一种多策略混合的粒子群优化算法

余伟伟[1] 谢承旺[2]
(北京工业大学软件学院 北京 100124)[1]
(广西师范学院科学计算与智能信息处理广西高校重点实验室 南宁 530023)[2]

摘 要 针对传统粒子群优化算法在解决一些复杂优化问题时易陷入局部最优且收敛速度较慢的问题,提出一种多策略混合的粒子群优化算法(Hybrid Particle Swarm Optimization with Multiply Strategies,HPSO)。该算法利用反向学习策略产生反向解群,扩大粒子群搜索的范围,增强算法的全局勘探能力;同时,为避免种群陷入局部最优,算法对种群中部分较差的个体实施柯西变异,以产生远离局部极值的个体,而对群体中较好的个体施以差分进化变异,以增强算法的局部开采能力。对这 3 种策略进行了有机结合以更好地平衡粒子群算法全局勘探和局部开采的能力。将 HPSO 算法与其他 3 种知名的粒子群算法在 10 个标准测试函数上进行了性能比较实验,结果表明 HPSO 算法在求解精度和收敛速度上具有较显著的优势。

关键词 反向学习,粒子群优化,柯西变异,差分进化
中图法分类号 TP301 **文献标识码** A

Hybrid Particle Swarm Optimization with Multiply Strategies

YU Wei-wei[1] XIE Cheng-wang[2]
(School of Software,Beijing University of Technology,Beijing 100124,China)[1]
(Science Computing and Intelligent Information Processing of Guangxi Higher Education Key Laboratory,Guangxi Teachers Education University,Nanning 530023,China)[2]

Abstract A hybrid particle swarm optimization with multiply strategies (HPSO) was proposed to solve the problem of being easy to get into the local optimum and slow convergence speed for particle swarm optimization algorithm(PSO) in dealing with some complicated optimization problems. The HPSO uses the opposition-based learning strategy to generate the opposition-based solutions,which enlarges the search range of particle swarm,and enhances the global exploration ability of the algorithm. At the same time,in order to jump out of the local optimum,the HPSO performs Cauchy mutation on some poorer particles to generate individuals that are far from the local optimum,and the differential evolution (DE) mutation is employed to remain individuals to improve the capacity of local exploitation. The above strategies are combined to balance the abilities of global exploration and local exploitation,which are expected to solve some hard optimization problems better. The HPSO and other three well-known PSOs were compared on 10 benchmark test instances experimentally. The results show that the HPSO performs significant advantages over the compared algorithms in the solution accuracy and the convergence speed.

Keywords Opposition-based learning,Particle swam optimization,Cauchy mutation,Differential evolution

1 引言

粒子群优化算法(Particle Swarm Optimization,PSO)是由 Kennedy 和 Eberhart 等[1]于 1995 年提出的一种群体智能优化方法,其通过模拟自然界中鸟群或鱼群的觅食行为使得群体达到最优。PSO 算法具有概念简单、易于实现以及参数较少等优势,备受研究人员的青睐。PSO 算法在求解一些最优化问题时表现出了较强的全局搜索能力,并在多个科学与工程领域中得到成功应用。虽然 PSO 算法已经表现出了较好的性能,但它在解决一些复杂的优化问题时容易陷入局部最优且收敛速度较慢。

随着群智能优化技术的发展,考虑不同学习机制或进化范例的特点,并将它们有机结合,相互取长补短,克服算法本身固有的缺陷渐,已成为群智能优化算法的一种新的发展趋势[2]。受此启发,本文提出一种多策略混合的粒子群优化算法(Hybrid Particle Swarm Optimization with Multiply Strategies,HPSO),该算法具有如下特点:1)将反向学习机制(Opposition-based Learning)引入 PSO 算法,产生当前群体对应的反向种群。该策略有助于搜索更多的有效区域来提高群体的多样性,以增强算法的全局搜索能力;2)对种群中较好的个体实施差分进化的变异,使粒子群能在较优个体的邻域空间内实施局部开采,以提高算法的求解精度;3)对种群中较差的个体实施柯西变异以产生较远区间内的新解,避免种群陷入局部最优而导致搜索进程停滞。在 11 个典型测试函数上进行数值实验,将本文算法与基本 PSO 算法以及 PSO 算法的变体进行对比,实验结果表明本文算法在大部分测试函数上具有更优的性能。

