DOI:10.11721/cqnuj20210612

基于 K-means 的邻域结合随机吸引的萤火虫算法*

李媛媛，魏　延，张文泷，王晶仪，蒋俊蕊
(重庆师范大学 计算机与信息科学学院，重庆 401331)

摘要:【目的】为解决传统萤火虫算法收敛速度慢，特别是对于复杂的优化问题，容易陷入局部最优，从而导致收敛精度低的问题，提出了基于 K-means 的邻域结合随机吸引的萤火虫算法。【方法】先将初始萤火虫种群进行 K-means 聚类，用聚类中心的萤火虫种群为寻优萤火虫，然后以提出的邻域与随机相结合的吸引模型进行寻优，在寻优过程中，还引入自适应步长策略。【结果】在减少算法复杂度的同时保证了算法的全局搜索能力，不仅提高了算法跳出局部最优的能力，还能够让算法在快速收敛的同时提升结果的精度。【结论】实验结果表明，提出的基于 K-means 的邻域结合随机吸引的萤火虫算法，无论是寻优结果的精度和稳定性，还是寻优速度上都有更好的效果。

关键词:萤火虫算法；K-means 算法；邻域结构；全局寻优
中图分类号:TP301　　**文献标志码**:A　　**文章编号**:1672-6693(2021)06-0114-08

剑桥大学的 Yang[1]以萤火虫为灵感，将萤火虫在求偶时发光行为的一些闪光特性理想化，开发出了萤火虫算法(Firefly algorithm)。萤火虫算法具有多模态特征，能够有效地处理多模态问题；具有快速的收敛速度，可以作为通用的全局问题求解器和局部搜索启发式算法，适用于所有问题域[2]。由于萤火虫算法操作简单，需要调整的参数少，易于实现，而且在一些测试函数的全局寻优仿真结果中表明，萤火虫算法在效率和成功率方面均优于粒子群算法和遗传算法[3]。因此，萤火虫算法已经成功地应用于图像处理[4]、路径规划[5]、运输调度[6]、优化算法[7]等多个科学与工程领域中。

虽然萤火虫算法表现出了良好的性能，但由于每只萤火虫在迭代寻优时都会向不同方向移动多次，这样不仅会导致搜索过程复杂，还会减缓算法的收敛速度。所以，与其他的仿生优化算法类似，标准萤火虫算法在解决复杂问题时存在局部最优、收敛速度慢的不足。此外，该算法还有对维数增加比较敏感，计算复杂度高等缺陷[8]。

为了提高萤火虫算法性能，近些年来很多学者已经对萤火虫算法进行了多角度的改进。文献[9]提出了 3 种策略来提高算法的适应性，介绍了一种自适应的萤火虫改进算法。文献[10]将混沌技术引入萤火虫算法，提高了算法的全局搜索能力。文献[11]提出了一种用环形拓扑结构模拟萤火虫的邻域结构的 NaFA 算法。文献[12]在随机吸引模型的基础上，引入了高斯扰动和局部搜索策略，有效地提升了算法的寻优能力。文献[13]提出了一种正交反向学习策略来改进萤火虫算法。

针对标准萤火虫算法中每只萤火虫被吸引的次数过多导致的搜索时间太长和算法精度不足问题，本文提出了一种基于 K-means 优化的萤火虫算法——基于 K-means 的邻域结合随机吸引的萤火虫算法(Firefly algorithm based on K-means combining neighborhood and random attraction，KNRFA)。先对萤火虫种群进行 K-means 聚类，将聚类中心作为寻优萤火虫，以邻域与随机相结合的吸引模型进行寻优，并且在寻优时引入了自适应步长策略。本算法在保证全局搜索能力的同时减少了计算复杂度，而且提升了算法的寻优精度和跳出局部最优的能力，从而提高了算法的寻优性能。

1 萤火虫算法与 K-means 算法

1.1 萤火虫算法

剑桥大学的 Yang 在开发萤火虫算法的过程中，将萤火虫的一些闪光特性理想化，使用了如下 3 个规则：

* **收稿日期**:2021-06-03　**修回日期**:2021-07-02　**网络出版时间**:2021-11-17 13:46
资助项目:重庆市技术创新与应用发展重点项目(No. cstc2019jscx-mbdxX0061)
第一作者简介:李媛媛，女，研究领域为教育大数据感知与应用、智能算法，E-mail：2217880681@qq.com；**通信作者**:魏延，男，教授，博士，E-mail：942503433@qq.com
网络出版地址:https://kns.cnki.net/kcms/detail/50.1165.N.20211116.1947.020.html

