自适应中心引力优化算法

钱伟懿 张桐桐
(渤海大学数理学院 锦州 121000)

摘 要 针对函数全局优化问题,提出了一种自适应中心引力算法,以平衡全局探测能力和局部搜索能力。首先定义粒子的适应值函数,然后根据与平均适应值的比较,更新粒子运动时间,并引进交叉操作更新当前粒子位置,从而提高算法的收敛速度。最后选择8个典型测试函数进行测试,并与中心引力优化算法和其他粒子群优化算法进行比较。结果表明,该算法得到的结果十分精确,鲁棒性强,优于其他算法。

关键词 中心引力优化算法,粒子群算法,自适应,全局优化
中图法分类号 TP18 **文献标识码** A

Adaptive Central Force Optimization Algorithm

QIAN Wei-yi ZHANG Tong-tong
(School of Mathematics and Physics, Bohai University, Jinzhou 121000, China)

Abstract The adaptive central force optimization(ACFO) algorithm was proposed for the global optimization problems in order to balance the abilities of global detective and local search. The particles fitness functions was defined. The particles movement time was updated based on the fitness value compared with the average fitness value, and the current position was updated by the crossover operation. As a result, the algorithm convergence speed was improved. 8 classic benchmark functions were used to test it. Simulation results show that, ACFO is accurate, has strong robustness, compared with several other particle swarm optimization algorithms and CFO algorithms.

Keywords Central force optimization algorithm, Particle swarm optimization algorithm, Adaptive, Global optimization

1 引言

在人类实际生活中,经常会遇到一些全局优化问题,用传统的优化算法,如梯度法、牛顿法、共轭梯度法及信赖域法等,解决这些问题具有难以克服的局限性。首先,这些算法除了凸规划外求出的最优解是局部最优解;其次是要求目标函数具有可微性,但是在实际问题中目标函数经常是不可微的、非凸的。因此许多学者都在探索解决全局优化问题的算法。到目前为止,全局优化问题已有许多算法,如填充函数法等,但比起局部解算法,效果还不是很好。因此,为了克服这些困难,许多学者从自然界和生物界的一些机制和现象出发,设计了许多仿生类算法来解决复杂的优化问题,如模拟退火算法[1]、遗传算法[2-4]、蚁群算法[5,6]、粒子群算法[7-10]等。最近,R. A. Formato[11]提出一种基于重力场粒子运动规律的新的启发式算法,称为中心引力优化算法(Central Force Optimization,CFO)。经测试发现,CFO与其他智能优化算法相比,虽然所得结果的精确度较高,但其具有不稳定性。为了促使算法达到全局探测与局部开采两者间的有效平衡,本文首先介绍中心引力优化算法,然后提出了一种自适应中心引力算法,把解空间的解看成带有质量的粒子,定义了每个粒子的适应值函数,把粒子运动的时间常数变成一个与平均适应值有关的自适应数。另外,引进交叉操作,更新当前粒子位置,这样保证了算法的探索能力和局部搜索能力,从而提高了算法的全局收敛性。最后用8个典型的测试函数对算法进行了验证,并与PSO算法、PSOGA算法、SVPSO算法和CFO算法进行了比较分析。数值结果表明,该算法是可行的、有效的。

2 中心引力优化算法

本节主要介绍文献[11]提出的中心引力优化算法。

2.1 中心引力优化算法理论

根据牛顿万有引力定律,任何两个物体之间的引力为

$$F=G\frac{m_1 m_2}{r^2} \tag{1}$$

式中,G是万有引力常量,m_1、m_2是两个物体各自的质量,r是两个物体之间的距离。于是质量m_1物体作用于质量m_2物体的加速度为

$$a_1=G\frac{m_2\mu}{r^2} \tag{2}$$

式中,μ是质量m_2物体和质量m_1物体连线的单位向量,方向指向m_2。假设在时刻t物体的位置和速度分别为R_0和V_0,则由物体运动规律,在时刻$t+\Delta t$,物体的位置是

$$R(t+\Delta t)=R_0+V_0\Delta t+\frac{1}{2}a\Delta t^2 \tag{3}$$

到稿日期:2011-07-05 返修日期:2011-09-21 本文受国家自然科学基金项目(10871033),辽宁省自然科学基金项目(20102003)资助。
钱伟懿(1963-),男,博士,教授,主要研究方向为最优化理论应用、智能计算,E-mail:qianweiyi2008@163.com;张桐桐(1986-),女,硕士生,主要研究方向为智能计算。

