基于异维变异的差分混合粒子群算法

李 俊[1,2] 罗阳坤[1,2] 李 波[1,2] 李乔木[2,3]
(武汉科技大学计算机科学与技术学院 武汉 430065)[1]
(智能信息处理与实时工业系统湖北省重点实验室 武汉 430065)[2]
(武汉科技大学城市建设学院 武汉 430065)[3]

摘 要 针对粒子群(Particle Swarm Optimization,PSO)算法和差分进化(Differential Evolution,DE)算法存在容易陷入局部极值、进化后期收敛速度慢和收敛精度低的局限性,提出了一种基于异维变异的差分混合粒子群(UDEPSO)算法。首先,为了提高群体多样性,使用熵度量初始化粒子;其次,在粒子迭代的过程中,根据粒子的分布特点,引入异维变异学习策略和维度因子以引导粒子及时跳出局部极值达到最优解;最后,将所提算法在10个典型的测试函数上进行了仿真,其在9个测试函数的收敛精度和标准差上取得了显著的效果,远优于PSO算法、DEPSO算法以及CDEPSO算法。实验结果表明,UDEPSO算法在优化收敛精度和效率上具有较强的优势。

关键词 熵,异维变异,维度因子,粒子群差分混合算法
中图法分类号 TP301.6 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.05.035

Differential Hybrid Particle Swarm Optimization Algorithm Based on Different Dimensional Variation

LI Jun[1,2] LUO Yang-kun[1,2] LI Bo[1,2] LI Qiao-mu[2,3]
(College of Computer Science and Technology,Wuhan University of Science and Technology,Wuhan 430065,China)[1]
(Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System,Wuhan 430065,China)[2]
(School of Urban Construction,Wuhan University of Science and Technology,Wuhan 430065,China)[3]

Abstract Considering the limitations that particle swarm optimization(PSO) algorithm and differential evolution(DE) algorithm are difficult to control the initial distribution of the particles and easily fall into local optimum to reduce the convergence accuracy at later process,a differential hybrid particle swarm optimization algorithm based on the different dimensional variation was proposed. Firstly,in order to improve the diversity of particle swarm,the entropy measure me-thod was introduced to initialize particles. Secondly,during the process of particle iteration,learning strategy of different dimensional variation and dimension factors were adopted to guide the immersed particles to jump out of the local optimum to reach the best solution in a timely manner according to the particle distribution. Finally,this algorithm was simulated on ten typical test functions. The convergence precision and standard deviation show the superior performance of the proposed method on nine testing functions comparing with PSO,DEPSO and CDEPSO. These experiments prove that the algorithm has a strong advantage in convergence accuracy and optimization efficiency.

Keywords Entropy,Different dimensional variation,Dimensionality factor,Differential evolution-particle swarm optimization algorithm

粒子群优化算法(PSO)是由Kennedy等[1]于1995年受鸟群觅食行为启发而提出的一种全局的优化进化算法,其主要思想是通过个体之间的合作和信息共享来实现对目标问题的求解。该算法具有概念简单、控制参数少、收敛速度快和易于编程实现等优点,已经被广泛应用于多个领域[2-3]。但粒子群(PSO)算法在早熟收敛和收敛速度上存在严重的缺陷,针对这两个方面的改进方法和策略随之诞生,其中最主要的是对权重进行改进以增强扰动效果并结合策略以扩充群体多样性。文献[4]提出以灰色为主的粒子群优化惯性权重和加速系数策略,以灰色关联度来评价惯性权重和加速度系数,提升了算法初期的搜索能力和后期的发掘能力。文献[5]提出了反向点和反向解的概念,通过扩大搜索区域的范围,增强了算法的全局探索能力。文献[6]在反向学习的基础上提出了精英的概念,利用精英粒子的反向解大大提升了种群多样性和算法的全局搜索能力。

差分进化算法是由Price[7]首先提出来的一种基于种群并行随机搜索的新型进化算法。该算法从原始种群开始,通过变异、杂交、选择几种遗传操作来衍生新的种群,然后通过

到稿日期:2017-03-09 返修日期:2017-06-05 本文受国家自然科学基金(61572381)资助。
李 俊(1978-),男,博士,副教授,主要研究方向为智能计算,E-mail:lijun@wust.edu.cn(通信作者);罗阳坤(1992-),男,硕士生,主要研究方向为智能计算;李 波(1975-),男,博士,教授,主要研究方向为智能计算、机器学习;李乔木(1996-),男,主要研究方向为智能计算。

逐步迭代不断进化,从而实现全局最优解的搜索。经过一系列学者的研究和优化[8-9],差分进化算法已经逐步完善,并在函数优化和实际工程领域中取得了较好的应用。文献[10]将差分进化算法用于优化大数据问题,将大问题分解成若干个小问题,从而有效地解决了演化算法所面临的可扩展性问题。但是其缺陷仍不可置否,即在优化迭代后期易陷入局部极值并使效率降低。

杜松等[12]研究了模拟退火算法,并引入了差分进化算法,从而一定程度上提高了算法的全局优化能力;胡旺[13]引入熵的概念来平衡粒子群算法在解决从单目标问题扩展到多目标问题时的全局和个体最优解;周殊等[14]将 PSO 算法和 GA 算法相结合,以获取更好的收敛性以及更强的连续空间搜索能力;栾丽君等[17]认真研究了 PSO 算法和 DE 算法,结合寻优算法中 PSO 算法前期收敛速度快和 DE 算法求解精度高的特点,引入了一种消息传递机制将 DE 算法中的优秀个体及时更新到算法中,以避免算法过早陷入局部极值;阳春华等[18]在初始化方面做了一些拓展,针对粒子在随机初始化中存在不确定因素干扰的困扰,引入了混沌机制,提高了算法的局部搜索能力。但是以上几种算法只是简单地结合两个算法的优点而未做更深一步的优化操作,未能针对性地解决算法本身的局限性,也未能从根本上对算法进行优化。

