文章编号:1000－2367(2017)06－0091－09 DOI:10.16366/j.cnki.1000－2367.2017.06.015

融合榜样学习和反向学习的粒子群优化算法

张新明[a,b], 王霞[a], 涂强[a], 康强[a]

(河南师范大学 a.计算机与信息工程学院;b.计算智能与数据挖掘河南省高校工程技术研究中心,河南 新乡 453007)

摘 要:为了提高粒子群优化算法(Particle swarm optimization, PSO)的优化效率,降低其陷入局部最优的概率,提出了一种融合榜样学习和反向学习的 PSO 算法(PSO based on combing Example learning and Opposition learning,EOPSO).首先,对粒子群中的非最优粒子采用新颖的榜样学习机制更新,以便提高全局搜索能力,避免算法陷入局部最优;其次,对粒子群中最优粒子采用反向学习混合机制更新,提升该粒子的搜索能力,进一步避免算法陷入局部最优;最后,对粒子群中的最优粒子还采用了自身变异机制更新,有利于搜索前期的全局搜索和后期的快速收敛.在 15 个不同维度的基准函数上进行了仿真实验,实验结果表明,与最先进的 PSO 改进算法 ELPSO、SRPSO、LFPSO、HCLPSO 相比,EOPSO 优化性能更好.

关键词:智能优化算法;粒子群优化算法;榜样学习;反向学习

中图分类号:TP181 **文献标志码**:A

粒子群优化(Particle swarm optimization,PSO)算法作为最重要的自然启发式算法之一,是 1995 年 Kennedy 和 Eberhart[1－2]首次提出的.因 PSO 收敛速度快、易于实施、操作简单等优点而在多个领域得到了广泛的应用[3－5].但是,经过学者研究发现,PSO 仍然存在早熟收敛、易陷于局部最优、种群多样性丢失等问题.一般来说,一个优化算法找到问题的最优解主要依靠探索和开采两种能力.其中探索即全局搜索,就是在整个搜索空间中寻找有希望的区域;开采即局部搜索,就是在所识别的有希望的区域中微调搜索最优解.当探索和开采达到适当平衡时,可以使算法获得良好的优化性能.因此,在基于种群的进化算法中,重要的是获得探索和开采之间的平衡.

因此许多学者对 PSO 进行改进以便提高算法的优化性能[6－8].其中不乏借助不同的学习策略来达到算法中探索和开采之间的平衡.文献[9]针对 PSO 的早熟收敛问题,引进 5 次连续变异策略,提出了一种增强领导者的 PSO 算法(Enhanced leader PSO,ELPSO).文献[10]将 PSO 与两种学习策略进行结合实现算法的快速收敛,提出了一种自我调节的 PSO 算法(Self-Regulating PSO,SRPSO).文献[11]结合 PSO 和 Levy flight,提出了一种 Levy flight 的 PSO 算法(PSO with Levy flight,LFPSO).文献[12]将 PSO 与综合学习策略相结合以增强算法的探索与开采能力,提出了一种异构综合学习的 PSO 算法(Heterogeneous Comprehensive Learning PSO,HCLPSO).文献[13]通过反向学习与局部学习的协同行为,提出了一种具有反向学习和局部学习能力的 PSO 算法.文献[14]通过使用含有多种全局最优粒子的榜样集来更新粒子的位置,提出了基于榜样学习的 PSO 算法.这些改进算法在不同程度上提高了 PSO 的优化性能,但在解决某些高维或者复杂的优化问题时仍然存在全局搜索能力不足、搜索效率低等缺点.针对 PSO 存在的多样性缺失、易陷于局部最优的问题,本文提出了一种融合榜样学习和反向学习的 PSO 算法(PSO based on combing Example learning and Opposition learning,EOPSO).

1 粒子群优化算法

PSO 是一种基于种群的优化算法.PSO 采用两个最优位置:单个粒子的最优位置(p_{best})和粒子群的最

收稿日期:2017-05-09;**修回日期**:2017-09-12.

基金项目:河南省重点科技攻关项目(132102110209);河南省基础与前沿技术研究计划项目(142300410295).

作者简介(通信作者):张新明(1963－),男,湖北孝感人,河南师范大学教授,研究方向为智能优化算法、数字图像处理和模式识别,E-mail:xinmingzhang@126.com.

优位置(g_{best})进行更新.PSO 中每个粒子包含两个属性:速度向量和位置向量.设 D 维空间中粒子 i 的速度向量和位置向量分别为 $V_i=(V_i^1,V_i^2,\cdots,V_i^D)$ 和 $X_i=(X_i^1,X_i^2,\cdots,X_i^D)$,则其更新公式为:

$$V_i^d=V_i^d+c_1\times r_1\times(p_{\text{best}i}^d-X_i^d)+c_2\times r_2\times(g_{\text{best}}{}^d-X_i^d),\tag{1}$$

$$X_i^d=X_i^d+V_i^d,\tag{2}$$

其中,c_1 和 c_2 是学习因子;r_1 和 r_2 是区间[0,1]中两个独立的均匀分布随机变量;X_i 表示第 i 个粒子的位置,d 表示维标号.