2.2 中心引力算法

R. A. Formato 把优化问题解空间中的解看成带有质量的粒子，把目标函数值 $f(x)$，$x\in R^D$ 定义为适应值函数，假设在 $t-1$ 时刻解空间有 N 个粒子，$R_{t-1}^1, R_{t-1}^2, \cdots, R_{t-1}^N$，其中 $R_{t-1}^p=(R_{t-1}^{p,1}, R_{t-1}^{p,2}, \cdots, R_{t-1}^{p,D})$ 称为粒子 p，$p=1,2,\cdots,N$，则在 $t-1$ 时刻粒子 p 相对于粒子 s 的加速度为

$$\frac{G\cdot U(M_{t-1}^s-M_{t-1}^p)\cdot(M_{t-1}^s-M_{t-1}^p)^\alpha\cdot(R_{t-1}^s-R_{t-1}^p)}{\|R_{t-1}^s-R_{t-1}^p\|^\beta} \tag{4}$$

式中，$M_{t-1}^p=f(R_{t-1}^p)$，$U(z)=\begin{cases}1, & z\geqslant 0\\ 0, & z<0\end{cases}$，式(4)中 $(M_{t-1}^s-M_{t-1}^p)^\alpha$ 相当于式(2)中的质量。于是作用于粒子 R_{t-1}^p 的整体加速度为

$$a_{t-1}^p=G\sum_{\substack{k=1\\k\neq p}}^{N_p}\frac{U(M_{t-1}^s-M_{t-1}^p)\cdot(M_{t-1}^s-M_{t-1}^p)^\alpha\cdot(R_{t-1}^s-R_{t-1}^p)}{|R_{t-1}^s-R_{t-1}^p|^\beta} \tag{5}$$

式中，$\alpha>0$，$\beta>0$ 是参数，其大小直接影响所求解的好坏。粒子 p 在 t 时刻的位置是

$$R_t^p=R_{t-1}^p+V_{t-1}^p\Delta t+\frac{1}{2}a_{t-1}^p\Delta t^2 \tag{6}$$

式中，V_{t-1}^p 是物体 p 在 $t-1$ 时刻的速度，在算法实现过程中把它取为 0；Δt 为时间常数。

中心引力优化算法的实现过程如下：

(1)初始化每个粒子的加速度和位置；

(2)把目标函数看成适应值函数，计算每个粒子的适应值；

(3)用式(6)更新粒子的位置；

(4)对于每个粒子，计算其适应值，将其与最优粒子进行比较，若优于最优粒子，则将其作为最优粒子；

(5)用式(5)更新粒子的加速度；

(6)若满足停止条件，则停止，否则返回(3)。

3 自适应中心引力优化算法

自适应中心引力算法(ACFO)根据目标函数定义了粒子适应值函数，并把粒子运动时间定义为自适应数，从而达到全局探测与局部开采两者间的有效平衡。

3.1 适应值函数

为了加强算法的有效性，把算法的适应值函数定义为

$$fit(R^p)=e^{-D^{\frac{f(R^p)-f(R^{pg})}{s}}} \tag{7}$$

式中，D 为粒子的维数，R^{pg} 为当前最优解，f 为目标函数值(本文考虑目标函数极小问题)，s 为所有粒子的目标函数分别与当前最优粒子的目标函数的差的和。显然，目标函数值越小，适应值函数值越大。

3.2 时间 Δt 的更新

CFO 把粒子运动时间看成常数，这不利于全局探索和局部搜索能力的提高，因此把粒子运动的时间变成一个自适应：

$$\Delta t=\begin{cases}\Delta t_{\min}+\dfrac{(\Delta t_{\max}-\Delta t_{\min})(f-f_{\min})}{(f_{avg}-f_{\min})}, & f\leqslant f_{avg}\\ \Delta t_{\max}-\dfrac{(\Delta t_{\max}-\Delta t_{\min})(f_{avg}-f_{\min})}{(f-f_{\min})}, & f>f_{avg}\end{cases} \tag{8}$$

