Journal of Computer Applications
计算机应用,2012,32(10):2732-2735
ISSN 1001-9081
CODEN JYIIDU
2012-10-01
http://www.joca.cn

文章编号:1001-9081(2012)10-2732-04
doi:10.3724/SP.J.1087.2012.02732

引力搜索算法中粒子记忆性改进的研究

李春龙,戴　娟,潘　丰*
(江南大学 轻工过程先进控制教育部重点实验室,江苏 无锡 214122)
(*通信作者电子邮箱 pan_feng_63@163.com)

摘　要:针对引力搜索算法(GSA)对一些复杂问题的搜索精度不高的问题,特别是高维函数优化性能不佳、优化过程容易出现早熟的现象,因此考虑将粒子群优化(PSO)算法中关于局部最优解和全局最优解的概念引入引力搜索算法中,对引力搜索算法中粒子的记忆性进行改进,这样使得粒子的进化不仅受空间中其他粒子的影响,还受到自身记忆的约束,以此来提高算法的搜索能力。通过对选用的10个基准函数测试,证明了该方法的有效性。

关键词:引力搜索算法;粒子群优化算法;记忆性;数值函数优化;群智能
中图分类号:TP301.6　　**文献标志码**:A

Analysis on improvement of particle memory in gravitational search algorithm

LI Chun-long, DAI Juan, PAN Feng*
(*Key Laboratory of Advanced Process Control for Light Industry Ministry of Education, Jiangnan University, Wuxi Jiangsu* 214122, *China*)

Abstract: As the gravitational search algorithm plays bad performance in search accuracy of the complex issues, especially the poor search quality of standard Gravitational Search Algorithm (GSA) in the high dimensional function optimization. It is easy to get into premature convergence in the optimization process. Therefore, the idea of the particle swarm optimization algorithm was introduced to gravitational search algorithm, which was used to improve the memory of particles. The particle evolution is not only influenced by other particles in the space, but also by its own memory constraint, which is used to improve the ability of exploitation. The test of the 10 benchmark functions confirms the validity of the method.

Key words: Gravitational Search Algorithm (GSA); Particle Swarm Optimization (PSO) algorithm; memory; numerical function optimization; swarm intelligence

Rashedi 等[1-3]在2009年底提出了引力搜索优化算法(Gravitational Search Algorithm,GSA),与粒子群优化(Particle Swarm Optimization,PSO)算法相似,是一种元启发式算法,该算法的基本思想是基于牛顿万有引力定律[4]:"在宇宙间,每一个粒子由于万有引力的作用而彼此相互吸引,引力的大小与粒子的质量成正比,与它们之间的距离成反比"。它通过群体中各粒子之间的万有引力相互作用产生的群体智能指导优化搜索,但是目前针对引力搜索算法的研究还是比较少。文献[1]研究证明,GSA的搜索能力明显优于PSO、遗传算法、中心引力算法等一些智能优化算法,但是该算法在优化过程中仍存在早熟收敛现象,易陷入局部最优。

为了解决以上的不足,本文对引力搜索算法中粒子的记忆性加以改进,主要采用了粒子群思想中的局部最优解和全局最优解的概念,并对该改进方法作简要的分析,最后通过测试基准函数验证改进算法的有效性,并分析测试的结果。

1　引力搜索算法

在GSA中,粒子的质量对它们的行为表现有着非常重要的作用。粒子之间都是相互吸引的,相互之间的作用力引起粒子都朝着质量大的粒子的方向移动,每个粒子有四个特征:位置、惯性权重、施力粒子、受力粒子。而粒子的位置就是问题的解[1-2]。

和经典的粒子群算法一样,重力引力搜索算法中粒子的初始位置和初始速度都是随机生成的。根据式(1)和式(2),可计算出每个粒子的惯性质量$M_i(t)$[1]:

$$m_i(t) = \frac{fitness_i(t) - worst(t)}{best(t) - worst(t)} \tag{1}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{2}$$

其中:$i = 1,2,\cdots,N$,N为粒子数目;$m_i(t)$为计算粒子质量的中间变量;$fitness_i(t)$是粒子i在t时刻的适应值;$worst(t)$和$best(t)$分别是指在t时刻整个粒子群的最坏的适应值和最好的适应值[1,5]。

