A Dynamic Multimodal Differential Evolution Algorithm

Dong XiaoGang[1], Xie Qing[1], Ke Lin[2]

[1)] School of Information Science and Technology, JiuJiang University, Jiujiang 332005(283381913@qq.com)
[2)] College of Science, JiuJiang University, Jiujiang

***Abstract t*—Differential evolution algorithm in solving complex function optimization problems, the problems of convergence rate and precision is not high. At the same time, there is a big difference in the performance of evolutionary algorithms for solving the different types of optimization problems. To solve above two problems, this paper proposes a dynamic multimodal differential evolution algorithm. Firstly, the dynamically population is used to improve the exploration ability of algorithms; In addition, the algorithm uses Four different types of mutation operator to Produce among individuals, choose the best among individuals to enter the next iteration , improved the algorithms's performance of solving different types of optimization problems. Through a variety of BenchMark functions to the algorithm simulation experiment, and comparing and several other classical differential evolution algorithm, show that this algorithm has better optimization performance.**

***Keywords*—Differential evolution, Convergence rate, Mutation Operator**

一种动态多模式差分演化算法

董小刚 [1]，谢清 [1]，柯林 [2]

[1)]九江学院信息科学与技术学院，九江 江西 中国

[2)]九江学院理学院，九江 江西 中国

摘 要 差异演化算法在求解复杂函数优化问题时, 存在收敛速率和求解精度不高的问题;同时, 针对不同类型的优化问题, 差异演化算法的求解性能存在较大差别。针对以上两方面问题，本文提出一种动态多模式差分演化算法。该算法首先采用动态种群的方式，加快了算法的全局搜索能力;另外，算法采用四种特点不同的变异算子产生中间个体，选择最优秀的中间个体进入下次迭代之中，使得算法在求解不同类型的优化问题是都能表现出较好的性能。通过多种 BenchMark 函数对该算法进行仿真实验, 并和其它几种经典差分演化算法进行比较，证明该算法拥有较好的寻优性能。

关键词 差分演化，收敛速率，变异算子

1．引言

差异演化[1] (Differential Evolution DE) 算法是 Rainer stone 和 Kenneth Price 于 1997 年为求解切比雪夫多项式问题而提出的。DE 算法是一种采用实数编码的全局优化算法，运行原理简单、控制参数少，非常易于理解和实现。但是由于算法的参数设置(种群规模、变异算子、交叉算子)存在较大的随机性和盲目性，导致算寻优速率慢、求解精度不高等问题，这些情况在复杂函数的优化中尤为突出。因此，对 DE 算法的改进研究十分必要。

国内外的学者对 DE 算进行了多方面的研究分析，提出了很多种改进的算法。文献[2]提出了一种动态的差异演化算法(Dynamic Differential Evolution, DDE),通过一种动态策略提高了算法的寻优效率，但在对复杂函数优化时其性能仍有待提高。文献[3]提出了一种 SaDE 算法,该算法通过对前期演化的学习,自适应的从策略池中选择算法的变异策略,并取符合正态分布的参数 F 和 CR，从而避免了 DE 算法在进行演化时由于选择适当变异模式和相关参数试错上的消耗；文献[4]提出了一种 JADE 算法，该算法采用一种的新的变异算子及自适应的参数更新模式，通过这两项改进有效的提高了 DE 算法的整体性能；文献[5]提出了一种阶段

This work is supported by National Natural Science Foundation of China under Grant No. 61364025; the State Key Laboratory of Software Engineering (SKLSE) under Grant No. SKLSE2012-09-39; the Science and Technology Foundation of Jiangxi province under Grant No.GJJ13729;the Scientific Research Projects of Jiujiang University under Grant No.2013KJ27.

波动的差分演化算法，该算法通过引入分段思想，在不同阶段采用不同的配置利用所设计的算子生成动态变异率，并使之波动，从而加快了算法的收敛速率，改善了算法的性能;上述对 DE 算法的改进都在不同程度上提高了算法的寻优能力。然而，在某些情况下（求解多种类型不同的优化问题时）DE 算法的性能任然存在提升的空间。

本文提出一种动态多策略的差分演化算法（Dynamic and Multi-Strategy Differential Evolution DMDE）。该算法通过动态化的种群，及时的把较为优秀的新个体引入到当代演化中去，提升了 DE 算法的全局寻优能力；另外，算法利用多种策略构建变异策略池进行算法的变异操作，然后择优选取最为优秀中间个体，有效的提高了算法对不同优化问题的求解性能,降低了算对单一变异算子的敏感性。

