	StdDev	4.80e−15	3.48e−15	9.33e−16	9.45e−10	1.79e−15	9.22e−13	3.55e−15	**0**
$f_{11}(0)$	Mean	1.10e−2	1.31e−2	9.04e−4	1.07e−2	1.31e−2	6.45e−13	1.67e−2	**0**
	StdDev	1.60e−2	1.35e−2	2.78e−3	1.14e−2	1.73e−2	2.07e−12	2.41e−2	**0**
$f_{12}(0)$	Mean	2.18e−30	3.46e−3	1.22e−31	7.07e−30	2.05e−32	1.59e−21	3.76e−31	**1.57e−32**
	StdDev	5.14e−30	1.89e−2	4.85e−32	4.05e−30	8.12e−33	1.93e−21	1.2e−30	**2.74e−48**

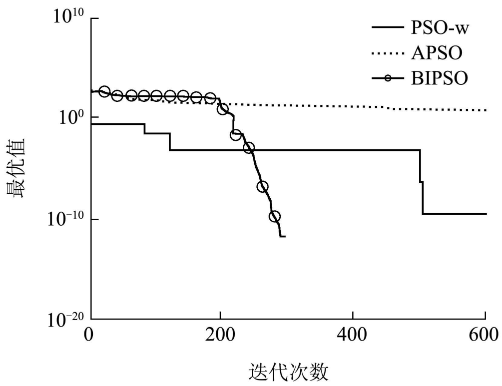
表 4 改进免疫算法对双态微粒群的影响(30次测试)

Table 4 Performance improvement of BPSO with AIS(30 tests)

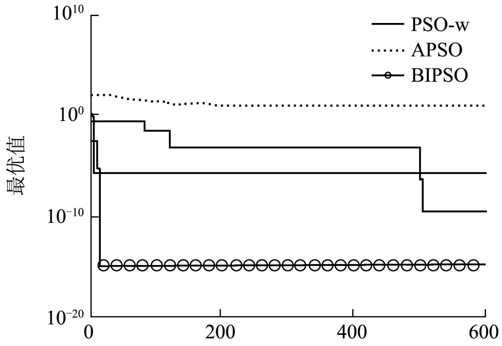
函数	方法	best	worst	mean	stdDev	ration/%	T/s
f_{13}	BPSO	2.1187634205	2.1187634193	2.1187634197	4.3e−10	90	3.8
	BIPSO	2.1187634205	2.1187634195	2.1187634198	4.5e−11	100	4.1
f_{14}	BPSO	200.0000	199.99999	199.99999	4.15e−14	68.33	4.0
	BIPSO	200.0000	200.00000	200.00000	0	100	6.0



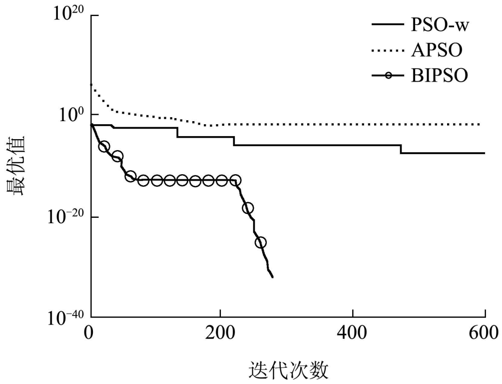
(a) f4 Rosenbrock



(b) f8 Rastrigin



(c) f10 Ackley



(d) f12 广义罚值

图 2 3种算法独立运行30次平均进化曲线比较图

Fig. 2 Comparison of three PSOs algorithms

4.3 多峰函数实验(Multimodal functions experiments)

$$f_{13}=1+x\sin(4\pi x)-y\sin(4\pi y+\pi)+\frac{\sin(6\sqrt{x^2+y^2})}{6\sqrt{x^2+y^2+10^{-15}}},-1\leqslant x,y\leqslant 1.$$

f_{13}函数有4个全局最大值2.118, 对称分布于(0.64, 0.64). 存在大量局部极大值, 尤其是在中间区域有一取值与局最大值接近的局部极大值(2.007)凸台.

$$f_{14}=200-(x^2+y-11)^2-(x+y^2-7)^2,$$
$$-6\leqslant x,y\leqslant 6.$$

F_{14}函数是改进的Himmelbau函数[17], 为典型不可分的、等高、非等距的二维多峰函数, 在定义域内有4个相等的峰值, 理论值均为200.0性能不佳的算法, 很难搜索到全部4个峰值.

为验证改进免疫算法对双态微粒群性能的影响选取上述多峰(模态)函数. 表4中best, worst, mean, stdDev, T分别表示测试获得的最优值, 最差值, 平均值, 及标准均方差, 求解时间, ration为同时找到全局多个极值点的百分比. BPSO为双态微粒群算法, 从表4中可以看出f_{13}, f_{14}函数优化中能够搜索多峰函数的所有全局最优解, 且每个最优解精度都达到了理想值, 针对多个个体极值进行克隆变异的克隆选择机制和受体编辑引入有利于BIPSO多峰全局寻优. 但BIPSO的求解时间大于BPSO.

