一种粒子群算法的多样性策略研究*)

王 芳 雷开友 邱玉辉
(西南大学智能软件与软件工程实验室 重庆 400715)

摘 要 本文提出了一种粒子群算法的多样性策略,即在搜索过程中,对部分适应值较差的粒子重新进行随机初始化。修改后的算法经过了大量测试函数上的模拟实验验证,并与其他已有算法进行了比较。实验结果表明,该算法能获得更高的收敛成功率和质量更好的解。在困难的多峰函数优化上具有很强的竞争力。

关键词 粒子群算法,多样性,多峰函数

A Diversity Strategy for Particle Swarm Optimization

WANG Fang LEI Kai-You QIU Yu-Hui
(Intelligent Software and Software Engineering Laboratory, Southwest University, Chongqing 400715)

Abstract In this paper, a diversity strategy for Particle Swarm Optimizer is proposed. The modified algorithm re-initializes part of particles with poorer fitness during the searching process. It is empirically tested and compared with other published methods on many famous benchmark functions. The experimental results illustrate that the proposed algorithm has the potential to achieve higher success ratio and better solution quality. It is very competitive for hard multimodal function optimization.

Keywords Particle swarm optimization, Diversity strategy, Multimodal function

1 引言

粒子群算法 PSO 是一种新的进化优化方法,其基本思想受启发于鸟群捕食的社会模型[1]。尽管易于实现,PSO 算法与传统优化技术和其他元启发式方法(如遗传算法 GA、禁忌搜索算法 TS 等)相比,在困难的全局优化问题上以及广泛的工程应用领域内,都具有很强的竞争力[2~4]。

自基本 PSO 算法于 1995 年被提出以来,该领域的众多学者发展了不同的变形算法,主要研究如何避免早熟问题和加快算法的收敛速度[5],这是随机的全局搜索算法中最重要的两个问题。在文[6]中,我们研究了 TS 算法在解决 TSP 问题时的一种多样性保持策略,改进后的算法在大量基准问题上都得到了满意的实验结果。

在本文中,我们对经典的 PSO 算法 CPSO 进行了修改,对部分适应度较差的粒子重新进行随机初始化操作。我们在多个著名测试函数实施的实验结果显示,新的算法能有效避免早熟问题,显著改善解的质量,并提高收敛成功率,在复杂的多峰函数优化问题上优势尤其明显。

第 2 节,我们简略介绍了 PSO 的一些背景知识,提出的多样性策略以及实验设计在第 3 节给出,第 4 节列出相应的实验结果并作了简要分析。最后是小结和未来工作的计划。

2 PSO 算法的背景

1995 年,James Kennedy 与 Russell Eberhart 基于生物学家 Frank Heppner 的鸟群觅食模型[1]最先提出了 PSO 算法的基本思想。他们修改了 Heppner 的模型以便用于解决优化问题。在粒子群算法中,"粒子"用来表示搜索空间的一个潜在解[7],种群即为问题潜在解的集合,初始种群通常是随机产生的、解空间上的均匀分布,然后种群在解空间中搜索(进化),其过程模拟鸟群觅食的聚集过程。在进化过程中,所有个体——粒子之间进行全局信息的交换,通过这种交换,每个粒子能共享其他粒子当前的搜索结果。

在最初的 PSO 进化公式中,粒子 i 表示为 $X_i=(x_{i1},x_{i2},\cdots,x_{iD})$,代表 D 维解空间中的一个点。每个粒子记忆到目前为止自己经过的最好位置 $P_i=(p_{i1},p_{i2},\cdots,p_{iD})$,以及它在各维的飞行速度 $V_i=(v_{i1},v_{i2},\cdots,v_{iD})$。在每次迭代中,由邻域中具有最好适应度的粒子的位置 P_g 和每个粒子的当前位置 X_i 对粒子在各维上的飞行速度进行更新,然后用新的飞行速度更新粒子的位置。

标准 PSO 的进化公式表示为[8]:

$$v_{id}=w*v_{id}+c_1*\mathrm{Rand}()*(p_{id}-x_{id})+c_2*\mathrm{Rand}()*(p_{gd}-x_{id}) \quad (1)$$

