文章编号:1001 - 9081(2010)09 - 2286 - 04

基于粒距和动态区间的粒子群权值调整策略

左旭坤[1],苏守宝[1,2]
(1. 皖西学院 计算机科学与技术系,安徽 六安 237012; 2. 安徽大学 计算智能与信号处理教育部重点实验室,合肥 230039)
(zxk_land@163.com)

摘 要:由于标准粒子群优化(PSO)算法把惯性权值作为全局参数,因此很难适应复杂的非线性优化过程。针对这一问题,提出了一种基于粒距和动态区间的权值调整策略(PSSIW),根据粒子的粒距大小在动态区间内选取不同的权值,并通过区间的动态变化来控制算法的收敛速度。设计了四种不同的动态区间,并采用三个常用的标准测试函数测试不同区间对算法性能的影响。通过与标准粒子群算法比较发现,该策略提高了算法摆脱局部极值的能力,是一种新型全局收敛粒子群算法。

关键词:粒子群优化算法;惯性权值;粒距;动态区间
中图分类号: TP301.6 **文献标志码**:A

Inertia weight adjustment strategy based on particle spacing and dynamic interval for PSO

ZUO Xu-kun[1], SU Shou-bao[1,2]
(1. *Department of Computer Science and Technology, West Anhui University, Lu'an Anhui* 237012, *China*;
2. *Key Laboratory of Intelligent Computing and Signal Processing, Ministry of Education, Anhui University, Hefei Anhui* 230009, *China*)

Abstract: The standard Particle Swarm Optimization (PSO) algorithm cannot adapt to the complex and nonlinear optimization process, because the same inertia weight is used to update the velocity of particles. In order to solve this problem, a strategy of inertia weight adjustment based on particle spacing and dynamic interval (PSSIW) was put forward. According to the particle spacing, the inertia weight was chosen, and the convergence rate of the algorithm were controlled by dynamic change of interval. Four different dynamic intervals were built in this paper. Sphere, Ackley and Rastrigrin functions were used to evaluate the intervals on the new PSO performance. Compared with the standard PSO algorithm, the new PSO algorithm has the ability to escape from the local minimum, so it is a global particle swarm optimization algorithm.

Key words: Particle Swarm Optimization (PSO) algorithm; inertia weight; particle spacing; dynamic interval

0 引言

粒子群优化(Particle Swarm Optimization, PSO)算法的思想来源于鸟群等社会群体生物的觅食现象,最初由 J. Kennedy 等人提出[1]。主要用来优化非线性、不可微和多峰值的复杂问题。由于 PSO 算法概念简单、易于编程、对计算机硬件性能要求不高,因此在多个学科和工程领域里得到广泛应用。

PSO 算法需要调整的参数很少,其中惯性权值的选取对算法性能具有很大的影响,因此国内外学者对此做了大量的研究工作,先后出现了线性递减策略(Linearly Decreased Inertia Weight, LDIW)[2]、模糊策略(Fuzzy Inertia Weight, FIW)[3]和随机策略(Random Inertia Weight, RIW)[4]等权值调整算法。其中 FIW 需要专家知识建立模糊规则,实现难度较大;RIW 主要用于求解动态系统;LDIW 算法简单且收敛较快,应用最广泛。

本文设计了一种基于粒距(粒子距当前最优点的距离)和动态区间(区间长度随迭代次数动态变化)的权值调整策略(Use Particle Spacing Selected Inertia Weight, PSSIW)。为了找到算法收敛速度和收敛精度的平衡点,本文构造了 4 种动态区间进行测试,并与 LDIW 策略进行了对比研究。

1 标准 PSO 算法

PSO 算法首先初始化种群随机粒子(particle),每个粒子都代表着优化问题的可能解。粒子 j 的信息可以用 D 维向量表示,位置表示为 $\boldsymbol{X}_j = (x_{1j}, x_{2j}, \cdots, x_{dj})^{\mathrm{T}}$,速度表示为 $\boldsymbol{V}_j = (v_{1j}, v_{2j}, \cdots, v_{dj})^{\mathrm{T}}$。每一次迭代,粒子通过跟踪两个"极值"来更新自己。一个是粒子本身所找到的最优解,用 *pbest* 表示其坐标。另一个是整个种群目前找到的最优解,用 *gbest* 表示其坐标。找到这两个最优值时,粒子根据如下的公式来更新自己的速度和位置:

$$v_{ij}(t+1) = \omega \times v_{ij}(t) + c_1 \times r_1 \times [pbest_{ij}(t) - x_{ij}(t)] + c_2 \times r_2 \times [gbest_i(t) - x_{ij}(t)] \quad (1)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t) \quad (2)$$