式中，$\Delta t_{\max}$、$\Delta t_{\min}$ 分别为 Δt 的最大值和最小值，f 为粒子当前的目标函数值，f_{avg}、$f_{\min}$ 分别为当前所有粒子的平均目标值和最小目标值。粒子的目标函数值越优于平均目标值，粒子越好，其对应的 Δt 越小，从而保护了该粒子；粒子的目标函数值越小于平均目标值，粒子越差，其对应的 Δt 越大，使得该粒子跳出当前搜索范围，向较好的搜索区域靠拢。同时，当各粒子的目标值趋于一致或趋于局部最优时，将使 Δt 值增大，从而增强了全局搜索能力；反之，当各粒子的目标值比较分散时，Δt 值减小，从而提高了局部搜索能力。

3.3 ACFO 算法的实现过程

(1)初始化粒子的加速度和位置：令 $a_0^{p,j}=0$，$R_0^{p,j}=a_j+rand(b_j-a_j)$，$p=1,2,\cdots,N$，$j=1,2,\cdots,D$，其中 b_j，a_j 是决策变量第 j 个分量的上、下限，令 $t=1$。

(2)计算每个粒子的目标函数值，并确定最好的粒子位置 R^{pg}，按式(7)计算每个粒子的适应值。

(3)按式(8)更新粒子运动时间，然后按下式更新粒子位置。

$$Q_t^p=R_{t-1}^p+\frac{1}{2}a_{t-1}^p\Delta t^2$$

$$令\ R_t^{p,j}=\begin{cases}Q_t^{p,j}, & r<CR\\ R_{t-1}^{p,j_1}, & r\geqslant CR\end{cases},\ j=1,2,\cdots,D,$$

其中 $CR\in[0,1]$ 是交叉概率。

若 $R_t^{p,j}<a_j$，则 $R_t^{p,j}=a_j+\frac{1}{2}(R_{t-1}^{p,j_1}-a_j)$；

若 $R_t^{p,j}>b_j$，则 $R_t^{p,j}=b_j-\frac{1}{2}(b_j-R_{t-1}^{p,j_1})$。

(4)计算 $f(R_t^p)$。若 $f(R_t^p)<f(R^{pg})$，则令 $R_t^p=R^{pg}$，按式(7)计算每个粒子的适应值，令 $M_p=fit(R_t^p)$，$p=1,2,\cdots,N_p$。

(5)按式(5)更新粒子加速度。

(6)若满足停止条件，则停止，否则令 $t=t+1$，返回(3)。

4 仿真实验

本文在 Window 7，Pentium(R)Dual-Core CPU T4300 @ 2.10GHz 2.10GHz RAM 2.00GB，MATLAB2009 的环境下，选取 8 个测试函数[10,11]，分别对 ACFO(本文算法)、CFO[11]、PSO[7]、PSOGA[12]、SVPSO[12]进行测试比较。所有算法的种群规模为 50，迭代次数为 100。CFO 算法中其他参数为 $G=2$，$\alpha=0.2$，$\beta=0.2$，$\Delta t=1$；ACFO 算法中参数 G，α，β 与算 CFO 算法取值一样，$\Delta t_{\max}=6$，$\Delta t_{\min}=1$，$CR=0.2$。所有粒子群法取 $c_1=c_2=2$，PSO 算法为标准粒子群算法，惯性权重由 0.9 动态递减为 0.4。PSOGA 算法中参数为 $w=0.7$，$pc=0.9$，$sp=0.2$。SVPSO 算法中参数为 $w=0.8$。每个算法独立运行 20 次，计算结果见表 1。