2．基本 DE 算法

2.1 算法简介

DE 算法是一种基于实数编码的演化算法，其主要操作包括随机产生初始种群、个体变异，交叉以及选择。

（1）初始种群：算法首先在一个 D 维空间上随机的产生 NP 个受上下界约束的个体 Xi0,i∈[1,NP]。

（2）变异操作：DE 算法的变异操作是利用当前种群中两个随机选择的个体之间差向量产生新的中间个体（记为 Xi,g+1）。DE 算法最常采用的两种经典变异策略是 DE/rand/1/bin 和 DE/best/2/bin,分别具体描述如下：

①DE/rand/1/bin：

$$v_{i,g+1} = x_{r1,g} + F\bullet(x_{r2,g} - x_{r3,g}) \tag{1}$$

②DE/best/2/bin：

$$v_{i,g+1} = x_{best,g} + F\bullet(x_{r1,g} - x_{r2,g}) + F\bullet(x_{r3,g} - x_{r4,g}) \tag{2}$$

其中 r1≠r2≠r3≠r4≠i, r1, r2, r3, r4∈{1, NP}。变异策略（1）具备较强的全局搜索能力。变异策略（2）具备较高的收敛速率。

（3）交叉操作

交叉操作是将中间个体$V_{i,g+1}$与$X_{i,g}$进行杂交产生候选个体$U_{i,g+1}$。基本DE算法往往采用二项式杂交模式,来得到候选个体，在杂交的过程中保证至少有一位是从$V_{i,g+1}$获得，其它位按照交叉概率CR决定来之$X_{i,g+1}$或者是$X_{i,g}$。其具体定义如公式（3）：

$$u_{ij,g+1} = \begin{cases} V_{ij,g+1} & if\ \ rand < CR\ \ or\ \ j = R(i) \\ x_{ij,g} & otherwise \end{cases} \tag{3}$$

其中，i = 1,2,…,NP; j = 1,2,…,D, rand ∈[0,1]是一个均匀分布的随机数；R(i)是[1,D]之间的随机整数； CR∈[0,1]为交叉概率。

（4）选择操作：

DE算法的选择操作采用了“贪梦”的方式，就是让变异和交叉所得的$U_{i,g+1}$和当前种群的$X_{i,g}$按照目标函数的适应度值进行竞争，从而决定更为优秀的个体进入下一代种群。其具体定义如(4)式所示：

$$x_{i,g+1} = \begin{cases} u_{i,g+1} & if\ \ f(u_{i,g+1}) \le f(x_{i,g}) \\ x_{i,g} & otherwise \end{cases} \tag{4}$$

2.2 算法分析

DE 算法的演化过程中，每一代演化时当前种群保持不变，这一特征使得每一次迭代新产生的优秀个体无法及时的进入当代演化中，这在一定程度上限制了算法对新产生的优秀个体所携带信息的及时利用，降低了算法的全局寻优能力；另外，传统的 DE 算法以及大多数改进的 DE 算法，往往采用固定的变异算子或者按照一定的概率来选择某种变异算子进行变异操作，产生中间个体。然而，特定的某种变异算子往往只在求解某些相似类型的优化问题时表现优秀，在其它不同类型的优化问题上往往性能不佳。所以研究如何提高 DE 算在求解不同类型问题时等寻优性能是非常必要的。

3．动态多策略的差分演化算法

3.1 动态种群的迭代策略

传统DE算法的演化过程中，每次迭代中，当前种群（记为pop）一直保持不变，直到一次迭代的完成。也就说，当前个体Xi经过变异、交叉和选择操作之后，不管新产生的Xi,g+1是否比Xi更为优秀，它只有等到下次迭代时才可以参与演化之中去。假设Xi,g+1的确比Xi更为优秀，那么这种当前种群保持不变的静态种群的演化策略，在同一代的演化中，对Xi之后的个体进行演化时就无法及时的将优秀个体Xi,g+1所携带的优秀基因利用起来。这就大大的限制了DE算法的全局寻优能力。为此，本文改进传统DE算法每次迭代种群保持不变的模式，采用一种动态种群的模式。动态种群的演化策略是指在每次迭代的过程中，Xi经过变异、交叉及选择操作之后产生新个体（暂且记为Xnew），如果Xnew评价后其适应度优于Xi，则对当前种群pop进行更新，即用Xnew替换当前种群pop中的对应个体Xi。这样在Xi之后的个体进行演化时就可以将Xnew所携带的优秀基因及时的在本轮迭代中利用起来，加快了算法的全局收敛速率。

