一种动态改变惯性权重的自适应粒子群算法

任子晖 王 坚
(同济大学CIMS研究中心 上海 200092)

摘 要 针对惯性权重线性递减粒子群算法(LDWPSO)不能适应复杂的非线性优化搜索过程的问题,提出了一种动态改变惯性权重的自适应粒子群算法(DCWPSO),在该算法中引入聚焦距离变化率的概念,并根据它对粒子群算法搜索能力的影响,将惯性因子表示为关于聚焦距离变化率的函数。在每次迭代时算法可根据当前粒子群聚焦距离变化率的大小动态地改变惯性权重,从而使算法具有动态自适应性。对6个典型函数的测试结果表明,DCWPSO算法的收敛速度明显优于LDWPSO算法,收敛精度也有所提高。

关键词 粒子群优化,惯性权重,聚焦距离变化率,自适应
中图法分类号 TP18 **文献标志码** A

New Adaptive Particle Swarm Optimization Algorithm with Dynamically Changing Inertia Weight

REN Zi-hui WANG Jian
(Research of CIMS Center in Tongji University, Shanghai 200092, China)

Abstract A new adaptive Particle Swarm Optimization algorithm with dynamically changing inertia weight (DCWPSO) was presented to solve the problem that the linearly decreasing weight (LDWPSO) of the Particle Swarm Optimization algorithm cannot adapt to the complex and nonlinear optimization process. The rate of cluster focus distance changing was introduced in this new algorithm and the weight was formulated as a function of this factor according to its impact on the search performance of the swarm. In each iteration process, the weight was changed dynamically based on the current rate of cluster focus distance changing value, which provides the algorithm with effective dynamic adaptability. The algorithm of LDWPSO and DCWPSO were tested with six well-known benchmark functions. The experiments show that the convergence speed of DCWPSO is significantly superior to LDWPSO, and the convergence accuracy is increased.

Keywords Particle Swarm Optimization (PSO), Inertia weight, Rate of cluster focus distance changing, Adaptability

1 引言

粒子群优化(Particle Swarm Optimization,简称PSO)是由Kennedy和Eberhart[1]等人提出的一类模拟群体(swarm)智能行为的优化算法。其思想来源于对鸟群捕食行为的研究,它与遗传算法和蚁群算法相比,PSO有着算法简单、容易实现、并且可调整参数少等特点,因此被广泛地应用于结构设计[2],电磁场[3]和任务调度[4]等工程优化问题。

在粒子群算法的可调整参数中,惯性权值是最重要的参数,较大的权值有利于提高算法的全局搜索能力,而较小的权值会增强算法的局部搜索能力,为了找到一种能在全局搜索和局部搜索之间取得最佳平衡的惯性权值选取方法,研究人员进行了大量的研究工作,先后提出了线性递减权值(LDIW)策略[5]、模糊惯性权值(FIW)策略[6]和随机惯性权值(RIW)策略[7]。其中,FIW策略需要专家知识建立模糊规则,实现难度较大,RIW策略被用于求解动态系统,LDIW策略相对简单且收敛速度快,因此被广泛应用。粒子群优化算法与其它随机优化算法(如遗传算法)一样,也存在早熟收敛现象,尤其是在比较复杂的多峰搜索问题中,目前解决这一问题的主要方法是增加粒子群的规模,虽然对算法的性能有一定的改善,但同样存在缺陷:一是不能从根本上克服早熟收敛问题,二是会大量增加算法的运算量。

本文在递减惯性权值的基本思想指导下,提出了一种新的基于聚焦距离变化率的自适应惯性权重,当聚焦距离变化率较大时表明粒子的最大聚焦距离和平均聚焦距离相差较大,此时粒子的全局搜索较差,要使粒子尽快地进入全局搜索,需提出一种新的动态自适应粒子群优化算法(DCWPSO)。通过对6个粒子的测试,表明这种改进的PSO算法既保持了搜索速度快的特点,又提高了全局搜索的能力,搜索成功的能力有了很大的提高。

