第 27 卷第 4 期
2010 年 4 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 27 No. 4
Apr. 2010

文章编号: 1000−8152(2010)04−0451−06

改进的吸引扩散微粒群算法

陈保娣, 曾建潮

(太原科技大学 复杂系统与计算智能实验室, 山西 太原 030024)

摘要: 为了避免微粒群算法存在的过早收敛问题, 在ARPSO的基础之上, 提出了一个简单的种群多样性度量函数和微粒最好飞行方向的概念, 引入了变异策略, 从而实现了一种改进的吸引扩散微粒群算法MARPSO, 并从理论上分析了MARPSO的局部收敛性和全局收敛性. 对四个经典函数进行了仿真测试, 测试结果表明: 与基本微粒群算法BPSO和ARSPO相比, 该算法能够有效的提高种群多样性, 并且具有较高的收敛速度.

关键词: 微粒群算法; 种群多样性; 微粒最好飞行方向; 收敛

中图分类号: TP301.6　　**文献标识码**: A

Modified attractive and repulsive particle swarm optimization

CHEN Bao-di, ZENG Jian-chao

(Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan Shanxi 030024, China)

Abstract: To avoid the premature convergence, based on the attractive and repulsive particle swarm optimizer(ARPSO), we propose a novel measure function for the population diversity, and a new concept of the particle's best flight direction. A modified ARPSO(MARPSO) is proposed by introducing a mutation strategy. Moreover, theoretical analysis has been made to prove that the algorithm can guarantee local convergence and global convergence. By comparing the simulation results of four classic testing functions with basic PSO(BPSO), ARPSO and MARPSO, this algorithm shows an effective increase in the diversity of population, and the improvement of convergence speed.

Key words: particle swarm optimization; swarm diversity; particle's best flight direction; convergence

1 引言(Introduction)

微粒群算法(particle swarm optimization, PSO) 是由Kennedy和Eberhart[1]于1995年提出的一种基于群智能的自适应进化计算技术. 算法最初受到飞鸟和鱼类集群活动的规律性启发, 用组织社会行为代替了进化算法的自然选择机制, 通过种群间个体协作来实现对问题最优解的搜索. 由于PSO算法概念简单、实现容易、参数较少、能有效解决复杂优化任务[2,3], 所以在过去几年中获得了飞速发展, 并在图像处理、模式识别、多目标优化和游戏设计等很多邻域得到广泛应用[4~6].

然而对于多模态函数优化问题, 如何有效地避免过早收敛是微粒群算法研究的一个热点. 过早收敛的主要原因是由于种群的多样性太低, 因此, 如何度量种群多样性、如何在微粒群的搜索过程中提高种群的多样性就成为解决过早收敛的关键. Riget和Vesterstrom提出了一种基于种群多样性的吸引扩散微粒群算法ARPSO(attractive and repulsive, PSO)[7], 该算法能够有效的提高种群多样性, 从而减少过早收敛的机会, 本文在ARPSO基础之上, 提出了一种更简单有效地种群多样性度量方法, 同时提出了微粒最好飞行方向的概念, 此外, 为了保证算法的局部收敛性能, 本文还引入了速度和位置变异策略.

2 ARPSO算法(The algorithm ARPSO)

为了避免微粒群算法所存在的过早收敛问题, J.Riget[7]提出了一种保证种群多样性的微粒群算法ARPSO. 该算法引入“吸引”(attractive)和“扩散”(repulsive)两个算子, 动态地调整“勘探”与“开发”比例, 从而能更好地提高算法效率. ARPSO算法的进化方程如下:

$$v_i(t+1) = \omega v_i(t) + dir \times c_1 r_1(p_i - x_i(t)) + dir \times c_2 r_2(p_{\rm g} - x_i(t)), \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1). \quad (2)$$

收稿日期: 2008−12−10; 收修改稿日期: 2009−04−26.
基金项目: 教育部重点科研资助项目(204018); 山西省自然科学基金资助项目(2008011027-2).