表5中Means(x,y)为平均坐标, Mean为均值, 从表中可以看出BIPSO能够搜索多峰函数的所有全局最优解, 且每个最优解精度都达到了理想值, BIPSO求解精度优于NAFSA[17](生境人工鱼群算法), 其中参数选取为F_{14}.

表 5 BIPSO与生境人工鱼群算法的比较

Table 5 Comparison between BIPSO and NAFSA

方法	Means(x,y)	Mean
NAFSA[17]	(3.5861,−1.8479)	199.9999
	(2.9989,1.9972)	199.9999
	(−2.8056,3.1331)	199.9998
	(−3.7769,−3.2818)	199.9997
BIPSO	(3.5844283310, −1.8481265036)	200.0000
	(3.0000000116, 2.0000000021)	200.0000
	(−2.8051180703, 3.1313125328)	200.0000
	(−3.7793102444,−3.2831859960)	200.0000

4.4 超高维函数实验(High-dimension function)

对函数f_1,f_4,f_8,f_{11}进行性能测试, 测试停止条件为达到全局最优值(f_1,f_4,f_8,f_{11}全局最小值为0). 从图3可以看出在测试条件一定的情况下, BIPSO的函数值计算次数并没有随着维数的增大而增加. 说明BIPSO在高维函数优化方面有明显的优势. 由于BIPSO算法很好地保持群体的多样性, 对超高维也能够在几乎相同代数下找到理想全局最优值.

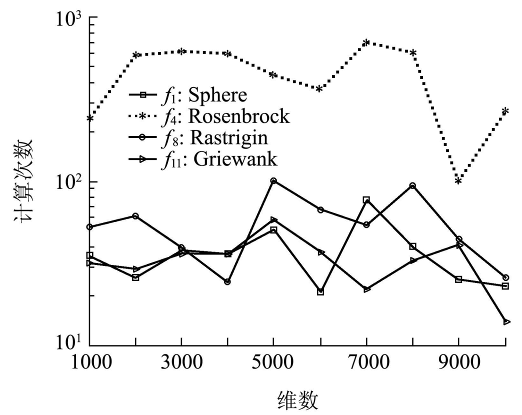


图 3 函数值计算次数与函数维数的关系曲线

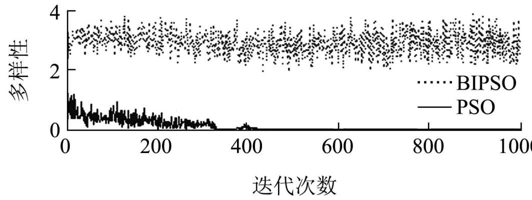
Fig. 3 The relation between average evaluations and dimensions

4.5 多样性分析(Diversity)

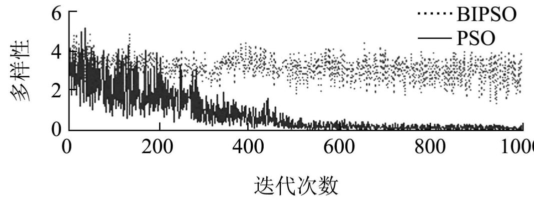
$$\text{diversity}(S)=\frac{1}{|L|.|S|}\sum_{i=1}^{|S|}\sqrt{\sum_{i=1}^{|D|}(x_{ij}-\bar{x}_j)^2},$$

其中S为种群, $|S|$为种群所含粒子的个数, $|L|$为搜索空间的最长半径, D为问题的维数, X_{ij}为第i个粒子的第j个分量, $\bar{x}_j$是所有粒子第j维的平均值.

由图4可看出基本PSO算法的种群多样性随着进化代数的增加而减少, 多样性在进化后期快速降低. 而BIPSO算法在进化过程使种群的多样性得到了很好的保证, 表明了BIPSO算法对改善种群多样性的有效性.



(a) f_{13}



(b) f_{14}-Himmelbau

图4 函数多样性变化曲线比较

Fig. 4 Change curve of diversity of function

5 结论(Conclusions)

分析了基本微粒群算法易陷入局部极值的原因, 结合鸟类, 鱼类捕食行为及人工免疫系统的特点, 提出了一种双态免疫微粒群算法. 一方面把微粒群分为"捕食"与"探索"两种状态, 扩大解的搜索空间, 平衡PSO算法的开采和勘探能力, 有效抑制了收敛过程中的早熟停滞现象, 另一方面引入免疫系统的克隆选择和受体编辑机制, 提高群体的多样性分布, 进一步增强群体逃离局部极值能力及多峰极值全局寻优能力. 用经典测试函数进行仿真实验并将结果与多种改进的PSO进行比较, 文中BIPSO算法在求解复杂优化问题时优势明显, 可以有效地避免粒子陷入局部最优, 同时也具有较快的收敛速度, 达到了非常好的优化效果, 群体多样性得到了很好的保证.