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

常量 c_1 和 c_2 表示粒子受社会知识和个体认知的影响程度(也称学习率),通常设为相同值以给两者同样的权重。rand()和 Rand()是(0,1)区间的随机数。一个常量 V_{max} 被用于限制粒子的最大飞行速度。参数 w 作为惯性因子,能随时间动态调整粒子速度,使粒子逐渐趋向于局部搜索[9,10]。

为了加快收敛过程并避免出现早熟,Shi YH 等[5,6]提出将惯性权重线性减小的方法(LDW):

$$w=w_{max}-(w_{max}-w_{min})*\frac{iter}{iter_{max}} \quad (3)$$

其中,w_{max} 和 w_{min} 分别表示惯性权重 w 的最大值和最小值,$iter$ 为当前迭代次数,$iter_{max}$ 是 PSO 算法的最大迭代次数。

*)本文受到国家 863 子专题(编号 863-511-910-101-01)和重庆市信息产业局重点项目(编号 200311014)的共同支持。王 芳 助教,博士研究生。

Maurice Clerc 提出用收缩因子 K 修改的 PSO，无须使用 V_{max}，从而减少了大量的反馈作用[11]。收缩因子由以下公式计算：

$$K=\frac{2}{|2-\varphi-\sqrt{\varphi^2-4\varphi}|},\varphi=c_1+c_2,\varphi>4 \quad (4)$$

用收缩因子 K 改写的速度更新公式为：

$$v_{id}=K*(v_{id}+c_1*\mathrm{Rand}()*(p_{id}-x_{id})+c_2*\mathrm{Rand}()*(p_{gd}-x_{id})) \quad (5)$$

Carlisle 和 Dozier 研究了 PSO 算法中各种参数的影响，选择了一组较为合理的参数（c_1 为 2.8，c_2 为 1.3，种群大小 PopSize 为 30），形成规范的 PSO 算法 CPSO[12]。

张丽平等人[13]提出了一种策略（RIW），根据最佳适应值的变化率对惯性权重的期望值进行自适应调节，该算法能有效平衡全局搜索和局部搜索能力，但不适合多峰函数的优化问题，并且在适应值零点附近将会产生较大的误差。

3 本文提出的多样性策略及实验设计

我们修改了 CPSO 算法。在每一次迭代中，先将所有粒子按适应值大小排序。然后，通过引入的集中性粒子比例参数 p，将种群划分为两个子种群：集中性子群体，由前 $p*$ PopSize 个粒子组成，它们按 CPSO 原有的方式更新速度和位置；多样性子群体，包括余下的 $(1-p)*$ PopSize 个粒子，它们的适应值较差，会被算法重新进行随机初始化。

修改后的算法（解决最小化问题）流程如表 1 所示。

算法的终止条件被定义为：

$$|f(Gbest)-GM|<ErrorGoal \quad (6)$$

其中，每个目标函数的理论最优值 GM 和优化精度 ErrorGoal 在式(7)～(11)中列出。为了在(6)难以满足的情况下提早结束算法，我们还设定了最大迭代次数作为算法的另一个终止条件，在这种情况下，我们认为算法收敛失败。

为了验证提出的混合算法，并与其他已有算法进行比较，我们在大量著名测试函数[4,14]上开展了广泛的实验，这些测试函数大部分都是多峰的、异常的或非常耗时的，用现有的优化技术很难获得满意的解。本文只列出其中有代表性的部分函数的实验结果。

$$Easom:f1(x)=-\cos(x_1)\cos(x_2)e^{-((x_1-\pi)^2+(x_2-\pi)^2)},GM=-1,ErrorGoal=10^{-8} \quad (7)$$

$$Hartmann(H_{6,4}):f2(x)=-\sum_{i=1}^{4}c_i\exp(-\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2),GM=-3.32237,ErrorGoal=10^{-5} \quad (8)$$

$$Shekel(S_{4,10}):f3(x)=-\sum_{i=1}^{10}((x_i-a_i)^T(x_i-a_i)+c_i)^{-1},GM=-10.53641,ErrorGoal=10^{-5} \quad (9)$$

$$Shubert:f4(x)=\sum_{i=1}^{5}(i\cos((i+1)x_1+i))\sum_{j=1}^{5}(j\cos((j+1)x_2+j)),GM=-186.7309,ErrorGoal=10^{-7} \quad (10)$$