2 融合榜样学习和反向学习的粒子群优化算法

2.1 PSO 算法存在问题

由(1)式可知,种群的每个粒子都向着最优位置方向趋近,如此模型使得 PSO 快速收敛,而且采用串行动态的贪婪算法更新种群更加快了收敛速度,但当最优位置为局部最优点时,整个粒子群趋向局部最优,尤其在 PSO 解决复杂优化问题时,由于其包含多个局部最优值,在经过一定的迭代次数之后粒子会集中于局部最优值附近,导致种群多样性降低,即全局搜索能力变差,从而易导致算法陷入局部最优.

2.2 榜样学习

为解决 PSO 易陷于局部最优的问题,本文采用了榜样学习策略[15],即将 PSO 中粒子向 g_{best} 和 p_{best} 学习改为向任意一个更优的粒子学习.具体操作过程是每个非最优粒子从比自己优秀的粒子池(即榜样池)里随机选择一个优者进行学习,以便提高自身的搜索能力,避免了 PSO 易限于局部最优的缺点.本文中的榜样学习策略伪代码见算法 1.

算法 1:榜样学习

for $t=1$ to $\text{Max}D_T$ do

根据适应度值大小由大到小的顺序进行排序;

for $i=2$ to N do

从榜样池中随机选取一个榜样粒子 c_n;

执行 $V_i^d=w_t\times V_i^d+c_1\times r_1\times(p_{\text{best}c_n}^d-X_i^d)$;

执行 $X_i^d=X_i^d+V_i^d$;

end for

end for

算法 1 中 t 表示当前迭代次数,$\text{Max}D_T$ 表示最大迭代次数,N 表示种群大小,i 表示非最优粒子($i=2,3,\cdots,N$),c_n 表示从榜样池中随机选择的榜样粒子序号,w_t 表示惯性权重.从算法 1 可以看出:速度更新公式不再趋向全局最优位置和个体历史最优位置,而是随机选择一个榜样学习,这不仅提高了算法的多样性(即提高了算法的探索能力),而且因为都向着优者靠近,故也注重了开采能力.另外此算法在未新增可调参数的情况下,仅使用一个学习因子 c_1,减少了一个学习因子参数,提高了算法的可操作性.

2.3 反向学习

反向学习(Opposition Learning, OL)是 2005 年由 Tizhoosh 提出的[6].在了解反向学习之前,首先引入反向数的概念.

定义 1 反向数[6].令 $\{x\in\mathbf{R}\mid a\le x\le b\}$;反向数 x_{op} 被定义为:

$$x_{op}=a+b-x.\tag{3}$$

定义 2 反向点[6].在 D 维搜索空间中,令 $P(x^1,x^2,\cdots,x^D)$ 是搜索空间中的点,其中$\{x^d\in\mathbf{R}\mid a^d\le x^d\le b^d,\ d=1,2,\cdots,D\}$.$P$ 的反向点由 $P_{op}(x_{op}^1,x_{op}^2,\cdots,x_{op}^D)$ 定义,其中:

$$x_{op}^d=a^d+b^d-x^d.\tag{4}$$

对于种群中的最优个体,由于它是其他个体的榜样,没有榜样用于学习,因此选择反向学习策略,一方面提高种群的多样性,另一方面提高最优粒子自身的搜索能力.在本文提出的算法中,采用两种反向学习策略:随机反向学习策略和一般反向学习策略.在迭代前期采用随机反向学习,随机反向学习为

$$x_1^d=a^d+r\times(b^d-x_1^d).\tag{5}$$

其中,x_1^d 表示每次迭代后根据适应度值排序后的最大适应度值粒子(排名第一),即全局最优者;a 和 b 表示粒子的边界;r 表示[0,1]中的均匀分布随机数.

而在迭代后期采用一般反向学习策略.一般反向学习为

$$x_1^d=a^d+(b^d-x_1^d).\tag{6}$$

为了进一步提高种群的多样性，对于全局最优个体，不仅采用一般反向学习，而且还采用自身变异机制，更有利于提高种群的多样性.