2 基本的PSO算法及相关概念

PSO以模拟鸟的群体智能为特征,以求解连续变量优化问题为背景。在PSO中,每只鸟被称之为一个粒子,每个粒

到稿日期:2008-03-20 本文受国家科技支撑计划项目(2006BAF01A46),上海市科技发展基金重点项目(061612058),上海市基础研究重点项目(06JC14066),上海市登山计划重点项目(061111006)资助。
任子晖(1983-),女,博士,研究方向为CIMS、计算智能及其应用,E-mail:renzihui2006@126.com;王 坚(1961-),男,博士生导师,研究方向系统工程、仿真技术研究等。

子用其几何位置和速度向量表示，在问题的求解中，每个粒子参考自己的既定方向，所经历的最优方向和整个鸟群所公共认识到的最优方向来确定自己的飞行。

PSO 是 Kennedy 和 Eberhart[1] 于 1995 年首先提出，采用下列的公式对粒子群进行操作：

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

$$x_{id}=x_{id}+v_{id} \tag{2}$$

其中 $i=1,2,\cdots,m,d=1,2,\cdots D$；学习因子 c_1,c_2 是非负常数；r_1,r_2 是[0,1]间的随机数，$v_{id}=[-v_{\max},v_{\max}]$；$v_{\max}$ 是常数。文献[8]对式(1)作了如下的改动：

$$v_{id}=wv_{id}+c_1r_1(p_{id}-x_{id})+c_2r_2(p_{gd}-x_{id}) \tag{3}$$

其中 w 为惯性系数，为非负数，第 i 个粒子用一个 D 维的向量 $x_i=(x_{i1},x_{i2},\cdots,x_{iD})$ 表示，它在空间的飞行速度其用 $v_i=(v_{i1},v_{i2},\cdots,v_{iD})$ 表示；第 i 个粒子迄今为止搜索到的最优位置用 $p_i=(p_{i1},p_{i2},\cdots,p_{iD})$ 表示，整个粒子群迄今为止搜索到的最优位置用 $p_g=(p_{g1},p_{g2},\cdots,p_{gD})$ 表示。

迭代终止条件根据具体问题一般选为最大迭代次数或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。方程(1)和(3)是基本的 PSO 算法迭代公式。

我们引入文献[8]定义的平均聚集距离和最大聚集距离：

$$MeanDist=\frac{(\sum_{i=1}^{m}\sqrt{\sum_{d=1}^{D}(p_{id}-x_{id})^2})}{m} \tag{4}$$

$$MaxDist=\max_{i=1,2,\cdots m}(\sqrt{\sum_{d=1}^{D}(p_{id}-x_{id})^2}) \tag{5}$$

其中 m 为子群的粒子数，D 为每个粒子的维数，p_{id} 为粒子群目前搜索到的最优位置，x_{id} 为每个粒子目前搜索到的最优位置。

下面我们给出聚焦距离变化率的概念。

定义 1 粒子目前的聚焦距离的变化率定义为式(6)：

$$k=\frac{MaxDist-MeanDist}{MaxDist} \tag{6}$$

每迭代一次就计算此次我们得到的平局聚焦距离和最大聚焦距离，这样我们就得到了此次的聚焦距离的变化率，据此我们可以判断此次的粒子是应该提高其全局搜索能力还是需要提高其局部搜索能力，我们更需要对它的惯性权重进行调整。

3 随机惯性权重的构造

基本的 PSO 算法可看作是 $w=1$ 的情况，可以为 w 选取合适的值，从而使算法的全局和局部的搜索能力之间达到最佳平衡，也可以在算法迭代过程中根据不同时期搜索的进展情况动态地调节 w 的取值。为了平衡算法的全局和局部搜索能力，Shi 等[5]进一步提出了 LDIW 策略，即在迭代过程中线性地减小 w 的值，并表示为：

$$w_0=(w_{\max}-w_{\min})\left[\frac{t_{\max}-t}{t_{\max}}\right]+w_{\min} \tag{7}$$

