DOI:10.11721/cqnuj20150504

一种改进的粒子群优化算法*

杨华芬[1], 董德春[1], 杨丽华[2], 李 丽[1]
(1.曲靖师范学院 计算机科学与工程学院;
2.曲靖师范学院 数学与信息科学学院, 云南 曲靖 655011)

摘要:针对粒子群优化算法中粒子容易聚集和收敛速度慢,提出一种改进的粒子群优化算法。该算法同时考虑到粒子进化的成功率和多样性程度对算法寻优性能的影响,当粒子集聚程度较高时,增大惯性权值,提高算法的全局搜索能力。为平衡算法全局和局部寻优能力,当进化速度较快时,提高算法局部搜索能力,以免错过较好的位置。在速度更新中,引入较差粒子,避免算法再次去搜索这些较差的位置,降低算法的搜索效率。将该算法用于优化 6 个经典测试函数,实验表明:该算法不仅可以平衡局部和全局的搜索能力,而且可以提高算法的搜索效率和精度。

关键词:粒子群优化;进化速度;集聚度;速度更新
中图分类号:TP183 **文献标志码**:A **文章编号**:1672-6693(2015)05-0114-05

James Kennedy 等人对鸟类、鱼类群集活动和捕食活动时形成的协同智能进行模拟,提出一种进化算法——粒子群优化算法(Particle Swarm Optimization,PSO)[1]。粒子群算法能快速收敛、需要设置的参数少,具有较强的全局搜索能力,因此,常用于求解多目标优化、不光滑和多峰值的问题。在 PSO 算法的可调整参数中,惯性权值是最重要的参数:权值越大,算法全局搜索能力越强;权值越小,全局搜索能力越弱。为了较好地平衡全局和局部搜索能力,很多学者进行了大量的研究工作[2]。文献[3-4]将惯性权重因子引入到粒子群算法中。文献[5-6]提出基于收缩因子和自适应变异算子的粒子群算法。文献[7]提出线性递减 PSO 算法,文献[8]提出模糊自适应粒子群算法。文献[9]将距离信息方法引入到 PSO 中。文献[10]提出带压缩因子的 PSO 算法。文献[11]提出自适应变异 PSO 算法。文献[12-15]将 PSO 算法和其他算法混合。目前对粒子群算法的改进主要从两个方面考虑[16-17]:1)设置参数;2)修改粒子结果和运动轨迹。目的是为了提高局部搜索效率或解决熟收敛速度慢等问题,最终能同时提高收敛速度和精度,提高算法性能。上述改进算法在性能和效率上有不同程度的改善,但难以同时避免“早熟收敛”和提高算法的搜索精度。

为提高粒子群算法的收敛速度和寻优精度,本文对惯性权重进行改进,提出一种克服粒子早熟且快速收敛的改进 PSO 算法。在改进惯性权重的同时还考虑粒子的聚集程度和算法的搜索速度:粒子的多样性越低,陷入局部最优的可能性越大,此时,应增大惯性权重;粒子的多样性越高,聚集程度越低,此时,应降低惯性权重,避免算法变成随机搜索。在粒子学习过程中,不仅要向群体找到的最优个体和自身找到的最优个体学习,还应该记住曾经找到的较差位置,在下一次更新时避免粒子再次飞向此类位置。该算法能较好地避免陷入局部最优,在收敛速度和精度上均有所提高。

1 改进的粒子群算法

基本粒子群算法中粒子位置和速度的更新公式如下:

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t), \tag{1}$$

$$v_{id}(t+1)=\omega v_{id}(t)+c_1r_1(p_{id}(t)-x_{id}(t))+c_2r_2(p_{gd}(t)-x_{gd}(t))。 \tag{2}$$

从(1),(2)式可以看出:3 部分决定粒子的速度:一是惯性权重,体现最新速度和原来速度之间的关系,权重

* **收稿日期**:2014-08-31 **修回日期**:2015-04-20 **网络出版时间**:2015-05-15 12:45
资助项目:云南省自然科学基金(No. 2013FZ098);云南省自然科学基金(No. 2013FZ114);曲靖师范学院科研基金资助项目(No. 2009MS006)
作者简介:杨华芬,女,副教授,研究方向为人工智能,E-mail:sunnyyhf@126.com;**通信作者**:董德春,副教授,E-mail:sunnyyhf@126.com
网络出版地址:http://www.cnki.net/kcms/detail/50.1165.n.20150515.1245.027.html

