一种新的改进粒子群优化方法

刘怀亮[1],高 鹰[1],许若宁[2],苏瑞娟[1]
LIU Huai-liang[1],GAO Ying[1],XU Ruo-ning[2],SU Rui-juan[1]

1.广州大学 计算机科学与教育软件学院,广州 510006
2.广州大学 数学与信息科学学院,广州 510006
1.Faculty of Computer Science and Educational Software,Guangzhou University,Guangzhou 510006,China
2.Faculty of Mathematics and Information Science,Guangzhou University,Guangzhou 510006,China
E-mail:huailiangliu@126.com

LIU Huai-liang,GAO Ying,XU Ruo-ning,et al.New improved particle swarm optimizer.Computer Engineering and Applications,2010,46(12):38-41.

Abstract: To solve the local-optima convergence problem of particle swarm optimization,two new methods are introduced to modify the Particle Swarm Optimization inertia weight in parallel:When particles' fitness values are better than or equal to the average,the introduced dynamic nonlinear equations are employed to modify the inertia weight,which can make particles retain favorable conditions and converge to the global optima continually;on the contrary,when fitness values are worse than the average,the dynamic Logistic chaotic map formula is introduced to modify the inertia weight,which can make particles break away from the local optima and search the global optima dynamically.Two methods coordinate dynamically,and make two dynamic subswarms cooperate to evolve.Experimental results demonstrate that the new introduced algorithm outperforms many other famous improved Particle Swarm Optimization algorithms on many well-known benchmark problems.

Key words: Particle Swarm Optimization(PSO);inertia weight;dynamic nonlinear equations;dynamic Logistic chaotic map

摘 要:为解决粒子群优化算法易于陷入局部最优问题,提出了两种新方法并行修改粒子群优化算法惯性权重:对好于或等于整体适应度平均值的粒子,用动态非线性方程调整惯性权重,在保存相对有利环境的基础上逐步向全局最优处收敛;对比平均值差的粒子,用动态 Logistic 混沌映射公式调整惯性权重,在复杂多变的环境中逐步摆脱局部最优,动态寻找全局最优值。两种方法前后相辅相成、动态协调,使两个动态种群相互协作、协同进化。实验结果证实:该算法在不同情况下都超越了同类著名改进算法。

关键词:粒子群优化算法;惯性权重;动态非线性方程;动态 Logistic 混沌映射公式
DOI:10.3778/j.issn.1002-8331.2010.12.010 **文章编号**:1002-8331(2010)12-0038-04 **文献标识码**:A **中图分类号**:TP18

1 引论

粒子群优化算法(Particle Swarm Optimization,PSO)是由 Kennedy 和 Eberhart[1]于 1995 年首次提出的一种群智能优化算法,它源于对鸟群等生物群体觅食行为和理论研究的结果。

粒子群优化算法提出之后引起了全世界学者的广泛关注。由于粒子群优化算法概念简单,收敛速度快,涉及参数少,易于实现,具有很强的全局优化能力,因而迅速得到了认可,在当前的优化应用中受到越来越多的关注。目前粒子群优化算法已广泛应用于函数优化、神经网络训练、模糊系统控制、水利、电力和经济等大规模组合优化问题求解[2-5]。

类似遗传算法和模拟退火算法[6-7],粒子群优化算法也容易陷入局部最优,因此许多学者提出很多著名改进方法[2-5,8-14]。该文构造了动态非线性方程与动态 Logistic 混沌公式, 分别修改两种不同适应度值的动态粒子群,实验结果证明该算法比同类著名改进算法效果更好。

2 粒子群优化算法及其重要改进

粒子群优化算法形式如下[1-2,8-9]:

$$v_{id}=wv_{id}+c_1r_1(p_{id}-x_{id})+c_2r_2(p_{gd}-x_{id}) \quad (1)$$

$$x_{id}=x_{id}+v_{id} \quad (2)$$

其中:x_{id} 代表第 i 个 D 维的粒子;p_{id} 代表第 i 个粒子目前所找到的最佳位置;p_{gd} 代表所有粒子目前找到的最佳位置;v_{id} 代表第 i 个粒子位置变换的速率(速度);c_1 和 c_2 为常量,分别称为认知因子和社会因子;r_1 和 r_2 为范围在 0 和 1 之间的两个随机数;w 为惯性权重。