本文受科学计算与智能信息处理广西高校重点实验室开放课题(GXSCIIP201604)资助。
余伟伟(1993-),男,硕士生,主要研究方向为智能计算及其应用,E-mail:ecjtu_yuweiwei@163.com(通信作者);**谢承旺**(1974-),男,博士,副教授,主要研究方向为智能计算,E-mail:chengwangxie@163.com。

2　PSO 算法

PSO 算法是一种基于群体的随机搜索算法，对于 D 维搜索空间的优化问题，种群中的每个粒子表征一个候选解，具有速度与位置两个属性，其在解空间中以一定的速度移动。在算法的进化过程中，粒子的自身最优值(pbest)与群体的全局最优值(gbest)均被记录下来，用于更新粒子的速度和位置。

假设当前第 t 代粒子群 $P(t)$ 中包含了 N 个粒子，第 i 个粒子 $x_i=(x_{i1},x_{i2},\cdots,x_{iD})$ 的第 j 维速度分量 v_{ij} 和位置分量 x_{ij} 在第 $t+1$ 代中的更新公式如下：

$$\begin{cases} v_{ij}(t+1)=\omega v_{ij}(t)+c_1 rand_1(pbest_{ij}-x_{ij}(t))+ \\ \qquad c_2 rand_2(gbest-x_{ij}(t)) \\ x_{ij}(t+1)=x_{ij}(t)+v_{ij}(t+1) \end{cases} \tag{1}$$

其中，$i\in[1:N]$，$j\in[1:D]$，ω 是惯性权值，且 $\omega\in[0,1]$；c_1 和 c_2 为加速系数，且 $c_1,c_2\in[0,2]$；$rand_1$ 和 $rand_2$ 为 $[0,1]$ 区间上均匀分布的随机数。

3　多策略混合的粒子群优化算法

3.1　反向学习机制

反向学习策略(Opposition-Based Learning，OBL)是一种新型计算智能方法[3]，其主要思想是：对问题的一个可行解，同时计算并评估其反向解，从中选择较优的解作为下一代个体。反向学习机制由于能拓展种群探索的空间，增加群体的多样性，因此有利于改善算法的全局勘探能力。反向学习技术自提出以来已被成功应用到多种优化算法中。下面给出反向点(也可视为反向解)的定义。

定义 1(反向解)　设在 D 维搜索空间中一个可行解 $x_i=(x_{i,1},x_{i,2},\cdots,x_{i,D})$，$x_{i,j}=[a_j,b_j]$，$j=1,2,\cdots,D$。设其反向解 $x_i'=(x'_{i,1},x'_{i,2},\cdots,x'_{i,D})$，且满足式(2)：

$$x'_{i,j}=a_{i,j}+b_{i,j}-x_{i,j} \tag{2}$$

定义 2(广义反向学习)　令 $x'_{i,j}=k(a_j+b_j)-x_{i,j}$，其中 $x_{i,j}=[a_j,b_j]$，$i=1,2,\cdots,|Popsize|$，$j=1,2,\cdots,D$，$|Popsize|$ 为粒子群的规模，D 为搜索空间的维度。

定义 2 中的 k 可以取不同的实数，当 $k=0$ 时其被称为基于解对称的广义反向学习；当 $k=0.5$ 时其被称为基于对称区间的广义反向学习；当 $k=1$ 时其被称为广义的反向学习；而当 k 为 $[0,1]$ 区间内的随机数时其被称为随机广义反向学习。