1）所有的萤火虫都是不区分性别的，所以一只萤火虫会被其他比它亮度高的萤火虫所吸引；

2）对任何两只闪烁的萤火虫来说，亮度低的那只萤火虫会向亮度高的那只移动。相对吸引力与亮度成正比，与距离成反比。如果根据比较结果，没有一只萤火虫比当前的萤火虫更亮，那它就是当前最亮的萤火虫，就会随机移动；

3）萤火虫的亮度受目标函数的值影响或决定。

萤火虫算法的相关定义的数学描述如下：

定义 1　萤火虫的相对荧光亮度为：

$$I=I_0\times e^{-\gamma r_{ij}^2}。\tag{1}$$

其中：I_0 是 $r=0$ 处荧光亮度最大的时候；γ 是光强吸收因子；r_{ij} 是萤火虫 i 和 j 之间的欧式距离，且有：$r_{ij}=\|x_i-x_j\|=\sqrt{\sum_{k=1}^{d}(x_{ik}-x_{jk})^2}$。

定义 2　萤火虫之间的相对吸引力为：

$$\beta=\beta_0\times e^{-\gamma r_{ij}^2},\tag{2}$$

其中：β_0 为最大吸引度，是光源处，即 $r=0$ 时的吸引度，γ，r_{ij} 的意义同上。

定义 3　萤火虫 i 被吸引向萤火虫 j 移动的位置更新公式为：

$$x_i^{t+1}=x_i^t+\beta_0\times e^{-\gamma r_{ij}^2}+\alpha\varepsilon_i,\tag{3}$$

其中：x_i，x_j 表示萤火虫 i 和 j 的位置；$\alpha\varepsilon_i$ 是扰动项，α 为步长因子，$\alpha\in[0,1]$，ε_i 是服从高斯分布或均匀分布的随机因子，且为[−0.5，0.5]区间上的随机数。目前最亮的萤火虫为 x_{best}，没有萤火虫可以吸引它，于是它的位置更新公式为：

$$x_{\text{best}}=x_{\text{best}}+\alpha\varepsilon_i。\tag{4}$$

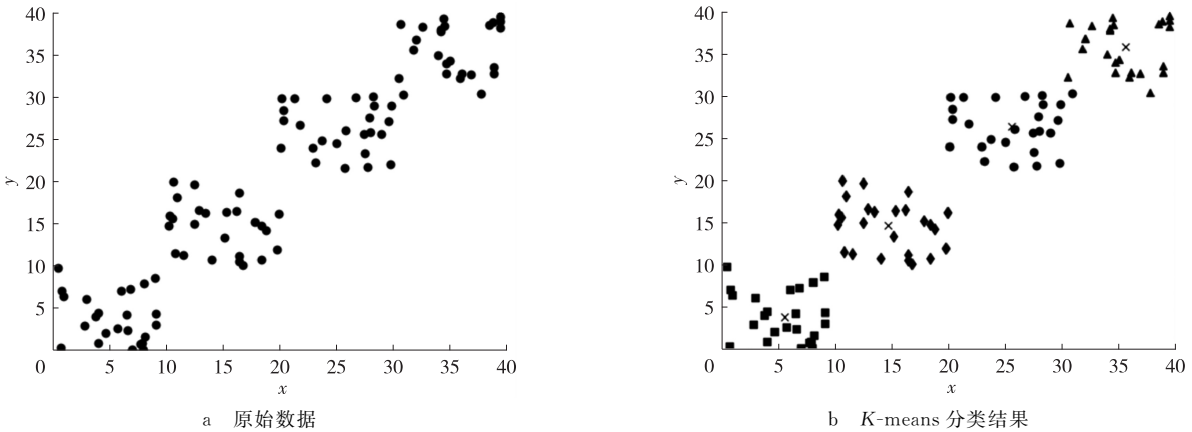
1.2 *K*-means 算法

K-means 算法是一种无监督学习，在 1967 年被 Queen[14] 首次使用，以效果较好、思想简单的优点在聚类算法中得到了广泛应用。*K*-means 算法就是对于给定的样本集，以样本之间的欧式距离作为相似度的指标，将样本集划分为 K 个簇。算法目标是：让簇内的样本距离尽量的小，而簇间的距离尽量的大。

K-means 算法的核心思想：首先，从给出的样本集中随机选取 k 个样本，将它作为初始聚类中心 C_i（$1\leqslant i\leqslant k$），并且计算出其余样本点 X_i 与聚类中心 C_i 的欧氏距离。比较这些欧式距离，找出离样本点 X_i 最近的聚类中心 C_i，并将样本点 X_i 分配到聚类中心 C_i 所对应的簇中。然后，计算每个簇中样本点的平均值作为新的聚类中心，进行下一次迭代，直到聚类中心不再变化或达到最大的迭代次数停止。

