计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(10):217−221 [doi: 10.15888/j.cnki.csa.007635]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

改进的蝴蝶优化聚类算法①

郑洪清

(广西外国语学院 信息工程学院, 南宁 530222)
通讯作者: 郑洪清, E-mail: zhq7972@sina.com

摘　要: 针对当前算法在求解聚类问题时存在精度低、速度慢及鲁棒性差等问题, 提出一种改进的蝴蝶优化聚类算法, 借鉴精英策略思想重新定义蝴蝶优化算法的局部搜索迭代公式, 然后融合遗传算法的选择、交叉和变异操作. 在 1 个人工数据集和 5 个 UCI 数据集上的测试结果表明所提出算法的性能, 且与其他算法相比具有一定优势.

关键词: 聚类算法; 蝴蝶优化算法; 遗传算法

引用格式: 郑洪清.改进的蝴蝶优化聚类算法.计算机系统应用,2020,29(10):217–221. http://www.c-s-a.org.cn/1003-3254/7635.html

Improved Butterfly Optimization Algorithm for Clustering

ZHENG Hong-Qing

(School of Information Engineering, Guangxi University of Foreign Languages, Nanning 530222, China)

Abstract: Aiming at the problems of low accuracy, slow speed, and poor robustness of the current algorithm in solving the clustering problem, an improved butterfly optimization clustering algorithm was proposed. Based on the idea of elite strategy, the local search iterative formula of butterfly optimization algorithm was redefined, and then the selection, crossover, and mutation operations of genetic algorithm were fused. Test results on one artificial dataset and five UCI datasets demonstrate that the performance of the proposed algorithm is superior to other algorithms.

Key words: clustering algorithm; butterfly optimization algorithm; genetic algorithm

聚类是将数据集中的样本划分为若干个通常不相交的子集, 每个子集称为“族”, 同一族中的对象具有较高的相似度, 不同族中的对象差别较大. 聚类分析已成为数据挖掘领域的研究热点问题, K 均值算法 (K-means) 是一种经典的聚类算法, 因其受初始聚类中心的影响容易陷入局部最优等缺陷而限制了其应用范围. 许多学者提出一系列智能聚类算法, 如蜜蜂交配优化聚类算法[1]、萤火虫聚类算法[2]、差分进化聚类算法[3]、布谷鸟聚类算法[4]、弹性网络聚类算法[5]、花朵授粉聚类算法[6]、蝙蝠聚类算法[7]、灰狼与郊狼混合优化算法[8]、自适应细菌觅食聚类优化算法[9]、拓展差异度的高维数据聚类算法[10]、无人仓系统订单分批问题及 K-max 聚类算法[11] 等. 各种改进算法均取得了一定成效, 但对于一些复杂问题仍存在精度不高和收敛速度慢等问题.

蝴蝶优化算法[12](Butterfly Optimization Algorithm, BOA) 是 2018 年 Sankalap Arora 提出一种新的全局优化算法, 其灵感来自于蝴蝶的觅食行为. 该行为由蝴蝶的合作向食物来源位置移动, 蝴蝶接收并感知空气中的气味以确定食物来源或交配伙伴的潜在方向, 但其本质不同于文献 [13,14]. 由于提出时间短, 国外可参考的文献很少, 国内暂无相关论文报道. 蝴蝶优化算法与其他群智能算法一样, 也存在收敛速度和易陷入局部最优等缺陷, 因此本文尝试提出了一种改进蝴蝶优化算法 (Improved Butterfly Optimization Algorithm, IBOA), 首先描述了蝴蝶优化算法的特点及实施步骤,

① 收稿时间: 2020-03-02; 修改时间: 2020-03-27, 2020-04-10; 采用时间: 2020-04-21; csa 在线出版时间: 2020-09-30

重新定义蝴蝶优化算法的局部迭代公式, 再将遗传算法的轮盘赌选择、交叉操作和变异操作融入蝴蝶优化算法中, 通过标准的数据集测试验证 IBOA 算法的有效性.

1 基本的蝴蝶优化算法

蝴蝶优化算法是模拟蝴蝶的觅食行为, 该思想的条件假设如下:

1) 所有的蝴蝶都应该散发出某种香味, 使蝴蝶能够互相吸引.

