第34卷 第17期 计 算 机 工 程 2008年9月
Vol.34 No.17 Computer Engineering September 2008
·人工智能及识别技术· 文章编号：1000—3428(2008)17—0205—03 文献标识码：A 中图分类号：TP301.6

一种局部与全局相结合的微粒群优化算法

胡乃平，宋世芳
(青岛科技大学信息科学技术学院，青岛 266042)

摘 要：提出一种基于局部与全局搜索相结合的粒子群算法。该算法结合全局和局部PSO算法的优点，摆脱局部极优点的束缚，保持前期搜索速度快的特性，提高全局搜索能力。仿真实验表明，与标准微粒群优化算法相比，该算法的全局收敛性能得到显著提高，有效地避免微粒群优化算法中的局部收敛问题，并快速搜索到全局最优解。

关键词：微粒群优化；局部；全局；仿真

Local and Global Combined Particle Swarm Optimization Algorithm

HU Nai-ping, SONG Shi-fang
(School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao 266042)

【Abstract】 This paper proposes a particle swarm optimization based on local and global combined search. Combining the strongpoint of local search and global search, the new algorithm reduces the possibility of trapping at the local optimum. The advanced algorithm maintains the characteristic of fast search in the early convergence phase, and improves the global search ability. Experimental results indicate that Local and Global Combined Particle Swarm Optimization (LGCPSO) has the advantage of convergence property over PSO. The algorithm avoids the local convergence problem effectively, and has the ability to get a convergence velocity quickly.

【Key words】 Particle Swarm Optimization(PSO); local; global; simulation

1 概述

微粒群优化(Particle Swarm Optimization, PSO)是由Kennedy 和Eberhart等人于1995年提出的一种新的进化计算算法[1-2]，它来源于鸟类或鱼类觅食过程中迁徙和群集的模拟，是群智能的代表性方法之一。该算法是一种基于群体寻优的启发式搜索算法。与基于达尔文“适者生存，优胜劣汰”进化思想的遗传算法不同，微粒群优化算法的核心思想是群体中个体之间信息的社会共享和协同进化。

PSO概念和参数调整都很简单而且容易编程，它既保持传统进化算法深刻的群体智慧背景，又有许多良好性能。在短短几年中，其获得快速发展，在诸多领域得到应用[3-4]。

虽然PSO算法发展迅速并取得了可观的研究成果，但其在中大规模问题中易陷入局部最优。根据PSO算法的这一缺点，本文提出局部和全局搜索相结合的粒子群算法(Local and Global Combine Particle Swarm Optimization, LGCPSO)，使其能跳出局部最优位置，进入解空间的其他区域进行搜索，增强PSO算法的全局搜索能力，加快搜索速度。将该算法应用于优化连续函数，优化效果更好。

2 基本微粒群优化算法

2.1 算法原理

微粒群中每一个个体代表待求解问题的一个潜在解，并具有位置和速度两个状态量。在算法开始时，先随机初始化一群微粒，然后通过群体的协作和迭代运算找到最优解。在每一次迭代过程中，各微粒根据简单的位置和速度更新规则改变自身的状态量，通过跟踪微粒自身的历史最优值和邻域内所有个体找到的最优值不断“完善”自己。下面给出微粒群优化算法的数学描述[5]。

假设在一个D维的目标搜索空间中，有m个粒子组成一个种群，其中，第i个粒子表示为一个D维的向量$\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{iD}),i=1,2,\cdots,m$，即第$i$个粒子在$D$维的搜索空间中的位置是$\boldsymbol{X}_i$；$\boldsymbol{V}_i=(v_{i1},v_{i2},\cdots,v_{iD})$是第$i$个粒子的飞行速度；$\boldsymbol{P}_i=(p_{i1},p_{i2},\cdots,p_{iD})$是第$i$个粒子迄今为止搜索到的最优位置$pbest$；$\boldsymbol{P}_g=(p_{g1},p_{g2},\cdots,p_{gD})$是整个粒子群迄今为止搜索到的最优位置$gbest$。基本PSO算法的递推方程如下：

$$\boldsymbol{v}_{id}(k+1)=\boldsymbol{v}_{id}(k)+c_1r_1(\boldsymbol{p}_{id}(k)-\boldsymbol{x}_{id}(k))+c_2r_2(\boldsymbol{p}_{gd}(k)-\boldsymbol{x}_{id}(k)) \quad (1)$$

$$\boldsymbol{x}_{id}(k+1)=\boldsymbol{x}_{id}(k)+\boldsymbol{v}_{id}(k+1)\ (i=1,2,\cdots,m;d=1,2,\cdots,D) \quad (2)$$

