文章编号: 1001-0920 (2015) 05-0923-06 DOI: 10.13195/j.kzyjc.2014.0387

自适应分组混沌云模型蛙跳算法求解连续空间优化问题

张 强, 李盼池

(东北石油大学 计算机与信息技术学院, 黑龙江 大庆 163318)

摘 要: 针对经典混合蛙跳优化算法寻优精度不高和易陷入局部收敛区域的缺点, 结合云模型在定性与定量之间相互转换的优良特性, 提出一种自适应分组混沌云模型蛙跳算法. 通过反向学习机制初始化种群, 应用云模型算法对优秀子群组的收敛区域进行局部搜索更优位置, 应用混沌理论在收敛区域以外空间探索全局最优位置. 典型复杂函数测试表明, 所提出的算法能有效找出全局最优解, 适用于多峰值函数寻优.

关键词: 混合蛙跳算法; 云模型; 混沌; 连续空间优化

中图分类号: TP301.6 **文献标志码:** A

Adaptive grouping chaotic cloud model shuffled frog leaping algorithm for continuous space optimization problems

ZHANG Qiang, *LI Pan-chi*

(School of Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China. Correspondent: ZHANG Qiang, E-mail: dqpi_zq@163.com)

Abstract: The shuffled flog leaping algorithm for optimization in function easily falls into local optimal solution and the premature quickly converges of such shortcomings. Combined with the excellent characteristics of cloud model transformation between qualitative and quantitative, an adaptive grouping chaotic cloud model shuffled frog leaping algorithm is proposed based on the cloud model theory. The population is initialized through reverse learning mechanism, the cloud model algorithm is used to local refinement in the region of convergence in order to explore the better position, and the chaos theory is used to obtain global optimization in the space outside the convergence region in order to explore the global optimum position. The simulation results show that the proposed algorithm has fine capability of finding global optimum, especially multi-peak function.

Keywords: shuffled flog leaping algorithm; cloud model; chaos; continuous space optimizing

0 引 言

一些群体动物在觅食过程中不断变换两种角色: 或是寻找猎物, 或是分享猎物. 发现当前最佳猎物的个体称为“探索者”, 参与觅食并分享“探索者”成果的个体称为“追随者”. 有时“探索者”所确定的目标不一定是猎物本身, 可能是猎物掉落的毛发、脚印等痕迹, 从而使整个觅食群体被“假象”所迷惑, 这种假象对应到智能优化问题中, 可以理解为算法陷入了局部最优解. 因此, 在觅食过程中可以引入“游荡”策略即“游荡者”, 负责在附近发现猎物或者新的猎物搜寻轨迹, 有利于避免“探索者”被“一些假象”所迷惑而使觅食群体陷入局部最优, 进而引导群体向更好的搜寻轨迹运动, 最终找到猎物.

Eusuff 等[1]提出的混合蛙跳算法 (SFLA) 具有便于实现、计算速度快和全局寻优能力强的优势, 在很多领域得到了广泛应用. Elbeltagi 等[2]证明了 SFLA 在解决某些连续优化问题时, 成功率和收敛速度优于遗传算法, 近似于粒子群算法. Babak 等[3]利用 SFLA 对 K 均值方法进行改进, 实验结果表明新算法的求解质量和运行速度优于基于蚁群等算法的聚类算法. 但 SFLA 存在早熟、收敛速度慢且求解精度不高的缺点, 使其在求解高维连续优化问题时效果不够理想. 导致该缺陷的主要原因是在进化后期, 种群多样性迅速下降, 缺乏局部细化搜索能力. 为了提高蛙跳算法的寻优性能, 许多学者[4-10]针对算法的参数调整、子群更新方式和与智能算法相结合等进行了改进, 取得了较

收稿日期: 2014-03-19; **修回日期:** 2014-08-18.

基金项目: 国家自然科学基金项目(61170132); 黑龙江省教育厅项目(12541086).

作者简介: 张强(1982−), 男, 讲师, 博士, 从事智能算法、神经网络的研究; 李盼池(1969−), 男, 教授, 博士后, 从事量子智能优化、过程神经网络等研究.

好的优化效果.

云模型[11-15]是一种定性知识描述和定性概念与其定量数值表示之间的不确定性转换模型, 在知识表达时具有不确定中带有确定性、稳定中有变化的特点, 体现了自然界物种进化的基本原理. 本文基于这种特性提出一种自适应分组混沌云模型蛙跳算法, 将整个蛙群子群划分为“探索者”群组、“追随者”群组和“游荡者”群组, 应用云模型算法对“探索者”群组收敛区域局部求精, 发掘全局更优位置; “追随者”群组按照改进的经典蛙跳算法进行寻优体现分享机制; “游荡者”群组通过混沌机制对未知区域的解空间进行随机探索全局最优解, 避免陷入局部最优解. 典型复杂函数测试表明, 所提出的算法能有效找出全局最优解, 适用于多峰值函数寻优.