样本集中样本点与聚类中心间的欧式距离为：$d(x,C_i)=\sqrt{\sum_{j=1}^{m}(x_j-C_{ij})^2}$，这里的 x 为样本点，C_i 为第 i 个聚类中心，m 为样本点的维度，x_j，C_{ij} 为 x 和 C_i 的第 j 个属性值。

使用 *K*-means 算法对数据集进行聚类，设置原始数据集有 4 个簇，图中 x 和 y 分别代表数据点的横纵坐标值，图 1 显示的是 100 个点的原始数据集和经过 *K*-means 聚类后的数据集。



a　原始数据　　b　*K*-means 分类结果

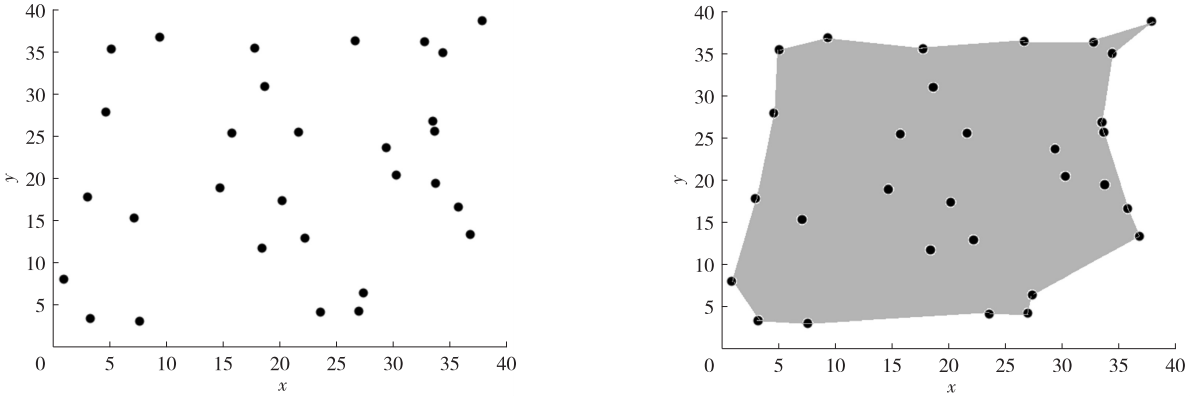
图 1　对 100 个点进行 *K*-means 聚类

Fig. 1　100 points are clustered by *K*-means

2 基于 K-means 的邻域结合随机吸引的萤火虫算法

2.1 以 K-means 聚类中心为寻优萤火虫

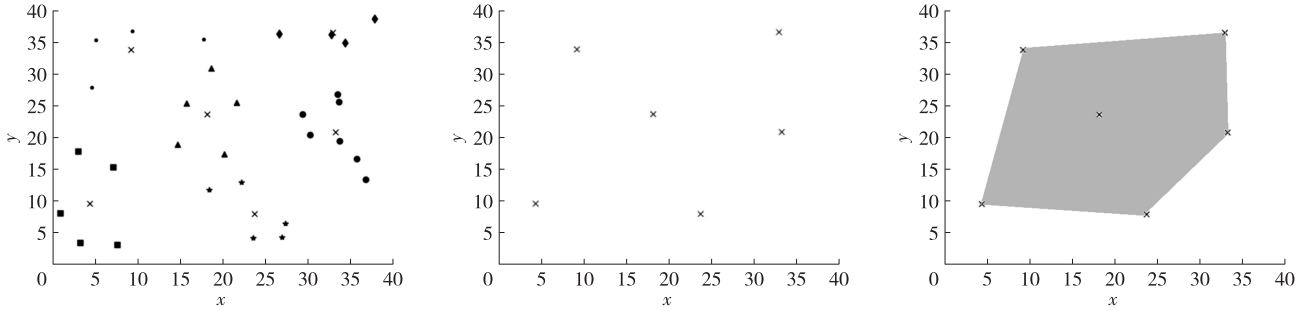
为了在减少算法复杂度的同时,能够保证萤火虫的寻优范围,本文先用 K-means 聚类算法对初始种群进行聚类,然后用聚类中心进行寻优。图2和图3分别展示了初始种群的覆盖范围、对初始种群进行 K-means 聚类后的覆盖范围。



a 随机生成30个点　　b 30个点的覆盖范围

图2 初始种群的覆盖范围

Fig. 2 Coverage of the initial population



a 在30个随机点里找6个 K-means 聚类中心　　b 6个聚类中心　　c 6个聚类中心的覆盖范围

图3 对初始种群进行 K-means 聚类后的覆盖范围

Fig. 3 Coverage of the initial population after K-means clustering

对比图2和图3的覆盖范围可知,使用本文提出的用 K-means 算法对初始种群进行聚类,保留了原始种群寻优范围的大体轮廓和覆盖面积。

2.2 邻域与随机吸引结合的吸引模型

标准萤火虫算法采用的是全吸引模型,即每只萤火虫都会被其他更亮的萤火虫所吸引,然后移动更新位置。在这种吸引模型中,如果一个种群有 N 只萤火虫,每只萤火虫最多会被 $N-1$ 只萤火虫吸引,平均每只萤火虫会被吸引$\frac{N-1}{2}$次。由此可见,全吸引模型的吸引次数过多,这让萤火虫算法收敛速度较慢,而且若每只萤火虫在每一代里向不同方向移动次数过多,则容易导致搜索的振荡[15]。