2.4　自身变异机制

在迭代后期，交叉使用自身变异机制与一般反向学习策略，提高算法的种群多样性.当随机产生的值小于等于 0.5 时，采用自身变异机制，即随机选择某一维的值赋给当前维，否则采用一般反向学习策略.对于变异概率采用自适应方式，即随着迭代次数的增加，变异概率由大变小，前期变异概率大，有利于搜索全局最优解，后期变异概率小，有利于提高收敛速度，而且避免设置参数之麻烦.反向学习与自身变异机制相结合的伪代码见算法 2.

```
算法 2：反向学习与自身变异机制
for t=1 to MaxD_T do
  if t>1 * MaxD_T/2        //迭代后期
    if rand<=0.5
      for j=1 to D
        if rand>与自身变异概率 w_t
          产生随机维；
          将随机维中的最优值赋给当前维；
        end if
      end for
    else
      执行一般反向学习策略；
    end if
  else      //迭代前期
    执行随机反向学习策略；
  end if
end for
```

其中，变异概率取值与惯性权重 w_t 相等.本文提出的算法中，惯性权重采用的是自适应线性下降的方法如(7)式所示，迭代中从 1 至 0 自适应线性递减，

$$w_t = 1 - t/\mathrm{Max}D_T. \tag{7}$$

2.5　EOPSO 算法

因榜样学习策略以及反向学习策略能够提高粒子的自身搜索能力和种群多样性，本文将二者有机融合，获得一种 EOPSO 算法，流程图如图 1 所示.

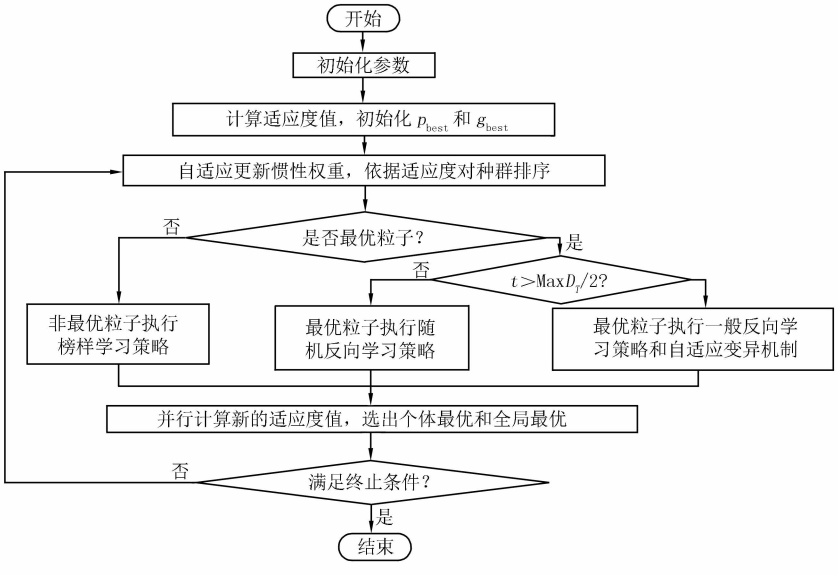

图 1　EOPSO 算法流程图

对于全局最优个体由于无榜样可供学习，前期采用随机反向学习策略，而后期则采用一般反向学习和自适应变异的结合机制.且在 EOPSO 中，减少了一个学习因子，并在每一次迭代的开始，自适应线性递减地更新惯性权重；根据粒子的适应度值对其进行排序，得到一个从大到小的序列，其中排名最前的则是最小函数值粒子，即全局最优者，排名最后的则是最大函数值的最差粒子.对于每一个非最优粒子，采用榜样学习策略.

在 PSO 中采用动态更新粒子方案，即每个粒子进行速度和位置更新时都要紧接着计算粒子的适应度值，接着更新最优粒子的位置，再用更新后的最优位置去引导随后的粒子速度和位置更新.这是一种串行的处理方式，这种方式使得种群更加向着最优位置快速聚集，加快收敛速度，但降低了种群的多样性，且不能采用并行计算方式；而在本算法中，在所有粒子的速度和位置更新之后，再同时计算所有粒子的适应度值，这是一种并行计算的方式，不仅可以提高算法的多样性，而且降低了算法的运行时间.

上述改进增加了算法的搜索能力,保持了种群多样性,避免算法陷入局部最优,也提高了算法的搜索效率.具体步骤描述如下:

步骤 1 随机初始化粒子的速度 V、位置 X.仅设置一个学习因子参数 c_1;

步骤 2 计算种群中每个粒子的适应度值,更新 p_{best} 和 g_{best};

步骤 3 在每次迭代中按自适应线性下降方式更新惯性权重,根据适应度值按由大到小的顺序排序;

步骤 4 对全局最优粒子,在迭代前期采用随机反向学习,迭代后期采用一般反向学习和自适应变异策略相结合的方式对粒子更新,并进行越界处理;

步骤 5 对每一个非最优粒子,从榜样池中随机选择榜样进行榜样学习,并更新粒子的速度和位置,且对粒子的速度和位置作越界处理;

步骤 6 判断是否满足终止准则,若满足则算法终止,否则返回至步骤 2.

