文章编号: 1001-0920(2007)01-0117-04

基于混沌变异的小生境粒子群算法

贾东立[1,2], 张家树[2]
(1. 河北工程大学 信息与电气工程学院, 河北 邯郸 056038;
2. 西南交通大学 信号与信息处理省重点实验室, 成都 610031)

摘 要: 针对粒子群算法早熟收敛和搜索精度低的问题,提出了基于混沌变异的小生境粒子群算法(NCPSO). 该算法结合小生境技术并加入了淘汰机制,使算法具有良好的全局寻优能力. 变尺度混沌变异具有精细的局部遍历搜索性能,使算法具有较高的搜索精度. 实验结果表明,NCPSO 算法可有效避免标准 PSO 算法的早熟收敛,具有寻优能力强、搜索精度高、稳定性好等优点,适合于工程应用中的复杂函数优化问题.

关键词: 混沌变异; 小生境; 粒子群优化算法
中图分类号: TP301.6 **文献标识码**: A

Niche particle swarm optimization combined with chaotic mutation

JIA Dong-li[1,2], *ZHANG Jia-shu*[2]
(1. Department of Information and Electrical Engineering, Hebei University of Engineering, Handan 056038, China;
2. Signal and Information Processing Key Laboratory of Sichuan Province, Southwest Jiaotong University, Chengdu 610031, China. Correspondent: JIA Dong-li, E-mail: jwdsli@163.com)

Abstract: Niche chaotic mutation particle swarm optimization (NCPSO) is proposed to overcome the problem of the premature and low precision of the standard PSO. In this algorithm, niching methods and eliminating strategy are introduced to improve the global optimizing ability. Further, shrinking chaotic mutation, which behaves well in local searching, is introduced to improve the solution. Simulations show that NCPSO can avoid premature effectively and has powerful optimizing ability, good stability and higher optimizing precision.

Key words: Chaotic mutation; Niche; Particle swarm optimization algorithm

1 引 言

粒子群优化(PSO)算法[1,2]是模拟鸟群、鱼群和人类社会行为规律的基于群智能的演化计算技术. PSO 算法具有设置参数少、计算简单、收敛速度快、鲁棒性好等优点,在函数优化、神经网络训练、模糊系统控制、组合优化、机器人路径规划等领域得到了广泛应用[3,4].

PSO 算法也是近年来演化计算的研究热点之一. 文献[5]对 PSO 算法的稳定性和收敛性作了初步分析,指出基本 PSO 算法无法保证收敛到全局最优. 目前已提出了多种改进的 PSO 算法,但这些算法大多着眼于 PSO 的参数选择或某个参数的动态修改策略,难以克服 PSO 算法易陷入局部极小的固有弱点. 文献[6]提出混沌优化与 PSO 相结合的 CPSO 算法,利用混沌优化提高 PSO 算法的搜索精度,在低维函数优化中效果良好,但在高维函数优化中效果不佳. 文献[7]指出粒子的趋同性是算法陷入局部最优的主要原因,并提出以多种群协同优化(PSCO)策略克服早熟收敛和提高全局收敛性,但没有给出如何将各个子种群汇聚收敛到同一个极值点的方法.

本文针对这一问题,提出以全局性更好的小生境策略代替协同策略,小生境中各个子种群互相排斥,动态形成自己独立的搜索空间,各自追逐自己搜索范围内的极值点,使小种群在搜索空间有效地分布,避免了协同中的种群汇聚;进一步引入淘汰更新机制,迭代一定次数后更新最劣子种群,以保证整个种群不断向前进化,直至搜索到全局最优点;引入变尺度混沌变异,进一步提高了本文算法的搜索精度.

2 基本粒子群算法

收稿日期: 2005-09-26; **修回日期**: 2005-12-03.
基金项目: 国家自然科学基金项目(60572027); 四川省杰出青年基金项目(0326ZQ026-033).
作者简介: 贾东立(1972 -),男,河北大名人,助教,硕士,从事智能计算、图像处理的研究; 张家树(1965 -),男,四川西充人,教授,博士生导师,从事通信信号处理、混沌信息工程等研究.