本文提出了一种将邻域吸引和随机吸引相结合的吸引模型(N-R 吸引模型)。在 N-R 吸引模型中,每只萤火虫 X 会被从它的邻域和随机挑选出来的萤火虫中更亮个体所吸引,X 的邻域和随机挑选出的种群数量远远小于$\frac{N-1}{2}$。所以,在 N-R 吸引模型中的萤火虫 X 被吸引移动的次数比标准萤火虫算法所采用的全连接吸引模型少,而且 N-R 吸引模型中的随机吸引策略还可以在一定程度上避免算法陷入局部最优。因此,本文提出的 N-R 吸引模型不仅减小了寻优的时间复杂度,还增强了算法跳出局部最优的能力。

为了使算法的对比更直观，将萤火虫种群排列成圆形拓扑结构。图 4 和图 5 分别展示全吸引模型和 N-R 吸引模型。

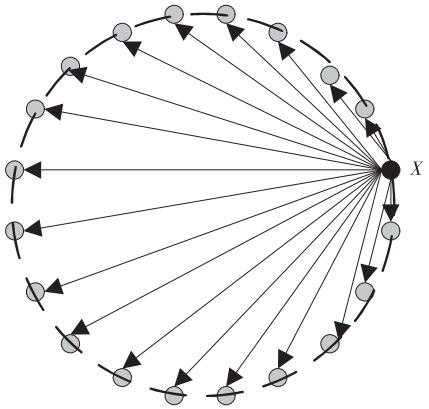


图 4　全吸引模型

Fig. 4　Full attraction model

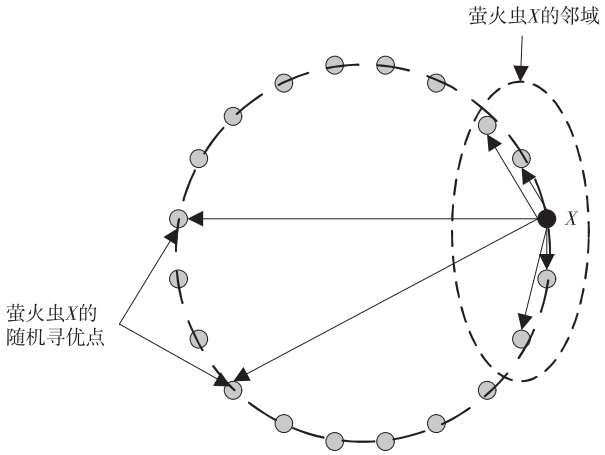


图 5　N-R 吸引模型

Fig. 5　Neighborhood and random attraction model

全吸引模型中所有萤火虫最多被吸引的次数 T_X 为：$T_X=(N-1)+(N-1)+\cdots+(N-1)=N(N-1)$。在 N-R 吸引模型中的所有萤火虫最多被吸引的次数 T_X 为：$T_X=(n+r)+(n+r)+\cdots+(n+r)=N(n+r)$。其中：$n$ 是萤火虫 X 的邻域种群数量，r 是随机挑选的种群数量。由于在本文中，$n+r$ 远远小于 $N-1$，所以 $N(n+r)\leqslant N(N-1)$，N-R 吸引模型的复杂度小于全吸引模型的。

2.3 自适应步长

为了满足算法的全局寻优能力，初始阶段步长应该尽可能大[16]，使算法在理论上具有更强的全局寻优能力。而在算法演化后期，则要求群体有更强的收敛性，此时就要求步长应该足够小[17]。当萤火虫算法收敛时，应该满足 $\lim\limits_{t\to\infty}\alpha=0$[18]，基于这一原理，本文使用了文献[18]设计的自适应参数策略来动态调整 α，在搜索过程中，α 更新如下：

$$\alpha(t+1)=\left(1-\frac{t}{T}\right)\alpha(t)。\tag{5}$$

2.4 KNRFA 描述

萤火虫算法的性能决定于初始种群、萤火虫间的吸引模型、参数控制等因素，本文对标准萤火虫算法进行了 3 方面的改进：1）用 K-means 算法对初始种群进行聚类；2）提出了一种新的吸引模型；3）引入了自适应步长策略。