基于此,本文在 PSO 和 DE 混合算法的基础上,提出了一种基于异维变异的差分混合粒子群算法 UDEPSO(Updated Differential Evolution Particle Swarm Optimization)。该算法采用熵度量的方式筛选达到均匀分布的粒子群体从而提高群体多样性,即初始化粒子;然后根据粒子分布的特点,在差分粒子群混合算法中结合异维变异策略并引入维度因子,以确保即便部分粒子陷入局部极值,算法也能及时跳出循环,从而提高算法后期的精度和效率。

1　粒子群-差分优化算法

差分粒子群混合优化算法是基于无免费午餐理论而提出的一种新的优化策略,即将原有的差分进化算法和基本的粒子群算法各自的优点相结合,以弥补单一算法求解全局优化问题时的缺陷,从而提高效率和收敛精度。

1.1　基本粒子群

PSO 算法[1]是一种基于种群的优化智能算法,所有的粒子都有一个由被优化的函数决定的适应值,以评价该粒子当前位置的优劣。粒子通过自身的迭代找到最优解,在每一次的迭代过程中,粒子通过追逐个体极值 *pbest* 和全局极值 *gbest* 来更新自己的位置。

设由 N 个粒子组成一个群体,在 D 维搜索空间中,向量 $x_i=(x_{i1},x_{i2},\cdots,x_{id})(i=1,2,\cdots,N)$ 表示第 i 个粒子在搜索空间中的位置,向量 $v_i=(v_{i1},v_{i2},\cdots,v_{id})$ 表示粒子的速度,第 i 个粒子搜索到的最优位置为 $p_i=(p_{i1},p_{i2},\cdots,p_{id})$,整个粒子群能搜索到的最优位置为 $p_g=(p_{g1},p_{g2},\cdots,p_{gd})$,则粒子的变化方程如下:

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

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t+1) \tag{2}$$

其中,$d=1,2,\cdots,D$;w 是非负常数,表示惯性权重;c_1 和 c_2 为学习因子;r_1 和 r_2 为在(0,1)两个范围内变化的随机函数;t 为当前迭代次数。式(1)的第 1 部分为粒子先前的速度;第 2 部分为“认知”部分,引导粒子向自身经历最优的方向运动;第 3 部分为“社会”部分,引导粒子向全局最优的方向运动。

1.2　差分进化

DE 算法是由 Stom 和 Price[7]于 1995 年提出的一类求解连续全局优化问题的演化算法。其基本思想是运用当前种群个体的差来重组得到中间种群,并将其与该群体中的第三个个体向量相加得到一个变异个体向量,然后运用直接的父子混合个体适应值来竞争获得新一代种群。

经典的 DE/rand/1/bin 算法的步骤如下:

步骤 1(初始化)　种群中的每个个体向量 X_i 可由式(3)的方式生成:

$$x_{ij}=x_j^{\min}+rand(0,1)\times(x_j^{\max}-x_j^{\min}) \tag{3}$$

其中,$x_i=(x_{i1},x_{i2},\cdots,x_{id})(i=1,2,\cdots,NP)$,$D$ 表示问题的维数,NP 表示种群的规模,$x_j^{\max}$ 和 $x_j^{\min}$ 分别表示个体向量第 j 维分量的上界和下界,$rand(0,1)$ 表示生成[0,1]之间的均匀分布随机数。

步骤 2(变异)　针对当前代中的每一个个体 x_i,从种群中随机选择 3 个个体向量 x_a,x_b,x_c,其中 $a,b,c\in[1,2,\cdots,NP]$,$a\neq b\neq c\neq i$,按照式(4)进行差分变异操作以生成变异个体 x_i:

$$x_{ij}=x_{aj}+F\cdot(x_{bj}-x_{cj}) \tag{4}$$

其中,$i=1,2,\cdots,NP$;$j=1,2,\cdots,D$;$F\in[0,2]$表示差分变异的缩放因子,用于控制差分向量(x_b-x_c)的缩放程度。

步骤 3(杂交)　采用随机重组变异个体和目标个体各维分量的方式来产生交叉个体,其目的在于提高种群的多样性。

$$U_{ij}=\begin{cases}v_{ij}, & \text{if } rand(0,1)\leqslant C_r \text{ or } j==j_{rand}\\ x_{ij}, & \text{others}\end{cases} \tag{5}$$

其中,j_{rand} 为[1,2,…,NP]之间的随机数,用于确保交叉个体至少有一维分量与目标个体不同;C_r 为交叉概率,$C_r\in[0,1]$。通常随着 C_r 的增大,算法的收敛速度加快,但经验表明 C_r 超过一定取值后,算法的收敛速度会下降。通常 C_r 的取值范围为[0.8,1]。

步骤 4(选择)　令

$$x_i(t+1)=\begin{cases}U_i(t), & \text{if } J(U_i(t))>J(X_i(t))\\ X_i(t), & \text{others}\end{cases} \tag{6}$$

