第 36 卷 第 4 期　江西师范大学学报(自然科学版)　Vol. 36 No.4
2012 年 7 月　Journal of Jiangxi Normal University (Natural Science)　Jul. 2012

文章编号: 1000-5862(2012)04-0403-04

求解布局问题的混合萤火虫群优化算法

方　瑛，曾　宇

(湖北工业大学理学院，湖北　武汉　430068)

摘要：为了充分发挥萤火虫算法的优点，将人工萤火虫群优化算法与启发式策略相结合，设计了一个新的求解布局问题的高效萤火虫优化算法．实例测试和实验对比结果表明：相对于已有文献中的算法，提出的混合布局方法更加有效．

关键词：矩形布局；启发式策略；萤火虫算法

中图分类号: O 626.4　　文献标志码: A

0　引言

矩形布局问题是寻找一种模式，将一系列大小不等的矩形置于一个大的平面形状中，要求没有相互覆盖现象．这个问题有实际的工程应用背景，如一个人造卫星舱的圆形隔板上，最优地放置一些不同形状、大小和质量的矩形物件(如仪器、设备等)，放置的物体不能超出卫星的圆形底面且不能相互挤压，就是不等矩形在圆形中的布局问题．当问题达到一定规模时，在合理的时间内无法求得问题的精确解．

对于矩形布局问题，国内外学者根据模型的不同做了很多相关的研究．主要采用的方法为启发式算法、元启发式算法和二者结合求解的方法．文献[1-5]直接利用各种演化算法在空间中进行搜索，由于当布局物增加时，直接利用元启发式算法搜索不仅搜索时间较长，布局的精度也达不到要求．近年来，航天器布局的物体由简单的圆形转而研究更为复杂的矩形，带平衡约束的矩形布局的数学模型已被建立起来[6]．分步定位法是求解矩形布局问题的启发式方法，在求解无平衡约束的小矩形在大矩形中的装箱问题上有广泛应用，如文献[7]提出的BL(Bottom-Left)算法，文献[8]提出的BLF(Bottom-Left-Fill)算法．由于布局位置是直接计算的，其速度和精度都非常好，但相对于圆形布局来说，两矩形的干涉函数在定义域内有较多的间断点，因此求解此类布局问题很困难．

萤火虫优化算法(glowworm swarm optimization, GSO)是近年提出的一种新型仿生群智能算法，该算法已在高维空间复杂函数优化方面得到成功的应用，且表现出良好的性能．目前萤火虫算法已成功应用于多模态函数优化、感应器噪音处理、信号源定位、有害气体泄漏检测等，并且表现出良好的性能．为了充分发挥GSO算法的优点，本文采用启发式算法(BL规则)和萤火虫算法相结合的混合算法来求解矩形布局问题．首先利用启发式信息设计了一种分区域分步布局策略，使之满足BL规则的条件，确定矩形件的布局位置，然后结合GSO算法优化矩形的布局顺序，找出问题的最优解，最后用粒子群算法和蚁群算法解决此类布局问题，并进行数值仿真比较．数值试验结果显示，本文提出的混合算法更加有效，本文利用启发式信息既克服了萤火虫算法直接搜索时时间长、精度不高的缺点，又发挥了萤火虫算法在全局寻优的能力．

1　矩形布局问题的启发式求解

1.1　矩形布局问题的数学描述

设在圆形容器上布n个均匀并大小不等的矩形$R_i, i\in I=\{1,2,\cdots,n\}$，质量为$m_i$，$\delta\geqslant 0$为静不平衡量的允许值．建立平面直角坐标系$X$-$O$-$Y$，以$O$为圆心，$R$

收稿日期：2012-01-09
基金项目：湖北省自然科学基金(2009CDB312)和武汉市科技攻关课题(201010621218)资助项目．
作者简介：方　瑛(1963-)，女，湖北武汉人，副教授，主要从事应用数学的研究．

为半径画圆. 设第i个矩形件R_i 的形心为$X_i=(x_i,y_i)$, 矩形的边长分别为a_{i1}与a_{i2}, 其长边所在直线与X轴正向逆时针所成的角为$\theta_i\in[0,\pi)$. 当x_i、θ_i、a_{i1}与a_{i2}确定后, 矩形件R_i在坐标系中的位置、大小就唯一确定了.