本文提出的 KNRFA 的算法思想是：用 K-means 算法把初始萤火虫进行聚类，然后以聚类中心为寻优萤火虫；寻优时，萤火虫用邻域与随机相结合的 N-R 吸引模型进行寻优，在寻优过程中用自适应步长进行位置更新和算法收敛。此算法通过用 K-means 的聚类中心为寻优萤火虫，在减小算法复杂度的同时保证了寻优范围的广度。以 N-R 吸引模型和自适应步长策略进行寻优，在提高了算法的收敛速度和精度的同时，增强了算法跳出局部最优解的能力。

本文提出的 KNRFA，具体步骤如下：

步骤 1：初始化算法基本参数。萤火虫的个数 N，聚类个数 K，N-R 吸引模型中的模型大小 R，光吸收系数 γ，步长因子 α，最大迭代次数 T。

步骤 2：随机生成 N 个初始萤火虫，将 N 个萤火虫进行 K-means 聚类，只保留下 K 个聚类中心的萤火虫的位置，将它们作为寻优萤火虫。

步骤 3：根据测试函数计算出 K 个寻优萤火虫的目标函数值，把目标函数值作为亮度值。

步骤 4：用(1)式和(2)式计算每只萤火虫与 N-R 吸引模型中的 R 个萤火虫之间的相对亮度和吸引力，然后根据它的相对亮度值的大小决定个体移动方向。根据(5)式算出萤火虫应该移动的步长，萤火虫再根据(3)式更

新自己的位置。

步骤5:重新计算每只萤火虫的荧光亮度,对处在最佳位置的萤火虫根据(4)式执行随机扰动,比较扰动前后的亮度值,保留结果较优的位置。

步骤6:判断是否满足循环结束的条件,若不满足,转步骤4。

若萤火虫的初始规模为N,空间维度为D,则初始化种群的时间复杂度为$O(ND)$。初始萤火虫K-means聚类簇的个数为K,K-means聚类的迭代次数为t,则初始种群进行K-means聚类的时间复杂度为$O(tKND)$。在萤火虫迭代寻优的过程中,迭代的次数为T,N-R吸引模型的大小为R,步骤3~5是计算种群的亮度和位置更新,复杂度为$O(KRDT)$。因此本文算法的时间复杂度为$O(KRDT)$。

3 实验仿真与结果分析

3.1 测试函数

本文选择了12个测试函数[19]来验证文章提出的算法性能,其中函数f_1~f_3为单峰函数,f_4是一个阶梯函数,函数f_5~f_9为多峰函数。函数f_1只有1个极值点,具有易于求解的特性,主要用来测试算法的寻优精度。函数f_3是很难优化的典型非凸病态函数,全局最优点位于一个狭长的抛物线谷的范围内,通常用来评价算法的优化性能。函数f_8是属于多模态最小化的一类测试函数,当在定义域内趋于无穷大时,该函数会沿着自变量方向产生大量可微的局部极值,具有较高的寻优难度,可用来评价算法跳出局部最优的能力。函数f_9是典型的欺骗函数,不仅有大量局部极小点,而且全局极小点和局部极小点之间的距离较远。另外,广泛的搜索空间也导致该函数的寻优过程十分困难。

所有的测试函数都需要求它的全局最小值。各个函数的具体描述如表1所示。

表1 测试函数

Tab.1 Test functions

函数名称	函数表达式	寻优范围	全局极小值
Sphere	$f_1(x)=\sum_{i=1}^{D}x_i^2$	$x_i\in[-100,100]$	0
Schwefel2.22	$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$x_i\in[-10,10]$	0
Rosenbrock	$f_3(x)=\sum_{i=1}^{D}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$x_i\in[-30,30]$	0
Step	$f_4(x)=\sum_{i=1}^{D}(x_i+0.5)^2$	$x_i\in[-100,100]$	0
Ackley	$f_5(x)=-20\exp\left(-0.02\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)+20+\mathrm{e}$	$x_i\in[-30,30]$	0
Rastrigin	$f_6(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	$x_i\in[-5.12,5.12]$	0
Griewank	$f_7(x)=1+\frac{1}{4\ 000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	$x_i\in[-600,600]$	0
Alpine	$f_8(x)=\sum_{i=1}^{D}\lvert x_i\sin(x_i)+0.1x_i\rvert$	$x_i\in[-10,10]$	0
Schwefel 2.26	$f_9(x)=418.982\ 9D-\sum_{i=1}^{D}x_i\sin(\sqrt{\lvert x_i\rvert})$	$x_i\in[-500,500]$	0

3.2 实验结果分析

本文进行仿真实验的硬件和软件环境为:CPU为i7-10870H,内存16 GB,Windows 10操作系统,所有算法利用python 3.7和Pycharm 2020平台编程实现。