其中 $t_{\max}$ 为最大迭代次数；t 为当前的迭代次数；$w_{\max},w_{\min}$ 分别是初始惯性权重要的最大值和最小值，$t_{\max}$ 是进化到最大迭代次数的取值。

这里，我们给出一种自适应的非线性惯性权值递减函数，具体表达式为：

$$\omega=\begin{cases}(\alpha_1+|r|/2.0)|\ln k|, & |k|>1\\ \alpha_1\alpha_2+|r|/2.0, & 0.05\leqslant|k|\leqslant1\\ (\alpha_2+|r|/2.0)\dfrac{1}{|\ln k|}, & |k|<0.05\end{cases} \tag{8}$$

其中 $\alpha_1=0.3,\alpha_2=0.2$，$r$ 为一个[0,1]间均匀分布的随机数。该选择策略即随机地选取 ω 值使 ω 随聚焦距离的变化率自适应地调整，而且改变式(7)这种单一的调节模式，使之较好地适应复杂的实际环境，从而可以更灵活地调节全局搜索与局部搜索能力。当聚焦距离变化率较大时表明粒子的最大聚焦距离和平均聚焦距离相差较大，此时粒子的全局搜索较差，故应使粒子尽快地进入全局搜索，相反我们即应该提高粒子的局部搜索能力。另外，惯性权取值的随机性在一定程度上与遗传算法的变异算子较为相似[6]，这将有助于保持种群的多样性。

4 动态自适应粒子群优化算法及其数值分析

算法 DCWPSO

Step1：随机初始化粒子群中粒子的位置与速度

Step2：将粒子的 p_b 设置为当前位置，p_g 设置为初始群体中最佳粒子的位置。

Step3：判断算法收敛准则是否满足，如果满足，转 Step5；否则，执行(4)。

Step4：对粒子群中的所有粒子，执行如下操作：

①根据式(1)，(2)，(3)更新粒子的位置与速度；

②根据式(4)，(5)，(6)计算出聚集距离的变化率，从而确定惯性权重 ω 的值。

Step5：输出 p_g，算法运行结束。

本文利用上述提出的 DCWPSO 方法对下述的 6 个函数进行了实验分析，在实验中，我们取粒子规模为 100，$c_1=c_2=2$，具体函数及参数为：

(1)Sphere 函数

$$f(X)=\sum_{i=1}^{D}x_i^2$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-100\leqslant x_i\leqslant100$，$x_{\max}=v_{\max}=100$，维数 $D=30$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=5000$。

(2)Shaffer'sf7 函数

$$f(X)=(x_1^2+x_2^2)^{0.25}[\sin^2(50(x_1^2+x_2^2)^{0.1})+1.0]$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-100\leqslant x_i\leqslant100$，$x_{\max}=v_{\max}=100$，维数 $D=30$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=2000$。

(3)Rastrigrin 函数

$$f(X)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-10\leqslant x_i\leqslant10$，$x_{\max}=v_{\max}=10$，维数 $D=30$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=1000$。

(4)Shaffer'sf6 函数

$$f(X)=0.5+\frac{(\sin\sqrt{x_1^2+x_2^2})^2-0.5}{[1.0+0.001(x_1^2+x_2^2)]^2}$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-100\leqslant x_i\leqslant 100$，$x_{max}=v_{max}=100$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=1000$。

(5)Griewank 函数

$$f(X)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}}+1$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-100\leqslant x_i\leqslant 100$，$x_{max}=v_{max}=100$，维数 $D=30$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=1000$。