采用$d(o,i)=\| OR_i \|$表示点O与矩形件R_i之间的距离, 并且$D(o,i)=\max\{d(o,i)\},i\in I$ 表示所有矩形件离点O 最远点的距离. 为了求解的方便, 本文将多目标规划问题转化为单目标规划问题, 即将优化目标定位于在不平衡量达到最优值0的情况下, 求搜索容器的最小半径. 对于任何一个布局, 如果要求不平衡量为0, 只需将系统的重心平移到容器的中心, 而且转化后模型精度损失不大.设平移后的重心为O', 则转化后的单目标优化模型为

$$\min f_1(X)=D(O',i),\quad \begin{cases} \mathring{R}_i\cap\mathring{R}_j=\varnothing, i\neq j, i,j\in I, \\ f_1(X)<R, \end{cases}$$

这里$\mathring{R}_i$表示矩形图元R_i的内部及其边界.

1.2 启发式确定布局位置

本节把n个矩形件的布局问题视为一个n步决策问题. 下面引入BL条件确定每个矩形的布局位置, 这样, 布局位置的解空间将变成了有限个, 可以大大地减少解的数量.

这里以原点O为中心画半径为R的圆, R为最大允许的半径, 以横坐标轴X与纵坐标轴Y为十字架将圆形区域分成4个区域, 按逆时针分别记为Ⅰ、Ⅱ、Ⅲ、Ⅳ. 在每一个区域内采用BL条件进行分步布局, 即先确定布局区域, 然后在每个区域中确定布局位置, 具体规则如下:

(i) 布局区域: 当待布矩形件确定后, 考虑当前已布局系统静平衡量的偏移量, 若系统中心在第Ⅰ象限, 则将矩形件i布置在第Ⅲ象限; 第Ⅱ、Ⅳ象限类似处理. 若布入的区域确定后, 布置矩形件i时, 当$i=1$时, 在Ⅰ区域采用BL条件直接给出其位置; 当$i>1$时, 若布局位置在Ⅰ区域, 则采用BL条件来确定该矩形的位置; 若为Ⅱ区域, 则将其映射至Ⅰ区域, 同样采用BL条件进行布局; Ⅲ和Ⅳ区域类似处理.

(ii) 布局位置: 当所布局区域内已经放置了i个矩形件, 第$i+1$个矩形件可以布入的位置个数有多个.其位置确定的方法为, 采用贪心策略, 即将矩形件$i+1$放置在所有可能的布入点上, 可以计算出每个临时布局方案的包络半径, 选择包络半径最小的作为矩形件$i+1$的位置.

2 确定矩形布局顺序的萤火虫算法

2.1 GSO 算法

萤火虫群优化算法(GSO)是由 K. N. Krishnanand 和 D. Ghose 于 2005 年提出的一种新的群智能优化算法[8]. 在 GSO 算法中, 萤火虫在其动态决策域内移向荧光素比自己高的个体, 最终聚集于动态域内荧光素高的地方. 目前, GSO 算法已成功应用于多模态函数优化、感应器噪音处理[9]等, 并且表现出良好的性能. 以下给出 GSO 算法的具体描述[10]: $x_i(t)$ 表示 t 时刻第 i 只萤火虫的位置, $f(x)$ 为适应度评价函数, $L_i(t)$ 表示 t 时刻第 i 只萤火虫的荧光素浓度, 运动过程中, 根据 $L_i(t)=(1-\rho)L_i(t-1)+\gamma f(x_i(t))$ 进行更新, 其中 ρ 为荧光素挥发系数, γ 为荧光素增强因子. $r_d^i(t)$ 表示 t 时刻第 i 只萤火虫的动态决策范围, 运动过程中, $r_d^i(t)$ 按

$$r_d^i(t+1)=\min\{r_s,\max\{0,r_d^i(t)+\beta(n_t-|N_i(t)|)\}\} \quad (1)$$

进行调整, 其中 r_s 表示萤火虫感知范围, 且 $0<r_d^i(t)\leqslant r_s$, β 表示邻域变化率, n_t 表示邻居阈值, 控制萤火虫的邻居数目. 这里, $N_i(t)$ 表示 t 时刻第 i 只萤火虫的邻居集合, 由

$$N_i(t)=\{j:\|x_j(t)-x_i(t)\|<r_d^i(t);L_i(t)<L_j(t)\} \quad (2)$$

决定. 萤火虫在运动过程中, 根据其邻居集合中各萤火虫的荧光素浓度来决定其移动方向. $P_{ij}(t)$ 表示 t 时刻第 i 只萤火虫向其邻居集合中第 j 只萤火虫移动的概率, 由该值根据轮盘法选择第 i 只萤火虫移动的方向, 其中

$$P_{ij}(t)=\frac{L_j(t)-L_i(t)}{\sum_{k\in N_i(t)}L_k(t)-L_i(t)}, \quad (3)$$

设移动步长为 s, 则根据下式决定第 i 只萤火虫在 $t+1$ 时刻的位置为

$$x_i(t+1)=x_i(t)+s\left(\frac{x_j(t)-x_i(t)}{\|x_j(t)-x_i(t)\|}\right). \quad (4)$$