3.2 多模式变异池

DE算法的演化利用具体的变异算子对个体进行变异产生中间个体。然而，充分的实验表明，不同类型的优化问题往往对变异算子是敏感的，同一变算子不可能在所有优化问题中表现出同样优秀的性能。甚至，对于一个具体的优化问题的求解，在演化的过程的不同阶段变异算子的性能也表现出不同的性能。

为了提高算法对不同优化问题搜索性能，避免单一变异策略无法满足多种有问题的缺陷，我们提出建立一个由多种变异模式构建的变异池，本文采用4个变异策略构建变异池。在算法的每次迭代中分别采用4种策略对当前个体$X_{i,g}$进行变异和交叉操作，依次产生4个候选个体，之后从4个候选个体中选择最为优秀的候选个体$u_{i,g+1}^{best}$作为最终的候选个体，并让其和当前个体$X_{i,g}$进行选择则操作,选择二者之中更为优秀的进入下次迭代。这种结余多模式变异池的演化策略使DE算法在对不同优化问题进行演化的各个阶段都可以自适应选择一种最为有效的变异策略，从而可以较好的避免单一变异策略无法满足多种优化问题的缺陷。我们对变异池中变异策略的构成进行了详尽的分析和筛选，最终选择了四种不同特点的变异策略构成算法的变异池。四种变异策略分别是：前文2.1节所述的公式1变异策略 DE/rand/1/bin、公式5变异策略DE/best/1、公式6变异策略DE/rand-to-best/1以及式7所示的一种混合型变异策略。

$$v_{i,g+1} = x_{best,g} + F\bullet(x_{r1,g} - x_{r2,g}) \tag{5}$$

$$v_{i,g+1} = x_{i,g} + F\bullet(x_{best,g} - x_{i,g}) + F\bullet(x_{r1,g} - x_{r2,g}) \tag{6}$$

$$v_{i,g+1} = (FF+0.5)\bullet x_{r1,g} + (FF-0.5)\bullet x_{i,g} - FF\bullet(x_{r2,g} - x_{r3,g}) \tag{7}$$

其中各式中r1≠r2≠r3≠r4≠i, r1, r2, r3, r4∈{1, NP}，公式（1）、公式（5）和公式（6）中的F采用了固定取值，公式（7）中的缩放因子（为区别与其他三个策略用FF代替）采用高斯分布动态的产生，具体的实现过程是在变异之前首先利用高斯序列产生均值为0.5，标准差也为0.5的缩放因子FF，并取其绝对值。其实现公式如式（8）所示：

$$\text{FF=abs(normrnd(0.5,0.5))} \tag{8}$$

另外，在公式（7）的变异算子中，为了使变异能朝着比$X_{i,g}$更优秀的方向进行，我们在实验中采用公式（7）进行变异时，选择的$X_{r1,g}$是优于当前个体$X_{i,g}$的。变异池中四种变异策略的选择，是基于满足不同优化问题的求解需要。其中公式（1）的变异算子的优势是能够较好的保持群体的多样性有利于算法增强算法的全局搜索能力,适合于求解多峰的函数优化问题；公式（5）和公式（6）都是基于当前最优个体的变异策略，具备较高的收敛速率，适合于求解单峰优化问题；公式（7）所示的变异算子优势在于减少了抽样的偏差性和参数的敏感性，所以既能较好的维持当前种群的多样性又能一定程度上提高算法的收敛速率，适合求解一些类似于Rosenbrock函数的复杂优化问题。

4. 实验与分析

4.1 测试函数

为比较 DMDE 算法的性能，本文仿真实验挑选了 8 个典型不同类型 Benchmark 函数进行计算比较分析。其中 f1、f2 为连续的单峰函数；f3 在 D=2 和 D=3 时是为单峰函数但在高维时可能存在多个最小值；f4 为非连续的阶梯函数；f5 为带有噪声的四次函数；f6、f7、f8 为多峰函数，它们的局部最优值数量随着维度的增加会成指数级的增长。

表 1 测试函数

函数	最优值
$f_1(x)=\sum_{i=1}^{n} x_i^2$	0
$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	0
$f_3(x)=\sum_{i=1}^{n-1}100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	0
$f_4(x)=\sum_{i=1}^{D}\lfloor x_i+0.5\rfloor^2$	0
$f_5(x)=\sum_{i=1}^{D} ix_i^4+rand[0,1)$	0
$f_6(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	0
$f_7(x)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$	0
$f_8(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	0