PSO 算法是一种基于迭代的优化方法,系统初始化为一组随机解,通过迭代搜寻最优值,粒子在解空间追随最优的粒子进行搜索. 在每次迭代中,粒子通过跟踪两个极值来更新自己:一个是粒子本身目前找到的最优解,即个体极值;另一个是整个种群目前找到的最优解,称为全局极值. 其数学描述及迭代公式如下:

设函数优化问题为

$$\min f(X), X = [x_1, x_2, \cdots, x_n],$$
$$\text{s.t. } x_i \in [a_i, b_i], i = 1, 2, \cdots, n. \quad (1)$$

其中:$f(X)$ 为目标函数,n 为自变量 x_i 的维数,$[a_i, b_i]$ 为 x_i 的搜索区间.

PSO 算法是基于群智能的迭代演化技术,群中每个粒子 X_i 代表了目标函数 $f(X)$ 的一种可能解. 粒子迭代计算公式如下:

$$V_i[t+1] = wV_i[t] + c_1\text{rand}(\cdot)(P_i - X_i) + c_2\text{rand}(\cdot)(P_g - X_i), \quad (2)$$
$$X_i[t+1] = X_i[t] + V_i[t+1]. \quad (3)$$

其中:$V_i = [v_{i1}, v_{i2}, \cdots, v_{in}]$ 称为粒子 i 的速度,代表了粒子 i 现在位置与其下一步目标位置之间的距离;$X_i = [x_{i1}, x_{i2}, \cdots, x_{in}]$ 是粒子 i 的目前位置;P_i 是粒子 i 迄今为止搜索到的最优解;P_g 是整个群体迄今搜索到的最优解;c_1 和 c_2 称为加速因子,控制粒子向极值追踪的速度,一般取 2;w 称为惯性因子,控制上一步速度对下一步的影响;rand 为 $[0,1]$ 间的随机数. 粒子群按上式进行迭代,直至搜索到满意解或达到一定搜索代数,停止并输出搜索结果.

3 基于混沌变异的小生境粒子群算法

3.1 RCS 小生境进化策略

小生境策略以其能有效解决多峰函数优化问题而广泛应用于进化算法[8]. 在遗传算法中,小生境技术主要采取排挤法和共享法. 这两种方法均需较多的个体参与竞争共享资源,造成了时间的浪费. 本文采用文献[9]提出的 RCS 策略作为构造小生境的基础. 该策略只需简单的几个个体即可实现,通过控制子种群之间的排挤和竞争,使各个子种群在进化中动态形成各自独立的搜索空间,从而实现对多个局部极值进行同步搜索,避免了算法的早熟收敛.

在 PSO 算法中,每个粒子都追逐种群中的最优个体,控制了种群中最优个体的搜索方向,也就控制了整个种群的搜索方向. 故本文 RCS 策略只对各个子种群中的最优个体进行控制,从而减少了计算量,加快了运算速度.

RCS 小生境算法具体实现如下:设 PSO 算法种群由 N 个子种群组成,每个子种群中最优个体为 $P_{n\text{best}}$.

Step1: For $i = 1$ to $N-1$, For $j = i+1$ to N;

Step2: When d_{ij} (两个子种群最优个体 $P_{i\text{best}}$ 与 $P_{j\text{best}}$ 之间的距离) $< R_{\text{niche}}$ (小生境半径),比较两个小生境最优个体的适应度,低者置零,高者保持不变;

Step3: 对置零的最优个体重新初始化,并在其所在的小生境内重新选择最优个体,转 Step1,直至每个小生境都具有最优个体.

算法中的小生境半径 R_{niche} 定义了各个子种群独立的搜索空间,一旦某个小生境最优个体进入了其他小生境的搜索空间,则重置该个体,并在其所在的小生境内重新选择最优个体. 从而使每个小生境子种群自然形成不同的独立搜索空间,追逐不同的局部极值,有效地减少了标准 PSO 算法的所有个体作为整体种群陷入局部最优的概率.