$$Sphere:f5(x)=\sum_{i=1}^{n}x_i^2,GM=0,ErrorGoal=10^{-8} \quad (11)$$

表 1 改进后的 PSO 算法流程

第 1 步：初始化
  对于种群中的每一个粒子 i：
    第 1.1 步：按搜索空间上的均匀分布随机初始化 X[i]；
    第 1.2 步：随机初始化 V[i]；
    第 1.3 步：计算 X[i]的目标函数值，存入 fitness[i]；
    第 1.4 步：将 Pbest[i]初始化为 X[i]；
    第 1.5 步：将 Pbest_fitness[i]初始化为 fitness[i]；
  然后：
    第 1.6 步：将 Gbest 初始化适应值最小的粒子的位置；
第 2 步：重复以下步骤直到终止条件被满足：
  第 2.1 步：对所有粒子按适应值进行排序，并根据参数 p 将粒子划分为多样性子群体和集中性子群体；
  第 2.2 步：对集中性子群体中的每一个粒子 i，根据公式 1 和 2 更新其 V[i]和 X[i]；
  第 2.3 步：对多样性子群体中的每一个粒子 i，按照第 1 步中的方法随机初始化其 V[i]和 X[i]；
  第 2.4 步：对每一个粒子 i，计算其适应值 fitness[i]；
  第 2.5 步：对集中性子群体中的每一个粒子 i，若 fitness[i]<Pbest_fitness[i]，则 Pbest[i]=X[i]，Pbest_fitness[i]=fitness[i]；
  第 2.6 步：对多样性子群体中的每一个粒子 i，令 Pbest[i]=X[i]，Pbest_fitness[i]= fitness[i]；
  第 2.7 步：更新 Gbest 为当前适应值最小的粒子的位置。

表 2 函数 $f2$ 的参数

a_{ij}						c_i	p_{ij}				
10.0	3.00	17.0	3.50	1.70	8.00	1.0	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
0.05	10.0	17.0	0.10	8.00	14.0	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3.00	3.50	1.70	10.0	17.0	8.00	3.0	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
17.0	8.00	0.05	10.0	0.10	14.0	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

表 3 函数 $f3$ 的参数

i	a_i^T				c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

我们让每个测试函数上的实验运行 200 次后取平均值，以消除不同初始种群的影响，得出更为公平的比较结论。PSO 的最大迭代次数设为 500，种群大小 PopSize 设为 30。在提出的混合算法（表示为 IADPSO）中，我们首先仔细研究了参数 p 对算法的影响，然后选择 $p=0.7$ 完成后面的比较实验。在 CFM 算法中，我们取 $c_1=c_2=2.05$。在标准的 PSO 算法中，我们取 $c_1=c_2=2.0$，$w=0.6$。在 LDW 算法中，取 $c_1=c_2=2.0$，惯性权重的初值设为 0.9，线性下降至 0.4。对于 RIW 方法，我们采用与文[13]中相同的参数。所有的算法在 Matlab7.0 下编程实现，运行环境为 Pentium IV 2.8G，512MB 内存，Windows2000 专业版操作系统。

4 实验结果

图 1～5 给出了集中性例子比例参数 p（从 0.1～1.0，间隔为 0.1）对算法的影响。从图中我们可以看到，当 p 值在 0.4 和 0.8 之间时，算法对大多数目标函数的优化在解的质量和收敛速度上都能得到较好的结果。较小的 p 值将加剧算法的随机行为，从而明显降低搜索速度，而较大的 p 值使得算法的性能趋近于经典的 PSO 算法。