其中:$v_{ij}(t)$ 是粒子 j 在第 t 次迭代中第 i 维的速度;$x_{ij}(t)$ 是当前粒子的位置;r_1, r_2 是(0,1)的随机数,用来模拟自然群体中的轻微扰动;c_1, c_2 是学习因子,分别表示粒子对自身记忆的依赖程度和其他粒子对它的影响;ω 是惯性权值,体现粒子对

收稿日期:2010 - 03 - 12;**修回日期**:2010 - 05 - 09。
基金项目:安徽高校省级自然科学研究项目(KJ2010B467);安徽高校省级自然科学研究重点资助项目(KJ2007A087)。
作者简介:左旭坤(1978 -),男,安徽六安人,讲师,主要研究方向:计算机测控与智能控制; 苏守宝(1965 -),男,安徽六安人,教授,主要研究方向:群体智能、模式识别。

自身速度的保留程度。

2　惯性权值调整策略研究

2.1　惯性权值对算法性能的影响

将式(1)、(2)简化为一维单个粒子,令$\varphi_1 = c_1 \times r_1$,$\varphi_2 = c_2 \times r_2$,$\varphi = \varphi_1 + \varphi_2$,$\varphi' = 1 - \varphi$,$\varphi_d = \varphi_1 \times pbest + \varphi_2 \times gbest$,若将$\varphi_1$、$\varphi_2$和$\varphi_d$看为常数,则$\omega$就成为式(1)、(2)中的唯一变量,利用状态空间将其描述为[5]:

$$\boldsymbol{Y}(t+1) = \boldsymbol{A}\boldsymbol{Y}(t) + \boldsymbol{B} \tag{3}$$

其中:$\boldsymbol{Y}(t) = \begin{bmatrix} x(t) \\ v(t) \end{bmatrix}$;$\boldsymbol{A} = \begin{bmatrix} \varphi' & \omega \\ -\varphi & \omega \end{bmatrix}$;$\boldsymbol{B} = \begin{bmatrix} \varphi_d \\ \varphi_d \end{bmatrix}$。

矩阵$\boldsymbol{A}$的特征方程为:

$$\lambda^2 - (\omega + \varphi')\lambda + \omega = 0 \tag{4}$$

可知其两个特征根为:

$$\lambda_{1,2} = (\omega + \varphi' \pm \sqrt{(\omega + \varphi')^2 - 4\omega})/2 \tag{5}$$

则式(3)的显性表达式可写为:

$$\begin{cases} x(t) = \{\varphi_d - [k_1\lambda_1^t(e_1 - \omega) + k_2\lambda_2^t(e_2 - \omega)]\}/\varphi \\ v(t) = k_1\lambda_1^t + k_2\lambda_2^t \end{cases} \tag{6}$$

由式(6)知,$x(t)$和$v(t)$的收敛速度取决于$\lambda_{1,2}$,而$\lambda_{1,2}$的大小又与ω成正比。因此权值ω对 PSO 算法的影响可总结为:

1)较大的权值有利于 PSO 的全局搜索;

2)较小的权值有利于 PSO 的局部所搜。

2.2　LDIW 策略分析

目前广泛采用的权值调整策略是 LDIW 策略,其算法公式描述如下(设$\Delta\omega = \omega_{max} - \omega_{min}$,全文同):

$$\omega(t) = \omega_{max} - \Delta\omega \times t/t_{max} \tag{7}$$

其中ω_{max}、ω_{min}分别为最大、最小权值,当$\omega_{max} = 0.95$、$\omega_{min} = 0.4$时,有助于提高 PSO 算法性能[2];t为当前迭代次数;t_{max}为迭代总次数。

线性递减的优势在于,算法早期的极值点未必是真正的极值点,所以较大的ω可使v_{ij}偏离当前极值点,从而有可能搜索到更好的极值点。而算法末期ω较小有助于全局极值点的收敛。但是该策略也存在两个缺陷:

1)所有粒子采用相同的权值,不利于种群的多样性,同时恒速递减也降低了求解效率;