越大，全局搜索能力越强；二是学习因子 c_1 决定粒子的搜索能力；三是学习因子 c_2 体现粒子之间的信息共享能力。合理调整学习因子，可以较好地平衡算法求精和求泛的能力。本文从改进惯性权值和速度更新两方面对粒子群算法进行改进。

1.1 改进的惯性权值策略

惯性权重对算法寻优的效果影响较大，权重应随着进化的成功率和多样性程度实时更新。在搜索过程中，各个粒子的搜索速度不全相同，为让权重随着粒子的搜索速度和多样性程度变化，本文采用动态惯性权重。

$$s_i^t=\left|\frac{\min(f(p_{\text{best}\,i}^{\,t-1}),f(p_{\text{best}\,i}^{\,t}))}{\max(f(p_{\text{best}\,i}^{\,t-1}),f(p_{\text{best}\,i}^{\,t}))}\right|, \tag{3}$$

$$c^t=\sqrt{\frac{(\sum_{i=1}^{n}(|f_i-\bar{f}_t|^2))}{n}}, \tag{4}$$

其中 s_i^t 为第 t 代的种群的第 i 个粒子进化速度因子，$f(p_{\text{best}\,i}^{\,t-1})$ 为到第 $t-1$ 代为止第 i 个粒子最佳位置的适应度，f_i 为第 i 个粒子的适应度，$\bar{f}_t$ 为第 t 代群体的平均适应度，n 为种群规模，c^t 为第 t 代种群的集聚度因子。

随着 c^t 的增大，粒子的集聚度越高，算法越容易陷入局部最优；c^t 越小，粒子的聚集程度越低，算法全局搜索能力越强。为了防止算法陷入局部最优，应增大惯性权重。为防止算法变成随机搜索，应减小惯性权重。将惯性权重表示为 c^t 和 s_i^t 的函数可以让权重自适应变化，具体如(5)式所示。

$$w_i^t=f(c^t,s_i^t)=w_{ini}-s_i^t w_v+c^t w_c, \tag{5}$$

其中 w_{ini} 为 w 的初始值，通常 $w_{ini}=1$；w_v 和 w_c 分别为 s_i^t 和 c^t 的比例因子。由于 $0<s_i^t\leqslant 1$，$0<c^t\leqslant 1$，因此 $w_{ini}-w_v<w<w_{ini}+w_c$。从(5)式可以看出 w_i^t 随着 s_i^t 增大而减小，随着 c^t 的增大而增大。

1.2 速度更新方法

从传统的粒子群算法可以看出，一个粒子的速度受到 3 个因素的影响：惯性权重 w，粒子个体的加速权重系数 c_1 和粒子群体的加速权重系数 c_2。从速度更新公式可以看出：①如果没有粒子个体和粒子群体的加速权重系数，即 $c_1=0$，$c_2=0$，粒子将一直以当前的速度飞行，直到飞至边界，此时，搜索范围非常有限，很难找到最优解；②如果惯性权重 $w=0$，粒子的速度由当前位置和它找到的最好位置决定，不能记住原来的速度，则容易改变速度，但难以把握飞行的方向，此时，粒子凭借自己和群体好的经验，将飞向它自身的最好位置和全体的最好位置的加权中心，有利于深度搜索；③如果删除速度更新公式的第二部分，即 $c_1=0$，则粒子没有自身学习能力，只有社会的认知能力，在粒子的相互影响下，可以达到新的搜索区域，此时收敛速度快，但是容易陷入局部最优；④如果删除速度更新公式的第三部分，即 $c_2=0$，则粒子缺乏社会认知能力，没有信息共享，一个群体的行为等于该群体中各个个体独立运动，难以寻找到最优解。

为了加快算法的收敛速度，提高算法的性能，粒子不仅应该记住自己好的经验和群体好的学习经验，还应该总结学习的教训(不好的经验)，在学习过程中尽量远离这些较差的位置，因此本文提出新的速度更新公式。

$$v_{id}(t+1)=\omega v_{id}(t)+c_1 r_1(p_{id}(t)-x_{id}(t))+c_2 r_2(x_{id}(t)-p_{\text{worst}}(t))+c_3 r_3(p_{gd}(t)-x_{gd}(t)), \tag{6}$$