2.2 确定矩形布局顺序的萤火虫算法实施步骤

根据实际情况希望能够有较小的包络圆半径, 考虑到优先布局边长长的、质量大的矩形会有较好的结果. 设

$$\varphi(F_i)=\gamma m_i+\alpha a_{i1}+\beta a_{i2}$$

为排序函数, 这里 a、β、$\gamma(0\leqslant a,\beta,\gamma\leqslant 1)$分别为矩形

长边、短边及质量的权重，本文将所有矩形按 $\varphi(F_i)$ 的值非减排序. 为了求得每个布局对应的最佳参数，本文采用 GSO 算法优化 α、β、γ.

设萤火虫个数为 L，第 i 个萤火虫位置用 $x_i=(x_{i1},x_{i2},x_{i3})$ 表示，其中 x_{i1},x_{i2},x_{i3} 表示 α,β,γ 的值. $R(x_i)$为(α,β,γ)取 $x_i=(x_{i1},x_{i2},x_{i3})$ 得到的包络圆半径. 算法步骤如下:

(1)初始化每个萤火虫的位置$x_{ij}(i=1,2,\cdots,L,j=1,2,3)$，令 $L_i(0)=L_0$，局部决策域 $r_d^i(0)=r_0$，设置迭代总次数.

(2) 每只萤火虫寻找最优路径：对每只萤火虫计算适应函数值 $f(x_i)=1/R(x_i)$，其中 $R(x_i)$为(α,β,γ)取 $x_i=(x_{i1},x_{i2},x_{i3})$ 时按照排序函数非减排序选择矩形，并且采用 BLF 规则放置矩形，所得到的包络圆半径. 进行包络半径评价，记录包络半径较小值为当前最优值.

(3) 进行萤光素更新：对每只萤火虫，用(3)式进行荧光素更新. 按照(1)式和(2)式计算 $r_d^i(t+1)$ 和 $N_i(t)$. 对每只萤火虫路径根据公式(3)式和(4)式更新. $t=t+1$ 迭代.

(4) 输出最小的$R(x_i)$和 x_i.

本文算法中设置参数为：$\rho=0.35$，$\gamma=0.57$，$s=0.02$，$n_t=5$，$\beta=0.078$，$\ell_0=6$，$L=10$.

3　数值试验和仿真分析

3.1　数值试验

本文算例用 C 语言实现，为了便于比较，计算时间均已折算到主频为 256 MHz 的微机上. 采用 3 组矩形布局物，待布入矩形件个数分别为 1 000、500、200，把本文提出的萤火虫混合算法与其它 2 种元启发式算法：蚁群算法(ACO)，粒子群算法(PSO)的计算结果进行比较. 表 1 为数值试验的结果，算例 1、2、3 的矩形布局物数目分别为 1 000、500、200.

3.2　结果分析

从表 1 中 3 个算例可以看出，若仅仅采用元启发式算法求解矩形布局问题，当矩形物的规模不是很大的时候，如算例 3，得到问题的空间利用率最高达到 0.340 536，而当矩形件规模增大时，算例 2 和算例 1，得到问题空间利用率便十分低下了，最高仅为 0.083 972(算例 1 的 PSO). 这 2 种算法不仅空间浪费严重，而且搜索时间长. 而采用本文算法在时间和空间利用率上求解效果有了明显的提高. 3 个算例结果中最低的空间利用率为 0.861 653(算例 3，本文算法)，最高位 0.975 280(算例 1，本文算法). 从算例 3 到算例 1 的空间利用率对比来看，随着问题规模的增大，算例 3 到算例 1 在空间利用率上明显增加，本文算法的空间利用率越来越好，算法也较稳定. 此外，本文算法也能大大缩短布局时间.