在搜索目标函数最小值问题时,最坏和最好的适应值分别为:

$$best(t) = \min_{j\in\{1,\cdots,N\}} fitness_j(t)$$

$$worst(t) = \max_{j\in\{1,\cdots,N\}} fitness_j(t)$$

在搜索目标函数最大值问题时,最坏和最好的适应值分别为:

$$best(t) = \max_{j\in\{1,\cdots,N\}} fitness_j(t)$$

$$worst(t) = \min_{j\in\{1,\cdots,N\}} fitness_j(t)$$

然后,使用变换后的万有引力公式,可以计算各个粒子在每一维空间上相互之间的引力,在d维空间上粒子i和粒子j之间的引力[1-2]:

$$F_{ij}^d(t) = G(t)\frac{M_i(t)\times M_j(t)}{R_{ij}(t)+\varepsilon}(x_j^d(t) - x_i^d(t)) \tag{3}$$

收稿日期:2012-04-17;**修回日期**:2012-05-31。
基金项目:国家863计划项目(2009AA05Z203);江苏高校优势学科建设工程资助项目(PAPD)。
作者简介:李春龙(1987-),男,江苏盐城人,硕士研究生,主要研究方向:群智能优化算法、图像处理;　戴娟(1989-),女,江苏盐城人,硕士研究生,主要研究方向:工业过程建模;　潘丰(1963-),男,江苏苏州人,教授,博士生导师,主要研究方向:工业过程建模与优化控制。

其中:$x_i^d(t)$ 是指粒子 i 在 d 维空间中的位置,$x_j^d(t)$ 是指粒子 j 在 d 维空间中的位置,ε 是指非常小的常数,$R_{ij}(t)$ 是粒子 i 和粒子 j 之间的欧氏距离:

$$R_{ij}(t) = \| x_i(t), x_j(t) \|_2$$

$G(t)$ 是引力系数:

$$G(t) = G_0 e^{-\alpha \frac{t}{T}} \tag{4}$$

其中:G_0 和 α 为常数,T 为最大迭代次数。

然后根据牛顿运动定律计算出每一维空间上在 t 时刻粒子 i 的加速度,在引力搜索算法中,为了增加算法的随机特性,在作用力 $F_{ij}^d(t)$ 前加 $rand_j$ 随机函数,假设在第 d 维空间上作用在第 i 个粒子上总的作用力是来自其他所有的粒子作用力的总和[1,5],粒子 i 的惯性质量为 $M_{ii}(t)$,那么粒子的加速度为:

$$a_i^d(t) = \frac{\sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t)}{M_{ii}(t)} \tag{5}$$

最后,粒子 i 在下一时刻的速度和位置的进化公式为[1,2,9]:

$$V_i^d(t+1) = rand_i \times V_i^d(t) + a_i^d(t) \tag{6}$$

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \tag{7}$$

2　粒子记忆性改进

2.1　GSA 和 PSO 粒子记忆性分析

对 GSA 搜索算法中的速度和位置进化式(6)、(7)分析可知,算法中仅仅只有当前的位置信息在迭代更新过程中起作用,可以得知 GSA 是一种缺乏记忆性的算法。由式(5)~(7)分析可知,当粒子 i 运动到最优解或者比较接近最优解前,粒子 i 的速度是不断增加的(根据万有引力公式,引力的大小是与距离成反比关系),当到达最优解或接近最优解时,粒子的速度可能会很大(存在随机性),根据运动学规律可知,这种情况会导致粒子在最优解的附近反复来回震荡,导致整个算法搜索精度不高(如图 1 所示),其中★代表全局最优解。

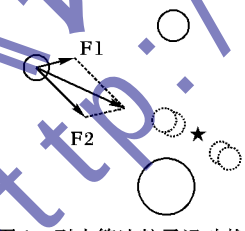

图 1　引力算法粒子运动状态

在一个 D 维的目标搜索空间中,有 m 个粒子组成一个群体,其中第 i 个粒子的位置表示为 $X_i = (x_i^1, x_i^2, \cdots, x_i^D)$,每个粒子的位置就是一个潜在解,将 X_i 代入目标函数就可以计算出其适应值,而粒子个体经历的最好值记为 $pbest_i = (p_i^1, p_i^2, \cdots, p_i^D)$,整个群体所有粒子经历过的最好位置记为 $gbest = (gbest^1, gbest^2, \cdots, gbest^D)$,而粒子的记忆性就体现在了 $pbest$ 和 $gbest$ 这两个量中。