为克服粒子群优化算法陷入局部最优,学者们提出了许多

基金项目:广东省自然科学基金(the Nature Science Foundation of Guangdong Province under Grant No.8451009101001040)。
作者简介:刘怀亮(1976-),男,讲师,主要研究方向:智能优化算法;高鹰(1963-),男,博士,教授,主要研究方向:智能优化算法;许若宁(1957-),男,博士,教授,主要研究方向:智能优化算法,模糊数学;苏瑞娟(1980-),女,硕士生,主要研究方向:智能优化算法。
收稿日期:2009-03-31 **修回日期**:2009-05-18

著名的改进方法。Clerc 引入了收缩因子 K[10-11],保障粒子群优化算法收敛。为满足优化复杂动态系统的要求,Eberhart 和 Shi 将惯性权重修改成:w=0.5+(Rnd/2.0)形式[3],其中 Rnd 为范围在 0 和 1 之间的一个随机数。Ratnaweera 引入了随时间变化的加速因子[12],对参数 c_1 和 c_2 进行修改。Chatterjee 引入了静态非线性修改惯性权重方法修改惯性权重[13]。

混沌与粒子群优化算法结合,也是一个很好的改进方法。目前有如下三类改进方法:第一类是加入混沌局部搜索作为内循环[14];第二类是用混沌映射取代粒子群优化算法中的 r_1 和 r_2[4];第三类是用混沌映射取代粒子群优化算法中的惯性权重 w[5]。其中,第一类方法加入内循环,时间复杂度增加了一个数量级,代价较大;第二、第三类方法更可取。

以上粒子群改进算法都是针对全部粒子用统一的方法修改,惯性权重指数是静态固定值,使用混沌也是不加任何调整的静态直接映射。针对不同适应度值的粒子,构造了动态非线性方程与动态 Logistic 混沌映射公式两种不同方法,分别修改两个动态种群惯性权重,大量实验结果证实,该算法性能在不同情况下都超越了同类著名改进算法。

3　动态非线性方程与动态 Logistic 混沌映射并行惯性权重调整机制

3.1　动态非线性方程惯性权重调整机制

为保障算法收敛,避免陷入局部最优,在保存粒子有利条件的基础上逐步寻找全局最优,提出了两种动态非线性方程惯性权重调整方法,第一种形式为:

$$dnl=dnl_{\min}+(dnl_{\max}-dnl_{\min})\left(\frac{iter}{iter_{\max}}\right) \tag{3}$$

$$w=w_{\min}+(w_{\max}-w_{\min})\left(\frac{iter_{\max}-iter}{iter_{\max}}\right)^{dnl} \tag{4}$$

第二种形式为:

$$dnl=dnl_{\max}-(dnl_{\max}-dnl_{\min})\left(\frac{iter}{iter_{\max}}\right) \tag{5}$$

$$w=w_{\max}-(w_{\max}-w_{\min})\left(\frac{iter}{iter_{\max}}\right)^{dnl} \tag{6}$$

其中,dnl 为动态非线性惯性权重调整因子,$dnl_{\min}$ 和 $dnl_{\max}$ 分别为 dnl 最小值和最大值;$iter$ 为当前循环次数,$iter_{\max}$ 为最大循环次数;$w_{\min}$ 和 $w_{\max}$ 分别为惯性权重最小值和最大值。

当 dnl 的值不是传统固定静态值 0 或 1,而是动态线性变化时,惯性权重表达式(4)和(6)呈动态非线性形态,且差异较大。

图 1 显示了惯性权重为表达式(3)与(4),dnl 从 0.4 到 1.8 逐步递增过程中非线性惯性权重 w 变化过程。图 2 显示了惯性权重为表达式(5)与(6),dnl 从 1.8 到 0.4 逐步递减过程中非线性惯性权重 w 变化过程。为了看得更清晰,这里只画出了 8 条曲线,实际上最大循环次数是多少,曲线就有多少条。