1 标准混合蛙跳算法

设蛙群 $\boldsymbol{P}$, 每个青蛙代表一个解向量 $\boldsymbol{X}=[x_1, x_2, \cdots, x_l]$. 按照文献 [1] 的方法进行分组, 寻找每个分组适应度最好的青蛙 x_b 和最坏的青蛙 x_w. 令 x_g 为整个蛙群中当代最好适应度的青蛙, 在每次迭代中对 x_w 按如下公式进行调整:

$$S_i=\text{rand}(\ )\times(x_b-x_w), \tag{1}$$

$$s_w=s_w+Si,\ S_{\max}\geqslant S_i\geqslant -S_{\max}. \tag{2}$$

其中: rand( ) 为 $[0,1]$ 的随机数, $S_{\max}$ 为允许青蛙移动的最大范围. 如果上述调整产生的解优于 x_w 则代替, 否则, 将 x_b 替换成 x_g, 用式 (1) 和 (2) 产生新解; 若仍不优于 x_w, 则随机产生一个新解取代 x_w.

2 云 模 型

2.1 云的定义

设 U 是一个用精确数值表示的定量论域, C 是 U 上的定性概念. 若定量值 $x\in U$, 且 x 是定性概念 C 的一次随机实现, 则 X 对 C 的确定度 $\mu(x)\in[0,1]$ 是具有稳定倾向的随机数. 若 $\mu: U\to[0,1], \forall x\in U$, $x\to\mu(x)$, 则 x 在论域 U 上的分布称为云, 每一个 x 称为一个云滴[14-15].

2.2 云的数字特征

云的数字特征用期望 E_x、熵 E_n 和超熵 H_e 表征. E_x 为在论域空间中最能够代表定性概念的值; E_n 为定性概念不确定性的度量, 一方面反应论域能被语言接受的范围, 另一方面反应定性概念云滴出现的随机性, 揭示模糊性和随机性的关联性; H_e 为熵的度量, 即熵的熵, 反映在论域空间代表该语言值所有点不确定度的凝聚性, 由熵的随机性和模糊性共同决定. 云模型可以通过两次正态随机生成云滴, 实现方法如下.

Step 1: 产生一个以 E_n 为期望值、H_e 为标准差的正态随机数 $E_{n'}$.

Step 2: 产生一个以 E_x 为期望值、$E_{n'}$ 的绝对值为标准差的正态随机数 x.

Step 3: 计算 $y=\exp\dfrac{-(x-E_x)^2}{2(E_{n'})^2}$, 令 y 为属于定性概念 C 的确定度.

Step 4: 重复 Step 1 ~ Step 3, 直到产生 n 个云滴为止.

3 自适应分组混沌云模型蛙跳算法

3.1 基于反向学习机制的种群多样化

在实际计算中, 进化算法都是从随机初始种群开始迭代, 如果能在算法初始时便选择一些靠近最优解的个体开始计算, 则可以在一定程度上加快算法的收敛速率, 改善算法的进化性能. 基于该思想, Tizhoosh[16]提出了一种反向学习机制的机器学习方法, 王燕[17]证明了反向学习算法具有较快的学习速度和更强的优化能力, 林娟等[18]表明了基于反向学习机制的种群初始化有助于改进优化过程中的收敛速率.

3.2 自适应混沌云模型蛙跳算法进化模式和变异策略

定义 1 (群组) 由蛙群分组后的若干个子群构成的集合称为群组, 群组中包含的子群个数称为群组规模. 按照在寻优过程中的不同分工可分为“探索者”群组、“追随者”群组和“游荡者”群组. 其中: “探索者”群组拥有的子群个数要多一些, 即越优秀的父代应产生更多的子代; “游荡者”群组的子群个数不应过多, 即进化过程中只有少数的个体会突变, 使子代以大概率继承父代的优秀特征, 保证种群的先进性, 朝着较优的方向进化、突变来保证进化的任何可能性.