其中,$J(X)$是个体 X 的适应度函数。

步骤 5(终止检验)　令由步骤 2 所产生的新种群为:

$$x(t+1)=(x_1(t+1),x_2(t+1),\cdots,x_N(t+1)) \tag{7}$$

并记 $x(t+1)$ 中的最优个体为 $x_{\text{best}}(t+1)$。如果满足精度要求或者达到进化时限,则停止并输出 $x_{\text{best}}(t+1)$作为近似解,否则置 $t=t+1$,并转步骤 2。

2　异维变异差分粒子群算法

2.1　初始化粒子

不论是粒子群算法还是差分算法,粒子的初始化都是随机分布在区域中,这将导致种群的多样性难以控制,随机性太强。如果初始群体集中分布在局部区域,则群体的多样性较

差，算法的搜索时间较长且难以找到最优值。而如果可以在初始化时就先控制好粒子的群体分布情况，让其尽可能均匀地分布在解空间中，那么将能够一定程度地辅助后面的优化。

设种群中已有 $n-1$ 个粒子满足要求，则第 n 个粒子在第 i 维的熵 s_i 为：

$$s_i=\sum_{j=1}^{n-1}(-\ln G_{jn}^i) \tag{8}$$

$$G_{mn}^i=1-\frac{|x_m^i-x_n^i|}{|C_{\max}^i-C_{\min}^i|} \tag{9}$$

其中，$C_{\max}^i$ 和 $C_{\min}^i$ 分别表示个体第 i 个分量的上界和下界；n 为群体中已经包含的粒子数目 $n\leqslant N$，N 为群体的规模；G_{mn}^i 表示第 m 个粒子和第 n 个粒子的第 i 个分量相同的概率；x_m^i 表示粒子 m 第 i 维的取值，x_n^i 表示粒子 n 第 i 维的取值。因此，两个粒子的第 i 维取值越近，G_{mn}^i 越大；当第 i 维取值非常接近，即 $G_{mn}^i\to 1$ 时，$\ln G_{mn}^i\to 0$，$\lim\limits_{G_{mn}^i\to 1}(-\ln G_{mn}^i)=0$；反之，当 $G_{mn}^i\to 0$ 时，$\ln G_{mn}^i\to -\infty$，$\lim\limits_{G_{mn}^i\to 0}(-\ln G_{mn}^i)=\infty$。群体的熵的定义为：群体中所有个体第 j 维熵的均值[15]。其算法流程如图 1 所示。

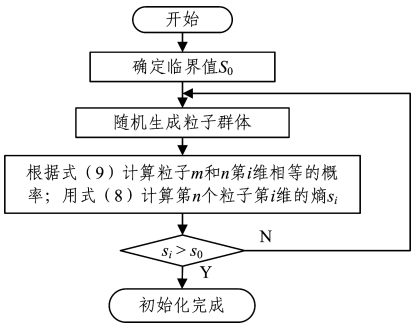


图 1 粒子初始化的流程图

Fig. 1 Flow chart of particle initialization

2.2 异维变异学习

定义 1（异维变异学习（Different Dimensional Variation Learning，DDVL））在一个 D 维搜索空间中，随机选出第 j 维和第 m 维，$j,m\in\{1,2,\cdots,D\}$，且 $m\neq j$；然后从种群中随机选择 3 个个体向量 x_a,x_b,x_c，且有 $a,b,c\in[1,2,\cdots,NP]$，$a\neq b\neq c$，进行变异操作以生成变异个体 x_i，则异维变异学习公式为：

$$x_{ij}=x_{am}+\varphi\cdot(x_{bj}-x_{cj}) \tag{10}$$

异维变异学习（DDVL）的本质就是通过 DE 算法引入一种新的信息交流机制，从而达到模拟 PSO 算法中粒子认知学习和社会学习的效果，增强粒子对错误信息的判断能力，降低陷入局部极值的概率。

相比于传统的 DE/rand/1/bin 算法，式(4)所采用的搜索策略是面向随机邻域下同一维之间的学习。对于某一维而言，父代位置的分布范围将直接影响子代的分布范围，当迭代到一定的次数后，父代的分布范围逐渐缩小并集中到某一个区域，这会使产生的子代集中在一个很小的范围区域内，由于粒子不具备“跳跃”能力，子代将在原始位置附近缓慢搜索，粒子在短时间内（有限的迭代次数或者固定的精度）难以逃脱此区域，这就是在原始算法中粒子在算法后期会陷入局部极值的原因。

若存在一种机制可以促进粒子在不同范围内进行跳跃且维持在既有社会群体之间，即使粒子在既定的 D 维之间具备跨越性的搜索能力，从而使粒子不再局限于父代的直接邻域，粒子间不再是限于自身位置附近的邻域学习，而是当前维在短时间内可以在较大的范围内进行尝试，则既可以充分发挥粒子本身的认知学习能力，又可以有效利用当前所处位置的社会学习环境，有效地解决子代受父代位置约束的问题，从而增强算法的全局搜索能力，利于算法逃离局部最优。同时由于进行了变异操作，将会产生双模式下的新位置，从而显著地提高种群的多样性，避免算法后期在邻近区域进行单一搜索，使算法更快地摆脱局部极值的困境。