(6)Rosenbrock 函数

$$f(X)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$$

其全局最优值 $f(X^*)=0$，函数计算时取 $-30\leqslant x_i\leqslant 30$，$x_{max}=v_{max}=100$，维数 $D=30$。此函数对应的程序运行的终止条件是函数值为零，若迭代次数超过 NC，认为失败。在此程序中 $NC=1000$。

本文分别用上述给出的 LDWPSO 算法和 DCWPSO 算法各自计算了 20 次，表 1－表 6 分别给出了 6 个函数在 AMD Athlon XP-M 1700+256MDDR/30G 计算机上运行计算的结果比较。

表 1　Sphere 函数(NC=5000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	2488	1952	132.0799	104.8695	100%	0
DCWPSO	1449	1434	54.5585	53.3367	100%	0

表 2　Shaffer'sf7 函数(NC=2000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	841	722	2.9042	2.4773	100%	0
DCWPSO	900	866	3.1145	2.9943	100%	0

表 3　Rastrigrin 函数(NC=1000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	70	67	4.0558	3.8856	100%	0
DCWPSO	62	59	2.3834	2.2252	100%	0

表 4　Shaffer'sf6 函数(NC=1000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	304	146	0.5908	0.2904	100%	0
DCWPSO	137	101	0.2704	0.2043	100%	0

表 5　Griewank 函数(NC=1000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	1000	1000	40.4381	40.2959	0%	2.8693×10^{-13}
DCWPSO	1000	1000	39.5769	39.3706	0%	3.0768×10^{-14}

表 6　Rosenbrock 函数(NC=1000)

Algorithm	n_{max}	n_{avg}	t_{max}	t_{avg}	κ	f_{avg}
LDWPSO	1000	1000	7.8613	7.8241	0%	8.9245
DCWPSO	1000	1000	10.4654	9.0303	0%	7.7351

上述表中 n_{max} 表示最大迭代次数，n_{avg} 表示平均迭代次数，t_{max} 表示最大迭代时间，t_{avg} 表示平均迭代时间，κ 表示搜索到最优解的成功率，f_{avg} 表示搜索得到最优值得平均值(前 3 个函数均搜索到最优值故此项均为零)。

下面我们给出当迭代次数不同时，用 LDWPSO 和 DCWPSO 方法求解上述 6 个函数，函数的平均值。

Sphere	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$	Shaffer'sf6	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$
n=100	0.5422	4.3866×10^{-22}	n=10	0.4845	0.3062
n=200	4.5483×10^{-30}	3.4976×10^{-47}	n=20	0.5171	0.1990
n=400	4.7392×10^{-60}	6.4985×10^{-86}	n=30	0.4898	3.3471×10^{-5}
n=500	4.2614×10^{-75}	3.0050×10^{-114}	n=40	0.3848	2.0473×10^{-6}
n=800	3.9937×10^{-130}	2.7821×10^{-177}	n=100	4.2581×10^{-8}	0

Shaffer'sf7	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$	Rastrigrin	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$
n=100	3.5991×10^{-5}	3.3004×10^{-8}	n=10	284.8632	166.5252
n=200	5.8284×10^{-18}	3.8421×10^{-17}	n=20	75.0527	0.1469
n=300	6.2565×10^{-31}	4.2044×10^{-27}	n=30	3.4646	5.7311×10^{-5}
n=400	5.0336×10^{-57}	2.9757×10^{-37}	n=40	4.4841×10^{-4}	6.0037×10^{-9}
n=500	0	2.0626×10^{-46}	n=100	0	0

Rosenbrock	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$	Griewank	$LDWPSOf_{avg}$	$DCWPSOf_{avg}$
n=200	8.7627	8.8667	n=200	4.5367×10^{-13}	3.8366×10^{-13}
n=400	8.7392	8.8706	n=400	1.6775×10^{-13}	4.8817×10^{-14}
n=600	9.0089	8.8351	n=600	3.9039×10^{-13}	4.0626×10^{-14}
n=800	8.6626	8.8029	n=800	1.5768×10^{-13}	5.0469×10^{-14}
n=1000	8.9245	7.7351	n=1000	2.8693×10^{-13}	3.0768×10^{-14}