其中 c_1，c_2 和 c_3 为加速度系数，c_1 加快粒子飞向自己的最佳位置，c_2 加快粒子远离自己曾经找到的最差位置，c_3 加快粒子飞向群体的最优位置，p_{worst} 是粒子 i 的最差位置。

2 算法描述及仿真实验

2.1 改进的算法流程

本文提出一种改进的粒子群算法，该算法在搜索过程中，惯性权重 w 的值随着 s_i^t 和 c^t 实时调整，将 w_v 和 w_c 均初始化为 0。算法的具体步骤如下：

步骤 1　初始化粒子的速度和位置，并计算所有粒子的适应度；

步骤 2　设置粒子的全局最优位置和个体最优位置；

步骤 3　如果算法满足结束要求，则转步骤 7，否则执行步骤 4；

步骤 4　根据(3)、(4)和(5)式计算进化速度因子 s_i^t 和聚集度因子 c^t；

步骤 5　分别根据(6)式和(1)式修改粒子的速度和位置，计算每一个粒子的适应度，修改粒子的全局最优位置和个体最优位置；

步骤6　修改迭代次数,并转步骤3;

步骤7　输出最优位置,结束算法。

改进算法同时考虑粒子的进化速度和多样性程度对算法寻优性能的影响:粒子的多样性越低,陷入局部最优的可能性越大,此时,应提高惯性权值,增强粒子的多样性,扩大算法寻优的范围。本文采用适应度方差来度量群体的多样性:随着方差的增大,粒子的多样性越好;随着方差的减小,粒子的多样性越差。为平衡算法求精和求泛的能力,进化速度越快,全局搜索能力越弱,避免算法错过较好的位置。考虑到各个粒子的进化速度不尽相同,因此,本文为每一个粒子提出自己的进化速度。粒子的学习过程中,不但要记住自身曾经找到的最佳位置和群体找到的最佳位置,而且还应该记住学习的教训。在后期的学习过程中,远离这些较差的位置,提高学习的效率,因此,在更新速度时,记住较差的粒子。本文算法不仅能够平衡算法求精和求泛的能力,而且可以提高算法的搜索效率,防止算法飞向曾经经过的最差位置。

2.2 仿真实验

2.2.1 实验参数的讨论　本文算法中引入的进化速度比例因子速度 w_v 和集聚度比例因子 w_c 对算法的性能影响较大。通过多次实验比较得到,当 w_c 的取值在0.4～0.6之间,w_v 的取值在0.05～0.2之间,算法的效果较好。用同样的实验方法确定 c_1,c_2 和 c_3 的取值,$c_1=2.0$,$c_2=0.000\ 16$ 和 $c_3=0.5$ 时算法性能较好。

2.2.2 实验结果　为验证本文算法的有效性,与文献[18]的APSO-VI算法、文献[19]的DDEPSO算法和文献[20]的MPSO算法进行比较。采用6个典型测试函数进行实验,分别为:Sphere,Rastrigrin,Branin,Ackly,Noisy Quadric和Schwefel函数,维度都为30。这6个函数的搜索区域分别为:$[-10,10]^{30}$,$[-5.12,5.12]^{30}$,$x_1\in[-5,10]$,$x_2\in[0,15]$,$[-32,32]^{30}$,$[-1.28,1.28]^{30}$,$[-10,10]^{30}$。Branin函数的最优值为0.397 887,其余函数的最优值为0,最优点都为$[0,\cdots,0]$。每一种算法都取函数值作为个体适应度值,对这6个函数进行寻优过程中,每一代的最优个体适应度变化情况分别如图1、图2、图3、图4、图5和图6所示。4种算法的粒子群体规模都为100,最大进化代数为1 000,为消除初始数据随机特性影响,优化每一个函数时,4种算法都采用同一组粒子进行实验。