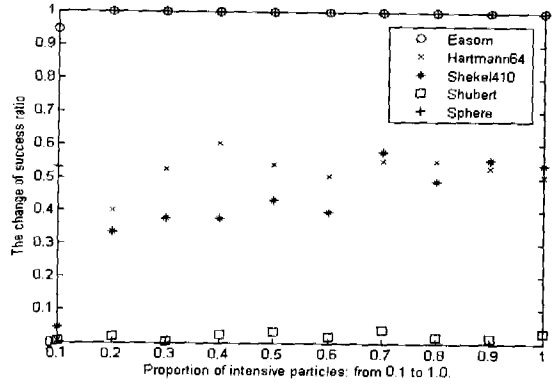

图 1　不同集中性粒子比例下的收敛成功率

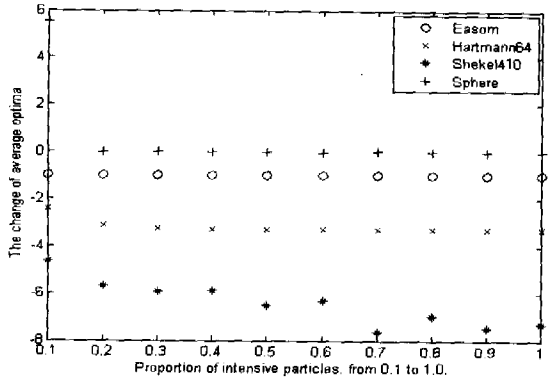

图 2　(a)不同集中性粒子比例下的平均最优值

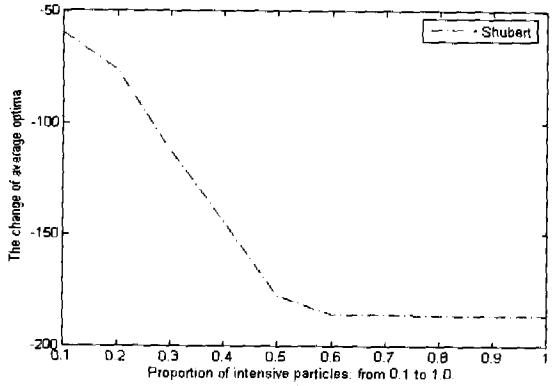

图 2　(b)不同集中性粒子比例下的平均最优值

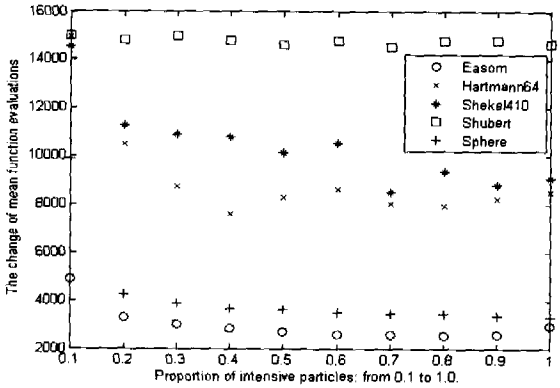

图 3　不同集中性粒子比例下的目标函数计算次数

在表 4～表 8 中，列出了每个测试函数的平均最优解、平均收敛步数、函数值平均计算次数、收敛成功率以及总的 CPU 时间。从表中可以看到，IADPSO 算法在以上性能指标的表现优于其他已有算法。在多数测试函数上(包括文中未列出的测试函数)，本文提出的改进算法都能以更少的迭代步数和目标函数值计算次数，获得质量更好的解。