2)迭代初期权值较大,局部搜索能力较弱,容易错过全局最优点;迭代后期权值较小,全局搜索能力变弱,容易陷入局部最优。

文献[6]提出了佳粒子距的概念,指出权值的选择应该与粒子在群中的位置相关。文献[7]指出在算法不同时期采取不同的权值递减速度有助于提高算法性能。受上述思想启发,本文提出了 PSSIW 权值调整策略。

2.3　PSSIW 算法思路

规定在第t次迭代时第j个粒子在i维中的粒距为:

$$d_{ij}(t) = |x_{ij}(t) - gbest_i(t)| \tag{8}$$

根据粒子的粒距对粒子排序,并根据排序顺序在区间$[W_{start}, W_{end}]$内取值。粒距较小的粒子取较小的权值,使其拥有较强的局部搜索能力;粒距较大的粒子取较大的权值,使其拥有较强的全局搜索能力。算法主要代码描述如下:

```
for i = 1: dimension                          //dimension: 维数
  for j = 1: size                             //size: 粒子数
    A((i-1) * size + j) = abs(gbest(i) - x(i,j));
                                              //数组 A 存放粒距
  end
end
[B, M] = sort(A);            //排序,数组 M 保存排序前元素下标
for k = 1: size * dimension
  for i = 1: dimension
    for j = 1: size
      if ((i-1) * size + j) == M(k)
        weight(i, j) = W(k);   //按粒距顺序,在 W 中取相应权值
      end
    end
  end
end
```

2.4　权值区间的设计

本文构造了 4 种权值区间($W[W_{end}, W_{start}]$)进行对比研究,如下。

1)静态区间。

$$\begin{cases} W_{end}(t) = \omega_{min} \\ W_{start}(t) = \omega_{max} \end{cases} \tag{9}$$

2)线性收缩的动态区间。

$$\begin{cases} W_{end}(t) = \omega_{min} \\ W_{start}(t) = \omega_{min} + \Delta\omega \times (t_{max} - t)/t_{max} \end{cases} \tag{10}$$

3)按凹函数非线性收缩的动态区间。

$$\begin{cases} W_{end}(t) = \omega_{min} \\ W_{start}(t) = \omega_{max} + \Delta\omega \times [(t/t_{max})^2 - (2t/t_{max})] \end{cases} \tag{11}$$

4)按凸函数非线性收缩的动态区间。

$$\begin{cases} W_{end}(t) = \omega_{min} \\ W_{start}(t) = \omega_{max} - \Delta\omega \times (t/t_{max})^2 \end{cases} \tag{12}$$

可知,区间2)随迭代次数匀速收缩;区间3)在算法前期加速收缩,后期放慢收缩;区间4)的收缩速度正好与区间3)相反。同时,这三个区间都能从W_{start}收缩到W_{end},正好满足文献[2]的结论。

2.5　粒子聚集程度的考虑

当粒子之间的距离过小时,种群的全局搜索能力降低,很容易陷入局部极值而无法跳出;而距离过大时,算法又难以收敛。因此需要对粒子的聚集程度进行跟踪,当聚集度过高(粒子间距较小)时应加大权值,使其有逃离局部极小的能力;当聚集度过低时减小权值,有利于算法收敛。基于以上考虑,本文引入了聚集度的概念:

$$\begin{cases} d_{mean}(t) = \left[\sum_{j=1}^{N}\sum_{i=1}^{D} d_{ij}(t)\right]/(N \times D) \\ d_{max}(t) = \max[d_{ij}(t)] \\ k = [d_{max}(t) - \overline{d(t)}]/d_{max}(t) \end{cases} \tag{13}$$

其中:$d_{mean}(t)$是全部粒子的平均粒距;$d_{max}(t)$是最大粒距;k被定义成此刻种群的聚集度,k小则种群聚集度高;k大则种群聚集度低,且$0 < k < 1$。根据聚集度,对粒子权值做如下规定(α和β是调整参数,本文分别取 0.95 和 0.05):

$$\begin{cases} \omega(t) = W_{end}(t), & k \geqslant \alpha \\ \omega(t) = W_{start}(t), & k \leqslant \beta \end{cases} \tag{14}$$

3 实验分析

3.1 标准测试函数

选取 Benchmark 的 3 个常用测试函数对 LDIW 和 PSSIW 进行测试对比，如表 1 所示。

表 1 中，Sphere 函数是一个单峰函数，只有一个全局最优解，整个优化过程是平滑过渡到最优解；Ackley 函数是多级函数，优化过程呈螺旋向四周扩散；Rastrigin 函数是一个多峰函数，在唯一的全局最优解四周有多个半球形的局部最优解。