3 仿真实验与结果分析

为了验证本文提出算法的有效性,采用 15 个常用的不同维基准函数进行优化实验.表 1 中给出了这 15 个 benchmark 测试函数的信息.其中,$f_1 \sim f_5$ 为高维单峰函数,使用单峰函数可以测试算法的收敛速度和精度,即算法的开采能力;$f_6 \sim f_{10}$ 是高维多峰函数,$f_{11} \sim f_{15}$ 是低维多峰函数,这些多峰函数主要用于测试算法的全局探索能力.函数的表达式及参数的具体取值见文献[16].

表 1 benchmark 函数信息表

序列	函数名称	取值范围	理论最优值
f_1	Sphere	[−100,100]	0
f_2	Rosenbrock	[−10,10]	0
f_3	Quartic	[−1.28,1.28]	0
f_4	Step	[−100,100]	0
f_5	Schwefel2.22	[−10,10]	0
f_6	Rastrigin	[−5.12,5.12]	0
f_7	Griewank	[−600,600]	0
f_8	Schwefel2.26	[−500,500]	0
f_9	Ackley	[−32,32]	0
f_{10}	Sum of different power	[−1,1]	0
f_{11}	Shekel's Foxholes	[−65.54,65.54]	0.998 0
f_{12}	Kowalik	[−5,5]	0.000 307 5
f_{13}	Shekel's Family 1	[0,10]	−10.153 2
f_{14}	Shekel's Family 2	[0,10]	−10.402 8
f_{15}	Shekel's Family 3	[0,10]	−10.536 4

3.1 参数设置与对比算法

本文选择了 4 种最新的 PSO 改进算法作为 EOPSO 的对比算法,这些对比算法分别为 ELPSO,SRPSO,LFPSO 和 HCLPSO 算法[9−12],均为 PSO 最先进的改进算法,其性能大幅度领先于标准的 PSO,具有一定的代表性.另外,选取高维(200 维)和低维(2～4 维)函数来验证 EOPSO 的普适性和有效性.5 种算法在高维和低维函数上的最大迭代次数分别均为 8 000 和 100,种群大小均为 20,每种算法独立运行均为 30 次.EOPSO 的学习因子 c_1 设置为 1.494 45,对比算法的参数与原文献相同.以下仿真实验的运行环境为:Windows 7 操作系统, CPU 为主频为 3.10 GHz 和内存为 4 GB 的 PC 机;编程语言采用 MATLAB R2014a.

3.2 优化性能比较

15 个 benchmark 函数的测试结果见表 2 和表 3,其中包含 30 次独立运行的寻优结果的最大值、最小值、均值、方差,粗体表示算法测试结果的最优者.

表 2　5 种算法在 200 维函数上的测试结果

函数	算法	最大值	最小值	均值	方差
f_1	EOPSO	**0**	**0**	**0**	**0**
	ELPSO	4.936 3e+02	7.622 0e+01	2.034 1e+02	1.093 2e+02
	SRPSO	8.562 8e+01	9.376 9	4.662 2e+01	2.313 1e+01
	LFPSO	6.637 8	3.323 0e-01	1.387 5	1.223 6
	IICLPSO	6.883 8	3.548 6	4.641 2	7.811 1e-01
f_2	EOPSO	**6.082 7e-04**	**3.049 6e-07**	**1.923 9e-04**	**1.996 1e-04**
	ELPSO	3.100 7e+03	6.405 7e+02	1.568 3e+03	5.855 9e+02
	SRPSO	1.005 6e+03	4.551 4e+02	6.928 2e+02	1.394 7e+02
	LFPSO	1.139 1e+03	5.116 0e+02	8.102 9e+02	1.439 9e+02
	IICLPSO	1.538 4e+03	1.026 6e+03	1.240 4e+03	9.778 9e+01
f_3	EOPSO	**1.687 3e-02**	**8.010 5e-05**	**3.191 4e-03**	**4.520 6e-03**
	ELPSO	3.207 6	1.366 4	2.196 5	4.679 1e-01
	SRPSO	6.468 8e-01	3.475 8e-01	4.791 3e-01	6.938 4e-02
	LFPSO	2.708 8	1.261 6	1.879	3.566 8e-01
	IICLPSO	3.031 2e-01	1.682 8e-01	2.383 2e-01	3.422 4e-02
f_4	EOPSO	**0**	**0**	**0**	**0**
	ELPSO	1.994e+03	4.74e+02	1.089 6e+03	3.662 2e+02
	SRPSO	5.76e+02	1.50e+02	2.866 3e+02	8.989 3e+01
	LFPSO	1.43e+02	4.60e+01	8.286 7e+01	2.100 5e+01
	IICLPSO	1.50e+01	5	9.033 3	2.809 9
f_5	EOPSO	**0**	**0**	**0**	**0**
	ELPSO	1.003 0e+01	1.801 9	5.069 2	2.153 4
	SRPSO	4.543 8	1.232 5	2.652 4	8.574 6e-01
	LFPSO	3.466	2.071 5e-01	9.118 6e-01	6.838 3e-01
	IICLPSO	1.594 2	9.730 3e-01	1.285 5	1.553 4e-01
f_6	EOPSO	**0**	**0**	**0**	**0**
	ELPSO	5.893 6e+02	4.073 7e+02	5.039 6e+02	4.069 1e+01
	SRPSO	9.083 6e+02	1.503 2e+02	4.499 3e+02	1.974 5e+02
	LFPSO	4.812 2e+02	3.055 0e+02	4.157 0e+02	4.334 9e+01
	IICLPSO	2.617 3e+02	1.918 2e+02	2.269 4e+02	1.652 7e+01
f_7	EOPSO	**5.551 1e-16**	**0**	**1.813 4e-16**	**1.253 5e-16**
	ELPSO	8.129 6	1.407 9	2.836 9	1.382
	SRPSO	2.064 2	1.112 5	1.470 5	2.414 0e-01
	LFPSO	1.012 5	8.436 7e-02	4.516 6e-01	2.853 9e-01
	IICLPSO	9.790 2e-01	6.297 0e-01	7.963 6e-01	8.186 9e-02
f_8	EOPSO	**5.093 2e-10**	**3.346 9e-10**	**3.652 5e-10**	**3.136 2e-11**
	ELPSO	4.606 5e+04	3.083 8e+04	3.807 0e+04	4.127 2e+03
	SRPSO	1.315 4e+04	9.475 3e+03	1.168 5e+04	7.046 3e+02
	LFPSO	4.101 7e+04	2.054 3e+04	3.194 5e+04	5.148 4e+03
	IICLPSO	2.633 1e+04	2.151 5e+04	2.396 3e+04	9.868 3e+02