在式(1)和式(2)中，k为迭代代数；学习因子c_1和c_2是非负常数，一般取值为2；r_1和r_2是均匀分布于[0,1]之间的两个随机数；$\boldsymbol{v}_{id}\in[-\boldsymbol{v}_{\max},\boldsymbol{v}_{\max}]$，$\boldsymbol{v}_{\max}$是用户自己设定的常数，$\boldsymbol{v}_{id}(k)$为第$k$次迭代时粒子$i$飞行速度矢量的第$d$维分量；$\boldsymbol{x}_{id}(k)$是第$k$次迭代时粒子$i$位置矢量的第$d$维分量；$\boldsymbol{p}_{id}(k)$是粒子$i$个体最好位置的第$d$维分量；$\boldsymbol{p}_{gd}(k)$为群体最好位置的第$d$维分量；迭代终止条件根据具体问题一般选为最大迭代次数或(和)粒子群迄今为止搜索到的最优位置满足预定最小适应阈值。

2.2 算法解析

在式(1)中，$\boldsymbol{v}_{id}(k)$为微粒先前的速度，它使微粒有在搜索空间中扩张的趋势，从而使算法具有全局搜索的能力；$c_1r_1(\boldsymbol{p}_{id}(k)-\boldsymbol{x}_{id}(k))$为“认知(cognition)”部分，表示微粒吸取

基金项目：山东省自然科学基金资助项目(Y2005G02)
作者简介：胡乃平(1968－)，男，教授，主研方向：计算机控制技术，工业信息化技术；宋世芳，硕士
收稿日期：2007-10-29 **E-mail：**s_shifang@163.com

自身经验知识的过程；$c_2r_2(\boldsymbol{p}_{gd}(k)-\boldsymbol{x}_{id}(k))$ 为“社会(social)”部分，表示微粒学习其他粒子经验的过程，表现了微粒间信息的共享与社会协作。没有 $c_1r_1(\boldsymbol{p}_{id}(k)-\boldsymbol{x}_{id}(k))$ 的模型被称为只有社会部分的模型(social-only model)，该模型收敛速度比较快，但对于复杂问题易陷入局部最优值。没有 $c_2r_2(\boldsymbol{p}_{gd}(k)-\boldsymbol{x}_{id}(k))$ 的模型被称为只有认知的模型(cognition-only model)，该模型在处理问题时，没有得到可以接受的解，因为个体间没有联系，一个规模为 m 的群体等价于 m 个微粒单独运行，所以得到最优解的几率非常小。

如果微粒的邻域包含整个群体，则上述算法为全局模式的微粒群优化算法；否则为局部模式的微粒群优化算法。本文所用微粒群算法为全局与局部相结合的微粒群优化算法。

3 局部与全局搜索相结合的粒子群算法

3.1 局部与全局搜索相结合的微粒群优化算法原理

PSO 算法初始化为一群随机粒子(随机解)，然后通过迭代找到最优解，在每一次迭代中，粒子通过跟踪两个“极值”来更新自己。第 1 个就是粒子本身所找到的最优解，这个解叫做个体极值 *pbest*，第 2 个极值是整个种群目前找到的最优解，这个极值是全局极值 *gbest*；或者只是用其中一部分最优粒子的邻居，那么在所有邻居中的极值就是局部极值 *lbest*。分别采用全局和局部不同极值更新粒子的迭代方程，称为全局版和局部版的 PSO 算法。全局版和局部版的 PSO 算法各有利弊，前者收敛速度快但有时会陷入局部最优值，后者收敛速度慢一点，但很容易跳出局部最优。吸取 PSO 算法的优点，将局部和全局搜索的 PSO 算法相结合，本文提出局部和全局搜索相结合的微粒群算法(Local and Global Combine Particle Swarm Optimization, LGCPSO)。