相比于文献[17-18]简单地结合两种优化算法，本文使用异维学习可以更深层次地弥补单个算法中固有的缺陷，能明显提高全局搜索能力并增强搜索精度。

2.3 改进的差分混合粒子群算法 UDEPSO

针对混合算法，为了提高混合算法中 PSO 算法的优化性能，提出了一种非线性惯性权重策略，惯性权重的更新公式如下：

$$w=w_{\min}+(w_{\max}-w_{\min})*\exp[-\lambda*\frac{iter}{Iter}] \tag{11}$$

其中，λ 为控制因子，取值范围为[2,3]；$iter$ 为当前迭代次数，$Iter$ 为总迭代次数；$W_{\min}$ 和 $W_{\max}$ 是惯性权重的两个临界值。考虑到粒子的特性以及算法本身的搜索能力，引入一种概率选择方式动态地选择需要执行的算法，算法描述如下。

1. 随机生成 NP 个粒子作为初始种群 p；
2. 按照图 1 中的初始化流程对随机生成的粒子进行筛选，最后将满足要求的粒子作为迭代的种群 p_0；
3. for t=1 to Iter do
4. if rand<pg，动态更换概率
5. //执行差分策略
6. for i=1 to NP
7. 根据式(5)和式(10)执行异维变异策略；
8. 根据式(6)进行边界处理；
9. endfor
10. else
11. //执行粒子群算法
12. for i=1 to NP
13. 动态更新最大速度和最小速度；
14. 根据式(1)更新速度，根据式(2)更新位置；
15. 计算适应度值并评估和筛选；
16. endfor
17. endif
18. 筛选并更新个体极值和群体极值；
19. 如果达到最大迭代次数或满足精度要求，则输出 g_{best}；
20. endfor

3 仿真实验及分析

3.1 参数的设置

将提出的 UDEPSO 算法与标准粒子群优化（PSO）算法、

差分粒子群混合(DEPSO)算法[17]以及混沌差分粒子群混合(CDEPSO)算法[18]进行对比。实验参数综合参照文献[6,17],具体设置如下:粒子数 $NP=40$,维数 $D=30$,最大评估次数为 200000,迭代次数为 5000 次,$c_1=1.193$,$c_2=1.193$,$C_r=0.8$,$F=0.5$,执行改进差分的概率 $pg=0.2$(此值是经过多次实验测试之后最终选定的一个概率值)。在标准粒子群算法中固定惯性权重 $w=0.712$,DEPSO 中惯性权重 $w_{max}=0.9$,$w_{min}=0.4$;粒子最大速度 v_{max} 为搜索空间的一半,在运行过程中速度动态变化;算法适应度的评估次数为 200000,即迭代次数 *Iter* 为 5000。DEPSO 算法和 CDEPSO 算法的参数分别参照文献[17]和文献[18]设置。为了避免算法的随机性,所有算法均运行 30 次,取运行结果的平均值进行比较。

3.2　测试函数

实验中算法采用 Matlab2012b 实现。运行环境如下:Win7 双核,4 GB 内存,3.2 GHz CPU。将改进的算法与标准的粒子群优化算法(PSO)、差分混合粒子群算法(DEPSO)[17]和混沌差分粒子群混合算法(CDEPSO)[18]进行对比。实验选取的 10 个 Benchmark 函数均来自文献[19],如表 1 所列。其中,$f1-f2$ 为单峰函数,$f3-f10$ 为多峰函数。这 10 个测试函数的全局最优值均为 0,维数均为 30 维。

表 1　实验中使用的 10 个测试函数

Table 1　Ten test functions used in experiment

	函数名	维数	变量范围	最优值
$f1$	Sphere	30	[−100,100]	0
$f2$	Schwefel 2.22	30	[−10,10]	0
$f3$	Quadric	30	[−100,100]	0
$f4$	Step	30	[−100,100]	0
$f5$	Quadric Noise	30	[−1.28,1.28]	0
$f6$	Rastrigin	30	[−5.12,5.12]	0
$f7$	Nocontinuous Rastrigin	30	[−5.12,5.12]	0
$f8$	Achley	30	[−32,32]	0
$f9$	Griewangk	30	[−600,600]	0
$f10$	Generalized Penalized	30	[−50,50]	0

3.3　实验结果及分析

本文主要采用以下两种方法进行算法性能的评估:1)固定评价次数,评估算法的收敛精度、运行时间和收敛速度;2)固定收敛精度目标值,评估算法达到该精度目标所需要的评估次数。