为方便记忆，采用 LDIW 策略的算法记为 W_0-PSO；在 PSSIW 策略中，采用式(9)～(12)取值区间的算法分别记为 W_1-PSO、W_2-PSO、W_3-PSO 和 W_4-PSO。所有算法的参数均一致，即：$\omega_{min}=0.4$，$\omega_{max}=0.95$，$c_{1,2}=2.05$，$t_{max}=2000$，维数 D 分别取 10 维和 30 维，种群规模 N 取 30，所有实验均重复 50 次。

表 1 基准函数

函数名	表达式	参数
Sphere	$f(x)=\sum_{i=1}^{D}x_i^2$	$v,x\in[-5,5]$
Ackley	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)^2\right]+20+e$	$v,x\in[-30,30]$
Rastrigin	$f(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	$v,x\in[-5.12,5.12]$

3.2 测试结果及分析

测试结果如表 2～4 所示。

表 2 Sphere 函数测试结果

维数	算法	最优值	最差值	平均值	收敛代数
10维	W_0-PSO	3.4234E-6	0.0222	0.0012	1.0391E+3
	W_1-PSO	8.2868E-17	1.1275E-4	4.5671E-6	1.5649E+3
	W_2-PSO	1.1173E-16	4.6895E-8	2.3497E-9	1.0444E+3
	W_3-PSO	1.7278E-13	9.0886E-4	4.8804E-5	275.54
	W_4-PSO	3.0094E-21	2.8600E-8	1.2144E-9	1.4964E+3
30维	W_0-PSO	0.0501	75.5279	10.4556	942.00
	W_1-PSO	0.0010	25.0496	1.0402	1.7528E+3
	W_2-PSO	2.0588E-4	25.0588	1.6268	930.78
	W_3-PSO	0.1546	13.6427	2.4521	240.12
	W_4-PSO	1.2549E-4	25.0031	0.6293	1.4246E+3

从表2～4可以看出，基于PSSIW策略的新型PSO算法收敛精度都明显高于标准 PSO 算法。当维数增加后，虽然新型 PSO 算法精度丢失较大，但仍优于标准 PSO 算法，说明新型 PSO 优化复杂函数的能力较好。按凸函数收缩区间的算法精度最佳，并且最后收敛代数(精确到 10^{-2})较大，说明在算法末期仍具有跳出局部最优的能力；凹函数收缩区间算法收敛最快，精度最低，容易陷入局部最优，其原因在于凹函数区间在算法早期收缩较快，从而使得所有粒子的平均权值快速减小，种群活性变低，不易逃离局部最优。

另外一个有趣的现象是当维数增加到 30 时，四种不同区间的新型 PSO 算法的最差值非常接近，说明当优化过程趋于复杂时，不同区间对算法性能的影响也趋于相同。

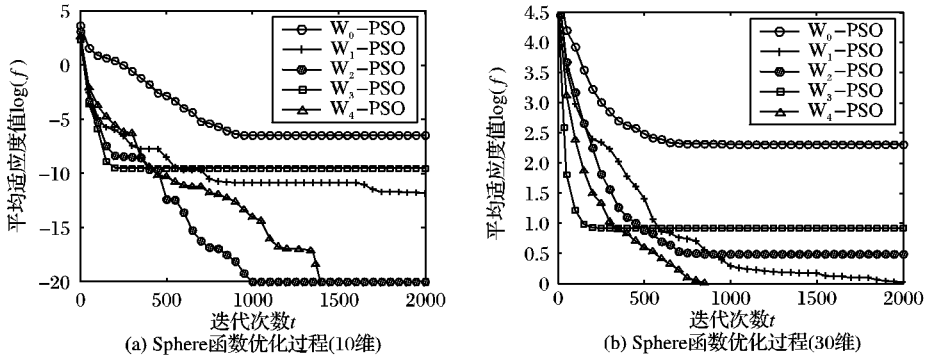
表 3 Ackley 函数测试结果

维数	算法	最优值	最差值	平均值	收敛代数
10维	W_0-PSO	0.0098	2.6175	1.2413	936.28
	W_1-PSO	1.0104E-6	1.5890	0.1599	1.6208E+3
	W_2-PSO	1.2144E-7	1.0066	0.1973	935.06
	W_3-PSO	2.8619E-4	3.7714	0.8913	208.92
	W_4-PSO	8.5864E-8	1.2340	0.1042	1.4198E+3
30维	W_0-PSO	2.1452	16.4859	8.1679	971.40
	W_1-PSO	1.1963	17.0057	5.1277	1.8036E+3
	W_2-PSO	2.2811	17.7824	6.4034	1.0392E+3
	W_3-PSO	3.8682	17.4656	11.2998	326.86
	W_4-PSO	0.8837	15.3155	3.4987	1.3952E+3

