CE\_FA

作者：刘桂元

# 1 引言

# 2 萤火虫算法文献简述

# 3 萤火虫算法改进策略

在标准萤火虫算法中，尚还存在一些需要改进的地方。一是算法步长不变导致种群后期动荡寻优，难以接近最优解，从而解的精度难以提高。二是同许多群智能算法一样，标准萤火虫算法在随着演化的进行，难以避免地失去种群多样性，但该算法本身没有增加种群多样性的策略。三是算法设计的个体间的吸引策略在一定程度上浪费了评估次数，导致算法收敛慢。

## 3.1 自适应步长策略

为了解决标准萤火虫算法中种群寻优后期动荡问题，提高算法所求得解的精度，我们提出了自适应步长改进策略。即步长随着迭代次数的增加而减小，并通过当前代的最优解与上一代的最优解的距离控制步长的减小。在图3.1中，系列同心圆为萤火虫个体在二维sphere函数空间中的亮度相等的位置集合，而中心位置为最优值，即为算法寻求的目标位置。标有字母标号的小圆则为萤火虫个体，且虚线个体为个体被吸引后可能的出现位置。其中a部分为步长不变的算法后期寻优状态，b部分为步长减小的算法后期寻优状态。从图3.1的a、b两部分的比较中可以知，适当的步长的减小有利于种群后期靠近最优值，提高求解的精度。

为此，我们设计了自适应步长策略来改进萤火虫算法。在步长随迭代次数减小的同时还增加了步长减小的控制结构，即当上一代最优个体与当代最优个体亮度不变时，再根据相关公式减小步长以适应下一代的对步长大小寻优需求。改进步长减小公式如3.1所示，其中*t*为迭代次数，和分别为解空间定义域的上限和下限。

|  |  |  |
| --- | --- | --- |
|  | | （3.1） |
|  |  | |
| a. 寻优步长不变(*α*=0.2) | b. 寻优步长减小(*α*=Eq.3.1) | |
| 图3.1 不同寻优步长在二维sphere函数中的状态 | | |

## 3.2 增加多样性的改进

为了提高算法种群的多样性，在我们提出的增加算法多样性的改进策略中，除了将初始吸引力参数由1变成0.4的小小改动外，主要是通过反向学习来增加萤火虫算法的多样性。其吸引力参数改进公式如公式3.2所示，目的为尽可能保留种群的多样性。

|  |  |
| --- | --- |
|  | （3.2） |

而在引入的反向学习策略中，为了提高萤火虫优化算法对求解空间的探索能力，我们在改进的萤火虫算法中将每次迭代的最差的三个个体进行反向学习，使其对解空间中心进行反转。反转变化公式定义如公式3.3所示，其中为种群某需要反转的个体位置，为该个体反转后的空间位置，和分别为解空间中维度的下限和上限。

|  |  |
| --- | --- |
|  | （3.3） |

如图3.2所示，其中三角形表示种群个体，且虚线三角形为个体经反向学习变化后可能的达到空间位置。叉表示某局部空间的适应值的最值，且大面积椭圆区域的最值要劣于小面积椭圆区域的最值。当问题为求解全局最小值时，则要求种群能在小面积椭圆区域探索以求得全局最优值。而如图3.2中的虚线三角形所示，通过反向学习使得种群中的个体有几率在小面积椭圆区域进行探索，极大的增加种群的多样性，跳出局部最优，提高算法求解问题全局最优的能力。

|  |
| --- |
|  |
| 图3.2 种群在某二维空间中反向学习探索示意图 |

## 3.3 萤火虫重心

为了改进萤火虫吸引策略，提高算法的收敛速度。我们在改进的萤火虫吸引策略中增加了萤火虫重心概念。即为了提高萤火虫吸引效率，减小评估次数，我们将每代萤火虫种群亮度进行排序，然后将连续亮度差小于设定阈值的三个萤火虫个体进行组合成一个虚拟个体以代替原来的个体在种群中吸引移动，且该虚拟个体在解空间中的位置为被代替个体的重心位置（我们称该过程为萤火虫的重心化，Centroidize），而在种群重心化之后，再根据种群个体亮度的优劣进行吸引移动。其中，连续个体亮度差小于如公式3.4所示阈值的萤火虫被认为亮度相近是可以组合的，式中和分别表示第*i*代种群中的最大亮度和最小亮度，*N*为种群大小，*Gen*为迭代次数，*a*为常系数一般设为0.001。