参考文献(References):

[1] EBERHART R, KENNEDY J A. A new optimizer using particle swarmtheory[C] //*Proceeding of International Symposium on Micromachine and Human Science*. Nagoya, Japan: IEEE, 1995: 39 – 43.

[2] HO S Y, LIN H S, LIAUH W H, et al. OPSO: Orthogonal particle swarm optimization and its application to task assignment problems[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: System and Humans*, 2008, 38(2): 288 – 298.

[3] RATNAWEERA A, HALGAMUGE S K, WWATSON H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients[J]. *IEEE Transactions on Evolutionary Computation* , 2004, 8(3): 240 – 255.

[4] EBERHART R C, SHI Y H. Guest editorial-special issue particle swarm optimization[J]. *IEEE Transactions on Evolutionary Computation*, 2004, 8(3): 201 – 203.

[5] 倪庆剑, 张志政, 王蓁蓁, 等. 一种基于可变多簇结构的动态概率粒子群优化算法[J]. 软件学报, 2009, 20(2): 339 – 348.
(NI Qingjian, ZHANG Zhizheng, WANG Zhenzhen, et al. Dynamic probabilistic particle swarm optimization based on varying multi-cluster structure[J]. *Journal of Software*, 2009, 20(2): 339 – 348.)

[6] CHEN Y P, PENG W C, ANDJIAN M C. Particle swarm optimization withrecombination and dynamic linkage discovery[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2007, 37(6): 1460 – 1470.

[7] ANDREWS P S. An investigation into mutation operators for particleswarm optimization[C] //*Proceeding of IEEE Congress on Evolutionary Computation*. Vancouver, BC, Canada, IEEE, 2006: 1044 – 1051.

[8] LIANG J J, SUGANTHAN P N. Dynamic multi-swarm particle swarm optimizer with local search[C] //*Proceeding of IEEE Congress on Evolutionary Computation*. Singapore, IEEE, 2005, 1: 522 – 528.

[9] LIANG J J, QIN A K, SUGANTHAN P N, et al. Comprehensivelearning particle swarm optimizer for global optimization of multimodalfunctions[J]. *IEEE Transactions on Evolutionary Computation*, 2006, 10(3): 281 – 295.

[10] ZHAN Z H, ZHANG J, LI Y, et al. Adaptive particle swarm optimization[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2009, 39(6): 1362 – 1380.

[11] DECASTRO L, VON ZUBEN F J. Learning and optimization using the clonal selection principle[J]. *IEEE Transactions on Evolutionary Computation*, 2002, 6(3): 239 – 251.

[12] HO S L, SHI Y Y, GUANG Z N, et al. A particle swarm optimization-based method for multiobjective design optimizations[J]. *IEEE Transactions on Magnetics*, 2005, 41(5): 1756 – 1759.

[13] 任子武, 伞冶. 实数遗传算法的改进及性能研究[J]. 电子学报, 2007, 35(2): 270 – 271.
(REN Ziwu, SAN Ye. Improvement of real valued genetic algorithm and performance study[J]. *Acta Electronica Sinica*, 2007, 35(2): 270 – 271.)

[14] GE H W, SUN L, LIANG Y C, et al. An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: System and Humans*, 2008, 38(2): 358 – 363.

[15] 李宏, 唐焕文, 郭崇慧. 一类进化策略的收敛性分析[J]. 运筹学学报, 1999, 3(4): 79 – 83.
(LI Hong, TANG Huanwen, GUO Chonghui. The convergence analysis of a class of evolution strategies[J]. *Operations Research Transactions*, 1999, 3(4): 79 – 83.)

[16] 郭崇慧, 唐焕文. 演化策略的全局收敛性[J]. 计算数学, 2001, 23(1): 105 – 110.
(GUO Chonghui, TANG Huanwen. Global convergence properties of evolution strategies[J]. *Mathematica Numerica Sinica*, 2001, 23(1): 105 – 110.)

[17] 张梅凤, 邵诚. 多峰函数优化的生境人工鱼群算法[J]. 控制理论与应用, 2008, 25(4): 773 – 776.
(ZHANG Meifeng, SHAO Cheng. Niche artificial fish swarm algorithm for multimodai function optimization[J]. *Control Theory & Applications*, 2008, 25(4): 773 – 776.)

作者简介:

刘朝华 (1983—), 男, 博士研究生, 主要从事计算机控制技术、智能控制理论与应用、智能系统的研究, E-mail: 163liuzhaohua@163.com;

张英杰 (1970—), 男, 博士, 副教授, 主要从事复杂工业系统计算机控制、智能计算、节能减排管控一体化研究, E-mail: zhangyj@hnu.edu.cn;

章 兢 (1957—), 男, 教授, 博士生导师, 主要从事智能控制理论与应用、复杂系统工业控制、节能减排管控一体化、智能系统的研究, E-mail: zhangj@hnu.cn;

吴建辉 (1970—), 男, 博士研究生, 主要从事智能计算、数字系统的研究, E-mail: wujianhui123@163.net.