表 4 Rastrigin 函数测试结果

维数	算法	最优值	最差值	平均值	收敛代数
10维	W_0-PSO	4.0713	43.8687	16.6694	956.96
	W_1-PSO	4.8143E-8	28.8539	2.0814	1.6978E+3
	W_2-PSO	4.4260E-6	29.9197	4.8776	753.96
	W_3-PSO	1.9899	34.8948	10.0245	235.54
	W_4-PSO	2.3036E-9	29.9197	3.6432	1.0694E+3
30维	W_0-PSO	82.2012	199.5821	129.3281	980.42
	W_1-PSO	18.8769	108.7946	52.8615	1.8674E+3
	W_2-PSO	17.2470	114.5069	63.2128	990.60
	W_3-PSO	26.8285	124.4267	67.6778	269.58
	W_4-PSO	14.9338	91.5270	48.8132	1.4741E+3

图 1～3 也反映出与表 2～4 相同的结论。可以直观地看出，凹函数区间在迭代初期(500 次之前)收敛快、性能佳，但随着迭代次数增加很快陷入局部最优。凸函数区间摆脱局部最优的能力最强，其次是静态空间和线性收缩空间。而基于这 4 种空间的新型 PSO 算法在收敛速度和收敛精度上都要优于标准 PSO 算法。

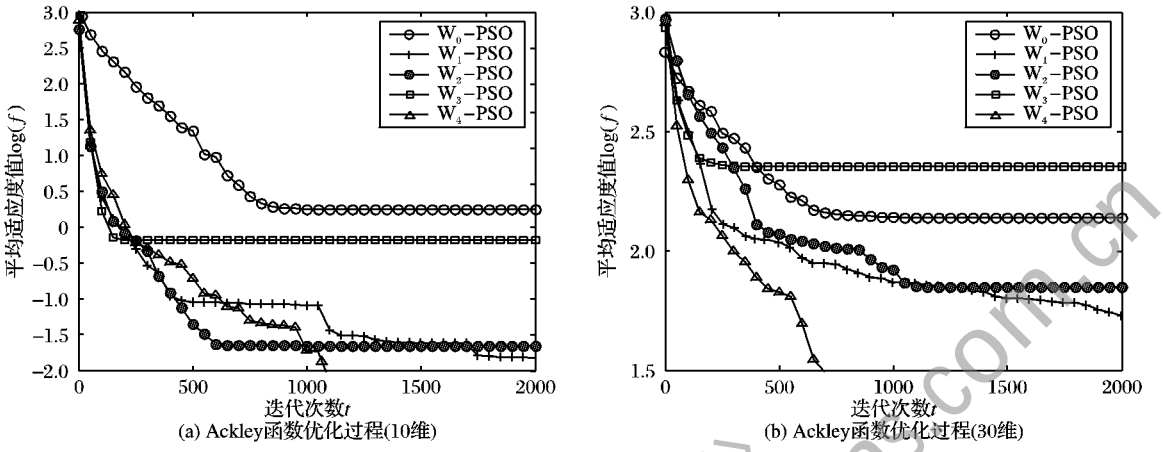


(a) Sphere函数优化过程(10维) (b) Sphere函数优化过程(30维)