与基本微粒子群算法不同，LGCPSO 算法的递推方程为

$$\begin{aligned}\boldsymbol{v}_{id}(k+1)=w\boldsymbol{v}_{id}(k)+c_1r_1(\alpha(\boldsymbol{p}_{id}(k)-\boldsymbol{x}_{id}(k))+\\(1-\alpha)(\boldsymbol{p}_{ld}(k)-\boldsymbol{x}_{id}(k)))+c_2r_2(\boldsymbol{p}_{gd}(k)-\boldsymbol{x}_{id}(k))\end{aligned}\tag{3}$$

$$\boldsymbol{x}_{id}(k+1)=\boldsymbol{x}_{id}(k)+\boldsymbol{v}_{id}(k+1)(i=1,2,\cdots,m;d=1,2,\cdots,D)\tag{4}$$

其中，$\boldsymbol{P}_l=(p_{l1},p_{l2},\cdots,p_{lD})$ 是每个种群搜索到的最优位置*lbest*。

LGCPSO 算法递推方程主要通过 3 个部分来更新粒子 i 的新速度：粒子 i 前一次迭代时刻的速度；粒子 i 当前位置与自己最好位置之间的距离；第 k 代种群中粒子最好位置与自己位置之间的距离，再结合粒子 i 当前位置与群体最好位置之间的距离。粒子 i 通过式(3)更新位置的坐标，通过递推方程决定下一步的运动位置。

3.2 LGCPSO 算法的流程

LGCPSO算法主要通过$\boldsymbol{X}_{pbest}$, $\boldsymbol{X}_{pbest}$, $\boldsymbol{X}_{gbest}$3 个部分来传递和共享信息，更新粒子i的新速度和位置，流程如下：

Step1 随机初始化微粒的位置与速度。

Step2 将微粒的 *pbest* 设置为当前位置，*gbest* 设置为初始群体中最佳微粒的位置，*lbest* 为种群最佳微粒位置。

Step3 判断算法收敛准则(群体最大适应度满足条件或超过最大迭代次数)是否满足，如果满足，转向 Step7；否则，执行 Step4。

Step4 对于所有微粒，求适应度，并根据适应度得到新的 *pbest, gbest, lbest*：

如果种群中某微粒适应度优于 *lbest* 的适应度，*lbest* 设置为新位置；如果微粒适应度优于 *pbest* 的适应度，*pbest* 设置为新位置；如果微粒适应度优于 *gbest* 的适应度，*gbest* 设置为新位置。

Step5 根据式(3)更新所有微粒的位置与速度。

Step6 判断算法收敛准则是否满足，如果满足，执行 Step7，否则，转向 Step4。

Step7 输出 *gbest*，算法运行结束。

从上述流程可以看出，差分进化算子变异的微粒群优化算法实际上是在微粒群优化算法的基本框架中增加了种群最优位置 *lbest*，来提高微粒群优化算法跳出局部最优解的能力，在跳出局部最优解的同时，式(3)中群体最佳微粒的位置 *gbest* 保证了收敛速度。

4 性能测试与结果讨论

4.1 测试的连续函数

本文采用 4 个经典的测试函数对 LGCPSO 算法的寻优性能进行测试，并与基本的 PSO 算法作比较。

Sphere model 为

$$f_1(x)=\sum_{i=1}^{n}x_i^2$$

其中，$x_i\in[-100,100]$；最小点 $x^*=(0,0,0,\cdots,0)$；最小值 $f(x^*)=0$。

$$f_2(x)=\sum_{i=1}^{n-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$$

其中，$x_i\in[-30,30]$；最小点 $x^*=(1,1,1,\cdots,1)$；最小值 $f(x^*)=0$。

Ackley's function 为

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

其中，$x_i\in[-32,32]$；最小点 $x^*=(0,0,0,\cdots,0)$；最小值 $f(x^*)=0$。

$$\begin{aligned}f_4(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\times[1+10\sin^2(\pi y_{i+1})]+\\(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)\end{aligned}$$

在 f_4 中，有

$$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\0 & -a\le x_i\le a\\k(-x_i-a)^m & x_i<-a\end{cases}$$