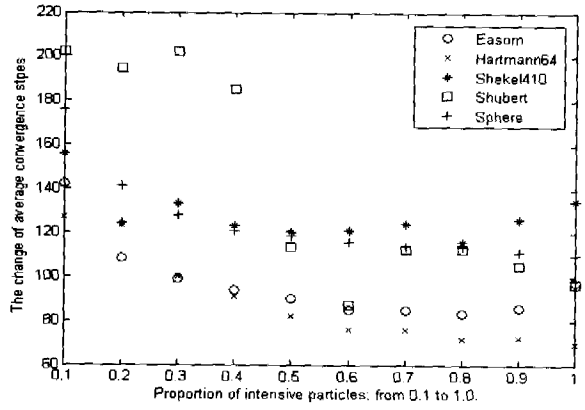

图 4　不同集中性粒子比例下的平均收敛步数

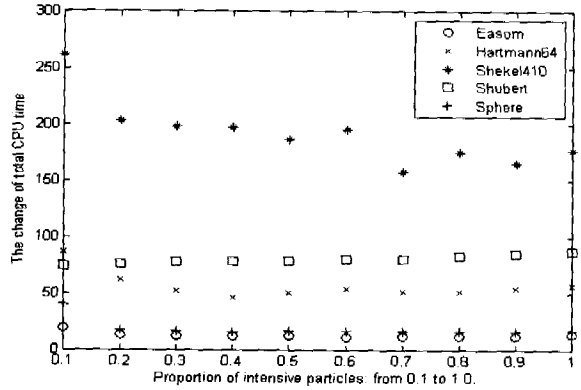

图 5　不同集中性粒子比例下的总的 CPU 时间

表 4　测试函数的平均最优解

函数	IADPSO	CPSO	SPSO	RIW	LDW	CFM
Easom	−1	−1	−0.88423	−0.98091	−1	−1
$H_{6,4}$	−3.2726	−3.2657	−3.0523	−3.2562	−3.2687	−3.2631
$S_{4,10}$	−7.6788	−6.4659	−5.1432	−7.0481	−6.1926	−7.6218
Shubert	−186.73	−186.61	−185.91	−185.3	−186.73	−186.04
Sphere	6.0405e−9	6.0442e−9	0.00395	5.7769e−9	6.0872e−9	6.056e−9

* 注：IADPSO：本文提出的算法　CPSO：经典 PSO 算法[12]
SPSO：标准 PSO 算法[8]　RIW：随机惯性权重策略[13]
LDW：权重线性下降策略[10]　CFM：带收缩因子的 PSO[11]

表 5　测试函数的平均收敛步数

函数	IADPSO	CPSO	SPSO	RIW	LDW	CFM
Easom	98.77	93.26	137.05	285.69	93.78	85.15
$H_{6,4}$	69.575	102.83	259.26	339.47	81.6	76.273
$S_{4,10}$	138.38	157.65	192.85	341.8	127.2	124.4
Shubert	87.667	132.33	150	321	103.8	112.75
Sphere	110.7	111.8	168.72	323.85	110.35	114.19

表 6　测试函数的目标函数平均计算次数

函数	IADPSO	CPSO	SPSO	RIW	LDW	CFM
Easom	2993.1	2827.8	5774.9	8761.5	2843.4	2584.5
$H_{6,4}$	7282.4	8774.5	14055	12887	8126.4	8038.5
$S_{4,10}$	8683.5	10665	12588	12799	10500	8494.5
Shubert	14659	14699	14873	14976	14733	14565
Sphere	3351	3383.8	5191.1	9745.5	3340.5	3455.8

(下转第 263 页)

样。算法用四元组集记录分解得到的所有子图。一般来说这些子图的个数越少、子图的平均节点个数越多，那么算法的效率越高。而子图的个数与子图的平均节点个数是相关的。如果子图的个数增加，那么子图的平均节点个数就会减少。反之，子图的平均节点个数就会增加。为了保证子图的平均节点个数增加，算法进行了如下的处理：

对待处理的模式图按节点数进行排序，并按照节点从小到大的顺序进行分解。这样能保证当模式图中存在"包含"关系时，一定能体现这种包含关系。

在分解一个模式图 G 时，首先在已经得到的子图中查找最大的子图 S_{max}。然后将图 G 分解为 S_{max} 和 $G-S_{max}$ 两部分。只有当找不到这样的 S_{max} 时，才将 G 随机分解为两部分。

和 Messmer 算法相比，分解算法增加了这样一种情况：S_{max} 与 *graph* 的节点数相同，但是 S_{max} 的边数()小于 *graph* 的边数。此时，求属于 G 而不属于 S_{max} 的边集 E，从 S_{max} 中删除这些边，并分解 *graph*，增加四元组(*graph*，S_{max}，NULL，E)。这样既能保证子图节点个数的最大化，又能解决引言中提到的问题。实验结果也反映了这一优点。

参 考 文 献

1 Ullman J R. An Algorithm for Subgraph Isomorphism. J. of the Assoc. for Computing Machinery, 1976,23(1):31～42

2 Cordella L P, Foggia P, Sansone C, et al. Evaluating Performance of the VF Graph Matching Algorithm. In: Proc. of the 10th Intl. Conf. on Image Analysis and Processing, IEEE Computer Society Press, 1999. 1172～1177

3 Kuner P, Ueberreiter B. Pattern Recognition by Graph Matching Combinatorial versus Continous Optimization. Int'l J. Pattern Recognition and Artificial Intelligence, 1988,2(3):527～542