其中dir为种群飞行方向. 并且提出了种群多样性函数, 描述如下:

$$DS(S)=\frac{1}{|S|\cdot|L|}\cdot\sum_{i=1}^{|S|}\sqrt{\sum_{j=1}^{N}(p_{ij}-\bar{p}_j)^2}. \quad (3)$$

其中: DS表示种群多样性, S为种群, $|S|$为种群所含微粒的个数, $|L|$为搜索空间的最长半径, N为问题的维数, p_{ij}为第i个微粒的第j个分量, $\bar{p}_j$表示所有微粒在第j维的平均值. 在算法运行过程中, 如果种群多样性函数满足$DS(S)<d_{\rm low}$, 则$dir=-1$, 从而种群不再向整体最优位置靠近, 而是纷纷远离该最优位置, 从而执行了“扩散”操作, 而当种群多样性逐步增大, 直至超出上限$d_{\rm high}$时, $dir=1$, 从而种群又开始向整体最优位置靠拢, 即执行了“吸引”操作. J.Riget没有给出参数选择策略[3], 但给出了经验取值, 例如: $d_{\rm high}$为0.25, $d_{\rm low}$为5.0×10^{-6}.

假设存在一个抽象的微粒$\bar{p}$, 它在搜索空间中的位置是$(\bar{p}_1,\bar{p}_2,\cdots,\bar{p}_j,\bar{p}_N)$, $\bar{p}_j$表示所有微粒在第j维的平均值, 那么$\sqrt{\sum_{j=1}^{N}(p_{ij}-\bar{p}_j)^2}$表示第$i$个微粒到微粒$\bar{p}$的距离,

$$1/|S|\sum_{i=1}^{|S|}\sqrt{\sum_{j=1}^{N}(p_{ij}-\bar{p}_j)^2}$$

表示所有微粒到微粒$\bar{p}$的平均距离, 即微粒群的平均半径,

$$1/(|S||L|)\sum_{i=1}^{|S|}\sqrt{\sum_{j=1}^{N}(p_{ij}-\bar{p}_j)^2}$$

表示微粒群的平均半径与搜索空间的最长半径之比, 即微粒群对于搜索空间的覆盖程度, 一般情况下, 覆盖程度越高, 表明微粒越分散, 即种群多样性较好, 反之则表示微粒较密集, 种群多样性较差. ARPSO算法正是在覆盖程度(种群多样性)的指导下进行搜索的. 然而, J.Riget提出的种群多样性函数有以下两个缺点:

1) 种群多样性是在综合计算N维情况下得出的, 而不是分别考虑每一维的多样性, 造成各维之间相互影响, 甚至误导算法的“扩散”和“收缩”运动, 使得算法在高维与低维情况下的效率差别极大, 例如, 对于Rastrigin函数, ARPSO在20维和50维下的平均适应值相差9个数量级, 对于AckleyF1函数, 在20维和50维下的平均适应值相差6个数量级.

2) ARPSO不能保证算法局部收敛.

由上述分析可知, 为了更好地发挥种群多样性对于微粒群进化过程的指导作用, 有必要克服上述种群多样性函数的缺点, 用运算更简单、各维之间互不影响的种群多样性函数来代替J.Riget提出的种群多样性函数.

3 MARPSO算法(Modified ARPSO)

3.1 算法基本思想(Basic idea)

对于全局优化问题:

$$(P)\ \ \min\{f(x):x\in\Omega\subset\mathbb{R}^n\},f:\Omega\subset\mathbb{R}^n\to\mathbb{R}^1.$$

一般而言, 让所有微粒在全局最好微粒$p_{\rm g}$附近进行搜索, 找到最值点的可能性较大, 因而收敛速度快. 然而对于多模态函数而言, 此策略则容易陷入局部极值点, 因此为了兼顾收敛速度和避免过早收敛, 有必要将微粒群分为两部分, 一部分在$p_{\rm g}$ 附近进行局部搜索, 从而找到更好的适应值, 加快收敛速度, 另一部分微粒在距离$p_{\rm g}$ 较远的区域进行全局搜索, 从而找到更好的搜索区域, 使得微粒群跳出局部极值点, 避免过早收敛. 显然, 可以$p_{\rm g}$为中心, δ为半径, 来划分微粒群, 在$p_{\rm g}$的delta邻域内的微粒进行局部搜索, 邻域外的微粒进行全局搜索. 一般情况下, $p_{\rm g}$的δ邻域外的微粒数量越多, 则种群的多样性越高, 因此可以把邻域外的微粒占整个群体规模的比重作为种群多样性的度量标准, 同时为了克服J.Riget提出的种群多样性的缺点, 避免各维之间的相互影响, 可以考虑在各维上分别定义种群多样性. 本文给出的种群多样性定义如下:

$$\begin{aligned}&k=\begin{cases}0,\ \text{如果}\ |X_{ij}-P_{gj}|<\delta;\\1,\ \text{其他},\end{cases}\\&DS(S_j)=1/|S|\sum_{i=1}^{|S|}k.\end{aligned}\quad (4)$$