定义 3(一般动态反向学习[4])　令 $x'_{i,j}=k(da_j+db_j)-x_{i,j}$，其中 da_j 和 db_j 分别表示当前代种群搜索空间中第 j 维上的最小值和最大值，即：

$$\begin{cases} da_j=\min(A_j) \\ db_j=\max(A_j) \end{cases} \tag{3}$$

其中，A_j 为种群中的个体在第 j 维上所有取值的集合，$k\in[0,1]$ 为一般化系数。

下文通过一个实例来说明反向学习具有较强的全局勘探能力[5]。假设待优化问题 $\min f(q)=\|q-A\|$，优化问题搜索空间的维度为 2，A 是优化问题的全局最优位置，并设其为 (14,15)。假设决策变量 x_1 和 x_2 的取值范围为 $[0,20]$，现从决策空间中取一点 $p=(9,8)$，其适应值为 $f(p)=\sqrt{74}$，根据定义 1 可得其反向解点 $p'=(11,12)$，其适应度值 $f(p')=3\sqrt{2}$，显然 $f(p')<f(p)$，说明反向点更加接近最优解的位置。

本文算法利用一般动态反向学习策略产生反向种群，并从当前种群和反向种群值中选择具有较好适应值的粒子组成下一代粒子群。该学习机制的好处是既扩大了种群搜索的范围，又避免了无效搜索，使得进化的群体能较快地收敛于全局最优解。

3.2　差分进化变异

差分进化算法(Differential Evolution，DE)最初是由 Storn 和 Price[6] 于 1995 年提出，与其他进化算法具有类似的计算步骤，特点是使用了个体的差异信息。本文结合周新宇提出的差分进化变异策略(Differential Evolution Mutation，DEM)[7] 来对种群值较优的个体实施 DEM 操作，以实现在优质个体附近的局部搜索。一般地，种群中较好的粒子往往可能更加靠近全局最优解，因此在这些优质个体附近进行搜索将更可能发现全局最优解。在 DEM 策略中，有几个至关重要的参数需要确定，DE 主要有 3 个版本，其差别主要体现在变异的方法上，根据变异个体的生成方式不同，DE 的变异方式也不同，常见的变异方式有 3 种：DE/rand/1/bin、DE/best/1/bin 和 DE/current-to-best/2/bin。本文采用“DE/current-to-best/2/bin”变异方式，设粒子群 $Pop=\{X_1,X_2,\cdots,X_{|Popsize|}\}$，则实施如式(4)所示的变异操作。

$$v_{i,j}=x_{i,j}+\lambda*(x_{best,j}-x_{i,j})+F*(x_{r1,j}-x_{r2,j}) \tag{4}$$

其中，$r_1\neq r_2\neq r_3\neq i$，$r_1,r_2,r_3\in\{1,2,3,\cdots,N\}$；变异算子 F ($F\in[0,2]$)是一个常数因子，用于控制偏差变量的放大；λ 是一个附加的控制参数，$x_{best,j}$ 是当前种群最好个体的第 j 维分量。然后采用如式(5)所示的交叉操作来生成新个体 $u_{i,j}$。

$$u_{i,j}=\begin{cases} v_{i,j}, & \text{rand}(0,1)\leqslant CR \text{ or } j=\text{rand}(1,D) \\ x_{i,j}, & \text{others} \end{cases} \tag{5}$$

其中，$CR\in[0,1]$ 为交叉概率，rand(0,1)为 $[0,1]$ 之间均匀分布的随机数，rand(1,D)是 [1,D] 之间的一个随机整数，这种交叉可以保证 u_i 中至少有一个分量由 v_i 提供。需要说明的是，本文只对各代粒子群中适应值排名前 50% 的个体实施 DEM 操作。

3.3　柯西变异策略

Yao 等[8] 在一种快速的进化规划算法中提出利用柯西变异(Cauchy Mutation)代替传统的高斯变异，实验结果表明了柯西变异算子能加强算法的全局搜索能力。图 1 给出了柯西分布($t=1$)和高斯分布($N(0,1)$)的密度函数分布。