粒子群算法采用下列公式对粒子所在位置不断更新[5-6,11]:

$$V_i^d(t+1) = w \cdot V_i^d(t) + c_1 \cdot rand_1(t) \cdot (pbest_i^d(t) - X_i^d(t)) + c_2 \cdot rand_2(t) \cdot (gbest_i^d(t) - X_i^d(t)) \tag{8}$$

$$X_i^d(t+1) = V_i^d(t+1) + X_i^d(t) \tag{9}$$

其中:$rand_1$,$rand_2$ 是(0,1)间均匀分布的相互独立的随机数序列;c_1 调节粒子飞向自身最好位置的步长;c_2 调节粒子飞向全局最好位置的步长[7-8,16]。

由此,考虑将粒子群优化中对粒子运动历史最优解进行记忆保存的思想引入到引力搜索算法中,对引力搜索算法中粒子的记忆能力进行改进,使其像 PSO 算法一样,能够记忆自身运动过程中的最优解和全局最优解[13-14]。以此调节粒子在演化过程的速度更新方式,提高算法的搜索能力。其表达式如下:

$$V_i^d(t+1) = rand_1 \cdot V_i^d(t) + c_1 \cdot rand_2(t) \cdot (pbest_i^d(t) - X_i^d(t)) + c_2 \cdot rand_3(t) \cdot (gbest_i^d(t) - X_i^d(t)) + a_i^d(t) \tag{10}$$

位置更新公式保持不变。其中:$rand_1$,$rand_2$,$rand_3$ 是(0,1)内均匀分布的相互独立的随机数序列;c_1 和 c_2 定义等同于式(7)中的定义。

2.2　算法改进分析

根据式(9)可将速度的进化方程改写为:

$$V_i^d(t+1) = T_1 + T_2 + T_3 \tag{11}$$

其中:

$$T_1 = c_1 \cdot rand_2(t) \cdot (pbest_i^d(t) - X_i^d(t))$$

$$T_2 = c_2 \cdot rand_3(t) \cdot (gbest_i^d(t) - X_i^d(t))$$

$$T_3 = rand_1 \cdot V_i^d(t) + a_i^d(t)$$

其中 T_1 和 T_2 为研究部分,对这两部分进行分析[10]。

为了进一步研究 T_1 和 T_2 在算法更新中所起的作用[10,15],分别研究以下的进化方程:

$$\begin{cases} V_i^d(t+1) = c_1 \cdot rand_2(t) \cdot (pbest_i^d(t) - X_i^d(t)) \\ X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \end{cases} \tag{12}$$

$$\begin{cases} V_i^d(t+1) = c_1 \cdot rand_2(t) \cdot (gbest_i^d(t) - X_i^d(t)) \\ X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \end{cases} \tag{13}$$

此时,设速度和位置信息的进化更新公式为式(11),令 $\lambda_1 = c_1 \cdot rand_2(t)$,$r(t) = 1(t)$ 为单位阶跃函数。

经过化简得到:

$$X_i^d(t+1) = X_i^d(t) + \lambda_1 \cdot pbest_i^d(t) - \lambda_1 \cdot X_i^d(t)$$

根据积分差分形式:

$$\frac{dX_i^d(t)}{dt} = X_i^d(t+1) - X_i^d(t)$$

从而上式可化为:

$$\frac{dX_i^d(t)}{dt} = \lambda_1 \cdot pbest_i^d(t) - \lambda_1 \cdot X_i^d(t) \cdot r(t)$$

并利用拉普拉斯变换和反变换,可得到:

$$X_i^d(t) = pbest_i^d(t)(1 - e^{-\lambda_1 t}) \tag{14}$$

对于 T_2 部分,令 $\lambda_2 = c_2 \cdot rand_3(t)$,$r(t) = 1(t)$ 为单位阶跃函数,用同样的方法可以得到:

$$X_i^d(t) = gbest_i^d(t)(1 - e^{-\lambda_2 t}) \tag{15}$$

由式(14)分析可知,该部分的运动过程是,逐渐逼近于局部最优解;式(15)的运动过程是逐渐逼近于全局最优解的。这两部分在动力学上可以理解为是物体在某一平衡点附近震荡运动的阻力,使得运动最终能接近于平衡点(即最优解)。