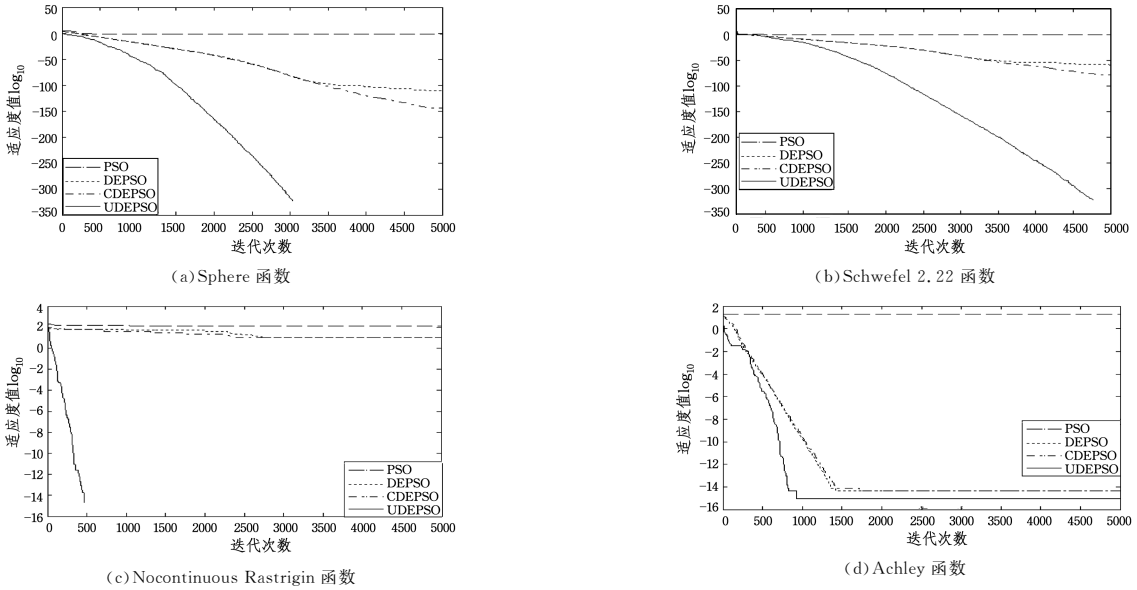
3.3.1　固定评价次数下的收敛精度和运行时间

表 2 列出了固定评价次数时 4 种算法的实验结果。由于算法的收敛速度较快,为了清晰地显示实验结果,适应度取以 10 为底的对数,限于篇幅,图 2 给出了 4 种算法在部分具有代表性的函数上的收敛效果。最后对算法的时间复杂度进行分析。

表 2　固定迭代次数时 4 种算法在 10 个测试函数上的均值、标准差和运行时间

Table 2　Mean, standard deviation and running time of four algorithms on ten test functions measured in fixed iteration times

函数	PSO			DEPSO			CDEPSO			UDEPSO		
	mean	std. dev	time	mean	std. dev	time	mean	std. dev	time	mean	std. dev	time
$f1$	2.10E−03	9.79E−05	2.90	1.48E−105	9.27E−107	6.91	3.51E−141	3.26E−148	7.23	**0**	0	3.94
$f2$	1.41E−01	7.06E+00	3.13	7.56E−58	9.44E−57	6.97	2.82E−71	5.35E−76	7.66	**0**	0	4.19
$f3$	8.70E−03	3.56E−02	4.92	3.39E−25	3.82E−24	9.91	1.27E−35	6.78E−39	10.8	**0**	0	5.98
$f4$	2.58E−03	2.76E−03	2.98	0	0	6.38	0	0	7.03	**0**	0	3.91
$f5$	6.17E−03	9.67E−03	3.47	4.15E−04	4.22E−05	7.48	3.95E−04	3.03E−04	8.27	**1.99E−05**	2.87E−06	4.48
$f6$	1.18E+01	3.74E+00	2.23	6.27E+00	1.41E+00	5.51	5.94E+00	2.11E+00	6.08	**0**	0	3.06
$f7$	4.20E+01	1.84E+01	3.24	9.57E+00	4.95E+00	6.95	9.03E+00	0	7.67	**0**	0	3.92
$f8$	1.77E+01	2.78E−01	3.67	4.44E−15	0	7.39	4.44E−15	0	8.22	**8.88E−16**	0	4.42
$f9$	1.80E−04	6.10E−02	3.58	0	0	7.68	3.29E−04	0	8.03	**0**	0	4.35
$f10$	8.63E+01	7.26E+00	5.29	2.36E−32	0	11.14	2.36E−32	0	11.23	9.90E−13	2.18E−15	6.18



(a) Sphere 函数　　(b) Schwefel 2.22 函数

(c) Nocontinuous Rastrigin 函数　　(d) Achley 函数

图 2　4 种算法在部分测试函数上的收敛曲线

Fig. 2　Convergence curve of four algorithms on some test functions

从表 2 中可以清楚地看到，对于算法的收敛精度而言，DEPSO 算法[17]在 10 个测试函数上均起到了优化效果，从侧面证明了混合算法在一定程度上优于单一算法的理论。同样 CDEPSO 算法[18]在 10 个测试函数上的效果均优于单一的 PSO 算法，同时由于添加了混沌、变异等操作，增强了算法的局部搜索能力，在 $f1$，$f2$，$f3$ 上的寻优效果更好。而改进算法(UDEPSO)在 10 个测试函数上除 $f10$ 外均超过了 CDEPSO 算法的效果，在 7 个函数上均达到了最优解 0，且在病态多峰函数($f5$ 和 $f8$)上效果有了一定的提升。结果表明，改进的混合算法具有较好的寻优能力。

利用 10 个测试函数进行实验测试，DEPSO 算法所得结果的标准差比 PSO 算法所得结果的标准差都小。同时，CDEPSO 算法的标准差小于 DEPSO(除了函数 $f5$，$f6$，$f7$)，UDEPSO 算法的标准差基本达到了 0(除了 $f5$ 和 $f10$，在 $f5$ 上也优于其他算法)。由于标准差可以在一定程度上比较客观地评测一个算法的稳定性和鲁棒性，即标准差值越小，算法的性能越好，算法越强壮。因此，改进的混合算法具有较强的稳定性和鲁棒性。

寻优时间指在固定且有限的迭代次数下进行对比，即在 200000 次评价中，算法完成整个过程所消耗的时间。由于差分算法(DE)中的粒子需要进行复杂的交叉变异操作，因此在时间消耗上高于单纯的粒子群算法(PSO)；而 CDEPSO 算法由于引入了混沌机制，因此需要额外消耗一部分时间，故它的耗时稍高于 DEPSO 算法。虽然改进算法在初始化时使用了熵度量的方式筛选粒子耗费了时间，但在迭代过程中考虑了粒子的分布特点，即种群中只有较小部分粒子随着时间的推进会逐渐陷入局部极值而无法自动跳出循环；而对于大部分经过“筛选”之后的粒子，粒子群算法的寻优能力已然满足。因此，本文采用策略优化对大部分粒子进行基本的粒子群算法引导，而对另一部分粒子进行差分变异操作，并且在这部分粒子中执行异维学习策略，确保了粒子可以在较短的时间内跳出局部邻域，从而增加了全局搜索能力，避免粒子陷入极值。下面对算法的时间复杂度进行分析。