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

其中，$x_i\in[-50,50]$；最小点 $x^*=(0,0,0,\cdots,0)$；最小值 $f(x^*)=0$。

在 4 个测试函数中，f_1,f_2是单峰函数；f_3,f_4是多峰函数。其中，局部最小值个数随着问题规模的增大而增多。

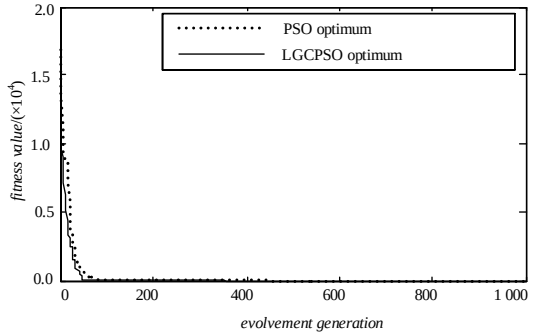
4.2 仿真结果与比较

由于在群体规模较小时，PSO 算法比较容易出现停滞现象，为了更好地验证改进算法的有效性，选取种群微粒数为 100；速度上限 *v*max 取各函数初始范围的上限；$c_1=c_2=2$；惯性权重采用变权重为 0.05~0.45；α 为在[0,1]之间的一个常数，本文在优化连续函数时不妨取 $\alpha=0.5$。对每个连续函数各运行 10 次，比较它们 10 次运行结果各自的最优(全局极小值下界)和平均值。采用 LGCPSO 和 OPSO 算法在不同惯性系数下，运行 10 次所求到最优极值用黑体表示，它们的优化结果比较如表 1 所示。

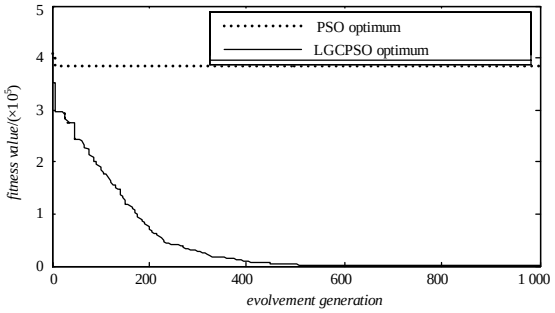
表 1 LGCPSO 与 OPSO 算法优化经典函数的比较结果

Fun	*dim*	The Best	Pop Gen	Minimum/Average		
				w	PSO	LGCPSO
f_1	50	0	100, 1 000	0.05	3.2185e-013/4.2512e-010	1.1961e-027/4.2531e-022
				0.15	6.3237e-006/7.2574e-002	**1.5815e-039/0.1471e-34**
				0.25	**2.0413e-032/4.1309e-022**	2.7802e-038/4.2451e-27
				0.35	2.2657e-028/6.3714e-024	3.2139e-0330.2341e-22
				0.45	7.1763e-020/6.2519e-015	1.7546e-023/4.2153e-15
f_2	30	0	200, 3 000	0.05	15.5102/7.8210e+004	7.6706e-004/1.9531e+003
				0.15	0.1650/ 0.0367e+004	9.3834e-012/23.4
				0.25	0.0031/93	9.8862e-017/10.9
				0.35	**4.8941e-004/53**	**8.8610e-019/8.3**
				0.45	0.0148/9	4.5280e-015/11.7
f_3	150	0	300, 3 000	0.05	20.6284 /28.8588	9.2870 /0.2504
				0.15	18.5687/27.3654	8.2064e-004/0.0453
				0.25	**8.8732/20.3587**	**1.9563e-007/1.0475e-5**
				0.35	19.3486/37.2581	4.2516e-006/7.36529e-3
				0.45	20.2677/47.2583	0.0148/0.5998
f_4	100	0	100, 1 000	0.05	2.3973e+009/2.9862e+009	0.8188/3.5424
				0.15	2.57533e+009/2.9875 e+009	0.0161e-002/7.6354
				0.25	2.76273e+009/3.1463 e+009	1.2544e-006/1.2313
				0.35	**2.31083e+009/2.5378e+009**	**2.3191e-007/1.0103**
				0.45	2.86563e+009/3.1457 e+009	5.6847e-004/25.3369

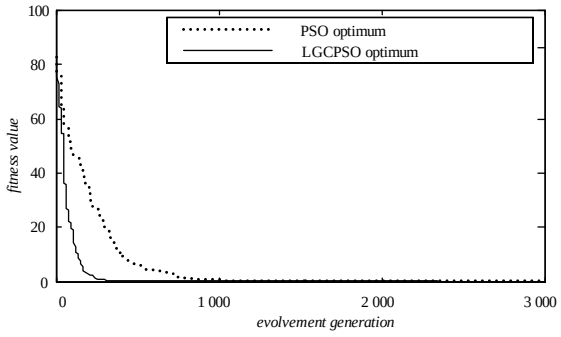
LGCPSO 和 PSO 算法优化 4 个典型连续函数，采用不同惯性系数，它们对各个例子在不同维数情况下搜索最优解过程的最优收敛速度曲线图比较见图 1、图 2。