4.2 实验结果与分析

DMDE算法实验所得结果将与采用DE/rand/bin的标准DE算法、文献[3]所提SaDE算法以及文献[4]所提JADE算法进行比较。为减少实验数据的偶然性，实验时对8个测试函数分别独立运行50次，统计最优解的平均值（mean）和标准方差（std）进行比较。实验结果如表2所示。表2的测试结果表明，DMDE算法在f1、f2、f3三个单峰函数、f4非连续的阶梯函数的求解中性能明显优于其它三种算法；对带噪声的四次函数f5求解性能和其它三种算法基本持平；在多峰分函数f6、f8的求解中相比其它三种算法表现更大的优势；而针对多峰函数f7的求解，DMDE算法在最大迭代次数为500代时相对于其它三种算法求解精度明显更为优秀，在最大迭代次数为2000代是和其他三种算法的求解精度相比略微优秀，这种结果说明算法DMDE在该函数的优化求解中能够较快的收敛，既算法的收敛速率好于其他三种算法。

表 2 实验结果

函数	Gen	DMDE Mean(std)	DE(DE/rand/bin) Mean(std)	SaDE Mean(std)	JADE Mean(std)
f1	1500	1.2525e-192 (0.0)	9.8E–14 (8.4E–14)	4.5E–20 (6.9E–20)	1.8e-60 (8.4e-60)
f2	2000	2.2007e-132 (3.4716e-132)	1.6E–09 (1.1E–09)	1.9E–14 (1.05E–14)	1.8e-25 (8.8e-25)
f3	3000	2.8565e-011 (1.1536e-010)	2.1E+00 (1.5E+00)	2.1E+01 (7.8E+00)	8.0e-02 (5.6e-01)
f4	100	0.0 (0.0)	4.7E+03 (1.1E+03)	9.3E+02 (1.8E+02)	2.9e+00 (1.2e+00)
f5	3000	1.9161e-004 (6.1867e-005)	4.7E–03 (1.2E–03)	4.8E–03 (1.2E–03)	6.4e-04 (2.5e-04)
f6	1000	0.0 (0.0)	1.8E+02 (1.3E+01)	1.2E–03 (6.5E–04)	1.0e-04 (2.3E-05)
f7	500	1.1724e-015 (9.7361e-016)	1.1E–01 (3.9E–02)	2.7E–03 (5.1E–04)	8.2e-10 (6.9E-10)
	2000	1.7764e-017 (1.2561e-016)	9.7E–11 (5.0E–11)	4.3E–14 (2.6E–14)	4.4e-15 (0.0e+00)
f8	500	0.0 (0.0)	2.0E–01 (1.1E–01)	7.8E–04 (1.2E–03)	9.9e-08 (6.0e-07)

5. 结论

本文提出了一种动态多策略的差分演化算法（DMDE），该算法采用动态种群的演化模式，将前期变异产生的优秀个体在后面个体的演化中及时的加以利用，增强了算法的全局搜索能力；同时，基于不同问题对变异算子的敏感性，建立了多模式的变异策略池，使得算法能够针对不同问题自适应的选择更好的变异模式，产生更好的个体，从而加快了算法的收敛速率，并降低了算法对某个变异算子的依赖性，提高了算法求解不同优化问题的能力。从8个不同类型的函数测试结果来看，DMDE算法是对DE算法的一种有效的改进。

参考文献(References)

[1] Storn R, Price K. Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, 1997, 11(4):341-359.

[2] H. Simpson, *Dumb Robots*, 3rd ed., Springfield: UOS Press, 2004, pp.6-9.

[3] Storn R, Price K. Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, 1997, 11(4):341-359.

[4] Qing Anyong. Dynamic Differential Evolution Strategy and Applications in Electromagnetic Inverse Scattering Problems. IEEE Transactions on Geosciences and Remote Sensing, 2006,44(1): 116-125

[5] Zhang Jingqiao, C. Sanderson. JADE: Adaptive Differential Evolution with Optional External Archive. IEEE Transactions on evolutionary computation, 2009, 13(5):948-958.

[6] Zheng Jian-Guo, Liu Rong-Hui. Segmental Waves Differential Evolution Algorithm for Function Optimization. Journal of Chinese Computer Systems, 2011, 32(10):2118-2123.