3.2 变尺度混沌变异

混沌是自然界广泛存在的一种非线性现象,具有随机性、遍历性、初始条件敏感性等特点,已被广泛应用于随机优化[10],在局部寻优领域具有优越的性能. 本文使用的混沌映射 Logistic 迭代方程为

$$\beta_j^{k+1} = \mu\beta_j^k(1-\beta_j^k), k = 1, 2, \cdots,$$
$$\beta_j \in (0,1), \beta_j \neq 0.25, 0.5, 0.75. \quad (4)$$

其中 β_j 对应于粒子 X_i 的第 j 个混沌变量. 当 $\mu = 4$ 时,Logistic 方程完全进入混沌状态. 图 1 展示了 $\beta_j = 0.112\ 3$,Logistic 方程迭代 200 次的混沌遍历特性. 实验证明,利用混沌的遍历性,可以很好地实现局部搜索.

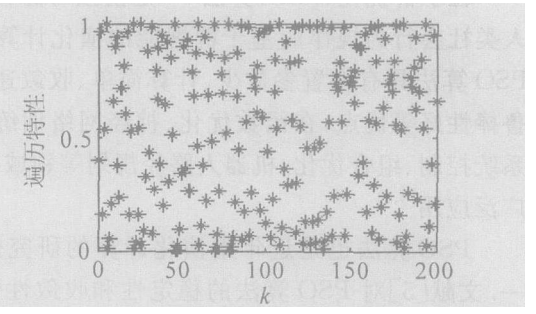

图 1 Logistic 映射的混沌遍历特性

在寻优过程中,对每个小生境子种群的最优个体进行混沌迭代变异,变异空间随着代数的增加而逐渐缩小. 对第 i 个子种群的最优个体 $P_{i\text{best}} = [x_1, x_2, \cdots, x_j, \cdots, x_n]$ 进行混沌迭代变异,步骤如下:

Step1: 随机设 β_j^k, $k = 1$;

Step2: 通过 Logistic 迭代方程得到 β_j^{k+1};

Step3: 由下式得到变异尺度:

$$P_c = x_{j,\min} + \beta_j^{k+1}(x_{j,\max} - x_{j,\min}); \quad (5)$$

Step4: 由下式得到变量 x_j 的新位置:

$$x_j^{k+1} = (1 - \lambda_g)\, x_j^k + \lambda_g P_c; \qquad (6)$$

Step5:重新计算 P_{ibest} 的适应值,如果大于原适应值或混沌迭代达到一定步数,则停止混沌搜索;否则转 Step2.

式(6)中 λ_g 称为收缩因子,它决定了变量 x_j 的变异空间,由下式得到:

$$\lambda_g = 1 - ((g - 1)/g)^m. \qquad (7)$$

其中:g 为粒子群的进化代数,m 用于控制收缩速度.

由式(6)可以看出,最优粒子 P_{ibest} 变量的搜索空间随着代数的增加而围绕小生境的极点逐渐缩小.这样,在进化初期变异尺度大,有利于算法在广阔的空间搜索全局最优解;在进化后期变异尺度小,在小空间内紧紧围绕局部极点精细搜索,有利于提高解的精度.

3.3　基于混沌变异的小生境粒子群算法

结合小生境策略全局优化与变尺度混沌变异精细搜索各自的优点,本文提出一种全新的粒子群算法,并在算法中引入了种群淘汰策略,结合小生境的子种群竞争策略一起使用.运行中首先利用 RCS 竞争策略,使各个小生境子种群形成独立的搜索空间,追逐不同的极值点;然后每隔一定代数,对陷入局部最优的最劣子种群进行随机初始化.这样可使种群在不断的竞争和更新中向前进化,从而避免了算法早熟收敛,保证了收敛到全局最优.而没有更新的小生境种群继续向前进化,又保证了搜索精度的连续提高.

在 PSO 算法中,惯性因子 w 对算法的性能有很大的影响,较大的 w 有利于大范围搜索,使种群跳出局部极值点,而较小的 w 则有利于种群快速收敛.本文采用自适应调整 w 的策略,随着迭代的进行,以下式动态递减 w 的值:

$$w = w_{\max} - g\frac{w_{\max} - w_{\min}}{T_{\max}}. \qquad (8)$$

其中:$w_{\max}$ 和 $w_{\min}$ 分别为 w 的最大值和最小值,g 为当前代数,$T_{\max}$ 为迭代截止代数.