图 1 Sphere 函数优化过程

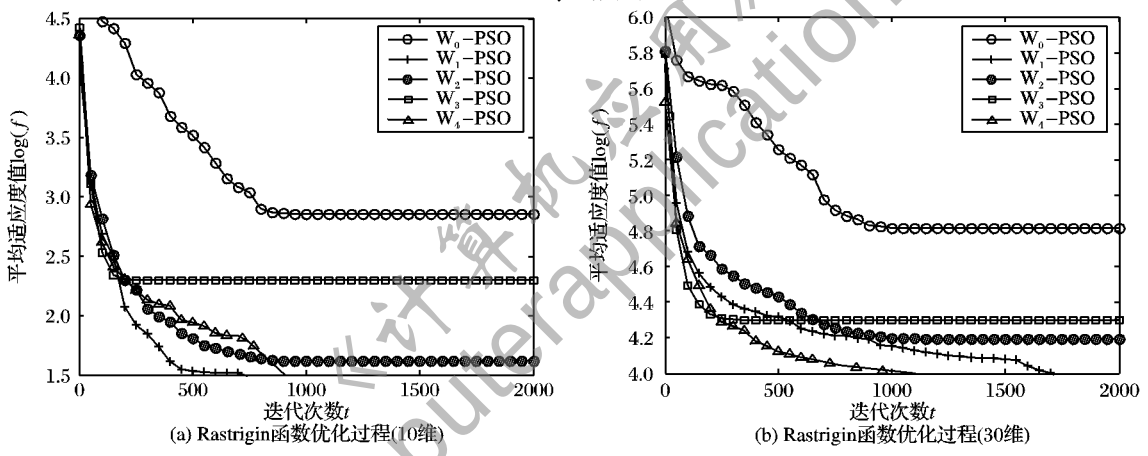
4　结语

本文分析了惯性权值对 PSO 算法的影响和线性递减策略的不足,提出了一种基于粒距和动态区间的 PSO 权值调整策略(PSSIW),并构造了 4 种不同的取值区间。通过与标准 PSO算法的比较发现,PSSIW策略能有效地提高算法的速度和精度。另外,不同区间对算法性能的影响也各不相同,凹函数区间在算法前期收缩较快,故算法收敛速度最快,但是容易陷入局部最优点;凸函数区间在迭代前期收缩慢,后期收缩快,能够在收敛速度和收敛精度之间找到较好的平衡点,故综合性能最优;静态区间和线性收缩区间性能介于上述两者之间。



(a) Ackley函数优化过程(10维)　(b) Ackley函数优化过程(30维)

图 2　Ackley 函数优化过程



(a) Rastrigin函数优化过程(10维)　(b) Rastrigin函数优化过程(30维)

图 3　Rastrigin 函数优化过程

参考文献:

[1] KENNDEY J, EBERHART R. Particle swarm optimization [C]// Proceedings of 1995 IEEE International Conference on Neural Networks. New Jersey: IEEE Press, 1995: 1942 – 1948.

[2] EBERHART R. Empirical study of particle swarm optimization [C]// Proceedings of 1999 IEEE International Conference on Evolutionary Computation. Washington, DC: IEEE Press, 1999: 1945 – 1950.

[3] EBERHART R. Fuzzy adaptive particle swarm optimization [C]// Proceedings of 2001 IEEE International Conference on Evolutionary Computation. San Francisco: IEEE Press, 2001: 101 – 106.

[4] EBERHART R, SHI Y. Tracking and optimizing dynamic systems with particle swarms [C]// Proceedings of 2001 IEEE International Conference on Evolutionary Computation. San Francisco: IEEE Press, 2001: 94 – 100.

[5] 黄翀鹏,熊伟丽,徐保国. 惯性权值对粒子群算法收敛性的影响及改进[J]. 计算机工程,2008,34(12):31 – 33.

[6] 郭文忠,陈国龙. 粒子群优化算法中惯性权值调整的一种新策略[J]. 计算机工程与科学,2007,29(1):70 – 75.

[7] 陈贵敏,贾建援,韩琪. 粒子群优化算法的惯性权值递减策略研究[J]. 西安交通大学学报,2006,40(1):53 – 61.

(上接第 2278 页)

[5] SUN Y J. Iterative RELIEF for feature weighting: Algorithms, theories, and applications [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(6): 1035 – 1051.

[6] BEZDEK J C. Convergence theory for fuzzy c-means: Counter example and repairs [J]. IEEE Transactions on System, Man and Cybernetics, 1987, 17(4): 873 – 877.

[7] KOHAVI R, JOHN G H. Wrapper for feature subset selection [J]. Artificial Intelligence, 1997, 97(1/2): 273 – 324.

[8] CHEN B, LIU H W, CHAI J, *et al.* Large margin feature weighting method via linear programming [J]. IEEE Transactions on Knowledge and Data Engineering, 2009, 21(10): 1475 – 1488.

[9] SUN Y J, LI J. Iterative RELIEF for feature weighting [C]// Proceedings of the 23rd International Conference on Machine Learning. Pennsylvania, USA: ACM Press, 2006: 913 – 920.

[10] GUYON I, ELISSEEFF A. An introduction to variable and feature selection [J]. Journal of Machine Learning Research, 2003, 3(1): 1157 – 1182.