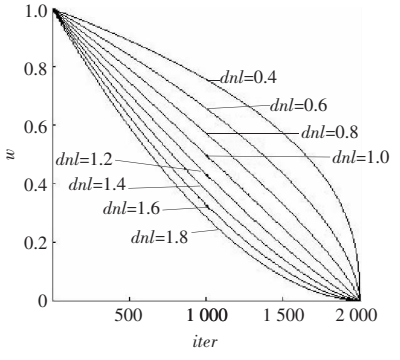

图 1　dnl 与 w 关系图 1

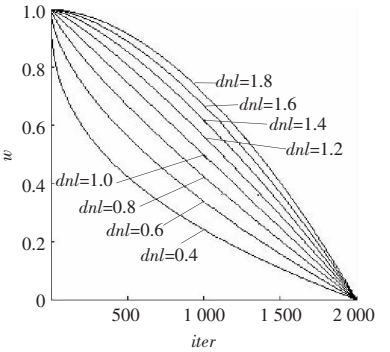

图 2　dnl 与 w 关系图 2

图 1 和图 2 左上角和右下角都比较平滑,算法执行区域正是从图的左上角区域逐步动态调整到右下角区域的,这保证了算法动态调整过程相对平稳。

图 1 和图 2 中惯性权重曲线由向上凸变为向下凸的逐步调整过程,反映了惯性权重由大到小的动态非线性调整过程。初始的向上凸,实际是要求算法进行大范围全局搜索,范围比传统情况下要大;最终的向下凸是要求算法进行小范围局部搜索,范围比传统情况下更细微;而由向上凸变为向下凸的逐步变化过程,反映了算法由大范围全局搜索逐步细微调整到小范围局部搜索的变化过程;惯性权重动态调整幅度大,范围广,且曲线相对平滑。

因此,动态非线性形式,比传统的静态形式更容易摆脱局部最优,做到大范围全局搜索和小范围的局部搜索的动态平衡,在逐步保存现有有利环境的基础上逐步向全局最优处收敛。

3.2　动态 Logistic 混沌映射惯性权重调整机制

混沌是一种有规律的动态非线性系统,具有非周期性、非收敛性、有界性、确定性、易于产生和存储、对初始值敏感、遍历性强等特征。著名的 Logistic 混沌映射形式如下[15]:

$$X_{n+1}=aX_n(1-X_n),n=0,1,2,\cdots,X_n\in(0,1) \tag{7}$$

虽然上述公式是确定的,但当 a=4,$X_0\notin\{0,0.25,0.5,0.75,1\}$的时候,就表现出混沌特性。当初始值不同的时候,会产生完全不同的混沌序列;初始值的一点微小变化,会导致后面混沌序列值的很大不同;使得该混沌映射可以不重复地遍历整个搜索空间。图 3 显示了在 X_0=0.000 1,循环次数 n=1 000 时 Logistic 混沌映射 $X(n)$遍历的轨迹。

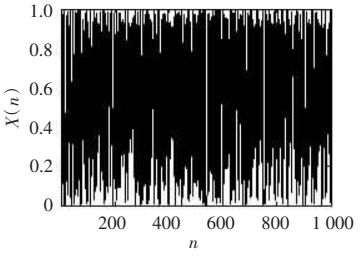

图 3　Logistic 混沌映射

为适应复杂多变的环境,迫使适应度值较差的粒子跳出局部最优,引入了动态惯性权重调节因子 α。因子 α 随算法执行而逐步动态下降,相应惯性权重平均值逐步动态减小,而混沌影响却逐步增大。动态 Logistic 混沌映射数学表达形式如下:

$$\alpha=\alpha_{\max}-(\alpha_{\max}-\alpha_{\min})\left(\frac{iter}{iter_{\max}}\right) \tag{8}$$

$$w=\alpha+(1-\alpha)Lmap \tag{9}$$

其中,α_{max} 为 α 的最大值,α_{min} 为 α 的最小值,$iter$ 为当前循环次数,$iter_{max}$ 为最大循环次数,$Lmap$ 为 Logistic 混沌映射的结果。

公式(8)和(9)中,α 是保证惯性权重 w 值相对稳定的重要因素;$(1-\alpha)Lmap$ 是使 w 值在一定范围内受混沌映射影响适应复杂多变环境的需要;α 值由 α_{max} 到 α_{min} 的线性递减,使得惯性权重 w 整体平均值由大到小动态非线性递减,是保证算法收敛,逐步从大范围全局搜索过渡到小范围局部搜索的必要条件;同时,惯性权重 w 整体平均值逐步下降,而混沌影响因素却在递增,因为 $(1-\alpha)Lmap$ 的平均值在增加,这可以防止算法运行到小范围搜索时陷入局部最优。因此,α、$(1-\alpha)Lmap$、α 的线性递减,这些因素是相辅相成、不可缺少的;它们共同促成了惯性权重动态非线性调整,使算法即使在复杂多变的环境中,依然能够摆脱局部最优,在动态变化中搜索全局最优。