3.2.1 “探索者”群组

社会学原理指出, 优秀个体附近往往存在着更优个体, 即局部最优值附近往往存在更优值, 在其附近更有机会发现最优值. 每组子蛙群更新时, 既更新组内最差个体, 又更新组内最优个体. 更新组内最差个体采用传统蛙跳算法, 更新组内最优个体采用第 2.2 节中的正态云模型算法. 将组内最优个体看作一个正态云滴 $C(E_x,E_n,H_e)$, 用它产生与本组数量相同的一组云滴, 即一组蛙, 如果新的这组蛙中有比原来组内最优个体更优秀的个体, 则用它替换旧的最优个体, 否则, 不作处理. 具体生成算法如下: E_x 是新个体生成的中心, 将组内最优个体作为 E_x; E_n 体现搜索范围, 将当代适应度方差 σ^2 作为 E_N, 动态改变搜索范围; 将 $E_n/5$ 作为 H_e, 初期加大算法的随机性, 后期加强算法的稳定性.

3.2.2 “追随者”群组

在 AGCCM-SFLA 中, “追随者”群组所占比例较

大, 但在经典蛙跳算法的进化过程中, 子群内最差个体先参考子群内最优个体, 若不能提高则再参考全局最优个体. 这种方式使得获取的信息过于单一, 没有同时考虑这两个最优个体对最差个体的影响, 导致算法在处理大规模复杂问题时易陷入局部最优. 同时, 依据最优觅食理论, 动物觅食行为中总是趋向于耗费更低的能量而获得更多的食物, 以达到能效最好[19-20]. 计算某一个体受到的能效吸引力为

$$d_{jk}=\sqrt{\sum_{s=1}^{n}(x_{js}-x_{ks})^2},$$
$$F_{jk}=\frac{f(x_j)-f(x_k)}{d_{jk}},\ F_{jk}=-F_{kj}. \tag{3}$$

这里仅考虑 $F_{jk}\geqslant 0$ 的情况. 设对更新个体产生最大能效作用力的个体所处的位置为 x_{nx}, 为了丰富“追随者”的参考信息, 改进经典算法的个体更新方式, 在更新位置时同时参考子群内最优个体、全局最优个体和 x_{nx}. 新的位置为

$$\begin{aligned}
&x_{wb}^{'}=x_w+\mathrm{rand}(\ )\times(x_b-x_w);\\
&x_{wg}^{'}=x_w+\mathrm{rand}(\ )\times(x_g-x_w),\\
&\qquad f(x_{wb}^{'})\geqslant f(x_w);\\
&x_{wbg}^{'}=x_w+\mathrm{rand}(\ )\times(x_b-x_w)+\\
&\qquad \mathrm{rand}(\ )\times(x_g-x_w)+w(x_{nx}-x_w),\\
&\qquad f(x_{wg}^{'})\geqslant f(x_w);\\
&x_w^{'}=\cos(k\arccos(x_w)),\ k\geqslant 2,\ f(x_{wbg}^{'})\geqslant f(x_w).
\end{aligned} \tag{4}$$

其中: rand( ) 为 $[0,1]$ 的随机数, w 为吸引系数, k 为 Chebyshev 混沌映射的阶.

3.2.3 “游荡者”群组

自然界和人类社会活动不仅有渐变的和连续光滑变化的现象, 还存在突变和跃迁的现象, 这种变化一般具有突发性、多向性和随机性等特点. 从某种意义上讲, 突变有利于新物种的产生. 混沌理论看似混乱, 却有着精致的内在结构, 具有随机性、遍历性和规律性等特点[21-22]. 应用 k 阶 Chebyshev 混沌映射对个体进行映射, 文献 [22] 证明了 k 为偶数时产生的序列的随机性较好, 且二阶 Chebyshev 混沌映射与 $\mu=4$ 时的 Logistic 强混沌映射的相图一致, 产生的个体新位置呈现遍历性、随机性和多样性, 可有效地在收敛区域以外空间搜索全局最优位置. 当映射产生的新个体的适应度优于当前全局最优个体的适应度时, 则更新全局最优个体.

3.3 算法群组自适应分组策略

在 SFLA 分组过程中, 适应度相对较差的青蛙总是被分在最后一组, 致使该组最差青蛙的学习效果不如前面的分组, 这种分组方式造成青蛙的学习具有一定的局限性. 但是, 这种局限性可以通过将前面几个子群定为“探索者”群组, 最后几个子群定为“游荡者”群组, 其他子群定为“追随者”群组进行化解. 因此, 每个群组包含的子群个数多少严重影响算法的寻优能力. 下面给出一种自适应分组方法.

Step 1: 求解所有子群中最优个体适应度值的平均值 f_{avg}.