续表

函数	算法	最大值	最小值	均值	方差
f_9	EOPSO	**2.753 4e-14**	**6.217 2e-15**	**1.521 7e-14**	**40545 2e-15**
	ELPSO	4.754 2	3.321 8	4.054 8	3.895 4e-01
	SRPSO	6.177 7	1.994 8	2.808 2	9.407 2e-01
	LFPSO	3.551 6	2.217 2	2.944 3	3.554 5e-01
	IICLPSO	2.276 6	1.561 1	1.999 9	7.130 7e-01
f_{10}	EOPSO	**1.045 7e-62**	**8.897 1e-93**	**4.318 2e-64**	**1.947 6e-63**
	ELPSO	3.402 5e-21	1.096 0e-27	2.917 8e-22	8.064 4e-22
	SRPSO	1.085 1e-33	9.698 1e-40	9.639 1e-35	2.168 6e-34
	LFPSO	5.443 1e-22	3.715 4e-29	2.263 6e-23	9.972 1e-23
	IICLPSO	1.839 8e-38	1.296 1e-42	2.205 6e-39	4.146 2e-39

表 3　5 种算法在低维函数上的测试结果

函数	算法	最大值	最小值	均值	方差
f_{11}	EOPSO	**9.980 0e-01**	**9.980 0e-01**	**9.980 0e-01**	**4.800 5e-10**
	ELPSO	1.076 4e+01	9.980 0e-01	2.523 8	2.488 5
	SRPSO	5.00e+02	9.980 2e-01	2.606 4e+02	2.335 0e+02
	LFPSO	1.550 4e+01	9.980 0e-01	2.969 6	3.246 6
	IICLPSO	2.123 4e+01	9.980 0e-01	6.213 1	5.467 5
f_{12}	EOPSO	**6.550 6e-04**	**3.157 9e-04**	**4.766 1e-04**	**1.079 0e-04**
	ELPSO	1.373 5e-03	3.081 2e-04	4.961 9e-04	2.635 3e-04
	SRPSO	1.454 9e-03	3.169 9e-04	6.096 1e-04	2.906 0e-04
	LFPSO	1.226 4e-03	3.244 3e-04	5.147 0e-04	1.738 7e-04
	IICLPSO	6.776 8e-04	3.100 5e-04	4.933 5e-04	1.062 4e-04
f_{13}	EOPSO	**−1.015 3e+01**	**−1.015 3e+01**	**−1.015 3e+01**	**8.787 1e-10**
	ELPSO	−2.630 5	−1.015 3e+01	−8.225 2	3.071 3
	SRPSO	−2.630 5	−1.015 3e+01	−8.306 8	2.771 5
	LFPSO	−2.630 5	−1.015 3e+01	−8.401 2	3.018 9
	IICLPSO	−1.013 6e+01	−1.015 3e+01	−1.015 3e+01	3.087 8e-03
f_{14}	EOPSO	**−1.040 3e+01**	**−1.040 3e+01**	**−1.040 3e+01**	**1.121 0e-09**
	ELPSO	−2.765 9	−1.040 3e+01	−9.639 2	2.330 3
	SRPSO	−2.751 9	−1.040 3e+01	−9.356	2.410 9
	LFPSO	−2.751 9	−1.040 3e+01	−9.032 6	2.833 4
	IICLPSO	−1.040 3e+01	−1.040 3e+01	−1.040 3e+01	1.992 2e-05
f_{15}	EOPSO	**−1.053 6e+01**	**−1.053 6e+01**	**−1.053 6e+01**	**1.674 4e-09**
	ELPSO	−2.421 7	−1.053 6e+01	−1.000 8e+01	2.010 6
	SRPSO	−3.758 9	−1.053 6e+01	−9.549 1	2.149 4
	LFPSO	−5.175 6	−1.053 6e+01	−1.035 8e+01	9.787 4e-01
	IICLPSO	−5.175 6	−1.053 6e+01	−1.035 8e+01	9.787 4e-01