3.3 动态非线性方程与动态 Logistic 混沌映射并行惯性权重调整算法

对粒子群中处于不同适应度值的粒子,采用不同的调整方法并行处理,形成两个动态种群相互协作、协同进化的形式。算法形式如下:

初始化粒子数、维数、权重等系统参数。

repeat

计算每个粒子适应度值。

if $f_i \leqslant favg$

调用公式(3)、(4)、(1)、(2),

或者,调用公式(5)、(6)、(1)、(2)。

elseif $f_i > favg$

调用公式(8)、(9)、(1)、(2)。

endif

until(满足退出条件)

其中,f_i 表示第 i 个粒子的适应度值,$favg$ 表示函数运行过程中的平均适应度值,这里以全局最小值表示最优状态。

运行过程中,当粒子进化速度比其他粒子快,由开始比平均值差,逐步超越平均值,或一直都处于较优环境中时,为避免这一动态种群粒子从当前相对较优环境,陷入较差区域,使用该文提出的动态非线性方程形式,在动态非线性因子 dnl 的作用下,逐步细微调整粒子惯性权重,保障这一动态种群粒子逐步收敛到较优值。当粒子进化速度较慢,由开始比平均适应度值好,慢慢落后到比平均值差,或者一直都处于较差情况时,为了使这一动态种群粒子逃离局部最优,使用该文提出的动态 Logistic 混沌映射惯性权重调整机制,在动态 Logistic 混沌权重调节因子 α 的作用下,逐步调整惯性权重,迫使粒子跳出局部最优,在复杂多变的环境中寻找全局最优。

算法前后两种形式针对两个不同适应度值的动态种群,动态、非线性、并行地修改粒子群惯性权重,前后两种方法动态协调、相辅相成,使两个动态种群互相协作、协同进化,逐步向全局最优处收敛。

4 实验方法、结果与分析

4.1 测试函数

使用多个著名的标准测试函数,包含 2 维和 30 维测试函数,测试算法在不同复杂程度下的寻优能力。2 维测试函数如下:

(1)$f_1(x)$:Branin

$$f_1(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{4\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)*\cos(x_1)+10,$$
$$-5\leqslant x_1\leqslant 10,0\leqslant x_2\leqslant 15$$

该函数最优值约等于 0.398,最优状态在(-3.142,12.275)、(3.142,2.275)和(9.425,2.425)三个地方。

(2)$f_2(x)$:Schaffer's F6 Function

$$f_2(x)=0.5+\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{[1+0.001(x_1^2+x_2^2)]^2},|x_i|\leqslant 1$$

该函数最优值等于 0,位置在(0,0)。此函数在全局最优值附近大约 3.14 范围内存在无数多个局部极小将其包围,并且函数强烈震荡,因此算法很难得到最优解。

30 维测试函数如下:

(3)$f_3(x)$:Sphere Model(又称为 De Jong's f1)

$$f_3(x)=\sum_{i=1}^{30}x_i^2,|x_i|\leqslant 100$$

(4)$f_4(x)$:Ackley's Function

$$f_4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30}x_i^2}\right)-$$
$$\exp\left(\frac{1}{30}\sum_{i=1}^{30}\cos(2\pi x_i)+20+e\right),|x_i|\leqslant 32$$

(5)$f_5(x)$:Generalized Griewank Function

$$f_5(x)=\frac{1}{4\,000}\sum_{i=1}^{30}x_i^2-\prod_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1,|x_i|\leqslant 600$$

(6)$f_6(x)$:Generalized Rastrigin's Function

$$f_6(x)=\sum_{i=1}^{30}[x_i^2-10\cos(2\pi x_i)+10],|x_i|\leqslant 5.12$$

以上 30 维标准测试函数最优状态和最优值均为:$\min(f(x_{id}))=f(0,0,\cdots,0)=0$。与 2 维测试函数相比,30 维测试函数更为复杂,通常会有数以千万计,甚至无穷多个极小值。