Step 2: 求解大于 f_{avg} 的所有子群中最优个体适应度值的平均值 $f_{\mathrm{avg}}^{'}$.

Step 3: 求解小于 f_{avg} 的所有子群中最优个体适应度值的平均值 $f_{\mathrm{avg}}^{''}$.

Step 4: 求解子群中最优个体适应度值大于 $f_{\mathrm{avg}}^{'}$ 的子群个数 $\mathrm{Num}_{\mathrm{cloud}}$.

Step 5: 求解子群中最优个体适应度值小于 $f_{\mathrm{avg}}^{''}$ 的子群个数 $\mathrm{Num}_{\mathrm{chaos}}$.

Step 6: 对 $\mathrm{Num}_{\mathrm{cloud}}$ 和 $\mathrm{Num}_{\mathrm{chaos}}$ 进行变换, 有

$$\begin{aligned}
\mathrm{Num}_{\mathrm{cloud}}&=\Big(\frac{M}{2}-\mathrm{Num}_{\mathrm{cloud}}\times\sin\Big(\frac{\pi}{2}\times\frac{n}{N}\Big)\times\\
&\qquad\Big(-LN\Big(\frac{\mathrm{Num}_{\mathrm{cloud}}}{M}\Big)\Big)\Big)\Big/2,\\
\mathrm{Num}_{\mathrm{chaos}}&=\Big(\alpha+\frac{\mathrm{Num}_{\mathrm{chaos}}-A}{B-A}\times(b-a)\Big)\times\\
&\qquad\sin\Big(\frac{\pi}{2}\times\frac{n}{N}\Big).
\end{aligned} \tag{5}$$

其中: M 为传统蛙跳算法分组后子群的个数; n 为算法当前运行的第几代; N 为算法的总迭代次数, 由于游荡者群组在算法进化中起到变异作用, 不宜取的过多, 需要将其进行区域变换并保持变化形态; $[A,B]$ 为运算过程中 $\mathrm{Num}_{\mathrm{chaos}}$ 的范围; $[a,b]$ 为实际需要定为“游荡者”群组的子群个数范围. 自适应分组混沌云模型蛙跳算法流程如图 1 所示.

4 实验仿真

算法采用 Java 语言实现, Windows 2003 运行平台, 双核 2.5 GHz 处理器, 2 G 内存. 以 8 个函数极值优化为例, 通过与粒子群算法 (PSO)、经典蛙跳算法 (SFLA)、云模型蛙跳算法 (CM-SFLA)、固定分组混沌云模型蛙跳算法 (GCCM-SFLA) 和自适应分组混沌云模型蛙跳算法 (AGCCM-SFLA) 进行对比, 以验证算法的优化性能. 粒子群参数设置如下: 惯性因子 0.729 8, 自身因子 1.496 18, 全局因子 1.496 18, 种群大小均为 200, 最大迭代代数均为 5 000, 每个函数独立运行 20 次, 误差精度为 10^{-10}. 蛙跳算法参数设置如下: Chebyshev 混沌映射参数 k 设置为 4, 种群大小均为 200, 分为 10 个子群, 每个子群 20 个青蛙, 子群内部迭代次数为 10 次, 最大迭代代数均为 5 000, 每个函数独立运行 20 次, 误差精度为 10^{-10}. 均匀设计是一种