为了验证 KNRFA 的性能，实验将 KNRFA 与萤火虫算法[1]、ApFA[18]、NaFA[11]等算法进行比较。为了公平的比较，这里将参与比较的萤火虫算法、ApFA、NaFA 算法的部分参数设置的与对应的参考文献一致，表 2 呈现了各个对比算法的部分参数值。

设置萤火虫的规模 $N=50$，其中 KNRFA 的聚类簇数 $K=10$。在仿真中，为了保证公平性，把萤火虫算法、ApFA、NaFA 和 KNRFA 在 3 维搜索空间和 30 维搜索空间上各自独立迭代 30 次。每次的最大迭代次数 $T=10\ 000$，这样计算出 30 次寻优结果的平均值、标准差以及各个算法的平均运行时间，再用平均值来比较算法的收敛精度，以标准差来衡量算法的稳定性，平均运行时间用来比较算法的寻优速度。

算法在 3 维和 30 维的测试函数中实验结果对比如表 3 所示。

表 2　对比算法的参数

Tab. 2　Parameter configurations of contrast algorithms

方法	参数
萤火虫算法	$\alpha=0.5,\beta_0=1.0,\gamma=1.0$
ApFA	$\alpha_0=0.5,\beta_0=1.0,\gamma=\frac{1}{\Gamma^m}$
NaFA	$\alpha=0.5,\beta_0=1.0,\beta_{\min}=0.2,\gamma=1.0$
KNRFA	$\alpha_0=0.5,\beta_0=1.0,\gamma=\frac{1}{\Gamma^m},R=6$

表 3　实验结果对比

Tab. 3　Comparison of experimental results

测试函数	算法	3 维			30 维		
		平均值	标准差	平均运行时间	平均值	标准差	平均运行时间
f_1	萤火虫算法	1.687e−04	5.170e−05	2.566e+02	4.084e−01	5.019e−02	2.649e+02
	ApFA	3.351e−06	2.010e−06	2.390e+02	2.117e−01	1.739e−02	2.415e+02
	NaFA	6.998e−06	2.950e−06	3.622e+01	2.650e−01	2.203e−02	3.623e+01
	KNRFA	**0.000e+00**	**0.000e+00**	**3.217e+00**	**3.694e−36**	**1.713e−35**	**3.726e+00**
f_2	萤火虫算法	5.257e−03	1.437e−03	3.371e+02	4.507e+04	8.970e+04	4.067e+02
	ApFA	2.112e−03	5.412e−04	3.186e+02	2.045e+00	7.506e−02	3.694e+02
	NaFA	3.468e−03	1.515e−03	4.692e+01	2.270e+00	1.015e−01	5.743e+01
	KNRFA	**0.000e+00**	**0.000e+00**	**5.279e+00**	**6.084e−02**	**1.573e−01**	**5.742e+00**
f_3	萤火虫算法	1.166e−03	4.941e−04	3.596e+02	4.202e+02	5.464e+02	3.660e+02
	ApFA	1.555e−04	6.602e−05	3.319e+02	7.544e+01	5.659e+01	3.474e+02
	NaFA	2.609e−04	**2.130e−04**	4.881e+01	5.251e+01	4.398e+00	4.948e+01
	KNRFA	**2.324e−05**	5.101e−04	**4.282e+00**	**5.089e+01**	**3.874e+01**	**4.398e+00**
f_4	萤火虫算法	2.336e−04	1.858e−04	2.672e+02	4.107e−01	1.826e−02	2.703e+02
	ApFA	1.271e−06	8.199e−07	2.443e+02	2.389e−01	1.259e−02	2.610e+02
	NaFA	6.159e−06	3.563e−06	3.596e+01	2.624e−01	2.445e−02	3.731e+01
	KNRFA	**0.000e+00**	**0.000e+00**	**3.164e+00**	**2.037e−31**	**9.623e−32**	**3.257e+00**
f_5	萤火虫算法	1.475e−01	1.151e−01	5.177e+02	5.373e+00	3.861e−01	5.394e+02
	ApFA	3.921e−04	**2.078e−04**	4.839e+02	**4.584e−01**	**3.551e−02**	5.338e+02
	NaFA	7.350e−04	2.141e−04	6.952e+01	7.440e−01	1.247e−01	7.313e+01
	KNRFA	**2.403e−04**	4.964e−04	**6.494e+00**	6.747e−01	4.792e−02	**6.946e+00**
f_6	萤火虫算法	1.501e+00	1.109e+00	3.775e+02	2.554e+02	1.471e+01	3.846e+02
	ApFA	**2.967e−04**	**1.941e−04**	3.482e+02	1.220e+02	2.864e+01	3.692e+02
	NaFA	1.007e−01	2.987e−01	4.767e+01	6.803e+01	5.826e+00	5.384e+01
	KNRFA	3.681e+00	3.271e+00	**3.871e+00**	**1.453e+01**	**5.077e+00**	**4.110e+00**