其中: $DS(S_j)$表示第j维的种群多样性, $|S|$为种群所含微粒的个数, δ是$p_{\rm g}$邻域的半径. 显然, $0\leqslant DS(S_j)\leqslant1$. 在微粒群进化过程中, 如果$DS$的值太低, 则让微粒群做扩散运动, 即微粒飞离全局最好微粒$p_{\rm g}$, 如果DS的值太高, 则让微粒群做收缩运动, 微粒飞向全局最好微粒$p_{\rm g}$.

MARPSO算法采用的微粒进化方程如下:

$$dir(t+1)=\begin{cases}-1, & \text{如果}dir(t)>0\text{且}DS<d_{\rm low};\\1, & \text{如果}dir(t)<0\text{且}DS>d_{\rm high};\\dir(t), & \text{其他},\end{cases}\quad (5)$$

$$d_i(t+1)=\begin{cases}\dfrac{v(t)}{|v(t)|}, & \text{如果}f(x(t))<f(p_i);\\d_i(t), & \text{其他},\end{cases}\quad (6)$$

$$\begin{aligned}v_i(t+1)=&\omega v_i(t)d_i(t+1)+dir(t+1)(c_1r_1(p_i-\\&x_i(t))+c_2r_2(p_{\rm g}-x_i(t))),\end{aligned}\quad (7)$$

$$x_i(t+1)=x_i(t)+v_i(t+1).\quad (8)$$

其中: $dir(t)$为第t代种群的飞行方向, $d_i(t)$是第t代

第i个微粒的飞行方向. $dir(t) = 1$, 表示种群执行收缩运动, $dir(t) = -1$, 表示种群执行扩散运动, $d_{\rm high}$和$d_{\rm low}$分别表示种群多样性的上下限, 对于$d_{\rm high}$和$d_{\rm low}$的取值将会影响算法的效率, $d_{\rm high}$设置过高, 种群保持较高的多样性, 则微粒群的收敛速度将会降低; $d_{\rm low}$设置过低, 种群将更多地执行“收缩运动”, 保持较高的收敛速度. 因此, $d_{\rm high}$和$d_{\rm low}$的取值不能过高也不能太低, 只能取经验值. 经实验测试, 对于$d_{\rm high} = 0.8$, $d_{\rm low} = 0.2$, 测试结果较好. $d_i(t) = 1$表示微粒的惯性部分, 为正对微粒的搜索有利, $d_i(t) = -1$表示微粒的惯性部分, 为负有利于微粒的搜索, 微粒的飞行方向与种群的飞行方向不一定相同, 这表明: 微粒之间既有共性, 也有差异性, 若共性较大则收敛速度快, 若差异性较大则全局搜索能力较强, 能更好的跳出局部最优, MARPSO在种群多样性指导下, 使得微粒的共性与差异性得到平衡, 从而进行更好的搜索.

MARPSO与ARPSO一样, 都不能保证算法局部收敛, 更不能保证全局收敛, 因此为了进一步提高算法的收敛性能, 本文引入了速度和位置变异策略, 该策略受启发于文献[11], 但有本质不同, 描述如下:

$$v(t+1) = \begin{cases} V_{\max} \times r_1^t, & \text{如果 } |v(t)| < V_{\min} \text{且 } r_2 < 0.5; \\ -V_{\max} \times r_1^t, & \text{如果 } |v(t)| < V_{\min} \text{且 } r_2 \geqslant 0.5, \end{cases} \tag{9}$$

$$x(t+1) = \begin{cases} p_{\rm g} + r_1^t, \text{如果 } |v(t)| < V_{\min} \text{且 } r_2 < 0.5; \\ p_{\rm g} - r_1^t, \text{如果}(|v(t)| < V_{\min} \text{且 } r_2 \geqslant 0.5. \end{cases} \tag{10}$$