NCPSO 算法具体实现如下:

Step1:初始化小生境粒子种群;

Step2:计算粒子适应度,找出每个小生境种群中的最优粒子;

Step3:实施 RCS 小生境淘汰选择进化策略,确定每个小生境独立搜索空间的最优个体;

Step4:如果迭代次数达到一定代数,则对最劣小生境子种群进行更新初始化;

Step5:对所有小生境最优个体实行变尺度混沌变异,进一步提高搜索精度;

Step6:对每一小生境种群独立进行 PSO 优化;

Step7:如果满足结束条件,则停止迭代,并输出最优解,否则转 Step2.

4　实验分析

4.1　实验方法

为了验证 NCPSO 算法对复杂高维函数的有效性,本文采用下列高维基准测试函数进行测试.

1) Rosenbrock 函数(RO)

$$f_1(\vec{x}) = \sum_{i}^{n-1}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2),$$

$$x_i \in [-30, 30], n = 30.$$

当全局最小为 $\vec{x} = (1, 1, \cdots, 1)$ 时,函数值为 0.

2) Rastrigin 函数(RA)

$$f_2(\vec{x}) = \sum_{i}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10),$$

$$x_i \in [-5.12, 5.12], n = 30.$$

当全局最小为 $\vec{x} = (0, 0, \cdots, 0)$ 时,函数值为 0.

3) Ackley 函数(AC)

$$f_3(\vec{x}) = -20\exp\left[-0.2\sqrt{\frac{1}{n}\sum_{i}^{n}x_i^2}\right] - \exp\left[\frac{1}{n}\sum_{i}^{n}\cos(2\pi x_i)\right] + 20 + e,$$

$$x_i \in [-30, 30], n = 30.$$

当全局最小为 $\vec{x} = (0, 0, \cdots, 0)$ 时,函数值为 0.

本文比较了 NCPSO 算法与标准 PSO 算法、CPSO 算法和 PSCO 算法对基准测试函数的效果.算法的种群粒子数都设为 25.其中 NCPSO 和 PSCO 设为 5 个子种群,每个子种群的粒子数为 5.CPSO 和 PSCO 的参数设置同文献[6,7].NCPSO 和标准 PSO 的参数设置如下:

$$c_1 = c_2 = 2, w_{\max} = 0.9, w_{\min} = 0.1.$$

4.2　实验结果

4.2.1　固定运行代数结果

算法运行代数统一设为 2 000,2 000 代后认为算法陷入停滞.表 1 为 50 次独立运行的测试结果.

从表 1 可以看出,NCPSO 算法能解决其他算法难以解决的高维多极函数的优化问题,并具有更强的寻优能力和较高的搜索精度.

4.2.2　稳定性分析

如果某次独立搜索的结果在最优值的半径为 1 的范围内,则认为是一次成功搜索.定义成功搜索率 $R_S = (N_v/50) \times 100$,其中 N_v 为 50 次运行中成功搜索的次数.

几种算法对基准测试函数的成功率如表 1 第 5 列所示.从中可以看出,NCPSO 算法的搜索成功率

表 1 高维复杂函数优化测试结果

F	Algorithm	Average	Standard error	R_S/ %
RO	NCPSO	0.261	0.313	96
	CPSO	1.741	2.858	50
	PSCO	18.184	5.364	0
	PSO	85.514	35.63	0
BA	NCPSO	0.15E-4	0.23E-4	100
	CPSO	6.462	8.343	32
	PSCO	9.580	10.220	8
	PSO	25.810	6.946	0
AC	NCPSO	0.31E-6	0.16E-5	100
	CPSO	0.262	0.682	88
	PSCO	0.502	0.431	92
	PSO	0.452	0.594	68

最高,是最稳定的一种算法.

为进一步说明 NCPSO 算法良好的寻优能力,本文给出了几种算法应用于 RO 函数的收敛曲线,如图 2 所示. 可以看出,NCPSO 算法比其他算法具有更强的寻优能力.