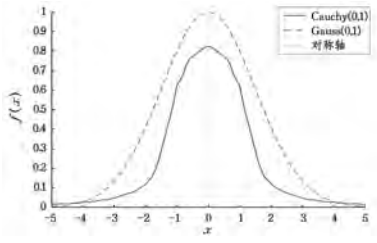

图 1　标准高斯和柯西分布概率密度函数曲线比较

从图 1 可以看出，在垂直方向上柯西分布略小于高斯分布，但在水平方向上柯西分布则更加平滑地靠近水平轴。因此，柯西分布产生的随机数分布更宽，离原点的距离较大，从而保证了其能有较大的概率来产生远离局部最优且与父代差异较大的解集粒子。

基于此，本文采用标准 Cauchy 变异策略($t=1$)产生远离局部最优点的新解，这样既可拓展粒子搜索的范围，又能防止算法陷入局部最优而早熟收敛。需要指出的是，这里对种群中适应值排名后 50% 的粒子实施扰动，式(6)给出了柯西变异的方法。

$$X_j^* = X_j + Cauchy(0,1) \quad (6)$$

其中，X_j 是粒子在第 j 维的分量，X_j^* 为变异后的值，$Cauchy(0,1)$ 是服从标准柯西分布的随机数。若变异后的个体的适应值优于变异前的个体，则用变异后的个体取代原个体。

3.4 HPSO 算法

在 3.1 节－3.3 节的描述的基础上，算法 1 给出了 HPSO 算法的流程。

算法 1 多策略混合的粒子群优化算法 HPSO

输入：种群规模|Popsize|，加速系数 c1，c2，惯性权重 w，最大迭代次数 T_{max}，一般化系数 k，一般动态反向学习概率 P_{OL}，差分进化的缩放尺度 F，差分进化的交叉概率 CR

输出：算法获得的最优解

Step1 随机初始化规模为|Popsize|的粒子群，评估种群的个体适应值，并设迭代器 t=0；

Step2 WHILE (t < T_{max})

Step3 IF (rand < P_{OL})

Step4 对种群 Pop(t)执行 3.1 小节所述的一般动态反向学习，产生反向种群 OP(t)；

Step5 合并种群 Pop(t)和 OP(t)，并从中选择|Popsize|个较好个体构成临时种群 TP(t)；

Step6 计算种群 TP(t)中的个体适应值，并依据适应值大小对种群排名前 50%的个体实施 3.2 小节所述的差分进化变异；对种余下的个体实施 3.3 小节所述的柯西变异；

Step7 从变异前后所有粒子中选择|Popsize|个较好的粒子构成新的种群；

Step8 ELSE

Step9 按照式(1)执行对粒子群的更新；

Step10 END IF

Step11 t=t+1

Step12 END WHILE

Step13 输出算法获得的最优解。

4 实验与结果分析

4.1 测试函数

实验采用表 1 所列的 10 个测试函数，将这些函数按其属性分为 3 类：单峰及简单的多峰函数(如 f_1-f_2)，它们主要用来测试算法的寻优精度；非旋转的多峰函数(如 f_3-f_6)，这些函数具有很多局部极值点，一般的算法较难找到全局最优值，因此它们可以用来检验算法的全局搜索性能和避免早熟的能力；带旋转的多峰函数(如 f_7-f_{10})，这些函数的全局最优值均为 0，维度 $D=30$。

表 1 数值实验使用的 10 个测试函数

函数	函数名	变量范围	最优值
f_1	Sphere	[－100,100]	0
f_2	Rosenbrock	[－2.04,2.04]	0
f_3	Griewanks	[－600,600]	0
f_4	Rastrigin	[－5.12,5.12]	0
f_5	Schwefel 1.2	[－100,100]	0
f_6	Schwefel 2.22	[－10,10]	0
f_7	Step	[－100,100]	0
f_8	Quartic	[－1.28,1.28]	0
f_9	Generalized P1	[－50,50]	0
f_{10}	Generalized P2	[－50,50]	0