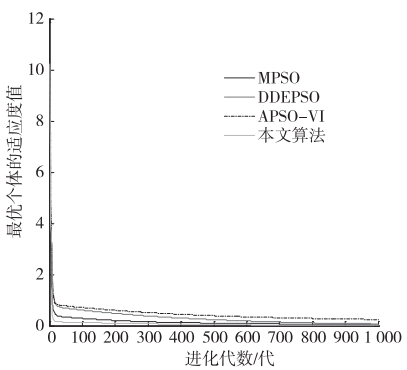

图1　优化Sphere函数所得最优粒子适应度变化曲线

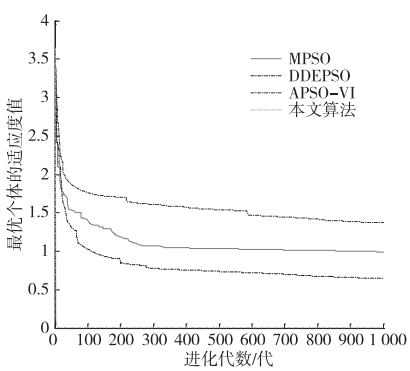

图2　优化Rastrigrin函数最优粒子适应度变化曲线

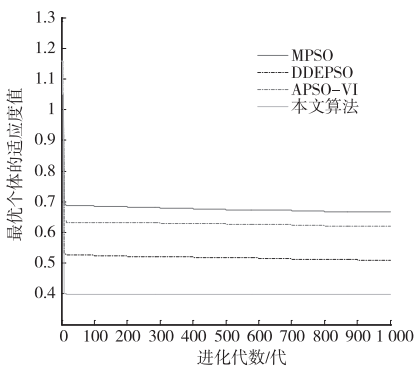

图3　优化Branin函数最优粒子适应度变化曲线

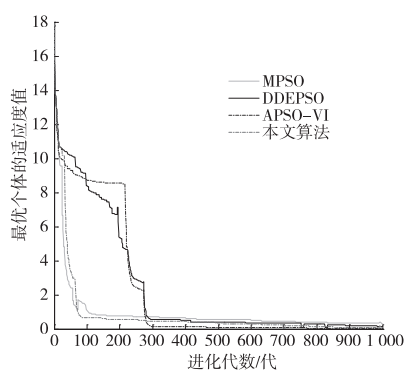

图4　优化Ackly函数最优粒子适应度变化曲线

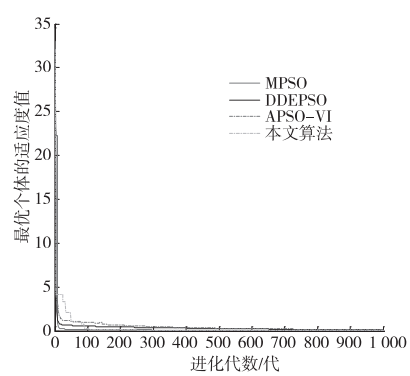

图5　优化Noisy Quadric函数最优粒子适应度变化曲线

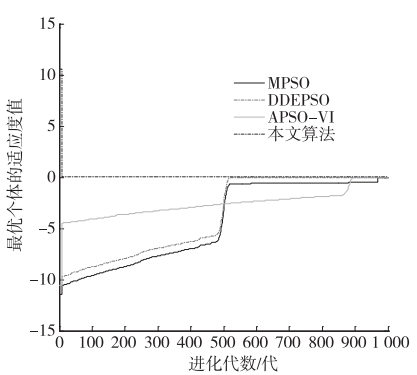

图6　优化Schwefel函数最优粒子适应度变化曲线

从图中可以看到4种算法对Sphere和Noisy Quadric函数寻优时,情况差不多,每一种算法进化1 000代找

到的最优值相差不是很大，收敛速度没有明显差异。对 Rastrigrin 和 Branin 函数寻优时，DDEPSO，APSO-VI 和 MPSO 算法收敛速度较慢，而且难以找到最优值。对 Ackly 函数寻优时，APSO-VI 算法大概在第 100～240 代之间陷入局部最优，但是后期的收敛速度较快；DDEPSO 算法在寻优过程中出现一点小的振荡，总体上收敛速度较慢；本文算法和 MPSO 算法整个进化过程比较类似，而且比另外两种算法进化速度快。对 Noisy Quadric 函数寻优时，DDEPSO，APSO-VI 和 MPSO 前期的进化速度较快，但后期难以找到最优值，而本文算法前期收敛速度相对较慢，最终找到的最优值 0.011 135 04 比其他 3 个函数要好。对 Schwefel 函数寻优时，4 种算法都能找到最优值，MPSO 和 DDEPSO 算法进化过程非常相像，本文算法进化速度较快。