续表3

测试函数	算法	3维			30维		
		平均值	标准差	平均运行时间	平均值	标准差	平均运行时间
f_7	萤火虫算法	6.921e+00	3.107e+00	4.410e+02	2.429e+02	4.254e+01	5.208e+02
	ApFA	1.194e+00	7.739e−01	4.199e+02	7.599e−02	7.141e−02	4.844e+02
	NaFA	8.335e−02	7.127e−02	6.398e+01	9.080e−02	1.020e−01	7.205e+01
	KNRFA	**3.937e−02**	**2.769e−02**	**5.449e+00**	**6.640e−02**	**6.709e−02**	**6.762e+00**
f_8	萤火虫算法	6.283e−04	3.070e−04	3.197e+02	1.460e+01	5.938e−01	3.237e+02
	ApFA	4.362e−04	4.016e−04	2.907e+02	5.567e+00	2.488e+00	3.001e+02
	NaFA	1.881e−04	5.684e−05	4.393e+01	4.623e−01	1.743e−01	4.492e+01
	KNRFA	**8.327e−17**	**1.878e−16**	**3.61e+00**	**1.432e−02**	**7.197e−03**	**4.514e+00**
f_9	萤火虫算法	**1.974e+01**	**6.003e+01**	3.360e+02	3.810e+03	8.206e+02	3.424e+02
	ApFA	1.456e+02	7.577e+01	3.293e+02	4.170e+03	**3.619e+02**	3.321e+02
	NaFA	9.080e+01	8.726e+01	4.555e+01	**3.751e+03**	4.966e+02	4.608e+01
	KNRFA	2.783e+02	1.237e+02	**3.200e+00**	6.364e+03	9.729e+02	**3.701e+00**

注:表3中数值加粗表示在不同算法下同类结果中的最优值

对表3的结果分析可知,无论是在低维还是高维的9个不同的测试函数中,KNRFA的寻优结果在大部分测试函数上的收敛精度和稳定性上都优于其他函数。在3维的测试函数中,KNRFA除了在f_6,f_9上陷入局部最优的情况外,在其他测试函数上都有着更优异的表现,尤其是在函数f_1,f_2,f_4中,KNRFA在30次测试得到的平均值就是测试函数的最小值0。在30维的测试函数中,KNRFA除了在f_5,f_9上陷入局部最优外,对其他测试函数的寻优精度和稳定性都有不同程度的优化。

无论是在低维还是高维的测试函数中,KNRFA的平均运行时间一直都远小于其他对比算法。由此可见,与其他萤火虫算法相比,本文提出的KNRFA复杂度较小,而且不会随着测试函数的维度增大而骤增,在寻优速度上明显优于其他3种对比算法。

基于以上实验结果和统计分析,可得出在这9个测试函数中,KNRFA的收敛精度和结果的稳定性都比较好,而且在寻优速度也比较快,所以KNRFA性能总体上明显优于其他3种对比算法。

4 结束语

本文在分析了萤火虫算法存在缺陷的原因后,提出了KNRFA。改进后的萤火虫算法用初始萤火虫种群的K-means聚类中心进行寻优,这一方法在保证萤火虫搜索范围的广度的同时减少了时间复杂度。邻域与随机相结合的吸引模型,降低了算法复杂度,还在一定程度上提高了跳出局部最优的能力。而且加入了自适应步长,这可以进一步提高寻优精度和速度。通过对9个测试函数的仿真实验可知,KNRFA无论是在低维还是高维的搜索空间上,都有较高的寻优精度和较强的稳定性,以及更快的寻优速率。

参考文献:

[1] YANG X S. Nature-inspired metaheuristic algorithms[M]. Beckington:Luniver Press,2008.

[2] FISTER I,YANG X S,BREST J. A comprehensive review of firefly algorithms[J]. Swarm and Evolutionary Computation,2013,13:34-46.

[3] YANG X S. Firefly algorithms for multimodal optimization[EB/OL]. (2010-03-07)[2021-06-03]. https://www.scienceopen.com/document?vid=55045187-6031-4f9e-81da-3f49ab04632e.

[4] 陈恺,陈芳,戴敏,等. 基于萤火虫算法的二维熵多阈值快速图像分割[J]. 光学精密工程,2014,22(2):517-523.
CHEN K,CHEN F,DAI M,et al. Fast image segmentation based on two-dimensional entropy and multi-threshold based on firefly algorithm[J]. Optics and Precision Engineering,2014,22(2):517-523.