4.2 对比算法

为了检测本文 HPSO 算法的性能，选择了 3 种经典的 PSO 算法与之比较，分别是标准的 PSO 算法[1]、精英反向学习的粒子群 EOBPSO 算法[7]以及混合了柯西变异和差分变异的 CDPSO 算法[9-10]。为了保证实验的公正性，将这 4 种算法所使用的共同参数都设为相同，而其他参数都与参考文献保持一致，即最大迭代次数 $T_{max}=1000$，种群规模 $Popsize=100$，搜索空间维度 $D=30$。为减少随机因素的误差，4 种算法在 10 个测试函数上均独立运行 30 次，统计其均值与方差。

这里借鉴文献[4]的数据统计与分析方法，即采用显著水平为 5%的双尾 t-test 检验标准来判断算法性能，结果如表 2 所列。表 2 中的“＋”“＝”和“－”分别表示本文算法获得的结果优于、相当于和劣于对应列的算法在对应行的测试函数上的结果。最后根据获得“＋”“＝”和“－”的个数计算算法获得的净胜分“Score”，即获得一个“＋”为 1 分，获得一个“＝”为 0 分，获得一个“－”为－1 分，“Score”则为得“＋”的数目减去得“－”的数目。净胜分 Score 越大表明 HPSO 算法于其他对比算法具有更显著的性能优势。

表 2 4 种算法在 10 个测试函数上的对比结果

测试函数		HPSO	PSO	EOBPSO	CDPSO
f_1	Mean	**3.96e－65**	2.67e－01	1.51e－12	1.35e－06
	Std.	6.60e－65	7.48e－02	2.01e－12	1.06e－06
	t-test		＋	＋	＋
f_2	Mean	7.15e＋00	4.67e＋01	3.44e＋01	**1.55e－01**
	Std.	7.63e＋00	6.75e＋00	9.24e＋00	1.94e－01
	t-test		＋	＋	－
f_3	Mean	**0.00e＋00**	9.65e－03	1.60e－13	8.64e－07
	Std.	0.00e＋00	2.39e－03	2.43e－13	8.15e－07
	t-test		＋	＋	＋
f_4	Mean	**3.28e＋00**	5.31e＋01	7.30e＋00	9.66e＋00
	Std.	3.03e＋00	8.75e＋00	6.28e－01	1.11e＋00
	t-test		＋	＝	＝
f_5	Mean	**1.63e－03**	3.68e－01	1.40e－01	6.21e－01
	Std.	2.39e－03	2.67e－01	1.31e－01	2.97e－01
	t-test		＋	＋	＋
f_6	Mean	**1.97e－24**	2.24e＋00	1.08e－04	5.14e－01
	Std.	4.83e－24	1.93e－01	1.17e－04	8.33e－02
	t-test		＋	＋	＋
f_7	Mean	2.17e－07	8.85e－01	**1.33e－10**	8.95e－06
	Std.	9.55e－08	2.11e－01	2.57e－10	1.66e－05
	t-test		＋	－	＋
f_8	Mean	4.68e－02	9.58e－01	**2.63e－04**	1.17e＋00
	Std.	1.05e－02	2.04e－01	9.57e－05	1.43e－01
	t-test		＋	－	＋
f_9	Mean	**1.91e－31**	6.85e－02	9.34e－13	1.25e－04
	Std.	3.74e－31	1.76e－02	1.29e－12	1.47e－04
	t-test		＋	＋	＋
f_{10}	Mean	**4.11e－04**	1.42e＋00	1.46e－03	1.24e－02
	Std.	2.19e－04	3.36e－01	1.61e－03	1.66e－02
	t-test		＋	＋	＋
	Better(＋)		10	7	8
	Same(＝)		0	1	1
	Worse(－)		0	2	1
	Score		10	4	7