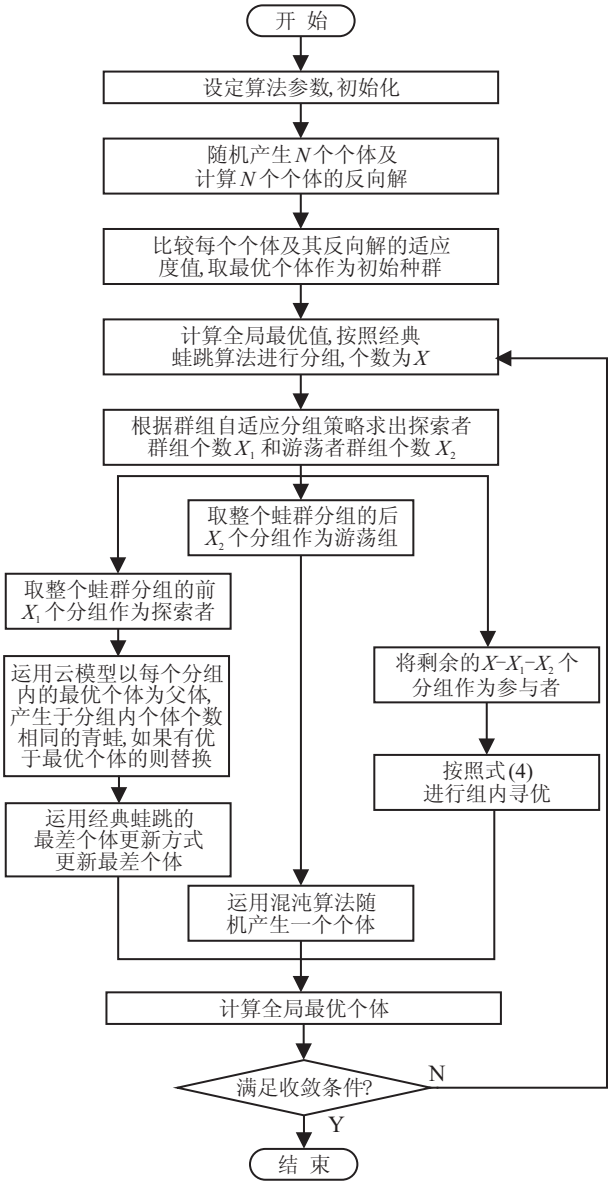

图 1 AGCCM-SFLA 算法流程

实验设计方法, 它舍弃了正交设计的整齐可比性, 只考虑实验点的均匀分布, 能用较少的实验点获得最好的均匀性[23-24]. 利用均匀设计方法设计 GCCM-SFLA 的分组比例为 2:7:1, 比较 5 种算法的最优结果、最差结果、平均结果、平均时间和方差, 有

$$f_1 = 0.5 + \frac{\sin^2(\sqrt{x_1^2+x_2^2}) - 0.5}{[1+0.001(x_1^2+x_2^2)]^2},$$
$$-100 \leqslant x_i \leqslant 100; \tag{6}$$

$$f_2 = \left[4 - 2.1x_1^2 + \frac{1}{3}x_1^4\right]x_1^2 + x_1x_2 + (-4+4x_2^2)x_2^2,$$
$$-3 \leqslant x_1 \leqslant 3,\ -2 \leqslant x_2 \leqslant 2; \tag{7}$$

$$f_3 = \left\{\sum_{i=1}^{5} i\cos[(i+1)x_1+i]\right\}\left\{\sum_{i=1}^{5} i\cos[(i+1)x_2+i]\right\},$$
$$-10 \leqslant x_i \leqslant 10; \tag{8}$$

$$f_4 = -\exp(-(x_1-\pi)^2-(x_2-\pi)^2)\cos(x_1)\cos(x_2),$$
$$-100 \leqslant x_i \leqslant 100; \tag{9}$$

$$f_5 = \sum_{i=1}^{n} x_i^2,\ -100 \leqslant x_i \leqslant 100,\ n=10; \tag{10}$$

$$f_6 = 20 + \exp(1) - 20\exp\left[-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right] -$$
$$\exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right],$$
$$-32.768 \leqslant x_i \leqslant 32.768,\ n=10; \tag{11}$$

$$f_7 = \frac{1}{4\,000}\sum_{i=1}^{n}x_i^2 - \prod_{i=1}^{n}\cos\frac{x_i}{\sqrt{i}} + 1,$$
$$-600 \leqslant x_i \leqslant 600,\ n=10; \tag{12}$$

$$f_8 = 100 + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)],$$
$$-5.12 \leqslant x_i \leqslant 5.12,\ n=10. \tag{13}$$

f_1 是二维的复杂函数, 具有无数个极小值点, 最小值为 0. f_2 有 6 个局部最优解, 全局最优值为 −1.031 628 453 489 88. f_3 是二维的复杂函数, 存在着 760 个局部极值点, 最小值为 −186.730 908 831 023 9. f_4 有一个全局最小值 −1. f_5 存在许多局部极小点, 全局最小值为 0. f_6 有一个全局最小值 0. f_7 存在许多局部极小点, 数目与问题的维数有关, 全局最小值为 0. f_8 是个多峰值的函数, 全局最小值为 0.

函数 $f_1 \sim f_8$ 的运行仿真结果对比如表 1～表 8 所示.