|  |  |
| --- | --- |
|  | （3.4） |

而种群重心化如图3.3所示，其中圆表示萤火虫个体且适应值从左到右依次减小，在变化后的种群中因部分连续个体亮度差小于设置阈值而被组合成一个虚拟的萤火虫个体。

|  |
| --- |
|  |
| 图3.3 种群重心化 |

种群中虚拟个体的移动对应于被代替个体的移动描述于图3.4，图中小圆为萤火虫个体并用字母标有相关编号，个体E为适应值最优萤火虫个体。个体A、B、C为三个被吸引个体，且因亮度小于设定阈值而被组合成的虚拟个体D。个体A’、B’、C’则为个体A、B、C根据虚拟个体D被E吸引移动到D’后做出相应变换的位置。因种群中小于设定阈值的萤火虫个体被整体移动而不再是逐个向更优个体移动，减少了代间评估，从而提高了算法种群的收敛速率。

|  |
| --- |
|  |
| 图3.4 三维空间中萤火虫个体组合后吸引移动 |

## 3.4 三种策略融合改进

如前所述，上述三种改进策略从不同角度对FA进行调优改进，从理论上将上述三种策略融合改进FA是没有冲突的。因此我们尝试将三种改进策略整合于FA中，并将其命名为SECFA（Self-adaptive-step Enhance-diversity Centroidize Firefly Algorithm），其算法框架如算法3.1所示。

|  |  |
| --- | --- |
| **算法3.1** Framework of SECFA | |
| 1 | Randomly initialize *N* fireflies (solutions) as an initial population ; |
| 2 | Calculate the fitness v of each firefly ; |
| 3 | FEs =0 and PS = *N*; |
| 4 | **while** *FEs* ≤*MAXFEs* |
| 6 | Sort and centroidize population; |
| 7 | **for** *i* = 1 to *N* |
| 8 | **for** *j* = 1 to *N* |
| 9 | **if** |
| 10 | Move towards according to Eq. 4; |
| 11 | Calculate the fitness value of the new ; |
| 12 | FEs++; |
| 13 | **end if** |
| 14 | **end for** |
| 15 | **end for** |
| 16 | Calculate the best brightness difference value; |
| 17 | **If** Δbrightness = 0 |
| 18 | Update the step factor *α* according to Eq.3.1; |
| 19 | **end if** |
| 20 | Reverse the three worst brightness of fireflies according to Eq. 3.3; |
| 21 | **end while** |

# 4 实验与分析

在这一部分中，我们设计了一些数值实验来验证上述改进算法的性能。 在实验中，对一些benchmark函数进行了测试，并将增加改进策略的萤火虫算法的仿真结果与其他版本的萤火虫算法进行了比较。

## 4.1 benchmak函数

为了验证我们提出的改进的萤火虫优化算法，在下一个实验中，使用了13个benchmark函数(Lin等人，2016)来测试算法的性能。在这些函数中，*f*1-*f*5是单峰函数，*f*6是一个具有最小值的跳跃函数，并且不连续，*f*7是一个具有噪声的二次函数，以及具有许多局部极小值的*f*8-*f*13多模态函数。 所有函数都是最小化问题且全局最优为0，它们具体数学表达式如表4.1所示，其中D为问题的维度大小。

|  |  |  |  |
| --- | --- | --- | --- |
| 表4.1 实验使用的benchmark函数 | | | |
| Name | Function | Search  Range | Global Optimum |
| Sphere |  | [-100, 100] | 0 |
| Schwefel 2.22 |  | [-10, 10] | 0 |
| Schwefel 1.2 |  | [-100, 100] | 0 |
| Schwefel 2.21 |  | [-100, 100] | 0 |
| Rosenbrock |  | [-30, 30] | 0 |
| Step |  | [-100, 100] | 0 |
| Quartic with noise |  | [-1.28, 1.28] | 0 |
| Schwefel 2.26 |  | [-500, 500] | 0 |
| Rastrigin |  | [-5.12, 5.12] | 0 |
| Ackley |  | [-32, 32] | 0 |
| Griewank |  | [-600, 600] | 0 |
| Penalized 1 |  | [-50, 50] | 0 |
| Penalized 2 |  | [-50, 50] | 0 |

## 4.2 实验参数设置

在我们的实验中，CeFA将与标准萤火虫优化算法和其他三种最近提出的改进萤火虫算法进行比较。参数的设计细节见表4.2。为了保证各比较算法实验数据的可比性，我们对其参数进行了相同的处理，将相关参数设置如下。