4 Messmer B T, Bunke H. Efficient Subgraph Isomorphism Detection: A Decomposition Approach. IEEE Tranactions on Knowledge and Data Engineering, 2000,12(2):307～323

(上接第 215 页)

表 7 测试函数的收敛成功率

函数	IADPSO	CPSO	SPSO	RIW	LDW	CFM
Easom	1	1	0.85	0.975	1	1
$H_{6,4}$	0.6	0.525	0.135	0.445	0.55	0.55
$S_{4,10}$	0.585	0.425	0.265	0.47	0.405	0.58
Shubert	0.03	0.03	0.015	0.01	0.025	0.04
Sphere	1	1	0.99	1	1	1

表 8 测试函数的总的 CPU 时间

函数	IADPSO	CPSO	SPSO	RIW	LDW	CFM
Easom	10.328	10.344	19.594	32.844	10.313	11.438
$H_{6,4}$	38.906	48.656	73.875	71.656	45.141	51
$S_{4,10}$	153.94	190.75	222.19	229.73	188.25	157.83
Shubert	66.625	71.531	68.375	75.094	70.578	80.484
Sphere	12.266	13.031	18.688	38.047	12.875	16.047

小结与未来工作 在本文中，我们提出了一种用于 PSO 算法的新的多样性策略，该策略的有效性在大量著名测试函数的优化上得到了验证。充分的模拟实验结果显示，基于多样性策略的混合算法在困难的多峰函数数优化问题上相比其他已有方法具有很强的竞争力。未来的工作将集中于分析集中性例子比例参数 p 的影响，对混合方法在高维函数上的性能进行研究，开展与其他多样性策略比较实验，并考虑将算法扩展至有约束多目标优化问题。

参 考 文 献

1 Kennedy J, Eberhart R C. Particle swarm optimization. In: Proc. of IEEE Intl. Conf. on Neural Networks, Piscataway, NJ, IEEE Press, 1995. 1942～1948

2 Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 2002,6(1):58～73

3 Hu X, Eberhart R C, Shi Y H. Engineering optimization with particle swarm. In: Proc. of the IEEE Swarm Intelligence Symposium, Indianapolis, Indiana, USA, 2003. 53～57

4 Fan S-K S, Liang Y-C, Zahara E. Hybrid Simplex Search and Particle Swarm Optimization for the Global Optimization of Multimodal Functions, Engineering Optimization, 2004,36(4):401～418

5 Parsopoulos K E, Vrahatis M N. Recent approaches to global optimization problems through particle swarm optimization. Natural Computing, 2002,1:235～306

6 Lei Kaiyou, Wang Kaiyou, et al. A Novel Forward Search Strategy to Automatically Harmonize Intensification and Diversification in Tabu Search. In: Proc. of Intl. Symposium on Autonomous Decentralized Systems, Chengdu, China, 2005. 513～519

7 Kennedy J, Eberhart R C. Swarm Intelligence. Morgan: Kaufmann Publishers, 2001

8 Shi Y H, Eberhart R C. A modified particle swarm optimizer. In: Proc. of the IEEE Congress on Evolutionary Computation, Piscataway, N J, IEEE Press, 1998. 69～73

9 Shi Y H, Eberhart R C. Parameter selection in particle swarm optimization. Evolutionary Programming VII. In: Proceedings of the Seventh Annual Conference on Evolutionary Programming, New York, 1998. 591～600

10 Shi Y H, Eberhart R C. Empirical study of particle swarm optimization. In: Proc. of the IEEE Congress on Evolutionary Computation, Piscataway, NJ, IEEE Press, 1999. 1945～1950

11 Clerc M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: Proc. of the IEEE Congress on Evolutionary Computation, 1999. 1951～1957

12 Carlisle A, Dozier G. An off-the-shelf PSO. In: Proc. of the Workshop on Particle Swarm Optimization, Indianapolis, USA, 2001. 1～6

13 张丽平，俞欢军，陈德钊，胡上序. 粒子群优化算法的分析与改进. 信息与控制，2004,33(5):513～517

14 Levy A, Montalvo A, Gomez S, et al. Topics in Global Optimization. New York: Springer-Verlag, 1981