其中: $V_{\max}$和$V_{\min}$分别用来表示微粒速度的上下限, r_1和r_2是[0,1]范围内服从均匀分布的随机变量, 当微粒速度小于给定阀值$V_{\min}$时, 对其速度和位置进行变异操作, 即表示微粒以50%左右的概率分布在$p_{\rm g}$两侧, 随着进化代数的无限增大, 微粒从$p_{\rm g}$左右两侧逐渐逼近于全局最好微粒. 公式(9)和(10)中阀值$V_{\min}$的大小将会影响算法的执行效果, 过大的阀值将会导致种群发生混乱, 使算法不能进行有效的局部搜索; 过小阀值将会导致微粒速度需要相对很长的时间下降, 不能有效提高算法的搜索速度. 经实验测试, $V_{\min}$的取值并不会严重影响算法效率, 因此, 取基本经验值即可.

3.2　算法流程(Algorithm flows)

MARPSO的算法流程如下:

Step 1　对所有微粒的位置和速度进行随机化设定;

Step 2　$j = 0$, $D =$ 搜索空间的维数;

Step 3　计算第t代第j维的种群多样性$DS_j(t)$;

Step 4　根据$DS_j(t)$, 设置第t代微粒群整体在该维上的飞行方向$dir_j(t)$;

Step 5　$j = j + 1$;

Step 6　如果j小于D, 则转Step 3;

Step 7　$i = 0$, $N =$种群规模;

Step 8　根据迭代公式(5)∼(10), 对第i个微粒的速度和位置进行进化操作;

Step 9　计算第i个微粒的适应值;

Step 10　对第i个微粒, 将其适应值与其所经历过的历史最好位置p_i的适应值进行比较, 若较好, 则将其作为新的历史最好位置, 同时更新微粒最好飞行方向$d_j(t)$;

Step 11　对第i个微粒, 将其适应值与全局所经历的最好位置$p_{\rm g}$的适应值进行比较, 若较好, 则将其作为新的全局最好位置;

Step 12　$i = i + 1$;

Step 13　如果i小于N则转Step 3;

Step 14　如未达到结束条件(通常为: 足够好的适应值或者达到最大进化代数GMax), 返回Step 2;

Step 15　算法结束.

4　MARPSO算法的收敛性分析(Convergence analysis of MARPSO)

假设 1　问题(P)的可行域Ω为$\mathbb{R}^n$中的有界闭区域, 且目标函数$f(x)$是Ω上的连续函数.

定义 1　对于随机序列$\xi_n(n = 1, 2, \cdots)$和随机变数ξ, 若$P\{\lim\limits_{n\to\infty} \xi_n = \xi\} = 1$, 或者对于$\forall\, \varepsilon > 0$, 有

$$P\{\bigcap_{n=1}^{\infty} \bigcup_{k\geqslant n} |\xi_k - \xi| \geqslant \varepsilon\} = 0,$$

则称随机序列$\{\xi_n\}$以概率为1收敛于随机变数ξ.

定义 2　设$\{S(k)\}$是由算法M产生的序列, 其中: $x^*(k) \in S(k)$是第k代种群中的全局最优个体, 且$f(x^*(k)) \leqslant f(x^*k - 1))$, 如果序列$\{x^*(k)\}$存在一个极限$x^*$, 且$x^*$是问题$(P)$的一个局部极小点, 那么称算法M是局部收敛的.

引理 1　波雷尔–坎特里引理[8]. 设$A_1, A_2, \cdots$是概率空间上的事件序列, 令$p_k = P\{A_k\}$, 若$\sum\limits_{k=1}^{\infty} p_k < \infty$, 则$P\{\bigcap\limits_{n=1}^{\infty} \bigcup\limits_{k\geqslant n} A_k\} = 0$, 若$\sum\limits_{k=1}^{\infty} p_k = \infty$且各$A_k$相互独, 则$P\{\bigcap\limits_{n=1}^{\infty} \bigcup\limits_{k\geqslant n} A_k\} = 1$.

引理 2　对于问题(P), 若其存在局部极小值x^*, 则MARPSO算法将收敛于x^*, 即$\lim\limits_{t\to\infty} p_{\rm g}(t) = x^*$.