2.3　算法流程

改进的 GSA 算法步骤如下:

步骤 1　明确整个问题的搜索空间;

步骤 2　初始化种群数目 n,最大迭代次数 max_it,并随机初始化粒子的位置;

步骤 3　对每个粒子,根据目标函数计算它的适应值;

步骤 4　根据式(1)和(2)更新每个粒子的惯性质量 $M_i(t)$,并根据式(4)更新引力系数函数 $G(t)$;

步骤 5　根据式(3)计算每个粒子不同方向上的力的总和;

步骤 6　根据式(5)和式(10)分别计算粒子加速度和速度;

步骤 7　根据式(7)更新每个粒子的位置;

步骤 8　循环迭代,直至达到迭代次数或满足要求精度为止。

3　仿真实验及分析

为了测试改进算法的性能,选用了 10 个测试函

数[1-2,17]，比较标准的 GSA（Standard Gravitational Search，SGSA）和记忆改进的 GSA（Memory Gravitational Search Algorithm，MGSA）的搜索能力表现，其中 n 表示测试函数的维数，$S \in \mathbf{R}^n$ 为搜索空间（如表 1 所示）。

前 4 个测试函数（$F_1 \sim F_4$）是单峰测试函数；$F_5 \sim F_7$ 是多峰测试函数；$F_8 \sim F_{10}$ 是低维多峰测试函数。表 2 为函数 F_{10} 中所使用的参数。所使用的实验仿真平台为 Windows XP，Matlab2011b 版本。

表 1 测试函数

测试函数	S	最优解
$F_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100,100]^n$	0
$F_2(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100,100]^n$	0
$F_3(x) = \max \lvert x_i \rvert, 1 \leqslant i \leqslant n$	$[-100,100]^n$	0
$F_4(x) = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2]$	$[-30,30]^n$	0
$F_5(x) = \sum_{i=1}^{n} [-x_i * \sin(\sqrt{\lvert x_i \rvert})]$	$[-500,500]^n$	$-418.9829 \times n$
$F_6(x) = \frac{1}{4000} \sum_{i=1}^{n} (x_i)^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	$[-600,600]^n$	0
$F_7(x) = 0.1\{\sin(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2[1 + \sin(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i,5,100,4)$	$[-50,50]^n$	0
$F_8(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	$[-65.53,65.53]^2$	1
$F_9(x) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$[-5,10] \times [0,15]$	0.398
$F_{10}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	$[-5,5]^4$	0.00030

表 2 测试函数 10 中参数

i	a_i	b_i^{-1}
1	0.1957	0.25
2	0.1947	0.50
3	0.1735	1.00
4	0.1600	2.00
5	0.0844	4.00
6	0.0627	6.00
7	0.0456	8.00
8	0.0342	10.00
9	0.0342	12.00
10	0.0235	14.00
11	0.0246	16.00

对每个基准测试函数运行 15 次，并统计运行结果的平均值、中值和方差。SGSA 和 MGSA 中粒子种群数目设为 50，前 7 个测试函数的维数取 30，最大迭代次数为 1 000；后 3 个测试函数的维数如表 1 中数据所示，最大迭代次数为 500，引力系数 G 如式(4) 所示，其中 G_0 设为 100，α 设为 20，其中 T 即为最大迭代次数，参数 $c_1 = c_2 = 0.5$，Average best-so-far 是指运行结果的平均值，Median best-so-far 是指运行结果的中值，Std best-so-far 指运行结果的方差。表 3 为标准测试函数优化结果比较，图 2 ~ 5 为测试函数的优化过程曲线图（限于篇幅，未全部给出）。

单峰高维函数 将表 3 中函数的优化结果与表 1 中的函数最优解进行比较，在函数 $F_1 \sim F_4$ 优化结果中，改进的 GSA（MGSA）的优化结果优于标准 GSA（SGSA），F_2 和 F_3 的优化效果比较明显，而改进的算法在 F_1 和 F_4 中的优化结果与标准算法优化效果相比，只是略有提高，大部分函数的收敛率明显提高（图 2 为 F_2 的优化过程曲线图）。