4　结论

粒子群算法主要适用于连续空间函数的优化问题，粒子群算法不能收敛于全局最优解，甚至于局部最优解，保证收敛的粒子群算法能够收敛于局部最优解，而不能保证收敛于全局最优解，这个算法能应用于求解大多数优化问题，比其它的算法更快地得到高质量的解，但当代数增加时，并不能进行更精确的搜索. 蚁群优化算法是一种针对难解的离散优化问题的元启发式算法，不太适合求解连续性优化问题，一般在离散集中比较有效，该算法在构造解的过程中，利用随机选择策略，这种选择策略使得进化速度较慢，正反馈原理旨在强化性能较好的解，却容易出现停滞现象. 萤火虫算法，是没有记忆性的，每一次朝着视线内被选的邻居移动，萤火虫算法处理多个局部极值问题比较好，求解等值

表 1　各种算法性能比较

算例	算法	包络半径最大值	包络半径最小值	平均包络半径	时间/s	空间利用率
1	ACO	715.417	710.651	713.114	2 356.07	0.072 581
	PSO	710.721	685.727	698.967	2 277.43	0.083 972
	本文算法	226.162	220.771	225.216	356.15	0.975 280
2	ACO	499.413	365.563	386.183	662.18	0.033 567
	PSO	380.722	365.722	372.275	551.61	0.041 432
	本文算法	61.164	65.775	63.525	219.69	0.935 271
3	ACO	56.022	43.186	49.068	239.56	0.202 571
	PSO	45.711	41.925	42.587	238.20	0.340 536
	本文算法	32.567	18.733	30.873	29.79	0.861 653

或不等值的多个局部最优.

本文针对矩形布局问题，采用启发式规则结合萤火虫算法进行寻优. 从实验结果可以看到，本文算法相对现有的蚁群算法和粒子群算法，无论从计算时间还是到解的质量，在容器半径、空间利用率等多个评价指标上均比现有算法有了较大的提高，而且算法稳定. 由于矩形件可用来包络其他不规则图形，本文提供的算法也为不规则图形件布局求解提供了一种新方法.

5 参考文献

[1] 李宁，刘飞，孙德宝. 基于带变异算子粒子群优化算法的约束布局优化研究 [J]. 计算机学报, 2004, 27(7): 897-903.

[2] 黄建江，须文波，孙俊，等. 量子行为粒子群优化算法的布局问题研究 [J]. 计算机应用, 2006, 26(12): 3015-3018.

[3] 刘国志，赵晓颖. 一个与信赖域搜索技术相结合的微粒群算法 [J]. 江西师范大学学报：自然科学版, 2007, 31(5): 463-466.

[4] 赵晓颖，刘国志，姜凤利. 求解一类不可微优化问题极大熵微粒群混合算法 [J]. 江西师范大学学报：自然科学版，2007, 31(2): 193-196.

[5] 鲁强，陈明. 平面布局的蚁群算法 [J]. 计算机应用，2005, 25(5): 1019-1022.

[6] 冯恩民，许广键，滕弘飞. 带平衡约束的矩形图元布局优化模型及不干涉性算法 [J]. 高校应用数学学报, 1993, 8(1): 53-60.

[7] Baker B S, Coffman J R, Rivest R L. Orthogonal packing in two dimensions [J]. Siam Journal on Computing, 1980, 9(4): 846-855.

[8] Chazelle B. The bottom-left bin packing heuristic: an efficient implementation [J]. IEEE Transactions on Computers, 1983, 32(8): 697-707.

[9] Krishnand K N, Ghose D. Detection of multiple source locations using a glowworm metaphor with applications to collective robotics [EB/OL].[2012-03-18]. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1501606.

[10] Krishnand K N, Ghose D. Glowworm swarm optimisation: a new method for optimising multi-modal functions [J]. International Journal of Computational Intelligence Studies, 2009, 1(1): 93-119.

[11] Eberhart R C, Shi Y. Particle swarm optimization: Developments, applications and resources [EB/OL].[2012-03-18]. http://www.nici. kun.nl/~aiweb/aicourses/mki44/slides/SwarmIntelligence/literature/Eberhart01 - PSO.pdf.

Hybrid Glowworm Swarm Optimization Algorithm for Solving Packing Problem

FANG Ying, ZENG Yu

(School of Science, Hubei University of Technology, Wuhan Hubei 430068, China)

Abstract: In order to give full play to the advantages of glowworm swarm algorithm, the paper designed a new efficient algorithm in which the glowworm swarm optimization algorithm has been combined with heuristic strategy to solve the packing problem. Through the comparison of experimental data, the results show that the proposed layout method is more effective than other known algorithms.

Key words: layout problem; heuristic tactics; glowworm swarm algorithm

(责任编辑：冉小晓)