由表 2 可知，HPSO 算法在 10 个测试问题上获得了 7 个最好的均值，EOBPSO 算法获得了 2 个最好的均值，CDPSO 获得了 1 个最好的均值，而基本 PSO 算法在 10 个测试问题中未能获得任何最好的均值。从表 2 的 t-test 结果来看，HPSO 算法与对比算法相比所获得的净胜分均大于 0，其中对比标准 PSO 算法的净胜分为 10 分，对比 EOBPSO 算法的净胜分为 4 分，对比 CDPSO 算法的净胜分为 7 分。由此可见，HPSO 算法在 10 个基准测试问题上获得了较显著的性能优势。

下面通过分析各算法在 10 个测试函数中获得的均值和方差的排名来评估各算法的准确性和稳定性[11-12]。由于 4 种算法在 10 个测试函数上均获得了求解结果的均值和方差，假设 $U_{Ag}=\{p_1,p_2,\cdots,p_{10}\}$ 表示某算法 Ag 在 10 个测试函数中获得的均值排名集合，那么算法 Ag 的准确性可以通过公式计算而获得，即算法的稳定性可以通过公式 $\mu_{Ag}=\frac{1}{10}\sum_{p\in U_{Ag}} p$

而获得，算法的稳定性通过公式 $\sigma_{Ag}=\frac{1}{10}\sum_{p\in U_{Ag}}(p-\mu_{Ag})^2$ 而获得。图 2 给出了 4 种算法的均值和方差的排名。

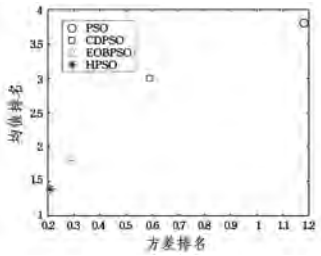

图 2　4 种算法在所有测试函数上的均值与方差的排名

由图 2 可知，本文算法 HPSO 获得的均值和方差排名处于最左下角的位置，即 HPSO 算法获得的均值和方差的排名均优于其他 3 种对比算法；而 EOBPSO 算法的均值排名优于 CDPSO 算法和基本 PSO 算法；CDPSO 算法的均值和方差的排名优于基本 PSO 算法。

进一步地，为检测算法的收敛速度，这里考察各算法在 4 个较难测试的函数上前 100 代所获得的求解结果，通过观察各算法获得结果的变化趋势可以直观地判断出算法收敛速度的快慢。需要说明的是，图 3 所使用的数据均来自各算法在 30 次独立运行中首次获得的结果。