证 由文献[9,10]知: $\lim\limits_{t\to\infty} x(t) = \lim\limits_{t\to\infty} p_{\rm g}(t)$, 对于标准PSO算法, $t\to\infty$时, 若$v(t) > 0$, 则$x(t)$从右侧逼近$p_{\rm g}(t)$, 反之, 若$v(t) < 0$, 则$x(t)$从左侧逼近$p_{\rm g}(t)$. 显然, 如果此时所有微粒的速度方向相同(即都大于零或者都小于零), 那么所有微粒将从同一侧逼近$p_{\rm g}(t)$, 因此, 不能保证另外一侧不存在更好的位置, 也就是说, $p_{\rm g}(t)$不一定是局部最优. 而对于MARPSO算法), 式(9和式(10)的引入, 使得$t\to\infty$时, $x(t)$是从左右两侧等概率的逼近$p_{\rm g}(t)$, 因此$t\to\infty$时, 有:

$$|x(t+1)-p_{\rm g}(t)| < \varepsilon_1 且 |x(t)-p_{\rm g}(t)| < \varepsilon_2,$$
$$|x(t+1)-p_{\rm g}(t)| + |x(t)-p_{\rm g}(t)| < \varepsilon_1+\varepsilon_2 = \varepsilon,$$
$$|x(t+1)-x(t)| < \varepsilon.$$

其中$\varepsilon_1, \varepsilon_2, \varepsilon$为任意小的正常数. 由此可得

结论 1 $t\to\infty$时, $x(t)$是连续随机变量, 它是连续的逼近$p_{\rm g}(t)$的, 在它逼近$p_{\rm g}(t)$的路径上, 若存在$x'(t)$, 使得$f(x'(t)) \leqslant f(p_{\rm g}(t))$, 则$p_{\rm g}(t)$将被更新, 使得$p_{\rm g}(t)$始终保持最优性.

假设$t\to\infty$时, $p_{\rm g}(t)$不是局部极小点, 那么根据局部极小点的定义, 可得

结论 2 $t\to\infty$时, 在$p_{\rm g}(t)$的任意小的ε邻域内, 至少存在一个点$x'(t)$, 使得$f(x'(t)) < f(p_{\rm g}(t))$.

显然, 结论2和结论1是相矛盾的, 因此假设不成立, 从而得出结论: $p_{\rm g}(t)$就是问题(P)的一个极小值点, 即$\lim\limits_{t\to\infty} p_{\rm g}(t) = x^*$.

引理 3 如果$\Delta x = x(t)-x(t-1), r_1, r_2 \sim N(0,1)$, 则, $\Delta x \sim N(\mu_1,\sigma_1), \Delta f(x) \sim N(\mu_2,\sigma_2)$.

证

$$\Delta x = wV + c_1r_1(p_i - x) + c_2r_2(p_{\rm g} - x).$$

设$wV = \varphi_1, c_1(p_i - x_i) = \varphi_2, c_2(p_{\rm g}-x_i) = \varphi_3$, 则有: $\Delta x = \varphi_1 + \varphi_2r_1 + \varphi_3r_2$, 对于MARPSO算法, 在微粒进化过程中, 由于式(9)和式(10)的引入, 使得$\varphi_1, \varphi_2, \varphi_3$不可能同时为0, 又由于$r_1, r_2$是服从正态分布的随机变量, 且正态分布的线性组合也服从正态分布, 由文献[11]知: $\Delta x \sim N(\mu_1,\sigma_1)$, 另外由于$f(x)$是连续函数, 所以: $\Delta f(x) \sim N(\mu_2,\sigma_2)$.

定理 1 MARPSO算法是局部收敛的.

证 由文献[9,10]知: $\lim\limits_{t\to\infty} x(t) = \lim\limits_{t\to\infty} p_{\rm g}(t)$, 又由引理2知: $\lim\limits_{x\to\infty} p_{\rm g}(t) = x^*$, 因此MARPSO算法收敛于问题$(P)$的局部极小点$x^*$, 且满足: $f(p_{\rm g}(t)) \leqslant f(p_{\rm g}(t-1)) \leqslant \cdots \leqslant f(p_{\rm g}(0))$, 因此有定义2知: MARPSO算法是局部收敛的. 证毕.

定理 2 如果问题(P)满足假设1, 那么MARPSO算法产生的全局最好微粒序列$\{x^*(k)\}$, 以概率为1收敛于问题(P)的全局最优解.