种群大小：30

最大评估次数：5.0E+05

运行次数：30

问题维度：30

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表4.2 各版本FA参数 | | | | | | |
|  |  |  |  |  |  |  |
| FA(Yang 2010) | 0.2 | - | - | 1.0or | 1.0 | - |
| MFA(Fister et al. 2012) | - | - | 0.5 |  | 1.0 | 0.2 |
| WSSFA(Yu et al. 2014) | - | 0.04 | - | 1.0 | 1.0 | - |
| VSSFA(Yu et al. 2015) | - | - | - | 1.0 | 1.0 | - |
| SECFA | Eq. 3.1 | - | Eq. 3.1 |  | 0.4 | - |

## 4.3 结果分析

为了验证我们提出的不同改进策略的有效性，本小节从策略单独改进FA与策略融合改进的FA两方面对实验结果数据分别进行比较分析，以进一步说明各改进策略的优化性能。

### 4.3.1 不同策略单独改进的实验结果

在表4.3中描述了本文提出的不同改进策略改进的FA与标准FA优化13各benchmark函数的实验结果比较，其中“Mean”为不同版本萤火虫算法30次求解测试函数的结果平均值，“Std dev”为标准差，“os”（Optimal Solution）为表中算法求得各目标函数解中平均值最优的个数。

并从表4.3中可以知道，在本文提出的改进策略单独改进FA后，相比于标准FA，其结果都有一定的优势，虽然有些目标函数优化结果优势不是很明显。而表4.3的实验数据中还表现出将FA融合不同改进策略时，算法似乎有更好的改进效果。因为算法在使用自适应策略加强解空间的局部搜索时容易陷入局部最优，而增加种群多样性策略则可以很好帮助种群跳出局部最优，以及在引入重心化策略后则可以从评估次数的利用效率方面提高算法寻优性能。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表4.3不同版本FA的相关实验数据 | | | | | | |
| Function | FA() | | FA() | | 步长自适应FA | |
| Mean | Std dev | Mean | Std dev | Mean | Std dev |
|  | 6.67E+04 | 1.83E+04 | 5.14E-02 | 1.36E-02 | **2.44E-64** | 1.26E-63 |
|  | 5.19E+02 | 1.42E+02 | 1.07E+00 | 2.65E-01 | 1.44E-02 | 7.90E-02 |
|  | 2.43E+05 | 4.85E+04 | 1.26E-01 | 1.86E-01 | **1.49E-41** | 2.81E-41 |
|  | 8.35E+01 | 3.16E+01 | 9.98E−02 | 2.34E−02 | **8.78E-35** | 1.61E-34 |
|  | 2.69E+08 | 6.21E+07 | 3.41E+01 | 6.23E+00 | **2.14E+01** | 3.35E+01 |
|  | 7.69E+04 | 3.38E+03 | 5.24E+03 | 1.08E+03 | 6.67E-02 | 2.54E-01 |
|  | 5.16E+01 | 2.46E+01 | 7.55E−02 | 1.42E−02 | 3.24E-02 | 1.64E-02 |
|  | 1.10E+04 | 3.77E+03 | 9.16E+03 | 1.78E+03 | **4.88E+03** | 1.21E+03 |
|  | 3.33E+02 | 6.28E+01 | 4.95E+01 | 2.39E+01 | 4.82E+01 | 1.70E+01 |
|  | 2.03E+01 | 2.23E−01 | 1.21E+01 | 1.96E+00 | 4.47E-02 | 2.45E-01 |
|  | 6.54E+02 | 1.69E+02 | 2.13E−02 | 1.47E−02 | 1.11E-02 | 1.16E-02 |
|  | 7.16E+08 | 1.82E+08 | 6.24E+00 | 4.62E+00 | 3.11E-02 | 9.89E-02 |
|  | 1.31E+09 | 4.76E+08 | 5.11E+01 | 1.28E+01 | 4.73E-03 | 9.37E-03 |
| os | 0 | - | 0 | - | 5 | - |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表4.3 不同版本FA的相关实验数据 （续表） | | | | |
| Function | 增加多样性FA | | 重心化FA | |
| Mean | Std dev | Mean | Std dev |
|  | 1.44E-02 | 1.40E-03 | 4.34E-07 | 1.64E-07 |
|  | 5.25E-02 | 3.62E-03 | **6.04E-04** | 1.12E-04 |
|  | 3.66E-02 | 7.01E-03 | 1.66E-06 | 8.10E-07 |
|  | 4.26E-02 | 3.43E-03 | 9.32E-04 | 1.82E-04 |
|  | 2.31E+01 | 8.09E-01 | 2.64E+01 | 1.99E+01 |
|  | **0.00E+00** | 0.00E+00 | **0.00E+00** | 0.00E+00 |
|  | **4.34E-06** | 3.15E-06 | 1.51E-03 | 6.23E-04 |
|  | 5.34E+03 | 4.20E+02 | 7.23E+03 | 5.98E+02 |
|  | **7.37E-03** | 1.71E-03 | 3.18E+01 | 9.28E+00 |
|  | 3.02E-02 | 1.96E-03 | **1.88E-04** | 4.92E-05 |
|  | 2.58E-02 | 4.58E-03 | **5.92E-03** | 6.26E-03 |
|  | 4.38E-05 | 7.01E-06 | **3.86E-09** | 1.86E-09 |
|  | 5.93E-04 | 9.22E-05 | **6.28E-08** | 3.40E-08 |
| os | 3 | - | 6 | - |