下面通过图 1－图 6，我们给出了用 LDWPSO 方法和 DCWPSO 方法搜索这几个函数的极值所需迭代次数的相对比较，其中横坐标为迭代次数，纵坐标为函数适应值的对数带 * 的表示 LDWPSO 方法，带 o 表示 DCWPSO 方法。

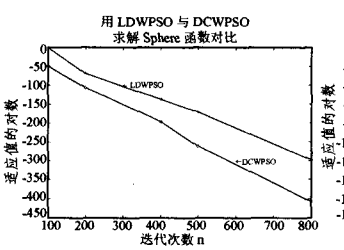

图 1　Sphere 函数的比较

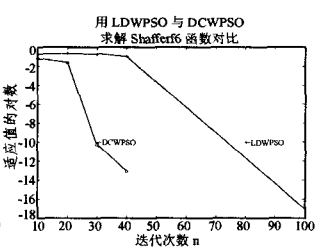

图 2　Shaffer'sf6 函数的比较

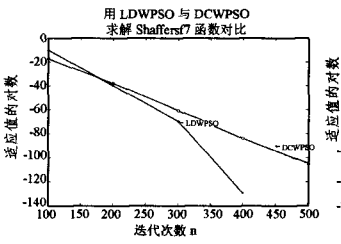

图 3　Shaffer'sf7 函数的比较

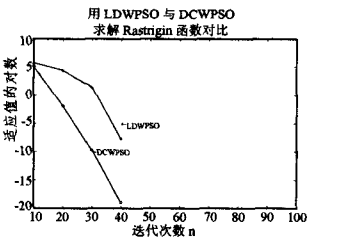

图 4　Rastriging 函数的比较

(下转第 256 页)

of International Workshop on Geometric Modeling. Tokyo. 1998:296-307

[9] Benedens O. Geometry-based watermarking of 3d models. Image Security, 1999:2-11

[10] Lee S, Kwon K. Watermark for 3d mesh model using patch CEGIs// ICCSA 2005, LNCS 3481. 2005:557-566

[11] Yin K, Pan Z, Shi J, et al. Robust mesh watermarking based on multiresolution processing. Computer & Graphics, 2001, 25: 409-420

[12] 尹康康,潘志庚,石教英.一种强壮的网格水印算法.计算机辅助设计与图形学学报,2001,13(2):102-107

[13] 周昕.三维几何模型数字水印技术及算法研究.硕士学位论文.浙江大学,2002

[14] Li Li, Zhan D, Pan Zhigeng, et al. Watermarking 3D mesh by spherical parametrization. Computer & Graphics, 2004, 28: 981-989

[15] 李黎.数字图像和三维几何模型水印技术研究.博士学位论文.浙江大学,2004

[16] Qiu J, Dai M, Bao H, et al. Watermarking 3 d mesh based on spherical wavelet trandform. International Journal of ZheJiang University, 2004, 5: 251-258

[17] Alface P R, Macq B. Shape quality measurement for 3D watermarking schemes // Electronic Imaging, Security and Watermarking of Multimedia Contents VIII. volume 6072, San Jose, California, Jan. 2006: 15-19

[18] Adelsbach A, Katzenbeisser S, Veith H. Watermarking Schemes Provably Secure Against Copy and Ambiguity Attacks // Proceedings of the 2003 ACM Workshop on Digital Rights Management. 2003: 111-119

[19] Kanungo T, Mount D M, Netanyahu N, et al. An efficient k-means clustering algorithm: Analysis and implementation. IEEE Transaction on Pattern Analysis and Machine Intelligence, 2002, 24: 881-892

[20] Renka R J. Algorithm 772: STRIPACK: Delaunay triangulation and Voronoi diagram on the surface of a sphere. ACM Transactions on Mathematical Software, 1997, 23(3): 416-434