2) 每只蝴蝶都会随机移动, 或朝着散发出更多香味的最佳蝴蝶移动.

3) 蝴蝶的刺激强度受目标函数值的影响或决定.

当蝴蝶能感觉到其他任何蝴蝶的香味时并朝它移动, 在该算法中, 该阶段称为全局搜索. 在另一种情况下, 当蝴蝶不能感觉周围的香味时, 然后它会随机移动这个阶段称为局部搜索. 利用转换概率 控制全局和局部搜索过程, 其迭代公式为:

$$f = c^t I^a \tag{1}$$

$$x_i^{t+1} = x_i^t + \left(r^2 \times g^* - x_i^t\right) \times f_i \tag{2}$$

$$x_i^{t+1} = x_i^t + \left(r^2 \times x_j^t - x_k^t\right) \times f_i \tag{3}$$

$$c^{t+1} = c^t + \left(b/c^t \times Ngen\right) \tag{4}$$

式 (1) 中的f是香味感知量, c是感觉形式, I是刺激强度, a是香味, 通常a和c的取值范围为 [0, 1] 之间; 式 (2) 中的x_i^{t+1}表示第i只蝴蝶在第$t+1$代的位置, $r \in [0,1]$的随机数, g^*表示全局最优解, f_i表示第i只蝴蝶的香味感知量; 式 (3) 中的x_i^t, x_j^t, x_k^t分别表示第i, j, k只蝴蝶在第t代的位置; 式 (4) 中的b为常数, c^t, 表示第t代的值, $Ngen$ 为最大迭代次数.

基于上述描述, 蝴蝶优化算法的实施步骤如下:

Step 1. 初始化种群规模n及转换概率p等参数.

Step 2. 利用式 (5) 计算每只蝴蝶的适应度值, 并求出当前最优值$f_{\min}$和最优解 Best.

Step 3. 利用式 (1) 计算香味感知量, 如果 $rand<p$, 则利用式 (2) 计算x_i^t, 否则利用式 (3) 计算x_i^t.

Step 4. 利用式 (5) 重新计算每只蝴蝶的适应度值 F_{new}, if $F_{\text{new}}<f_{\min}$, 则替换之前的最优值和最优解.

Step 5. 利用式 (4) 更新c.

Step 6. 判断是否达到最大迭代次数, 如果是输出最优值和最优解, 否则跳至 Step 3.

2 改进的蝴蝶优化算法

由于基本的 BOA 算法聚类效果差, 本文对其进行改进, 提出一种改进的蝴蝶优化聚类算法. 重新定义蝴蝶的局部搜索方式, 同时结合轮盘赌选择、交叉操作和变异操作, 提高算法的寻优能力, 使聚类效果稳定.

2.1 编码方法

采用实数编码, 一只蝴蝶的位置表示一组聚类中心, 假设有m个聚类中心, 数据集的属性有d个, 则每只蝴蝶的维数为$nd = d \times m$. 则第t代蝴蝶i的位置编码为$x_i(t) = [c_1(t), c_2(t), \cdots, c_m(t),]$, 其中$c_j(t)$表示第$t$代蝴蝶的第$j$个聚类中心, $j = 1, 2, \cdots, m$.

2.2 评价函数

本文采用如下的聚类准则作为适应度值:

$$f(x_i(t)) = \min \sum_{i=1}^{n_c} \sum_{j=1}^{m} \|y_i - c_j(t)\| \tag{5}$$

其中, n_c为聚类样本数, y_i为第i个样本, $f(x_i(t))$表示所有数据到聚类中心的最小值, 值越小表示聚类效果越好.

2.3 重新定义迭代公式

鉴于基本的蝴蝶优化算法局部寻优能力较差, 故结合精英策略将式 (3) 重新定义如下:

$$x_i^{t+1} = x_i^t + r \times (x_j^t - x_k^t) + r \times (g^* - x_i^t) \tag{6}$$

式 (6) 中g^*为全局最优解即为精英, 其他蝴蝶在精英附近进行搜索, 因此能提高算法的精度.

为了提高聚类效果和鲁棒性, 融合遗传算法相关操作.