表 2 给出了 200 维情况下 benchmark 函数的测试结果，从表 2 可以看出，使用 f_1，f_4，f_5，f_6 函数进行测试时，EOPSO 相比较于 4 个对比算法取得了绝对的优势，均值和方差均达到了理想的最优值 0；而其他 benchmark 函数的测试结果虽未达到最优值 0，但是相比较于其他算法的结果，无论是最大值、最小值、均值或方差，均占优势，大幅度领先于其他 4 种对比算法.虽然 EOPSO 在其他的函数上没有完全达到理想的最

优值 0，但是其数值上远优于 4 种对比算法的测试结果.这说明 EOPSO 不仅在高维单峰函数上有较好的收敛精度(这是由于采用新型的榜样学习策略使得非最优个体向着自己的榜样学习，具有较好开采能力的结果)，而且由于榜样学习采用随机选择榜样学习的方式和最优粒子采用反向学习和变异机制使得粒子在高维多峰函数上获得了更好的全局搜索能力.

表 3 表示的是低维情况下 benchmark 函数的测试结果，这些函数虽具有较低的维数，但其局部最优解的个数有多个.在使用 f_{11}，f_{13}，f_{14} 和 f_{15} 函数测试算法的优化性能时，EOPSO 均获得了最好值，且大幅度领先于 4 个对比算法的测试结果.仅在 f_{12} 上，EOPSO 获得的最小值稍差于 ELPSO.

总之，从以上函数优化结果看，EOPSO 算法采用榜样学习和反向学习策略提高算法的全局搜索能力和局部搜索能力，这说明本文提出的方法是可行的.

3.3　运行时间比较

为考查 EOPSO 的运行效率，统计 5 种算法的平均运行时间.篇幅所限，仅给出 200 维测试函数的运行时间结果见表 4，其中黑体字表示 5 种算法的最优者.

表 4　5 种优化算法的运行时间

函数	运行时间/s				
	EOPSO	ELPSO	SRPSO	LFPSO	HCLPSO
f_1	**2.765 6**	3.629 0	8.216 3	5.657 3	5.868 9
f_2	**2.256 6**	4.710 5	8.389 2	6.793 9	6.997 9
f_3	**4.279 8**	9.223 3	10.500 1	11.416 9	11.526 8
f_4	**2.324 8**	4.035 4	8.044 3	6.267 0	6.308 4
f_5	**2.775 4**	4.100 8	8.229 6	6.148 3	6.343 0
f_6	**2.736 0**	5.050 8	9.619 9	8.307 5	7.474 2
f_7	**2.936 9**	7.503 9	9.457 5	9.845 8	9.799 3
f_8	**3.054 3**	5.480 6	9.277 4	8.057 6	7.980 0
f_9	**2.906 3**	5.689 6	9.415 9	7.695 1	8.035 4
f_{10}	**4.317 0**	8.973 8	10.741 0	11.408 5	11.451 0
平均运行时间	**3.035 3**	5.839 8	9.189 1	8.159 8	8.178 5

从表 4 中可以看出，当优化 $f_1 \sim f_{10}$ 10 个函数时，EOPSO 在每个函数上的耗时是 5 种算法中最少的，且运行时间约为 ELPSO 的 1/2.从算法的运行时间的平均值来分析，EOPSO 的平均运行时间为 3.035 3 s，而 ELPSO、SRPSO、LFPSO 和 HCLPSO 的平均运行时间分别为 5.839 8 s、9.189 1 s、8.159 8 s、8.178 5 s. EOPSO 的平均运行时间是测试算法中的最小值，分别约为 4 种对比算法的 51.98%、33.03%、37.20%、37.11%.这主要有两个原因：第一、EOPSO 采用并行方式搜索节省了时间；其二、榜样学习中的速度更新公式计算复杂度低于(1)式的计算复杂度.由此可得，EOPSO 用最少的时间使优化性能最大化，故有较高的优化效率.