4 种算法对 6 个函数寻优 1000 代，最终找到的最优个体和最优个体的平均适应度值如表 1 所示。从表中可以看出对 Sphere 函数寻优时，用文献[18-20]算法得到的平均值和最优值相差不大，而本文算法得到值较为接近全局最优值。对 Rastrigrin 和 Branin 函数寻优时，本文算法的寻优速度较快，找到的最优值比另外 3 种算法要好得多。对 Ackly 函数寻优时，本文算法找到的最优值稍微比 APSO-VI 算法找到略差一点。对 Noisy Quadric 函数和 Schwefel 函数寻优时，本文算法进化速度相对较快。由此可以看出，采用自适应的方法改变权值并记录下个体曾经找到的最差值对算法跳出局部极值，提高算法性能效果较好。

表 1　4 种算法用于 6 个函数进化 1 000 代后得到的最优值和平均最优适应度值

算法 / 函数	DDEPSO		MPSO		APSO-VI		本文算法	
	平均值	最优值	平均值	最优值	平均值	最优值	平均值	最优值
Sphere	0.332 248 92	0.066 899 06	0.166 132 41	0.049 496 86	0.477 877 84	0.238 774 73	0.078 898 49	0.008 362 38
Rastrigrin	0.822 191 12	0.643 398 80	1.126 313 55	0.991 796 20	1.572 982 23	1.369 067 90	0.085 535 72	0.000 000 74
Branin	0.521 233 89	0.509 089 17	0.678 113 89	0.666 187 16	0.629 987 89	0.619 089 17	0.402 393 89	0.398 089 17
Ackly	2.298 925 33	0.156 808 19	0.943 040 56	0.303 630 16	0.209 256 02	0.026 483 80	0.817 102 83	0.053 630 16
Noisy Quadric	0.392 763 90	0.128 487 90	0.317 181 05	0.033 949 18	0.518 251 00	0.151 884 49	0.275 176 13	0.011 135 04
Schwefel	−4.404 893 87	0.000 000 00	−3.751 293 87	0.000 000 00	−2.613 756 54	0.000 000 00	0.084 681 84	0.000 000 00

3 结论

在 PSO 算法中，惯性权值对算法的性能影响较大。本文同时考虑粒子的多样性程度和进化速度，提出权重修改方法，让权重随着算法的搜索状态动态变化，实时改变粒子的搜索范围，以便让粒子具有较好的平衡求精和求泛的能力。粒子在进化过程中不仅记下自己曾经找到的最优值和群体找到的最优值，而且还记录了自己找到的最差值，在后期的学习中要远离这些较差的位置。为验证本文算法的有效性，对几个经典的测试函数进行优化。从实验结果看出，本文算法不仅能保持传统 PSO 算法的简单、快速的特点，还能避免陷入局部最优，提高算法的搜索效率和精度。

参考文献：

[1] Kennedy J，Eberhart R. Particle swarm optimization[C]// Proceedings of 1995 IEEE，international conference on neural networks. Perth，Australia：IEEE Computational intelligence Society，1995：1942-1948.

[2] Akay B. A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding[J]. Applied Soft Computing，2013，13：3066-3091.

[3] Zhan Z H，Zhang J，Li Y，et al. Adaptive particle swarm optimization[J]. IEEE Transactions on Systems Man and Cybernetics Part B-cybernetics，2009，39：1362-1381.

[4] 赵志刚，黄树运，王伟倩. 基于随机惯性权重的简化粒子群优化算法[J]. 计算机应用研究，2014，31(2)：361-363.
Zhao Z G，Huang S Y，Wang W Q. Simplified particle swarm optimization algorithm based on stochastic inertia weight[J]. Application Research of Computers，2014，31(2)：361-363.

[5] 申元霞，王国胤，曾传华. 相关性粒子群优化模型[J]. 软件学报，2011，22(4)：695-708.
Shen Y X，Wang G Y，Zeng C H. Correlative particle swarm optimization model[J]. Journal of Software，2011，22(4)：695-708.

[6] 高卫峰，刘三阳. 一种高效粒子群优化算法[J]. 控制与决策，2011，26(8)：1158-1162.
Gao W F，Liu S Y. An efficient particle swarm optimization [J]. Control and Decision，2011，26(8)：1158-1162.

[7] Khare A，Rangnekar S. A review of particle swarm optimization and its applications in solar photovoltaic system[J]. Applied Soft Computing，2013，13：2997-3006.

[8] Valdez F, Melin P, Castillo O. An improved evolutionary method with fuzzy logic for combining particle swarm optimization and genetic algorithms[J]. Applied Soft Computing, 2011, 11: 2625-2632.

[9] Marinakis Y, Marinaki M. Particle swarm optimization with expanding neighborhood opology for the permutation flowshop scheduling problem[J]. Soft Computing, 2013, 17: 1159-1173.