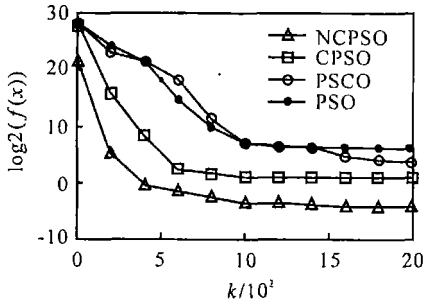

图 2 几种算法对 RO 函数的测试比较

本文同时给出了 NCPSO 算法对 RO 函数的测试曲线,如图 3 所示. 可以看出,小生境淘汰选择策略不断重置最劣小生境子种群,对粒子群摆脱局部最小和避免早熟的效果明显.

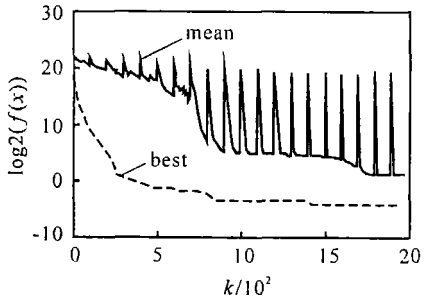

图 3 RO 函数收敛曲线

5 结 论

本文算法利用广泛应用于多目标函数优化的小生境策略掌控粒子群的全局搜索方向,其竞争策略使子种群在搜索过程中自动追寻不同的极值点,形成不同的搜索空间;淘汰策略则使子种群不断更新. 两种策略的有机结合,成功地避免了算法的早熟收敛,提高了全局寻优能力;变尺度混沌变异的引入,则显著提高了算法的搜索精度. 实验结果表明,与传统的 PSO 算法相比,本文算法寻优能力强,搜索精度高,稳定性好,适用于处理高维多极连续函数的优化问题.

参考文献(References)

[1] Kennedy J, Eberthart R. Particle swarm optimization [C]. Proc IEEE Int Conf on Neural Networks. Piscataway, 1995: 1942-1948.

[2] Eberhart R, Kennedy J. A new optimizer using particle swarm theory [C]. Proc of Int'1 Symp on Micro Machine and Human Science. Piscataway: IEEE Service Center, 1995: 39-43.

[3] Eberthart R, Shi Y. Particle swarm optimization: Developments, applications and resources [C]. Proc of the Congress on Evolutionary and Computation. Piscataway: IEEE Service Center, 2001: 81-86.

[4] 高尚,韩斌,吴小俊,等. 求解旅行商问题的混合粒子群优化算法[J]. 控制与决策,2004,19(11):1286-1289.
(Gao S, Han B, Wu X J, et al. Solving traveling salesman problem by hybrid particle swarm optimization algorithm[J]. Control and Decision, 2004, 19(11): 1286-1289.)

[5] Van den Bergh. An analysis of particle swarm optimizers [D]. South Africa: University of Pretoria, 2002.

[6] Liu B, Wang L, Jin Y H, et al. Improved particle swarm optimization combined with chaos [J]. Chaos, Solitons and Fractals, 2005, 25(5): 1261-1271.

[7] 李爱国. 多粒子群协同优化算法[J]. 复旦大学学报,2004,43(5):923-925.
(Li A G. Particle swarms cooperative optimizer[J]. J of Fudan University, 2004, 43(5): 923-925.)

[8] 陈辉,张家树,张超. 实数编码混沌量子遗传算法[J]. 控制与决策,2005,20(11):1300-1303.
(Chen H, Zhang J S, Zhang C. Real-coded chaotic quantum-inspired genetic algorithm [J]. Control and Decision, 2005: 20(11): 1300-1303.)

[9] Lee C G, Cho D H, Jung H K. Niche genetic algorithm with restricted competition selection for multimodal function optimization [J]. IEEE Trans on Magnetics, 1999, 35(3): 1122-1125.

[10] 贾东立,张家树,张超. 基于混沌遗传算法的基元提取[J]. 西南交通大学学报,2005,40(4):496-500.
(Jia D L, Zhang J S, Zhang C. Geometric primitive extraction using chaos genetic algorithm [J]. J of Southwest Jiaotong University, 2005, 40(4): 496-500.)