### 4.3.2 综合策略改进实验结果

表4.4记录了SECFA的13个基准函数的实验数据，其中“Mean”是算法运行30次的平均结果，“Std dev”是标准差。从表中可以看出，SECFA除在优化*f*5、*f*8等目标函数时陷入局部最优解外，其它函数结果取得了较好的精度解。特别是对于*f*6、*f*9、*f*11目标函数，在30次检验中，每一次都能找到全局最优解。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表4.4 SECFA实验计算结果 | | | | |
| Function | Worst | Best | Mean | Std Dev |
|  | 1.68E-139 | 2.66E-151 | 3.11E-140 | 1.30E-139 |
|  | 1.02E-73 | 8.28E-82 | 1.03E-74 | 3.60E-74 |
|  | 8.52E-176 | 5.83E-202 | 2.84E-177 | 0.00E+00 |
|  | 2.56E-70 | 3.61E-81 | 8.64E-72 | 4.67E-71 |
|  | 2.82E+01 | 2.56E+01 | 2.66E+01 | 5.22E-01 |
|  | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
|  | 1.06E-05 | 2.64E-07 | 3.39E-06 | 2.91E-06 |
|  | 4.55E+03 | 2.65E+03 | 3.16E+03 | 5.10E+02 |
|  | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
|  | 4.44E-16 | 4.44E-16 | 4.44E-16 | 0.00E+00 |
|  | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
|  | 7.02E-18 | 2.09E-32 | 2.05E-18 | 2.38E-18 |
|  | 7.80E-17 | 4.80E-32 | 2.27E-17 | 2.60E-17 |

而为了进一步验证SECFA的有效性，我们将其与近年来其它改进的FA进行对比，因为我们没有实现其它改进的FA，因此对比数据主要来源于Wang 2017 [Firefly algorithm with neighborhood attraction]文献。表4.5描述了近年来各学者提出的不同改进版本的FA求解13个benchmark函数的实验结果，其中“w/t/l”表示相比于CESFA该版本FA求解目标函数的平均最优结果的劣势、均势及优势个数。从表4.5可以看出，相比于其它版本FA，CESFA寻优结果的优势个数都在8个以上，因此，本文提出的改进策略对FA的优化性能有较大的提高。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 表4.5 各萤火虫算法最好亮度平均结果对比 | | | | | | | | |
| Function | FA  Mean | VSSFA  Mean | WSSFA  Mean | MFA  Mean | CFA  Mean | RaFA  Mean | NaFA  Mean | CESFA  Mean |
| F1 | 5.67E-02 | 5.84E+04 | 6.34E+04 | 1.56E-05 | 3.27E-06 | **5.36E-184** | 4.43E-29 | 3.11E-140 |
| F2 | 1.00E+00 | 1.13E+02 | 1.35E+02 | 1.85E-03 | 8.06E-04 | 8.76E-05 | 2.98E-15 | **1.03E-74** |
| F3 | 1.23E-01 | 1.16E+05 | 1.10E+05 | 5.89E-05 | 1.24E-05 | 4.91E+02 | 2.60E-28 | **2.84E-177** |
| F4 | 1.01E-01 | 8.18E+01 | 7.59E+01 | 1.73E-03 | 8.98E-04 | 2.43E+00 | 3.43E-15 | **8.64E-72** |
| F5 | 8.42E+01 | 2.16E+08 | 2.29E+08 | 2.29E+01 | **2.06E+01** | 2.92E+01 | 2.39E+01 | 2.66E+01 |
| F6 | 5.30E+03 | 5.48E+04 | 6.18E+04 | **0.00E+00** | **0.00E+00** | **0.00E+00** | **0.00E+00** | **0.00E+00** |
| F7 | 6.74E-02 | 4.43E+01 | 3.24E-01 | 1.30E-01 | 9.03E-02 | 5.47E-02 | 2.91E-02 | **3.39E-06** |
| F8 | 8.14E+03 | 1.07E+04 | 1.06E+04 | 4.84E+03 | 4.36E+03 | **5.03E+02** | 6.86E+03 | 3.16E+03 |
| F9 | 4.49E+01 | 3.12E+02 | 3.61E+02 | 6.47E+01 | 5.27E+01 | 2.69E+01 | 2.09E+01 | **0.00E+00** |
| F10 | 1.25E+01 | 2.03E+01 | 2.05E+01 | 4.23E-04 | 4.02E-04 | 3.61E-14 | 3.02E-14 | **4.44E-16** |
| F11 | 2.94E-02 | 5.47E+02 | 6.09E+02 | 9.86E-03 | 7.91E-06 | **0.00E+00** | **0.00E+00** | **0.00E+00** |
| F12 | 1.25E+01 | 3.99E+08 | 6.18E+08 | 5.04E-08 | 8.28E-09 | 4.50E-05 | **1.36E-31** | 2.05E-18 |
| F13 | 5.28E+01 | 8.12E+08 | 9.13E+08 | 6.06E-07 | 1.69E-07 | 8.25E-32 | **2.13E-30** | 2.27E-17 |
| w/t/l | 13/0/0 | 13/0/0 | 13/0/0 | 11/1/1 | 11/1/1 | 8/2/3 | 8/2/3 | --- |