证 对于$\forall\,\varepsilon > 0$, 令$p_k = P\{|f(x^*(k)) - f^*| \geqslant \varepsilon\}$, 其中$f^*$为全局最优解, 则显然有

$$\bar{p}_k = \prod_{T=1}^{k} P\{|f(x^*(T)) - f^*| \geqslant \varepsilon\}.$$

又因为: 在极小化搜索的过程中, 总是有$f(x^*(T)) \geqslant f^*$. 因而有

$$\bar{p}_k = \prod_{T=1}^{k} P\{|f(x^*(T)) - f^*| \geqslant \varepsilon\} =$$
$$\prod_{T=1}^{k} P\{f(x^*(T)) - f^* \geqslant \varepsilon\} =$$
$$\prod_{T=1}^{k} P\{\Delta f(x(T)) \geqslant f^* + \varepsilon - f(x^*(T-1))\}.$$

由引理3可知, $\Delta f(x) \sim N(\mu_2,\sigma_2)$, 因此

$$\bar{p}_k = \prod_{T=1}^{k} \int_{f^*+\varepsilon-f(x^*(T-1))}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma_2} {\rm e}^{\frac{-y^2}{2\sigma_2^2}}\,{\rm d}y.$$

令$c = \max\limits_{1\leqslant T\leqslant k} \int_{f^*+\varepsilon-f(x^*(T-1))}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma_2} {\rm e}^{\frac{-y^2}{2\sigma_2^2}}\,{\rm d}y$, 显然, $0 < c < 1$, 所以

$$\sum_{k=1}^{\infty} \bar{p}_k \leqslant \sum_{k=1}^{\infty} c^k = \frac{c}{1-c} < \infty.$$

由引理1可知, $P\{\bigcap\limits_{n=1}^{\infty} \bigcup\limits_{k\geqslant n} |f(x^*(k) - f^*| \geqslant \varepsilon\} = 0$.

由定义1可知, $f(x^*(k))$以概率为1收敛于f^*, 即$x^*(k)$以概率为1收敛于问题(P)的全局最优解.

5 仿真测试(Simulation tests)

为了验证算法有效性, 利用Griewank函数、AckleyF1函数、Rosenbrock函数和Rastrigin函数对本文实现的MARPSO算法进行了测试, 函数表达式如下:

F1 Griewank 函数

$$f_1(x) = \sum_{i=1}^{n} x_i^n/4000 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1.$$

其中$-600 \leqslant x_i \leqslant 600$, 在$x_i = 0$时达到极小值0.

F2 Rastrigrin 函数

$$f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10).$$

其中$-5.12 \leqslant x_i \leqslant 5.12$, 在$x_i = 0$时达到极小值0.

F3 AckelyF1函数

$$f_2(x) = e + 20 - 20\exp(-0.2\sqrt{1/n\sum_{i=1}^{n} x_i^2}\ ) -$$
$$\exp(1/n\sum_{i=1}^{n}\cos(2\pi x_i)).$$

其中$-30 \leqslant x_i \leqslant 30$, 在$x_i = 0$时达到极小值0.

F4　Rosenbrock 函数

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

其中$-30\leqslant x_i\leqslant 30$, 在$x_i=1$时达到极小值0.

为了与文献[7]的实验数据相比较, 故本文采用与文献[7]相同的参数, 即: 对于每个测试函数, 算法均运行50次, 每个测试函数分别考虑维数为20, 50, 100的情形, 且相应的最大进化代数分别为40000, 100000, 200000, 误差为1.0×10^{-10}, $C_1=2.0$, $C_2=2.0$, 惯性权重W从0.9线性减小到0.4. 此外, 对于MARPSO算法, δ的取值影响p_{g}邻域的大小, 进而影响种群多样性的计算, δ取值过高或过低, 都将降低计算出来的种群多样性的可信度, 应当根据收敛精度适当调整, 本文取$\delta=1.0\times10^{-10}$. 测试结果如表1～表4所示.

从表4可以看出, 对于Rosenbrock函数, MARPSO的收敛精度略优于BPSO和ARPSO, 由表1, 表2和表3可知, 对于AckleyF1, Griewank和Rastrigin函数, MARPSO的收敛精度都远远超过了BPSO算法和ARPSO算法, 尤其对于高维情形, 由此可见, 本文定义的种群多样性函数对于微粒群的进化过程的指导效果明显优于文献[7]定义的种群多样性函数.