由表 1 的平均值可以看出，对于函数 Sphere，Griewank，Schaffer，Zakharov，ACFO 算法优于其他算法；对于函数 Rastrigin，ACFO 算法与 CFO 算法一样，优于其他算法；对于函数 Goldstein-Price，ACFO 算法劣于 PSO，优于其他算法；对于函数 Rosenbrock，ACFO 算法劣于 PSOGA，优于其他算法；对于函数 Branin，ACFO 算法与 CFO、PSOGA 相差无几，优于其他算法。因此可以说明 ACFO 算法是一个有效算法。另外在相同的种群规模与迭代次数的条件下，对 8 个测试函数采用 5 种算法分别运行 20 次，得到前 100 步平均最优值的对数(为了结果比较直观，图 3 与图 5 使用 $\ln(f+c)$ 函数，其中 c 为

常数)的收敛情况,见图1—图8。如图所示,ACFO算法寻优能力优于其他4种算法。因此ACFO算法是有效的、可行的。

表1 PSO,PSOGA,SVPSO,CFO,ACFO计算统计结果

函数名	算法	最好值	最差值	平均值	函数名	算法	最好值	最差值	平均值
Sphere (f1)	PSO	0.326072	0.670782	0.449166	Goldste-inPrice (f5)	PSO	3.000000	3.000000	3.000000
	PSOGA	0.141545	0.246157	0.173384		PSOGA	3.005448	3.071178	3.021674
	SVPSO	0.018278	0.567117	0.131079		SVPSO	3.000616	3.013049	3.007523
	CFO	0.038358	0.072918	0.054837		CFO	3.000005	3.001037	3.000327
	ACFO	0.008760	0.046249	0.031589		ACFO	3.000000	3.000086	3.000018
Griewank (f2)	PSO	0.010028	0.027572	0.019565	Rosenbrock (f6)	PSO	4.93E+00	9.59E+00	7.74E+00
	PSOGA	0.007398	0.012983	0.010405		PSOGA	6.76E+00	1.33E+01	1.01E+01
	SVPSO	0.001051	0.028684	0.008960		SVPSO	8.98E+00	1.04E+01	9.35E+00
	CFO	0.001239	0.008168	0.005740		CFO	0.005652	0.756237	0.377947
	ACFO	0.003181	0.005696	0.004899		ACFO	0.010598	0.485978	0.214054
Rastrigin (f3)	PSO	−1.99988	−1.96686	−1.99466	Branin (f7)	PSO	0.398094	0.415197	0.399765
	PSOGA	−1.99992	−1.99833	−1.99930		PSOGA	0.397887	0.397887	0.397887
	SVPSO	−2.00000	−1.96841	−1.99543		SVPSO	0.398142	0.399788	0.398959
	CFO	−2.00000	−1.99991	−1.99997		CFO	0.397887	0.397901	0.397891
	ACFO	−2.00000	−1.99957	−1.99994		ACFO	0.397897	0.397887	0.397889
Schaffer (f4)	PSO	1.36E−13	5.18E−11	1.35E−11	Zakharov (f8)	PSO	0.001045	0.023087	0.004866
	PSOGA	5.77E−11	5.99E−09	2.53E−09		PSOGA	2.27E−06	9.28E−05	3.02E−05
	SVPSO	4.01E−11	8.46E−08	1.83E−08		SVPSO	7.80E−12	3.70E−07	1.09E−07
	CFO	6.93E−14	7.68E−11	2.84E−11		CFO	4.18E−13	1.33E−09	1.41E−10
	ACFO	0.00E+00	6.27E−11	8.53E−12		ACFO	5.25E−18	3.54E−11	4.38E−12