表 1 函数 f_1 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	1.410 6e-12	0.009 7	9.72e-04	0.276	9.44e-06
SFLA	8.123 4e-12	9.341 9e-11	5.794 8e-11	0.192	4.08e-21
CM-SFLA	5.788 9e-12	9.760 0e-11	4.414 8e-11	0.117	2.58e-21
GCCM-SFLA	4.120 9e-12	7.176 8e-11	3.573 8e-11	0.106	2.05e-21
AGCCM-SFLA	5.428 4e-13	8.115 7e-11	2.764 0e-11	0.087	1.51e-21

表 2 函数 f_2 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	−1.031 6	−1.031 6	−1.031 6	1.715	1.37e-19
SFLA	−1.031 6	−1.031 6	−1.031 6	1.456	3.69e-21
CM-SFLA	−1.031 6	−1.031 6	−1.031 6	1.034	3.25e-21
GCCM-SFLA	−1.031 6	−1.031 6	−1.031 6	0.973	3.18e-21
AGCCM-SFLA	−1.031 6	−1.031 6	−1.031 6	0.871	1.61e-21

表 3 函数 f_3 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	−186.730 9	−186.730 9	−186.730 9	2.605	1.64e-14
SFLA	−186.730 9	−186.730 9	−186.730 9	2.537	3.26e-21
CM-SFLA	−186.730 9	−186.730 9	−186.730 9	1.051	2.73e-21
GCCM-SFLA	−186.730 9	−186.730 9	−186.730 9	0.635	2.14e-21
AGCCM-SFLA	−186.730 9	−186.730 9	−186.730 9	0.467	1.24e-21

表 4　函数 f_4 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	−1	−1	−1	2.384	5.10 e-21
SFLA	−1	−1	−1	1.935	4.62 e-21
CM-SFLA	−1	−1	−1	1.097	3.57 e-21
GCCM-SFLA	−1	−1	−1	0.896	2.96 e-21
AGCCM-SFLA	−1	−1	−1	0.714	1.68 e-21

表 5　函数 f_5 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	9.660 5 e-11	0.013 9	9.81 e-04	132.875	1.07 e-05
SFLA	1.088 0 e-10	9.977 3 e-10	7.727 6 e-10	1.463	6.59 e-20
CM-SFLA	5.254 9 e-11	3.751 8 e-10	9.941 1 e-11	1.308	4.29 e-21
GCCM-SFLA	1.659 6 e-11	9.937 7 e-11	6.206 3 e-11	0.723	3.91 e-21
AGCCM-SFLA	1.238 0 e-12	6.231 9 e-11	4.978 39 e-11	0.605	3.07 e-21

表 6　函数 f_6 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	8.245 5 e-5	3.404 2	1.954 3	144.748	4.90
SFLA	1.840 8	1.840 8	1.840 8	6.981	3.39
CM-SFLA	3.308 5 e-10	9.057 5 e-10	7.220 4 e-10	3.327	6.76 e-21
GCCM-SFLA	5.502 7 e-11	9.996 5 e-11	8.132 2 e-11	1.365	5.51 e-21
AGCCM-SFLA	3.613 5 e-12	8.983 3 e-11	5.004 3 e-11	1.041	3.48 e-21

表 7　函数 f_7 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	0.073 9	0.518 9	0.265 5	147.376	8.41 e-02
SFLA	0.029 5	0.120 6	0.066 9	5.341	5.21 e-03
CM-SFLA	2.651 3 e-10	9.858 6 e-10	5.488 3 e-10	3.507	4.69 e-21
GCCM-SFLA	3.917 6 e-11	9.089 2 e-11	7.145 8 e-11	0.883	3.78 e-21
AGCCM-SFLA	5.946 9 e-12	7.710 3 e-11	4.773 1 e-11	0.812	2.34 e-21

表 8　函数 f_8 的运行仿真结果对比

算法	最优结果	最差结果	结果	时间/s	方差
PSO	5.969 7	40.793 2	17.710 2	139.393	4.12 e+02
SFLA	4.767 7 e-11	1.989 9	0.597 2	3.671	9.91 e-01
CM-SFLA	6.815 6 e-11	5.152 8 e-10	8.182 4 e-10	1.891	4.42 e-21
GCCM-SFLA	2.418 2 e-11	9.805 5 e-11	6.895 3 e-11	0.652	3.98 e-21
AGCCM-SFLA	9.379 2 e-12	8.876 1 e-11	5.182 4 e-11	0.513	3.12 e-21

由仿真结果可见, 本文所提出的自适应分组混沌云模型蛙跳算法具有较好的求解精度和求解速度, 原因主要有以下两个方面:

1) 从最优结果、最差结果、平均结果可见, PSO 的寻优效果最差, SFLA、CM-SFLA 和 GCCM-SFLA 次之, AGCCM-SFLA 最优, 虽然 $f_2 \sim f_4$ 保留 4 位小数的计算结果是一样的, 但从实际计算精度看, AGCCM-SFLA 更接近全局最优值. 这是因为算法采用反向学习机制的种群初始化增加了个体接近最优解的机会, 且云模型的稳定倾向性可以较好地保护最优个体, 从而实现对周围更优值的自适应定位. 随机性可以保持个体多样性, 加快算法的进化速度和寻优效率, 同时引入的混沌理论可以使算法进化后期很好地避免陷入局部最优, 有利于获得全局最优解, 解决算法在一些复杂函数时易陷入早熟收敛、收敛速度慢、易陷入局部最优的缺陷.