表 1　Griewank函数测试结果

Table 1　Test results of Griewank

维数	算法	平均适应值
20	BPSO	1.74E-2
	ARPSO	2.50E-2
	MARPSO	4.03E-3 (86%概率找到0)
50	BPSO	1.35E-2
	ARPSO	1.97E-4
	MARPSO	1.74E-2 (97%概率找到0)
100	BPSO	1.25E-2
	ARPSO	9.84E-2
	MARPSO	0 (100%概率找到0)

表 2　Rastrigin函数测试结果

Table 2　Test results of Rastrigin

维数	算法	平均适应值
20	BPSO	9.71
	ARPSO	0
	MARPSO	0
50	BPSO	47.14
	ARPSO	0.02
	MARPSO	0
100	BPSO	96.59
	ARPSO	0.44
	MARPSO	0

表 3　AckleyF1函数测试结果

Table 3　Test results of AckleyF1

维数	算法	平均适应值
20	BPSO	0.018
	ARPSO	0.33E-7
	MARPSO	0
50	BPSO	0.668
	ARPSO	0.027
	MARPSO	2.39E-10
100	BPSO	0.830
	ARPSO	0.218
	MARPSO	3.99E-9

表 4　Rosenbrock函数测试结果

Table 4　Test results of Rosenbrock

维数	算法	平均适应值
20	BPSO	11.16
	ARPSO	2.34
	MARPSO	0.13
50	BPSO	30.08
	ARPSO	10.43
	MARPSO	1.28
100	BPSO	122.14
	ARPSO	103.46
	MARPSO	0 16.93

为了比较3种算法的收敛性能, 本文分别运行算法50次, 取其平均值, 并用MATLAB作出函数进化曲线, 图1～4分别给出了4个测试函数(50维)在种群规模为20, 最大迭代次数为100,000 的迭代曲线(注: 为了方便进化曲线的显示和观察, 本文对函数的适应值取以10为底的对数). 从图中可以看出: 与BPSO相比, 在进化的后期, MARPSO算法的迭代曲线下降很快, 迅速收敛到全局最优解, 与ARPSO算法相比, MARPSO的收敛适应值较好, 但收敛速度比ARPSO低, 原因是MARPSO的种群多样性度量函数能更好的指导算法进行扩散, 从而较好地提高了算法的收敛性能, 但同时降低了算法的收敛速度.