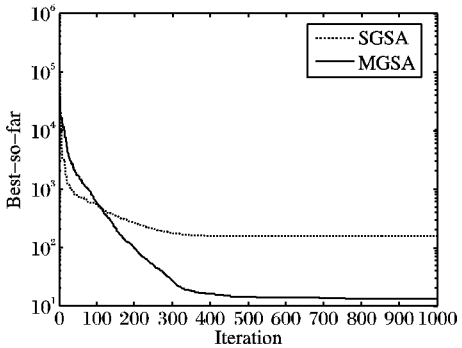

图 2 单峰高维函数 F_2 优化过程曲线

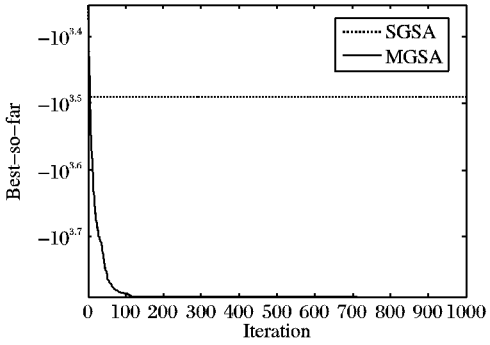

图 3 多峰高维函数 F_5 优化过程曲线

表 3　标准测试函数优化结果比较

函数	统计数据类型	MGSA	SGSA
F_1	Average best-so-far	6.49E-16	2.55E-15
	Median best-so-far	6.44E-16	9.68E-16
	Std best-so-far	2.10E-16	2.98E-15
F_2	Average best-so-far	15.4351	347.2984
	Median best-so-far	12.4208	338.2847
	Std best-so-far	12.3858	112.2059
F_3	Average best-so-far	1.03E-08	1.20E-01
	Median best-so-far	1.00E-08	2.89E-01
	Std best-so-far	1.41E-09	0.4584
F_4	Average best-so-far	25.8890	32.2866
	Median best-so-far	25.5805	27.6022
	Std best-so-far	1.0442	18.1056
F_5	Average best-so-far	-6.97E+03	-2.71E+03
	Median best-so-far	-6.94E+03	-2.82E+03
	Std best-so-far	743.6123	303.8504
F_6	Average best-so-far	1.4475	4.3318
	Median best-so-far	1.306	3.9004
	Std best-so-far	0.3676	1.3061
F_7	Average best-so-far	6.49E-17	0.1302
	Median best-so-far	6.36E-17	1.70E-16
	Std best-so-far	1.17E-17	0.3518
F_8 $n=2$	Average best-so-far	1.5979	4.0464
	Median best-so-far	1.5122	3.0258
	Std best-so-far	0.5006	1.9964
F_9 $n=2$	Average best-so-far	0.3979	0.3979
	Median best-so-far	0.3979	0.3979
	Std best-so-far	0	0
F_{10} $n=3$	Average best-so-far	7.02E-04	0.0074
	Median best-so-far	7.11E-04	0.0074
	Std best-so-far	7.02E-04	0.0074

多峰高维函数　函数 F_5 ~ F_7 的优化结果的比较中,MGSA 的优化性能明显优于 SGSA(图 3 和 4 分别为 F_5 和 F_6 的优化过程曲线图),表明该改进方法在部分高维优化函数中有提高搜索精度的作用。

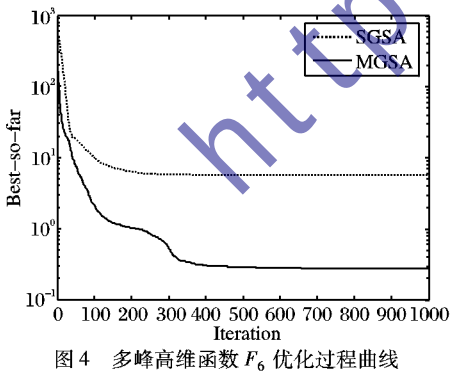

图 4　多峰高维函数 F_6 优化过程曲线

多峰低维函数　MGSA 在 F_8 和 F_{10} 中的优化结果优于 SGSA,在函数 9 中,两种算法的优化结果相同,都达到了函数的最优解,且优化结果非常稳定(图 5 为 F_{10} 的优化过程曲线图)。