(a)LGCPSO(*e*=0.3) and PSO for f_1(*dim*=30)

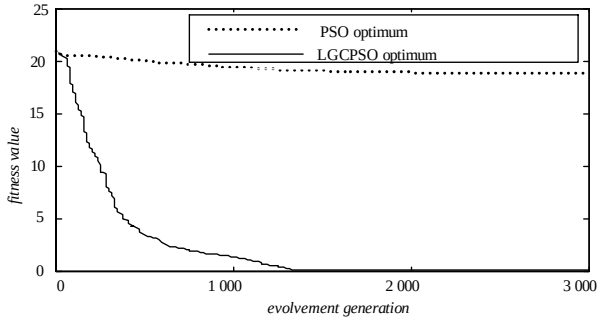


(b)LGCPSO(*e*=0.3) and PSO for f_1(*dim*=150)

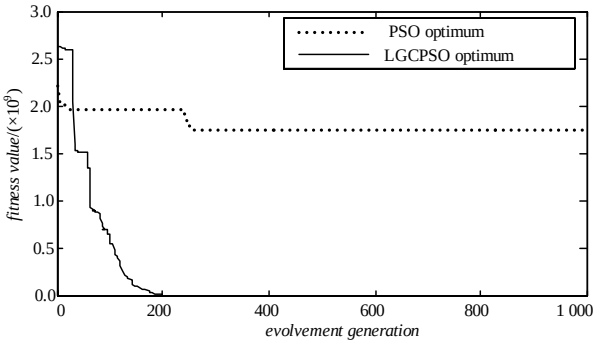


(c)LGCPSO(*e*=0.3) and PSO for f_2(*dim*=30)

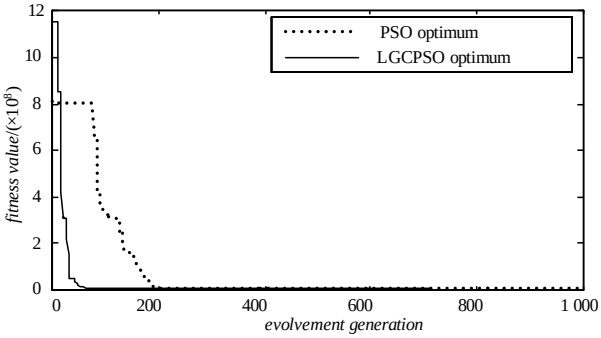
图 1　LGCPSO 与 OPSO 算法优化经典函数的最优收敛速度比较 1



(a)LGCPSO(*e*=0.3) and PSO for f_3(*dim*=50)



(b)LGCPSO(*e*=0.3) and PSO for f_4(*dim*=100)



(c)LGCPSO(*e*=0.3) and PSO for f_4(*dim*=50)

图 2　LGCPSO 与 OPSO 算法优化经典函数的最优收敛速度比较 2

从表 1 中的优化比较结果和曲线图来看，LGCPSO 明显比 PSO 算法优化经典连续函数结果优，LGCPSO 算法优化得到的 10 次最小极值下界、上界和平均值都小于 PSO 所得到的，尤其在优化、大规模问题中，其更为有效。另外从表 1 可以发现，惯性系数 $\omega \in [0.15, 0.45]$ 时，效率最高，此时它优化经典连续函数获得的极值下界，上界和平均值都比采用其他惯性系数得到的值更优。通过单峰、多峰经典函数在不同维数情况下的仿真，发现 LGCPSO 算法优化低维连续函数效果不明显，甚至没有 PSO 效率高，但对相同的问题，当为中、大规模时它的优势就比较明显，如优化函数 f_1, f_4。

5　结束语

针对基本微粒群优化算法存在搜索后期由于搜索策略效率低下导致的后期搜索速度慢、搜索效率低、优化性能不够好的缺陷，提出一种改进的微粒群优化算法——局部和全局搜索相结合的粒子群算法，它能有效地搜索到全局最优解，搜索速度和优化性能比基本微粒群优化算法有明显提高，是一种快速、有效、很有应用前景的优化算法。

（下转第 210 页）