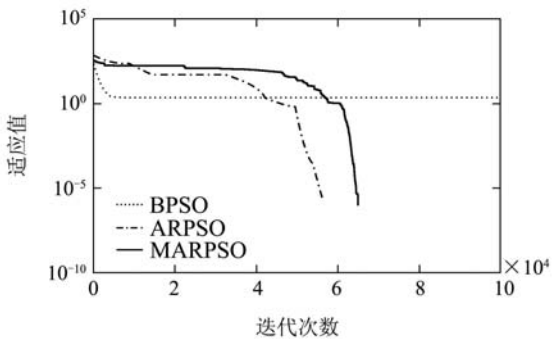

图 1　Griewank函数进化曲线

Fig. 1　Evolution curve of Griewank

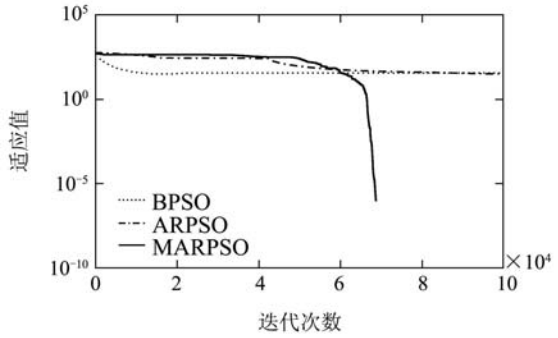

图 2 Rastrigin函数进化曲线

Fig. 2 Evolution curve of Rastrigin

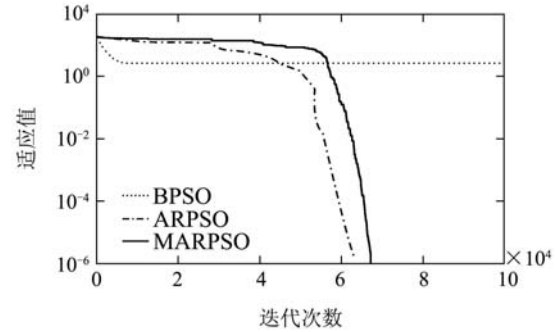

图 3 AckleyF1函数进化曲线

Fig. 3 Evolution curve of AckleyF1

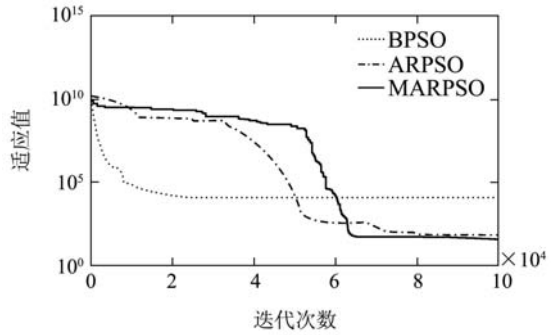

图 4 Rosenbrock函数进化曲线

Fig. 4 Evolution curve of Rosenbrock

6 结论(Conclusion)

众所周知, 维持较高的种群多样性和较快的收敛速度是一个矛盾体, ARPSO算法是一种由种群多样性驱动的微粒群算法, 通过种群的“扩散运动”和“收缩运动”, 来达到种群多样性和收敛速度的平衡, 既保证了种群的较高多样性, 避免陷入局部极值点, 又维持了较高的收敛速度, 是一种较好的优化策略, 但是其种群多样性函数的运算相对复杂. 本文提出了一个运算简单的种群多样性函数, 实验证明它能更好的指导种群的“扩散运动”和“收缩运动”, 同时本文提出了微粒最好飞行方向的概念, 并引入了速度和位置变异策略, 从而大大提高了算法的收敛性能. 但是, 在某些极端情况下, 这两种种群多样性度量方法都不能真正的表示微粒群的离散程度(即种群多样性), 因此, 有必要探索更为合理、高效的种群多样性度量方法, 更好的发挥种群多样性对于微粒群进化过程的驱动作用.

参考文献(References):

[1] KENNEDY J, EBERHART R C. Particle swarm optimization[C] //*Proceedings of IEEE International Conference on Neural Networks*. Piscateway, NJ: IEEE Service Center, 1995, IV: 1942 – 1948.

[2] PARSOPOULOS K E, VRAHATIS M N. On the computation of all global minimizers through particle swarm optimization[J]. *IEEE Transactions on Evolutionary Computation*, 2004, 8(3): 211 – 224.

[3] 曾建潮, 介靖, 崔志华. 微粒群算法[M]. 北京: 科学出版社, 2004.
(ZENG Jianchao, JIE Jing, CUI Zhihua. *Particle Swarm Optimization*[M]. Beijing: Science Press, 2004.)

[4] KENNEDY J. Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance[C] //*1999 Conference on Evolutionary Computation*. ashington, DC: IEEE, 1999.

[5] VAN DEN F B. *An analysis of particle swarm optimizers*[D]. South Africa: Department of Computer Science, University of Pretoria, 2001.

[6] XIE X F, ZHANG W J, YANG Z L. A dissipative particle swarm optimization[C] //*Congress on Evolutionary Computation(CEC)*. Hawail, USA: IEEE, 2002.

[7] RIGET J, VESTTERSTROM J S. *A diversity-guided particle swarm optimizer-the ARPSO*[R]. EVAlife: Department of Computer Science, University of Aarhus, Denmark, 2002.

[8] 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. Evolution of the global convergence strategy[J]. *Computational Mathematics*, 2001, 23(1): 105 – 110.)

[9] CLERC M, KENNEDY J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space[J]. *IEEE Transactions on Evolutionary Computation*, 2002, 6(1): 58 – 73.

[10] CRISTIAN T I. The particle swarm optimization algorithm: convergence analysis and parameter selection[J]. *Information Processing Letters*, 2003, 85(6): 317 – 325.

[11] 赫然, 王永吉, 王青, 等. 一种改进的自适应逃逸微粒群算法及其实验分析[J]. 软件学报, 2005, 16(12): 2036 – 2044.
(HE Ran, WANG Yongji, WANG Qing, et al. An improved particle swarm optimization based on self-adaptive escape velocity[J]. *Journal of Software*, 2005, 16(12): 2036 – 2044.)

作者简介:

陈保娣 (1981—), 女, 硕士研究生, 从事智能计算的研究, E-mail: cbdngc@sina.com.cn;

曾建潮 (1963—), 男, 教授, 博士生导师, 主要研究方向为系统建模与仿真、智能计算等, E-mail: zengjianchao@263.com.