假设种群规模为 NP，问题维数为 D，迭代次数为 T，PSO 算法主要包括对参数的初始化和对速度和位置的更新，因此时间复杂度为 $O(T \cdot NP \cdot D)$；DE 算法中也有参数的初始化，但是该算法主要将时间消耗在变异、交叉和选择上，因此 DE 算法的时间复杂度为 $O(T \cdot NP \cdot D)$。

根据 2.3 节中的算法步骤分析 UDEPSO 算法的时间复杂度。算法中第 3－20 步为迭代，因此算法的复杂度为 $O(n)$，依赖迭代循环。迭代循环中最复杂且最耗时的部分为第 4－15 步，所需时间为 $T_{混和}$。其中执行 if…else…的时间为 T_0；第 5－9 步执行差分策略，所需时间为 $T_{异维}$；第 11－16 步执行粒子群算法，所需时间为 $T_{粒子群}$。对于每个个体，执行异维变异及杂交的时间为 $D * T_1$，NP 个个体的总时间为 $T_{异维} = NP * D * T_1$。位置、速度及适应度的更新时间为 T_2，$T_{粒子群} = NP * D * T_2$。

分析第 4－17 步可知 $T_{混和} = T_0 + pg * NP * D * T_1 + (1-pg) * NP * D * T_2$，即 $T_{混和} - T_{粒子群} = NP * D * pg * (T_1 - T_2) + T_0$。由上式可得，$O(T_{混和}) = O(T_{粒子群}) = O(NP \cdot D)$。综合迭代循环，UDEPSO 算法的时间复杂度为 $O(T \cdot NP \cdot D)$。

虽然 4 种算法的时间复杂度一样，但 UDEPSO 算法加入了概率选择，异维变异时需要选择维度和个体，因此 UDEPSO 算法消耗的总时间比 PSO 算法多。由表 2 得证。

3.3.2 收敛速度分析

图 2 给出了 4 个函数 30 维的进化过程曲线图，其中图 2(a)和图 2(b)为单模态函数，图 2(c)和图 2(d)为多模态函数，横轴为迭代次数(评估次数＝迭代次数 * NP)，纵轴为适应度值的对数。由收敛曲线可以看出，与 PSO 相比，DEPSO 和 CDEPSO 在收敛速度上有非常明显的提高，但精度不够；而 UDEPSO 不仅最终找到了最优解，同时拥有最快的收敛速度。同时，图 2 表明，不论对于单峰函数还是多峰函数，UDEPSO 算法的收敛性能均优于对比算法。

3.3.3 指定精度下的评价次数

在总评价次数为 200000，其他参数设置不变的前提下(因为 $NP=40$，即迭代次数为 5000)，比较 DEPSO 算法、CDEPSO 算法和 UDEPSO 算法在达到指定的精度 10E－15 时所需的评价次数(单纯的 PSO 算法无法达到该精度，因此不列入对比对象)。对 10 个测试函数进行了测试，算法独立测试 30 次后的平均评价次数如表 3 所列，在部分测试函数上的收敛效果如图 2 所示。

表 3 给定目标精度时的平均评价次数

Table 3 Average evaluation number with given target accuracy

(单位：次)

函数名	DEPSO	CDEPSO	UDEPSO
$f1$	31600	31400	21200
$f2$	52640	52400	33480
$f3$	107480	99040	33400
$f4$	31560	31480	22320
$f5$	200000	200000	200000
$f6$	200000	200000	23600
$f7$	200000	200000	26360
$f8$	54480	54280	32900
$f9$	35336	34993	20720
$f10$	29320	29440	198040

由表 3 可以看出，差分和粒子群的混合算法大大提升了整个进化算法的精度。为了更准确地反映算法的寻优能力，表 3 中选定目标精度为 10E－15。可以清楚地看出，DEPSO 算法和 CDEPSO 算法的平均评价次数相差不大，而且在测试函数 $f5$，$f6$，$f7$ 上达到总评价次数(200000 表示当前进化过程出现死循环，无法达到设定的精度要求)，但是改进的 UDEPSO 算法能够很好地解决该问题，对于大部分函数则需要更少的评价次数，只在 $f5$ 和 $f10$ 上由于未达到给定的目标精度而出现“死循环”的现象。

3.3.4 在 30 维上的 Friedman 检验结果

为了在统计意义上比较多个算法的性能，本文在 30 维的基础上采用 Friedman 检验对实验结果进行分析。表 4 列出

了改进算法 UDEPSO 和 4 种经典算法的秩均值。算法性能越好,其秩均值越小。由表 4 可以看出,UDEPSO 算法的总体性能最好。其中 FIPSO[20],CLPSO[21] 和 APSO[22] 都是按照原文的参数进行设置,同样保持在 30 维上进行操作。