2) 从平均收敛迭代代数、运行时间和方差可见, SFLA 迭代次数相对 PSO 的迭代次数要少, 若考虑蛙跳算法子群内部的迭代次数, 则 SFLA 的迭代次数要多, 但从运行时间看, 还是 SFLA 较快, 原因是 SFLA 在迭代时只更新最差粒子, 最好情况下一次迭代只计算一次, 最坏情况才计算 3 次, 相对于粒子群每次迭代更新所有粒子计算量要少, 所以速度快, 且 AGCCM-SFLA 的运行时间和方差都优于前几种算法, 对于函数 $f_5 \sim f_8$ 的计算结果更优, 表明 AGCCM-SFLA 对高维多峰函数的求解具有很好的适应性.

5　结　论

本文应用云模型算法对“探索者”群组收敛区域局部求精, 发掘全局最优位置; 利用混沌变量有利于克服智能算法中因位置分布不合理带来的局限性的优势, 对“游荡者”空间进行全局寻优, 避免陷入全局最优位置, 从而可以快速收敛到最优解, 较好地避免了经典蛙跳算法易陷入局部最优解和选择压力过大造成的早熟收敛等问题. 仿真结果表明, 所提出的算法具有精度高、收敛速度快等优点. 云模型和进化计算思想的有效结合拓宽了云模型的应用领域, 也为进化计算的研究进行了新的探索和尝试.

参考文献(References)

[1] Eusuff M M, Lansey K E. Optimization of water distribution network designusing shuffled frog leaping algorithm[J]. J of Water Resources Planning and Management, 2003, 129(3): 210-225.

[2] Elbeltagi E, Hegazy T, Grierson D. Comparison among five evolutionary-based optimization algorithms[J]. Advanced Engineering Informatics, 2005, 19(1): 43-53.

[3] Amiri B, Fathian M, Maroosi A. Application of shuffled frog-leaping algorithmon clustering[J]. Int J of Advanced Manufacturing Technology, 2009, 45(1/2): 199-209.

[4] Rahimi-Vahed A, Mirzaei A H. A hybrid multi-objective shuffled frog leaping algorithm for a mixed-model assembly line sequencing problem[J]. Computersand Industrial Engineering, 2007, 53(4): 642-666.

[5] Rahimi-Vahed A, Mirzaei A H. Solving a bi-criteria permutation flow-shopproblem using shuffled frog-leaping algorithm[J]. Soft Computing, 2008, 12(5): 435-452.

[6] Bhaduri A. A clonal selection based shuffled frog leaping algorithm[C]. IEEE Int Conf on Advance Computing.

Piscataway: IEEE Press, 2009, 3: 125-130.

[7] 李英海, 周建中, 杨俊杰, 等. 一种基于阈值选择策略的改进混合蛙跳算法[J]. 计算机工程与应用, 2007, 43(35): 19-21.
(Li Y H, Zhou J Z, Yang J J, et al. Modified shuffled frog leaping algorithm based on threshold se-lection strategy[J]. Computer Engineering and Applications, 2007, 43(35): 19-21.)

[8] 罗雪晖, 杨烨, 李霞. 改进混合蛙跳算法求解旅行商问题[J]. 通信学报, 2009, 30(7): 130-135.
(Luo X H, Yang Y, Li X. Modified shuffled frog-leaping algorithm to solve traveling salesman problem[J]. J on Communications, 2009, 30(7): 130-135.)

[9] 赵鹏军. 优化问题的几种智能算法[D]. 西安: 西安电子科技大学数学与统计学院, 2009.
(Zhao P J. Several intelligent algorithms for optimization problems[D]. Xi'an: School of Mathematics and Statistics, Xidian University, 2009.)

[10] 唐德玉, 蔡先发, 齐德昱, 等. 基于量子粒子群搜索策略的混合蛙跳算法[J]. 计算机工程与应用, 2012, 48(29): 29-33.
(Tang D Y, Cai X F, Qi D Y, et al. Shuffled frog leaping algorithm based on particle swarm optimization searching strategy[J]. Computer Engineering and Applications, 2012, 48(29): 29-33.)