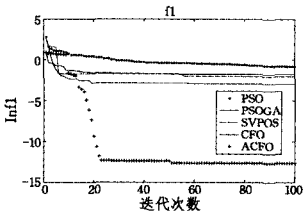

图1 函数 $f1$ 采用5种算法的平均收敛过程

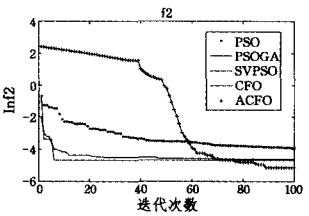

图2 函数 $f2$ 采用5种算法的平均收敛过程

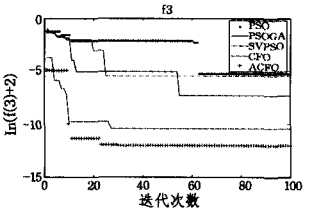

图3 函数 $f3$ 采用5种算法的平均收敛过程

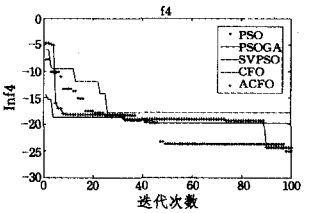

图4 函数 $f4$ 采用5种算法的平均收敛过程

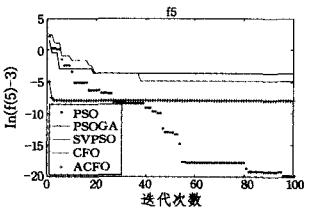

图5 函数 $f5$ 采用5种算法的平均收敛过程

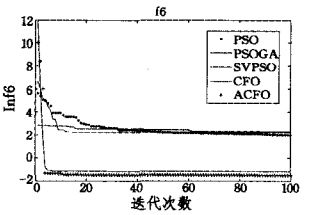

图6 函数 $f6$ 采用5种算法的平均收敛过程

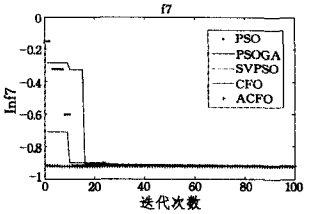

图7 函数 $f7$ 采用5种算法的平均收敛过程

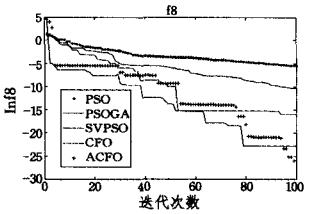

图8 函数 $f8$ 采用5种算法的平均收敛过程

结束语 本文介绍了中心引力优化算法,并对其进行了改进。通过改变适应度函数以及将常数 Δt 变为随着迭代的进行而不断变化的变量,以改进中心引力最优化算法。通过测试表明,ACFO所得结果更加精确,且相对稳定。

参考文献

[1] Kirkpatrick S,Gelatt C D,Vechi Jr M P. Opti-mization by simulated annealing[J]. Science,1983(220):671-680

[2] Holland J H. Adaptation in natural and artificial systems[M]. Ann Arbor,USA:Univ of Michigan Press,1975:110-125

[3] 黄利,丁立新,杜伟伟. 带有分级思想的自适应遗传算法[J]. 计算机科学,2010,37(5):165-167

[4] 王文义,秦广军,王若雨. 基于粒子群算法的遗传算法研究[J]. 计算机科学,2007,34(8):145-147

[5] Dorigo M,Caro G D,Gambardella L M. Ant algorithms for discrete optimization[J]. Artificial Life,1999,5:137-172

[6] 吴广潮,黄翰. 基于离散数字编码的蚁群连续优化算法[J]. 计算机科学,2008,35(3):146-148

[7] Kennedy J,Eberhart R C. Particle swarm optimization[C]// Proc of the IEEE International Conference on Neural Networks. Piscataway,NJ:IEEE Service Center,1995:1942-1948

[8] Shi Y,Eberhart R C. A modified particle swarm optimizer[C]// Proc of the IEEE International Conference on Evolutionary Computation. Piscataway,NJ:IEEE Service Center,1998:69-73

[9] 胡建秀,曾建潮. 二阶振荡微粒群算法[J]. 系统仿真学报,2007,19(5):997-999

[10] Shelokar P S,Siarry P,Javaraman V K,et al. Particle swarm and antolony algorithm shybridized for improved continuous optimization[J]. Applied Mathematics and Computation,2007,188(1):129-142

[11] Formato R A. Central force optimization:a new metaheuristic with applications in applied electromagnetics[J]. Progress in Electromagnetics Research,2007,77:425-491

[12] Valle Y D,Venayagamoorthy G K,Mohagheghi S,et alParticle swarm optimization:basic concepts,variants and applications in power systems [J]. Transactions on Evolutionary Computation,2008,4:171-195