4.2 实验参数设置

文中提出的改进粒子群优化算法有两种形式:采用公式(3)、(4)、(8)、(9)的简称为 NPSO1(New PSO 1),采用公式(5)、(6)、(8)、(9)的简称为 NPSO2(New PSO 2)。参数设置方式如下:动态非线性因子 $dnl_{max}=1.8$,$dnl_{min}=0.4$;惯性权重 $w_{max}=0.9$,$w_{min}=0.4$;动态 Logistic 混沌映射因子 $\alpha_{max}=0.9$,$\alpha_{min}=0.4$;采用动态 Logistic 混沌映射惯性权重调整机制时 c_1 和 c_2 取值为1.494 45;采用动态非线性方程调整机制时 c_1 和 c_2 取值为 2.0。

算法 PSOCF[10-11]、PSORW[3]、PSOTC[12]、PSONW[13]、PSOCR[4]、PSOCW[5]惯性权重 w、认知因子 c_1 和社会因子 c_2 参数值完全按照文中最好的设置方式。

为保证公平,实验中所有算法粒子数统一设定为较少的 20 个;循环最多不超过 2 000 次;循环退出条件为:达到最大循环次数,或者误差小于 0.000 01;速度最大值 $V_{max}=(X_{max}-X_{min})/4$,最小值 $V_{min}=-V_{max}$,其中 X_{max} 和 X_{min} 表示函数所允许搜索的最大值和最小值;粒子初始化为整个函数允许的空间。

4.3 实验结果

实验在 MATLAB 7.4.0(R2007a)软件环境下完成,计算机 CPU 为奔腾双核 1.87 GHz 主频,内存为 2 GB,将每种算法针对每个函数运行 100 次,结果如表 1 所示,图 4 显示了各种算法针对著名的高维复杂函数 $f_5(x)$动态进化过程对比。

表 1　实验结果

函数	指标	PSOCF	PSORW	PSOTC	PSONW	PSOCR	PSOCW	NPSO1	NPSO2
	平均	0.397 9	0.397 9	0.398 7	0.397 9	0.398 4	0.397 9	0.397 9	0.397 9
$f_1(x)$	最优	0.397 9	0.397 9	0.397 9	0.397 9	0.397 9	0.397 9	0.397 9	0.397 9
	方差	4.469 3E−5	3.665 6E−5	8.164 1E−4	4.739 3E−5	4.822 1E−4	4.579 0E−5	4.297 9E−5	4.223 6E−5
	平均	0.005 3	0.004 2	0.008 6	0.004 4	0.008 2	0.001 3	7.998 0E−4	2.043 0E−4
$f_2(x)$	最优	3.150 0E−7	2.263 7E−7	9.638 3E−7	2.076 3E−7	2.311 0E−4	2.829 6E−8	1.686 6E−7	2.294 6E−8
	方差	0.004 8	0.004 8	0.002 6	0.004 7	0.002 8	0.003 3	0.002 7	0.001 4
	平均	9.415 9E−6	9.434 0E−6	1.050 8E+4	712.746 9	6.076 4E+3	10.145 8	9.359 7E−6	9.468 2E−6
$f_3(x)$	最小	6.600 4E−6	7.358 5E−6	7.876 8E+3	6.902 3	4.445 5E+3	5.500 1E−5	7.444 0E−6	7.694 1E−6
	方差	5.800 5E−7	5.800 1E−7	1.072 7E+3	897.257 6	704.937 5	14.933 5	6.009 6E−7	4.951 6E−6
	平均	2.762 4	2.463 7	15.660 2	5.171 0	13.65 14	1.627 2	0.459 6	0.486 7
$f_4(x)$	最小	9.830 6E−6	9.676 3E−6	14.062 7	0.023 1	11.702 2	0.001 0	8.493 7E−6	8.620 9E−6
	方差	1.145 0	1.270 5	0.418 8	2.256 7	0.499 9	1.249 8	1.520 4	1.509 8
	平均	0.089 3	0.042 9	97.094 3	8.463 1	56.682 9	0.767 1	0.012 8	0.011 2
$f_5(x)$	最优	6.853 8E−6	8.450 3E−6	68.954 5	0.365 3	32.914 8	9.617 8E−5	8.167 3E−6	6.970 1E−6
	方差	0.177 5	0.077 1	9.488 7	7.113 4	6.255 9	0.532 4	0.014 4	0.015 9
	平均	78.159 2	66.392 0	248.745 5	67.059 8	233.348 4	65.148 4	55.750 6	53.217 1
$f_6(x)$	最优	31.838 7	25.868 9	208.387 0	33.679 6	179.567 0	17.100 1	21.889 1	21.889 1
	方差	24.604 1	18.499 3	13.945 1	18.520 0	17.413 1	26.325 4	22.822 9	20.535 9