2.4 轮盘赌选择

1) 计算每只蝴蝶的适应度值 *Fitness*;

2) 计算每只蝴蝶被遗传到下一代的概率 p;

3) 计算每只蝴蝶的累加概率 q;

4) 在蝴蝶种群中对每一个体产生一个随机数$r \in [0,1]$, 执行 $temp$=find(r<q), $x_i^{t+1} = x_i^t(temp(1),:)$.

2.5 交叉操作

1) 在蝴蝶种群中随机选择两个父类x_{j+1}^t, x_j^t, 设置交叉概率 p_c;

2) 若 r<pc, 随机产生一个交叉点 $point$; $temp = x_j^t$;

3) $x_j^t(:, point+1:nd) = x_{j+1}^t(:, point+1:nd)$ $x_{j+1}^t(:, point+1:nd) = temp$.

2.6 变异操作

设 $x_i=(x_{i1},x_{i2},\cdots,x_{id},x_{id+1},\cdots,x_{ie},\cdots,x_{in})$, 随机选取两个位置 x_{id} 和 x_{ie}, 将两个位置之间的元素进行逆转操作, 变换后的位置为 $x_i'=(x_{i1},x_{i2},\cdots,x_{ie},x_{ie-1},\cdots,x_{id},\cdots,x_{in})$.

2.7 IBOA 实施步骤

IBOA 算法求解聚类问题的步骤:

Step 1. 初始化种群规模、转换概率、迭代次数 *N_iter* 和交叉概率 p_c 等参数.

Step 2. 计算每只蝴蝶的适应度值, 并求出当前最优值 $f_{\min}$ 和最优解 Best.

Step 3. 利用式 (1) 计算香味感知量, 如果 *rand*<*p*, 则利用式 (2) 计算, 否则利用式 (6) 计算.

Step 4. 重新计算每只蝴蝶的适应度值 F_{new}, if $F_{\text{new}}<f_{\min}$, 则替换之前的最优值和最优解.

Step 5. 执行轮盘赌选择、交叉操作和变异操作. 重新计算每只蝴蝶的适应度值 F_{new}, if $F_{\text{new}}<f_{\min}$, 则替换之前的最优值和最优解.

Step 6. 利用式 (4) 更新.

Step 7. 判断是否达到最大迭代次数, 如果是输出最优值和最优解, 否则跳至 Step 3.

3 实验分析

3.1 实验环境与参数设置

为了测试 IBOA 算法的正确性与有效性, 选取 6 个基准测试函数来验证算法, 包括 1 个人工数据集和 5 个从 UCI (http://archive.ics.uci.edu/ml/index.php) 数据库中选取了 Iris、Wine、Glass、Cancer、Cintraceptive Method Choice (简称 CMC) 5 组实验数据, 所有的实例均运行在处理器为 Celeron(R) 双核 CPU T3100, 1.90 GHz、内存为 4G 的 PC 上, 以 Matlab R2010a 编写代码. 参数设置为: 种群规模 =50、转换概率 =0.1、*c* 的初值为 0.01, 迭代次数 *N_iter*=200 和交叉概率 *pc*=0.85. 在问题规模一致的情形下, 这些算法的复杂度是相同的.

3.2 测试实例结果比较

(1) 人工数据集 1 (*set_data*=250, *d*=3, *K*=5): 为了展示 IBOA 的求解过程, 分别计算第 10 代、第 50 代的求解结果如图 1 和图 2 中. 并将算法独立运行 20 次的结果于表 1 中, Best 表示最优解, Average 表示平均解, Worst 表示最差解, Std 表示标准差.表 1 中其他算法与数据来源于文献 [7], 从表 1 中的计算结果可知 IBOA 的求解精度及鲁棒性均优于其他算法.