表 4　优化算法在 Friedman 检验上的排名

Table 4　Optimization algorithm ranking in Friedman's test

算法	秩均值
DEPSO	5.28
FIPSO	3.88
CLPSO	3.54
APSO	2.67
UDEPSO	0.82

结合表 4 以及图 2 中的结果,无论是在固定评价次数方面,还是在达到给定的固定精度方面,UDEPSO 都能找到最优解零值,同时其寻优速度有了较大的提高,算法的稳定性和鲁棒性也有了一定的提升。

结束语　本文针对 DE 和 PSO 单个算法在求解最优问题时易陷入局部极值以及 DEPSO 混合算法寻优效果不佳的情况,引入熵度量方式对随机初始的粒子进行筛选,然后通过异维变异学习策略引导粒子在陷入局部极值时能够及时跳出循环,以趋于最优解空间,最终提出了一种基于异维变异的差分混合粒子群算法。本文从多个角度对算法进行仿真实验,实验结果表明,算法在优化效率、优化性能方面有了较大的改善。但是影响混合算法的因素众多,如何优化混合算法粒子的初始化和筛选过程以及提高混合算法的协同优化,进一步提升算法的寻优能力、鲁棒性和缩短算法的运行时间,都将是下一步的研究方向。

参 考 文 献

[1] KENNEDY J,EBERHART R. Particle swarm optimization[C]// IEEE Conference on Neural Networks. New York:IEEE Press,1995:1942-1948.

[2] 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.

[3] LIN Z H,GAO W,WU C M,et al. Data Center Network Flow Scheduling Based on DPSO Algorithm[J]. Acta Electronica Sinica,2016,44(9):2197-2202. (in Chinese)
林智华,高文,吴春明,等. 基于离散粒子群算法的数据中心网络流量调度研究[J]. 电子学报,2016,44(9):2197-2202.

[4] LEU M S,YEH M F. Grey particle swarm optimization[J]. Applied Soft Computing,2012,12(9):2985-2996.

[5] WANG H,WU Z,RAHNAMAYAN S,et al. Enhancing particle swarm optimization using generalized opposition-based learning[J]. Information Sciences,2011,181(20):4699-4714.

[6] ZHOU X Y,WU Z J,WANG H,et al. Elite opposition-based particle swarm optimization[J]. Acta Electronica Sinica,2013,41(8):1647-1652. (in Chinese)
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.

[7] PRICE K V. Differential evolution: a fast and simple numerical optimizer[C]// IEEE Conference on North American Fuzzy Information Processing. New York:IEEE Press,1996:524-527.

[8] DAS S,ABRAHAM A,CHAKRABORTY U K,et al. Differential Evolution Using a Neighborhood-based Mutation Operator[J]. IEEE Transactions on Evolutionary Computation,2009,13(3):526-553.

[9] SAYDJARI R,TOWNSEND C M,BARRANCO S C,et al. Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems[J]. Journal of Parallel & Distributed Computing,2013,73(1):62-73.

[10] SABAR N R,ABAWAJY J,YEARWOOD J. Heterogeneous Cooperative Co-Evolution Memetic Differential Evolution Algorithm for Big Data Optimization Problems[J]. IEEE Transactions on Evolutionary Computation,2017,21(2):315-327.

[11] HO Y C,PEPYNE D L. Simple Explanation of the No-Free-Lunch Theorem and Its Implications[J]. Journal of Optimization Theory and Applications,2002,115(3):549-570.

[12] DU S,ZHOU J Y. Hybrid Algorithm of Differential Evolution and Simulated Annealing Particle Swarm Optimization[J]. Computer Simulation,2015,32(12):218-221. (in Chinese)
杜松,周健勇. 一种差分进化和模拟退火粒子群混合算法[J]. 计算机仿真,2015,32(12):218-221.

[13] HU W,YEN G G,ZHANG X. Multiobjective Particle Swarm Optimization Based on Pareto Entropy[J]. Journal of Software,2014,25(5):1025-1050. (in Chinese)
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.

[14] ZHOU S,PAN W,LUO B,et al. A Novel Quantum Genetic Algorithm Based on Particle Swarm Optimization Method and Its Application[J]. Acta Electronica Sinica,2006,34(5):897-901. (in Chinese)
周殊,潘炜,罗斌,等. 一种基于粒子群优化方法的改进量子遗传算法及应用[J]. 电子学报,2006,34(5):897-901.

[15] YANG H F,YANG Y,YANG L H,et al. Swarm optimization algorithm with particle selection and memory[J]. Application Research of Computers,2016,33(4):1039-1043. (in Chinese)
杨华芬,杨有,杨丽华,等. 筛选和记忆相结合的粒子群算法[J]. 计算机应用研究,2016,33(4):1039-1043.

[16] LI B,SUN H,ZHAO J,et al. Artificial bee colony algorithm with different dimensional learning[J]. Application Research of Computers,2016,33(4):1028-1033. (in Chinese)
李冰,孙辉,赵嘉,等. 异维学习人工蜂群算法[J]. 计算机应用研究,2016,33(4):1028-1033.

[17] LUAN L J,TAN L J,NIU B. A Novel Hybrid Global Optimization Algorithm Based on Particle Swarm Optimization and Differential Evolution[J]. Information & Control,2007,36(6):708-714. (in Chinese)
栾丽君,谭立静,牛奔. 一种基于粒子群优化算法和差分进化算法的新型混合全局优化算法[J]. 信息与控制,2007,36(6):708-714.

[18] YANG C H,QIAN X S,GUI W H. Hybrid algorithm of chaotic differential evolution and particle swarm optimization[J]. Application Research of Computers,2011,28(2):439-441. (in Chinese)
阳春华,钱晓山,桂卫华. 一种混沌差分进化和粒子群优化混合算法[J]. 计算机应用研究,2011,28(2):439-441.