[10] 王晓佳,张宝霆,徐达宇. 含有压缩因子的粒子群优化灰色模型在智能电网中的应用[J]. 运筹与管理, 2012, 21(3): 114-118.
Wang X J, Zhang B T, Xu D Y. Compression factor with gray model of particle swarm optimization and its application in smart grid[J]. Operations Research and Management Science, 2012, 21(3): 114-118.

[11] 聂瑞,章卫国,李广文,等. 一种自适应混合多目标粒子群优化算法[J]. 西北工业大学学报, 2011, 29(5): 695-701.
Nie R, Zhang W G, Li G W, et al. A more useful AH-MOPSO (Adaptive Hybrid Multi-Objective Particle Swarm Optimization) algorithm[J]. Journal of Northwestern Polytechnical University, 2011, 29(5): 695-701.

[12] Sinha A K, Zhang W J, Tiwari M K. Co-evolutionary immuno-particle swarm optimization with penetrated hypermutation for distributed inventory replenishment[J]. Engineering Applications of Artificial Intelligence, 2012, 25: 1628-1643.

[13] Cheng M Y, Huang K Y, Chen H M. K-means particle swarm optimization with embedded chaotic search for solving multidimensional problems[J]. Applied Mathematics and Computation, 2012, 219: 3091-3099.

[14] Li M W, Kang A G, Zhou P F, et al. Hybrid optimization algorithm based on chaos, cloud and particle swarm optimization algorithm[J]. Journal of Systems Engineering and Electronics, 2013, 24: 324-334.

[15] 于海平,刘会超,吴志健. 基于模拟退火的自适应粒子群优化算法的改进策略[J]. 计算机应用研究, 2012, 29(12): 4448-4450.
Yu H P, Liu H C, Wu Z J. Strategy of adaptive simulated annealing particle swarm optimization algorithm[J]. Application Research of Computers, 2012, 29(12): 4448-4450.

[16] 贾文生,向淑文,杨剑锋,等. 基于免疫粒子群算法的非合作博弈 Nash 均衡问题求解[J]. 计算机应用研究, 2012, 29(1): 28-31.
Jia W S, Xiang S W, Yang J F, et al. Solving Nash equilibrium for N-persons'non-cooperative game based on immune particle swarm algorithm[J]. Application Research of Computers, 2012, 29(1): 28-31.

[17] Liu B, Wang L, Jin Y Y. Improved particle swarm optimization combined with chaos[J]. Chaos Solitons & Fractals, 2005, 25: 1261-1271.

[18] Gang X. An adaptive parameter tuning of particle swarm optimization algorithm[J]. Applied Mathematics and Computation, 2013, 219: 4560-4569.

[19] Nezami O M, Bahrampour A, Jamshidlou P. Dynamic diversity enhancement in particle swarm optimization (DDEPSO) algorithm for preventing from premature convergence[J]. Procedia Computer Science, 2013, 24: 54-65.

[20] Hung J C. Modified particle swarm optimization structure approach to direction of arrival estimation[J]. Applied Soft Computing, 2013, 13: 3153-3172.

An Improved Particle Swarm Optimization Algorithm

YANG Huafen[1], DONG Dechun[1], YANG Lihua[2], LI Li[1]

(1. Department of Computer Science and Engineering, Qujing Normal College; 2. School of Information Science and Engineering, Department of Mathematics and Information Science, Qujing Yunnan 655011, China)

Abstract: To overcome the problem of loss of diversity and poor convergence, an improved particle swarm optimization algorithm is proposed in this paper. The effect of agglomeration degree and evolution velocity on the optimization ability of algorithm is considered in the improved algorithm. To improve the global searching capacity of the presented algorithm, the inertia weight increases when agglomeration of particles is high. In order to balance global and local optimization ability of algorithm, local optimization ability should be increased when algorithm has higher evolution velocity, so as not to miss a good location. To avoid algorithm to search these poor location repeatedly and increase the search efficiency of algorithm, worst particle is introduced in velocity updating formula. To verify the validity of algorithm, six classical test functions are optimized by the improved algorithm proposed in this paper. The results show that the proposed algorithm can not only balance the global and local search ability, but also improve the search efficiency and accuracy of the algorithm.

Key words: particle swarm optimization; evolution velocity; agglomeration degree; velocity updating

(责任编辑 游中胜)