3.4　收敛性分析

为了更清晰地了解 EOPSO 的收敛性能，图 2 展示了 5 种算法在 200 维函数上的收敛曲线图，它能够更加清晰地显示出算法的收敛性.

从 200 维函数的测试实验结果中可以看出提出的 EOPSO 在测试函数 f_1，f_2，f_3，f_4，f_6，f_7，f_8，f_9，f_{10} 上展示出了明显的优势，其收敛性能更加优异.而在测试函数 f_5 上，收敛性与其他 4 种对比算法的收敛性比较相似，前期由于算法主要在于全局搜索，故收敛曲线趋于平缓，即处于慢速收敛状态，后期算法主要属于开采状态，故收敛速度快捷.从整体来看，EOPSO 的收敛性能远远优于对比算法的收敛性能.

本文提出的算法性能大幅度优于 4 种最先进的对比算法，根据测试函数的结果收敛图分析算法的收敛性.由收敛图 2 中可知，EOPSO 算法采用榜样学习和反向学习，使粒子及时跳出局部最优，收敛速度快.因此，该算法的收敛速度得到提升，同时避免了种群多样性的丢失.

4 结 论

针对PSO存在的多样性缺失、早熟收敛等缺点,本文提出了一种融合榜样学习和反向学习的PSO算法.该算法将榜样学习和反向学习有机融入粒子群算法中,不仅发挥了PSO局部搜索能力强的特点,而且也克服其易限于局部最优的缺点,整体优化性能优秀;虽然嵌入了榜样学习和反向学习机制,但未增加可调参数,且减少了一个学习因子,并对于PSO中惯性权重参数采用自适应线性下降方法,提高了算法的可操作性;采用并行方式,不仅未降低优化性能,而且克服了串行算法运算速度慢的不足,算法的计算效率高;EOPSO的普适性强,不管在低维还是在高维的函数优化上,其优化性能大幅度领先于4种state-of-the-art的PSO改进算法.