为了比较所有八种FA变体在目标测试函数的性能，进行了Friedman测试[18]。表4.6列出了CESFA和其他七个FAs的平均排名。最好的排名（排名值最小）用粗体表示。根据平均秩值，所有FA变量按以下顺序排序：CESFA、NaFA、RaFA、CFA、MFA、standard FA、VSSFA和WSSFA。而CESFA的排名最高。因此，CESFA是八个FA变体中最好的算法。

|  |  |
| --- | --- |
| Table 4.6Friedman 检验 | |
| Algorithm | Mean rank |
| FA\_Mean | 5.54 |
| VSSFA\_Mean | 7.31 |
| WSSFA\_Mean | 7.69 |
| MFA\_Mean | 4.31 |
| CFA\_Mean | 3.54 |
| RaFA\_Mean | 3.23 |
| NaFA\_Mean | 2.38 |
| CESFA\_Mean | **1.85** |

表4.7总结了CESFA和其他FA变体之间的Wilcoxon测试结果。低于0.05（显着性水平）的*p*值以粗体显示。可以看出，CESFA明显优于标准FA，VSSFA和WSSFA。 尽管CESFA并不比MFA，CFA和RaFA好得多，但就其平均等级而言，它的表现要优于它们。

|  |  |
| --- | --- |
| Table 4.7Wilcoxon 检验 | |
| CGPOFA vs. | *p* - values |
| FA\_Mean | **1.00E-03** |
| VSSFA\_Mean | **1.00E-03** |
| WSSFA\_Mean | **1.00E-03** |
| MFA\_Mean | 2.30E-02 |
| CFA\_Mean | 2.30E-02 |
| RaFA\_Mean | 9.10E-02 |
| NaFA\_Mean | 1.31E-01 |

图4.1示出SECFA和其它七个FA变体的一些收敛图。如前所述，在搜索中当前实验阶段未找到最优时，SECFA普遍比其他算法收敛得更快且精度高。在找到全局最优解时，SECFA的收敛也是最快的，如在图f、i、k中。MFA、CFA以及RaFA在大多数函数上都取得了相似的性能。

|  |  |
| --- | --- |
| SECFAGraphF1.jpg | SECFAGraphF2.jpg |
| (a) Sphere (*f*1) | (b) Schwefel 2.22 (*f*2) |
| SECFAGraphF3.jpg | SECFAGraphF4.jpg |
| (c) Schwefel 1.2 (*f*3) | (d) Schwefel 2.21 (*f*4) |
| SECFAGraphF5.jpg | SECFAGraphF6.jpg |
| (e) Rosenbrock (*f*5) | (f) Step (*f*6) |
| SECFAGraphF7.jpg | SECFAGraphF8.jpg |
| (g) Quartic with noise (*f*7) | (h) Schwefel 2.26 (*f*8) |
| SECFAGraphF9.jpg | SECFAGraphF10.jpg |
| (i) Rastrigin (*f*9) | (j) Ackley (*f*10) |
| SECFAGraphF11.jpg | SECFAGraphF12.jpg |
| (k) Griewank (*f*11) | (l) Penalized 1 (*f*12) |
| SECFAGraphF13.jpg |  |
| (m) Penalized 2 (*f*13) |  |
| 图4.1 选定版本FA求解目标函数收敛图 | |

# 5 结论

# 参考文献