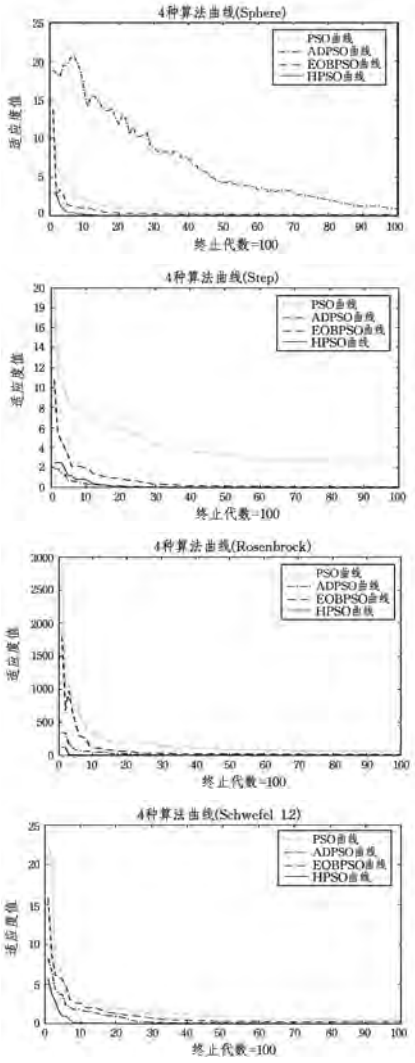

图 3　4 种算法在 4 个测试函数上的收敛性曲线

从图 3 可以看出，HPSO 算法在 Sphere 函数、Rosenbrock 函数和 Schwefel 1.2 函数上具有相对最快的速度，该算法在前 20 代就即将靠近水平轴。而 HPSO 算法在 Step 函数上的收敛速度仅次于 EOBPSO 算法。因此，HPSO 算法相对其他 3 种 PSO 算法及其变种具有更优越的收敛性能。

结束语　本文提出一种混合多策略的粒子群优化算法 HPSO，以应对复杂优化问题的挑战。HPSO 算法将一般动态反向学习策略引入 PSO 算法框架，通过产生反向种群来增加粒子的多样性以增强算法的全局勘探能力；其次，该算法对种群中较优的部分粒子实施差分进化的变异，使算法能在较好的解点附近进行局部开采，以提高算法求解的精度；最后，HPSO 算法对种群中较差的个体实施柯西变异，以产生远离较劣个体的解点，有利于算法跳出局部最优的“陷阱”。上述 3 种策略有机结合，优势互补，能够较好地改善 PSO 算法的性能。在 10 个经典的测试函数上进行了对比实验，结果表明本文算法具有较好的解题精度和收敛速度，是一种颇具前途的算法。

未来将利用更多、更复杂的测试函数以及实践中的优化问题来全面检验 HPSO 算法的性能，并将一些新型的进化范例引入基本 PSO 算法的框架中，不断提高算法的性能。

参 考 文 献

[1] KENNEDY J，EBERHART R C. Particle Swarm Optimization [C]// Proc. of The IEEE International Conferenceon Neural Networks. Piscataway：IEEE Press，1995：1942-1948.

[2] TANG L X，WANG X P. A Hybrid Multiobjective Evolutionary Algorithm for Multiobjective Optimization Problems[J]. IEEE Trans. on Evolutionary Computation，2013，17(1)：20-46.

[3] RAHNAMAYAN S，TIZHOOSH H R，SALAMA M M A. Opposition-based Differential Evolution[J]. IEEE Trans. on Evolutionary Computation，2008，12(1)：64-79.

[4] 谢承旺，王志杰，夏学文. 应用档案精英学习和反向学习的多目标进化算法[J]. 计算机学报，2017，40(3)：757-772.

[5] 王立平，谢承旺. 一种带反向学习机制的自适应烟花爆炸算法[J]. 计算机科学，2016，43(11A)：103-107.

[6] STORN R，PRICE K. Differential evolution：A simple and efficient adaptive scheme for global optimization over Continuous spaces：TR-95-012[R]. ICSI，USA，1995.

[7] 周新宇，吴志健，王晖，等. 一种精英反向学习的粒子群优化算法[J]. 电子学报，2013，41(8)：1647-1652.

[8] YAO X，LIU Y，LIN G M. Evolutionary Programming Made Faster[J]. IEEE Trans. on Evolutionary Computation，1999，3(2)：82-102.

[9] 康岚兰，董文永，田降森. 一种自适应柯西变异的反向学习粒子群优化算法[J]. 计算机科学，2015，42(10)：226-230.

[10] 池元成，方杰，蔡国飙. 基于差分进化和粒子群优化算法的混合优化算法[J]. 计算机工程与设计，2009，30(12)：2963-2980.

[11] 谢承旺，许雷，赵怀瑞，等. 应用精英反向学习的多目标烟花爆炸算法[J]. 电子学报，2016，44(5)：1180-1188.

[12] 胡旺，YEN G G，张鑫. 基于 Pareto 熵的多目标粒子群优化算法[J]. 软件学报，2014，25(5)：1025-1050.