4　结语

引力搜索算法是一种全局优化算法,但在文献[1]中看出一些基准测试函数中的优化表现仍然不理想,因此本文将粒子群算法中关于粒子记忆性的思想引入到引力搜索算法中,用以提高该算法中粒子的个体记忆性,对粒子的速度更新方式进行修正,这样粒子的速度信息更新不仅得取决于整个系统中其他粒子的共同作用,还受到它自身记忆的影响,通过测试证明,该方法提高算法的搜索能力。

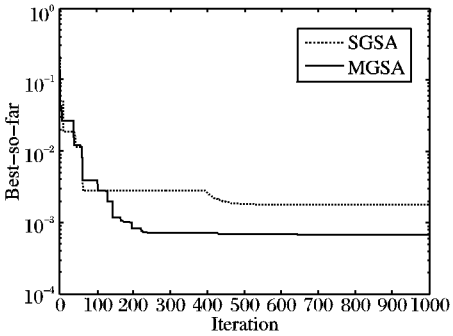

图 5　多峰低维函数 F_{10} 优化过程曲线

参考文献:

[1] RASHEDI E, NEZAMABADI-POUR H, SARYAZDI S. GSA: A gravitational search algorithm [J]. Information Science, 2009, 179(13): 2232 – 2248.

[2] RASHEDI E, NEZAMABADI-POUR H, SARYAZDI S. BGSA: Binary gravitational search algorithm [J]. Natural Computing, 2010, 9(3): 727 – 745.

[3] SARAFRAZI S, NEZAMABADI-POUR H, SARYAZDI S. Disruption: A new operator in gravitational search algorithm [J]. Scientia Iranica, 2011, 18(3): 539 – 548.

[4] SCHUTZ B. Gravity from the ground up [M]. London: Cambridge University Press, 2003.

[5] HASSANZADEH H R, ROUHANI M. A multi-object gravitational search algorithm [C]// The Second International Conference on Computational Intelligence. Washington, DC: IEEE Computer Society, 2010: 7 – 12.

[6] KANOVIC Z, RAPAIC M R, JELICIC Z D. Generalized particle swarm optimization algorithm – Theoretical and empirical analysis with application in fault detection [J]. Applied Mathematics and Computation, 2011, 217(24): 10175 – 10186.

[7] 居上游. 动态扩散粒子群算法及其应用[J]. 计算机工程与应用, 2011, 47(36): 61 – 67.

[8] KROHLING R A. Gaussian swarm: A novel particle swarm optimization algorithm[C]// The International Conference on Cybernetics and Intelligent Systems. Washington, DC: IEEE Computer Society, 2004: 372 – 376.

[9] NOBAHARI H, NIKUSOKHAN M, SIARRY P. Non-dominated sorting gravitational search algorithm[C]// ICSI: International Conference on Swarm Intelligence. Washington, DC: IEEE Computer Society, 2011: 1 – 10.

[10] 崔志华, 曾建潮. 基于微分模型的微粒群算法分析与改进[J]. 小型微型计算机系统, 2006, 05: 849 – 853.

[11] KENNEDY J, EBERHART R C. Particle swarm optimization[C]// Proceedings of IEEE International Conference on Neural Networks. Washington, DC: IEEE Computer Society, 1995: 1942 – 1948.

[13] 王丽芳, 曾建潮. 基于微粒群算法与模拟退火算法的协同进化方法[J]. 自动化学报, 2006, 32(4): 630 – 635.

[14] 俞欢军, 张丽平, 陈德钊, 等. 基于反馈策略的自适应粒子群优化算法[J]. 浙江大学学报: 工学版, 2005, 39(9): 12 – 17.

[15] RIGET J, VESTERSTROEM J. A diversity-guided particle swarm optimizer – the ARPSO[R]. Aarhus, Denmark: University of Aarhus, Department of Computer Science, Evalife Project Group, 2002.

[16] 孙俊, 方伟, 吴小俊, 等. 量子行为粒子群优化: 原理及其应用[M]. 北京: 清华大学出版社, 2011.

[17] ANDREI N. An unconstrained optimization test functions collection [J]. Advanced Modeling and Optimization, 2008, 10(1): 147 – 161.