4.4　实验分析

从运行结果的平均值、最优值和方差 3 个方面去考查可以看出:对于相对简单的 2 维函数 $f_1(x)$,所有改进算法都可以达到最优;对于复杂的 2 维函数 $f_2(x)$,NPSO1 和 NPSO2 结果最好,其中 NPSO2 效果更好;对于相对简单的 30 维函数 $f_3(x)$,算法 PSOCF、PSORW、NPSO1 和 NPSO2 表现都较好;对于非常复杂的 30 维函数 $f_4(x)$、$f_5(x)$和 $f_6(x)$,只有 NPSO1 和 NPSO2 结果最好。

从图 4 的各种算法针对 $f_5(x)$动态进化过程对比图可以看出,NPSO1 和 NPSO2 能够很好地平衡全局和局部的关系,使粒子摆脱局部最优,具有较好的全局收敛性能。

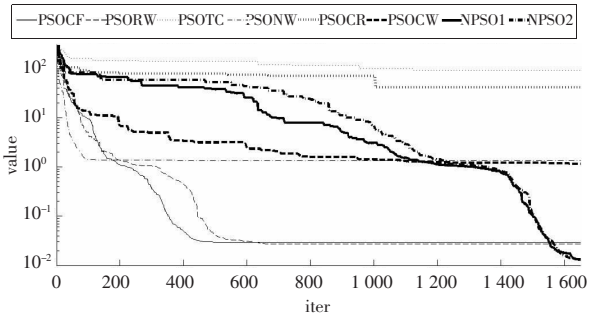

图 4　各种算法针对 $f_5(x)$动态进化过程对比图

综合以上内容发现:当函数比较容易时,所有改进算法表现均较好;但当函数很复杂的时候,NPSO1 和 NPSO2 算法显示出卓越的性能。采用其他一些标准测试函数测试,实验结果与以上分析一致。

5　结论

提出了两种新型方法有针对性、并行、动态、非线性地修改粒子群优化算法惯性权重:对适应度值比平均值好或等于平均值的粒子,为使这一动态种群在保存当前有利条件的基础上,逐步由大范围到小范围向全局最优处收敛,采用所构造的动态非线性方程形式,逐步修改粒子惯性权重;对适应度值比平均值差的粒子,为使这一动态种群跳出局部最优,逐步动态寻找全局最优,采用动态 Logistic 混沌映射公式,修改粒子惯性权重;两种方法相辅相成、相互协调、并行处理,使处于两种不同情况的两个动态种群相互协作、协同进化。实验结果验证该文提出的算法比同类改进算法效果更好,是一个很有发展前景和应用潜力的群智能优化算法。

参考文献:

[1] Kennedy J,Eberhart R C.Particle swarm optimization[C]//Proceedings of the IEEE Conference on Neural Networks,Perth,Australia,1995:1942-1948.

[2] Eberhart R C,Shi Y.Particle swarm optimization:Developments,applications and resources[C]//Proc Congress on Evolutionary Computation,Seoul,Korea,2001:81-86.

[3] Eberhart R C,Shi Y.Tracking and optimizing dynamic systems with particle swarms[C]//Proceedings of IEEE Congress on Evolutionary Computation,Seoul,Korea,2001:94-97.

[4] dos Santos Coelho L,Lee Chu-Sheng.Solving economic load dispatch problems in power systems using chaotic and Gaussian particle swarm optimization approaches[J].Electrical Power and Energy Systems,2008,30:297-307.

[5] Jiang Chuan-wen,Etorre B.A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization[J].Mathematics and Computers in Simulation,2005,68:57-65.

[6] 刘怀亮.模拟退火算法及其改进[J].广州大学学报:自然科学版,2005,4(6):503-506.

[7] 刘怀亮,刘淼.一种混合遗传模拟退火算法及其在 TSP 问题中的应用[J].广州大学学报:自然科学版,2005,4(2):141-145.

[8] Shi Y,Eberhart R C.A modified particle swarm optimizer[C]//Proceedings of IEEE International Conference on Evolutionary Computation,Anchorage,USA,1998:69-73.

[9] Shi Y,Eberhart R C.Empirical study of particle swarm optimization[C]//Proc IEEE Int Congr Evolutionary Computation,Washington,DC,1999:1945-1950.

(下转 45 页)