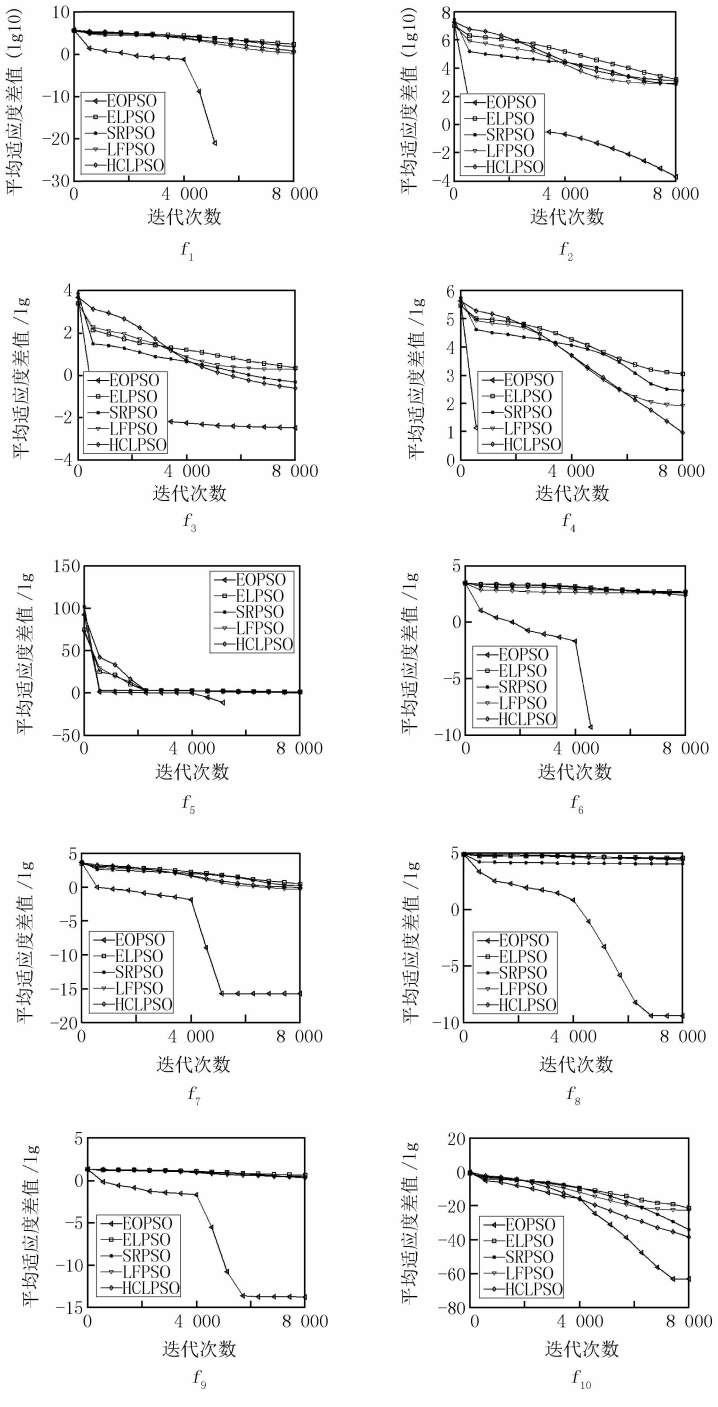

图2 200维(高维)函数的收敛图

参 考 文 献

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

[2] 张新明,涂强,尹欣欣,等.嵌入趋化算子的PSO算法及其在多阈值分割中的应用[J].计算机科学,2016,43(2):311-315.

[3] 刘欢,刘志刚.基于改进粒子群算法的牵引变电所维修优化研究[J].电力系统保护与控制,2015,43(11):87-94.

[4] 吴辰斌,李海明,刘栋,等.一种改进型粒子群优化算法在电力系统经济负荷分配中的应用[J].电力系统保护与控制,2016,44(10):

44-48.
[5] 许昆.涞水河流域生态环境需水量计算和预测 [J].灌溉排水学报，2016，35(11)：107-110.
[6] Liu H，Xu G，Ding G Y，et al.Integrating opposition-based learning into the evolution equation of bare-bones particle swarm optimization [J].Soft Computing，2015，19(10)：2813-2836.
[7] Shang J L，Sun Y，Li S J，et al.An improved opposition-based learning particle swarm optimization for the detection of SNP-SNP interactions [J].Biomed Research International，2015.DOI：10.1155/2015/524821.
[8] Wang H，Wu Z J，Rahnamayan S，et al.Enhancing particle swarm optimization using generalized opposition-based learning [J].Information Sciences，2011，181(20)：4699-4714.
[9] Jordehi A R.Enhanced leader PSO (ELPSO)：A new PSO variant for solving global optimisation problems [J].Applied Soft Computing，2015，26(26)：401-417.
[10] Tanweer M R，Suresh S，Sundararajan N.Self regulating particle swarm optimization algorithm [J].Information Science，2015，294：182-202.
[11] Hakli H，U guz H.A novel particle swarm optimization algorithm with Levy flight [J].Applied Soft Computing，2014，23(5)：333-345.
[12] Lynn N，Suganthan P N.Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation [J].Swarm & Evolutionary Computation，2015，24：11-24.
[13] 夏学文，刘经南，高柯夫，等.具备反向学习和局部学习能力的粒子群算法[J].计算机学报，2015，38(7)：1397-1407.
[14] Huang H，Qin H，Han Z F，et al.Example-based learning particle swarm optimization for continuous optimization [J].Information Sciences，2012，182(1)：125-138.
[15] 张新明，涂强，尹欣欣.混合迁移的高效 BBO 算法及其在图像分割中的应用 [J].计算机科学与探索，2016，10(10)：1459-1468.
[16] 纪震，廖慧连，吴青华.粒子群算法及应用 [M].北京：科学出版社，2009.

Particle Swarm Optimization Algorithm Based on Combing Example Learning and Opposition Learning

Zhang Xinming[a,b]，Wang Xia[a]，Tu Qiang[a]，Kang Qiang[a]

(a. College of Computer and Information Engineering；b. Engineering Technology Research Center for Computing Intelligence & Data Mining of Henan Province，Henan Normal University，Xinxiang 453007，China)

Abstract：In order to improve the optimization efficiency of the particle swarm optimization algorithm and prevent the algorithm from trapping into the local optima. Based on combing Example learning and Opposition learning (EOPSO). This paper proposes a PSO Firstly，all non-optimal particles in the particle swarm are updated by a novel example learning mechanism to improve their search ability and to prevent the algorithm from trapping into the local optima. Secondly，the optimal particle is updated by a hybrid opposition learning way to improve its search ability and further avoid the algorithm′s trapping into the local optima. Finally，a self-mutation mechanism is also adopted to update the optimal particle to increase the population diversity. In addition，the self-mutation mechanism adopts an adaptive mutation rate to provide the good global search ability at the early search phase and accelerate the convergence speed at the late search phase in the algorithm process. The simulation experiments are made on 15 benchmark functions with different dimensions. The experiment results show that，compared with the state-of-the-art PSO variants such as ELPSO，SRPSO，LFPSO and HCLPSO，EOPSO obtains better optimization performance.

Keywords：intelligent optimization algorithm；particle swarm optimization algorithm；example learning；opposition learning

[责任编辑　陈留院]