[19] LI C,YANG S,NGUYEN T T. A Self-Learning Particle Swarm Optimizer for Global Optimization Problems [J]. IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society,2012,42(3):627-646.

[20] MENDES R,KENNEDY J,NEVES J. The fully informed particle swarm:simpler,maybe better[J]. IEEE Transactions on Evolutionary Computation,2004,8(3):204-210.

[21] LIANG J J,SUGANTHAN P N. Dynamic Multi-Swarm Particle Swarm Optimizer[C]// IEEE Conference on Swarm Intelligence Symposium. New York:IEEE Press,2005:124-129.

[22] ZHAN Z H,ZHANG J,LI Y,et al. Adaptive particle swarm optimization [J]. IEEE Transactions on Systems,Man,and Cybernetics,2009,39(6):1362-1381.

(上接第 207 页)

参 考 文 献

[1] SBALZARINI I F,MÜLLER S,KOUMOUTSAKOS P. Multiobjective optimization using evolutionary algorithms [C] // Summer Program. CiteSeer,2001:63-74.

[2] HUGHES E J. Evolutionary many-objective optimisation:many once or one many? [C] // The 2005 IEEE Congress on Evolutionary Computation,2005. IEEE,2005:222-227.

[3] ZHANG Q,LI H. MOEA/D:A Multiobjective Evolutionary Algorithm Based on Decomposition [J]. IEEE Transactions on Evolutionary Computation,2008,11(6):712-731.

[4] LI Y L,ZHOU Y R,ZHAN Z H,et al. A Primary Theoretical Study on Decomposition-Based Multiobjective Evolutionary Algorithms[J]. IEEE Transactions on Evolutionary Computation,2016,20(4):563-576.

[5] ZHAO S Z,SUGANTHAN P N,ZHANG Q. Decomposition-Based Multiobjective Evolutionary Algorithm With an Ensemble of Neighborhood Sizes[J]. IEEE Transactions on Evolutionary Computation,2012,16(3):442-446.

[6] LI H,DING M,DENG J,et al. On the use of random weights in MOEA/D[C] // Evolutionary Computation. IEEE,2015:978-985.

[7] CHENG R,JIN Y,OLHOFER M,et al. A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization [J]. IEEE Transactions on Evolutionary Computation,2016,20(5):773-791.

[8] XIE C W,DING L X. Study on Selection Strategies of Multiobjective Evolutionary Algorithms [J]. Computer Science,2009,36(9):167-172. (in Chinese)
谢承旺,丁立新. 多目标进化算法中选择策略的研究[J]. 计算机科学,2009,36(9):167-172.

[9] ZHENG J H,ZHANG Z F,ZOU J. An adaptive multi-objective evolutionary algorithm directed by objective space decomposition [J]. Chinese High Technology Letters,2013,23(7):671-678. (in Chinese)
郑金华,张作峰,邹娟. 基于目标空间分解的自适应多目标进化算法[J]. 高技术通讯,2013,23(7):671-678.

[10] WANG Z,ZHANG Q,GONG M,et al. A replacement strategy for balancing convergence and diversity in MOEA/D[C]// IEEE Congress on Evolutionary Computation. IEEE,2014:2132-2139.

[11] LI K,ZHANG Q,KWONG S,et al. Stable Matching-Based Selection in Evolutionary Multiobjective Optimization [J]. IEEE Transactions on Evolutionary Computation,2014,18(6):909-923.

[12] WANG Z,ZHANG Q,ZHOU A,et al. Adaptive Replacement Strategies for MOEA/D[J]. IEEE Transactions on Cybernetics,2015,46(2):474-486.

[13] MUNKRES J. Algorithms for the Assignment and Transportation Problems[J]. Journal of the Society for Industrial & Applied Mathematics,1957,5(1):32-38.

[14] ZHANG Q,ZHOU A,ZHAO S,et al. Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition[J/OL]. http://xueshu.baidu/com/s?wd=paper.

[15] LI H,ZHANG Q. Multiobjective Optimization Problems With Complicated Pareto Sets,MOEA/D and NSGA-II [J]. IEEE Transactions on Evolutionary Computation,2009,13(2):284-302.

[16] HUBAND S,BARONE L,WHILE L,et al. A Scalable Multiobjective Test Problem Toolkit[C]// EMO. 2005:280-295.

[17] DEB K,THIELE L,LAUMANNS M,et al. Scalable Test Problems for Evolutionary Multiobjective Optimization[M]// Evolutionary Multiobjective Optimizatio. 2005:105-145.

[18] ZHANG Q,LIU W,LI H. The performance of a new version of moea/d on cec09 unconstrained mop test instances[C]// Evolutionary Multiobjective,2009. IEEE,2009:203-208.

[19] GENG H T,ZHAO Y G,CHEN Z,et al. Multi-objective Particle Swarm Optimization Algorithm with Balancing Each Speed Coefficient[J]. Computer Science,2016,43(12):248-254. (in Chinese)
耿焕同,赵亚光,陈哲,等. 一种均衡各速度项系数的多目标粒子群优化算法[J]. 计算机科学,2016,43(12):248-254.

[20] DEB K,PRATAP A,AGARWAL S,et al. A fast and elitist multiobjective genetic algorithm:NSGA-II[J]. IEEE Transactions on Evolutionary Computation,2002,6(2):182-197.