[5] 刘鹏,刘弘,郑向伟,等. 基于改进萤火虫算法的动态自动聚集路径规划方法[J]. 计算机应用研究,2011,28(11):4146-4149.
LIU P,LIU H,ZHENG X W,et al. Dynamic automatic aggregation path planning method based on improved firefly algorithm[J]. Application Research of Computers,2011,28(11):4146-4149.

[6] 刘志勇,蔡延光,戚远航.集装箱物流运输调度问题的改进萤火虫算法[J].东莞理工学院学报,2017,24(3):27-32.
LIU Z Y,CAI Y G,QI Y H. Improved firefly algorithm for container logistics scheduling problem[J]. Journal of Dongguan University of Technology,2017,24(3):27-32.
[7] 陈小雪,尉永清,任敏,等.基于萤火虫优化的加权 *K*-mean 算法[J].计算机应用研究,2018,35(2):466-470.
CHEN X X,YU Y Q,REN M,et al. Weighted *K*-means algorithm based on firefly optimization[J]. Application Research of Computers,2018,35(2):466-470.
[8] 臧睿,李晶.基于维度加权的改进萤火虫算法[J].计算机科学,2017,44(S1):123-125.
ZANG R,LI J. Improved firefly algorithm based on dimension weighting[J]. Computer Science,2017,44(S1):123-125.
[9] YAN X H,ZHU Y L,WU J W,et al. An improved firefly algorithm with adaptive strategies[J]. Advanced Science Letters,2012,16(1):249-254.
[10] GANDOMI A H,YANG X S,TALATAHARI S,et al. Firefly algorithm with chaos[J]. Communications in Nonlinear Science and Numerical Simulation,2013,18(1):89-98.
[11] WANG H,WANG W J,ZHOU X Y,et al. Firefly algorithm with neighborhood attraction[J]. Information Sciences,2017,382:374-387.
[12] LI L,JIA Z. The firefly algorithm with Gaussian disturbance and local search[J]. Journal of Signal Processing Systems,2018,90:1123-1131.
[13] 周凌云,丁立新,马懋德,等.一种正交反向学习萤火虫算法[J].电子与信息学报,2019,41(1):202-209.
ZHOU L Y,DING L X,MA M D,et al. An orthogonal reverse learning firefly algorithm[J]. Journal of Electronics and Information Technology,2019,41(1):202-209.
[14] QUEEN J M. Some methods for the classification and analysis of multivariate observations[C]//Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability. Berkeley:University of California Press,1967:281-297
[15] HUI W, WANG W, HUI S, et al. Firefly algorithm with random attraction[J]. International Journal of Bio-Inspired Computation,2016,8(1):33-41.
[16] YU S H,ZHU S L,MA Y,et al. A variable step size firefly algorithm for numerical optimization[J]. Applied Mathematics and Computation,2015,263:214-220.
[17] 汪靖,刘桂元.基于动态步长变化的萤火虫算法[J].计算机工程与设计,2019,40(4):108-114.
WANG J,LIU G Y. Firefly algorithm based on dynamic step size change[J]. Computer Engineering and Design,2019,40(4):108-114.
[18] WANG H,ZHOU X Y,SUN H,et al. Firefly algorithm with adaptive control parameters[J]. Soft Computing,2017,21(17):5091-5102.
[19] JAMIL M,YANG X S. A literature survey of benchmark functions for global optimization problems[J]. International Journal of Mathematical Modelling & Numerical Optimisation,2013,4(2):150-194.

The Firefly Algorithm Based on *K*-means Combining Neighborhood and Random Attraction

LI Yuanyuan, WEI Yan, ZHANG Wenlong, WANG Jingyi, JIANG Junrui
(College of Computer and Information Science, Chongqing Normal University, Chongqing 401331, China)

Abstract: [**Purposes**]In order to solve the problem of slow convergence speed of traditional firefly algorithm, especially when solving complex optimization problems. It is easy to fall into the local optimal value, which leads to the problem of low convergence accuracy. [**Methods**]The firefly algorithm based on K-means combining neighborhood and random attraction (KNRFA) is proposed. First, the initial firefly population was K-means clustering, and the firefly population in the clustering center was taken as the optimal firefly. Then, the attraction model combining neighborhood and randomness proposed was used for optimization. In the optimization process, the self-adaptive step size strategy was also introduced. [**Findings**]The global searching ability of the algorithm is guaranteed while the complexity of the algorithm is reduced. It not only improves the ability of the algorithm to jump out of the local optimum, but also enables the algorithm to converge quickly and improve the accuracy of the results. [**Conclusions**]The experimental results show that the firefly algorithm based on K-means combining neighborhood and random attraction proposed has better results both in terms of precision and stability of the optimization results and in terms of optimization speed.

Keywords: firefly algorithm; *K*-means algorithm; neighborhood structure; global optimization

(责任编辑 许 甲)