[11] 付斌, 李道国, 王慕快. 云模型研究的回顾与展望[J]. 计算机应用研究, 2011, 28(2): 420-426.
(Fu B, Li D G, Wang M K. Review and prospect on research of cloud model[J]. Application Research of Computers, 2011, 28(2): 420-426.)

[12] 刘明周, 张玺, 张铭鑫, 等. 基于损益云模型的制造车间重调度决策方法[J]. 控制与决策, 2014, 29(8): 1458-1464.
(Liu M Z, Zhang X, Zhang M X, et al. Rescheduling decision method of manufacturing shop based on profit-loss cloud model[J]. Control and Decision, 2014, 29(8): 1458-1464.)

[13] 张英杰, 邵岁锋, Niyongabo J. 一种基于云模型的云变异粒子群算法[J]. 模式识别与人工智能, 2011, 24(1): 90-94.
(Zhang Y J, Shao S F, Niyongabo J. Cloud hypermutation particle swarm optimization algorithm based on cloud model[J]. Pattern Recognition and Artificial Intelligence, 2011, 24(1): 90-94.)

[14] 张光卫, 何锐, 刘禹, 等. 基于云模型的进化算法[J]. 计算机学报, 2008, 31(7): 1082-1090.
(Zhang G W, He R, Liu Y, et al. An evolutionary algorithm based on cloud model[J]. Chinese J of Computers, 2008, 31(7): 1082-1090.)

[15] 马颖, 田维坚, 樊养余. 基于云模型的自适应量子免疫克隆算法[J]. 计算物理, 2013, 30(4): 627-632.
(Ma Y, Tian W J, Fan Y Y. Quantum adaptive immune clone algorithm based on cloud model[J]. Chinese J of Computational Physics, 2013, 30(4): 627-632.)

[16] Tizhoosh H. Opposition-based learning: A new scheme for machine intelligence[C]. Proc of the Int Conf on Computational Intelligence for Modeling Control and Automation. Piscataway: IEEE Press, 2005: 695-701.

[17] 王燕. 反向粒子群算法理论及及其应用研究[D]. 西安: 西安工程大学理学院, 2011.
(Wang Y. The research of opposition-based particle swarm optimization and its application[D]. Xi'an: College of Science, Xi'an Polytechnic University, 2011.)

[18] 林娟, 钟一文, 马森林. 改进的反向蛙跳算法求解函数优化问题[J]. 计算机应用研究, 2013, 30(3): 760-763.
(Lin J, Zhong Y W, Ma S L. Improved opposition-based shuffled frog leaping algorithm for function optimization problems[J]. Application Research of Computers, 2013, 30(3): 760-763.)

[19] 孙儒泳. 动物生态学原理[M]. 第 3 版. 北京: 北京师范大学出版社, 2001: 430-431.
(Sun R Y. Principles of animal ecology[M]. 3rd ed. Beijing: Beijing Normal University Press, 2001: 430-431.)

[20] 崔志华, 曾建潮. 微粒群优化算法[M]. 北京: 科学出版社, 2011: 65-66.
(Cui Z H, Zeng J C. Particle swarm optimization[M]. Beijing: Science Press, 2011: 65-66.)

[21] 胥小波, 郑康锋, 李丹, 等. 新的混沌粒子群优化算法[J]. 通信学报, 2012, 33(1): 24-37.
(Xu X B, Zheng K F, Li D, et al. New chaos-particle swarm optimization algorithm[J]. J on Communications, 2012, 33(1): 24-37.)

[22] 刘金梅, 屈强. 几类混沌序列的随机性测试[J]. 计算机工程与应用, 2011, 47(5): 46-49.
(Liu J M, Qu Q. Randomness tests of several chaotic sequences[J]. Computer Engineering and Applications, 2011, 47(5): 46-49.)

[23] 王元, 方开泰. 关于均匀分布与试验设计 (数论方法)[J]. 科学通报, 1981, 26(2): 65-70.
(Wang Y, Fang K T. Uniform distribution and experimental design(number theoreticm ethod)[J]. Chinese Science Bulletin, 1981, 26(2): 65-70.)

[24] 梁昌勇, 陆青, 张恩桥, 等. 基于均匀设计的多智能体遗传算法研究[J]. 系统工程学报, 2009, 24(1): 109-113.
(Liang C Y, Lu Q, Zhang E Q, et al. Research on multi-agent genetic algorithm based on uniform design[J]. J of Systems Engineering, 2009, 24(1): 109-113.)

(责任编辑: 郑晓蕾)