[21] Press W H, Teulkolsky S A, Vetterling W T, et al. Numerical Recipes in C. 2nd ed. Cambridge University Press, 1996

[22] 余鄂西.矩阵论.北京:高等教育出版社,1995

[23] Faugeras O D. The representation, recognition, and locating of 3-d objects. International Journal on Robotic Research, 1986, 5: 27-52

[24] Besl P J, McKay N D. A method for registration of 3-D shape. IEEE Transaction on Pattern Analysis and Machine Intelligence, 1992, 14: 239-256

[25] Woods K. Generating ROC curves for artificial neural networks. IEEE Transactions on medical imaging, 1997, 16(3): 329-337

[26] Garland M, Heckbert P. Surface Simplification Using Quadric Error Metric // Computer Graphics (SIGGRAPH '97 Proceedings). 1997: 209-216

[27] Biermann H, Levin A, Zorin D. Piecewise smooth subdivision surfaces with normal control // Computer Graphics (SIGGRAPH'2000 Proceedings). 2000: 113-120

[28] Taubin G. A signal processing approach to fair surface design// ACM SIGGRAPH 95 Conference Proceedings. 1995: 351-358

(上接第 229 页)

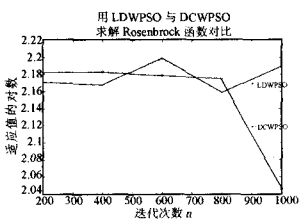

图 5 Rosenbrock 函数的比较

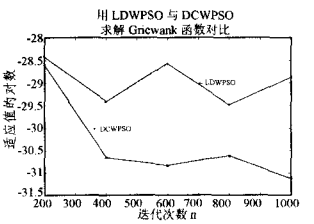

图 6 Griewank 函数的比较

从上面的表和图可以看出本文给出的 DCWPSO 方法比 LDWPSO 方法无论是在计算时间上还是在效率上都大大提高了,也改善了基本 PSO 可能出现的震荡现象。

结束语 本文提出了一种动态改变惯性权重的自适应粒子群算法(DCWPSO),在该算法中引入聚焦距离变化率的概念,并根据它对粒子群算法搜索能力的影响,将惯性因子表示为它的函数。在每次迭代时算法可根据当前粒子群聚焦距离变化率的大小动态地改变惯性权重,从而使算法具有动态自适应性。对 6 个典型函数的测试结果表明,DCWPSO 算法的收敛速度明显优于 LDWPSO 算法,收敛精度也有所提高。

参 考 文 献

[1] Kennedy J, Eberhert R. Particle swarm optimization// IEEE International Conference on Neural Networks. 1995: 1942-1948

[2] Elegbede C. Structural reliability assessment based on particles swarm optimization[J]. Structral Safety, 2005, 27(10): 171-186

[3] Pobinson J, Rahmat - Samii Y. Particle swarm optimization in electromagnetics[J]. IEEE Transactions on Antennas and Propagation, 2004, 52(2): 397-406

[4] Salman A, Ahmad I, Al-Madani S. Particle swarm optimization for task assignment problem[J]. Microprocessors and Microsystems, 2002, 26(8): 363-371

[5] Shi Y, Eberhart R. Empirical study of particle swarm optimization[A]// International Conference on Evolutionary Compution [C]. Washington, USA: IEEE, 1999, 1945-1950

[6] Shi Y, Eberhart R. Fuzzy adaptive particle swarm optimization [A]. The IEEE Congress on Evolutionary Compution[C], San Francisco, USA: IEEE, 2001: 101-106

[7] Eberhart R, Shi Y. Tracking and optimizing dynamic systems with particle swarm[A]. The IEEE Congress on Evolutionary Computatiion[C], San Francisco, USA: IEEE, 2001: 94-100

[8] 李宁,孙德宝,等.带变异的粒子群优化算法.计算机工程与应用,2004,17