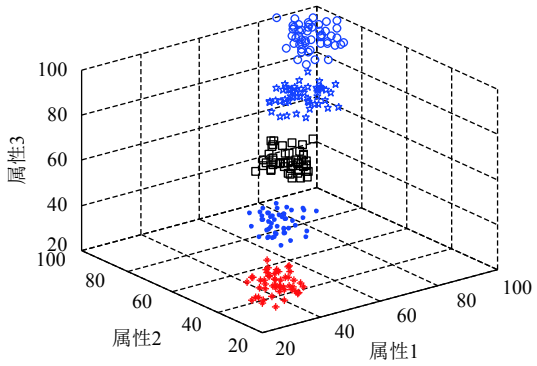

图 1 人工数据集 1 第 10 代的聚类结果

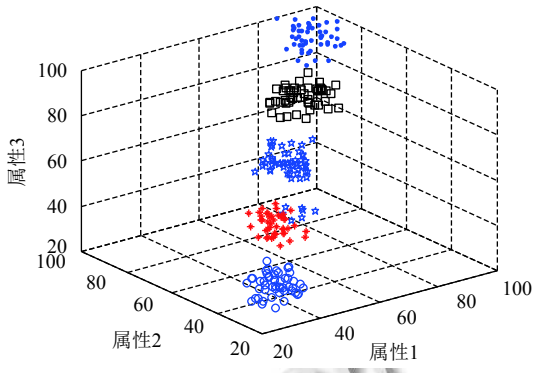

图 2 人工数据集 1 第 50 代的聚类结果

表 1 人工数据集 1 的 20 次独立运行结果比较

算法	Best	Average	Worst	Std
K-means	1747.3859	1991.9351	2507.9091	342.2974
PSO	1718.2538	2153.7595	2444.8930	344.4796
ABC	1718.2069	1899.4503	2059.1212	77.4967
IBA	1718.2538	1784.2538	2285.1103	150.9430
BOA	2720.5851	3288.8383	3705.1412	217.1649
IBOA	**1718.2538**	**1718.9507**	**1732.0992**	**3.0948**

(2)UCI 数据集: 将算法独立运行 20 次, 与近几年多种算法比较如表 2–表 7 所示, 其中表 2–表 6 中的 K-means、GA、ACO、PSO、HBMO、IDE 算法数据来源于文献 [3] 且迭代次数为 500 时的计算结果, IBA 算法数据来源于文献 [7], IGSO 算法数据来源于文献 [2], BPFPA 算法数据来源于文献 [6]; 表 7 中 K-means、PSO、ABC、BA 和 IBA 算法数据来源于文献 [7]. 从表 2 可知, IBOA 算法求解效果与 IBA、BPFPA 相当, 但比其余 8 种算法效果较好; 从表 3 可知, IBOA 算法与

BPFPA 求解效果相当, 但比其余 7 种算法效果优越许多; 文中的“—”表示未有相关数据. 从表 4 可知, IBOA 的求结果在迭代次数为 200 时优于 IDE 和其他算法; 从表 5 可知, IBOA 算法的精度和方差均优于其他算法; 从表 6 可知, IBOA 算法的求解效果差于 IDE, 与 IBA、BPFPA 效果相当, 但优于其他算法; 从表 7 可知, IBOA 算法的求解效果与 IBA、BPFPA 相当, 但优于其他算法. 另外, 图 3 展示了 IBOA 算法和 BOA 算法在 Survival 数据集的最优解收敛曲线图, 从图 3 易知, IBOA 算法的求解速度和精度较 BOA 算法高. IBOA 求解 Iris、Survival、CMC 数据集的聚类效果图如图 4–图 6 所示.

表 2 11 种算法在 Iris 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	97.33	106.05	120.45	14.631
GA	113.98	125.19	139.77	14.563
ACO	97.10	97.17	97.81	0.367
PSO	96.89	97.23	97.89	0.347
HBMO	96.75	96.95	97.75	0.531
IDE	97.22	97.22	97.22	0
IBA	96.6555	96.6555	96.6555	0
IGSO	96.6831	97.3242	97.7154	—
BPFPA	96.6555	96.6555	96.6555	8.2e–13
BOA	131.0493	144.2562	156.8839	7.3156
IBOA	**96.6555**	**96.6555**	**96.6555**	**4.374e–014**

表 3 10 种算法在 Wine 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	16 555.68	18 061	18 563.12	390
GA	1653	16 530.53	16 530.53	0
ACO	16 530.53	16 530.53	16 530.53	0
PSO	16 345.97	16 417.47	16 562.32	85.497
HBMO	16 357.28	16 357.28	16 357.28	0
IGSO	16 309	16 348	16 376	—
IDE	16 530.54	16 530.54	16 530.54	0
BPFPA	16 292.1847	16 292.9230	16 294.172	0.7663
BOA	16 604.0170	16 886.9516	17 348.4400	180.729
IBOA	**16 292.1847**	**16 293.0218**	**16 294.171**	**0.7940**

表 4 9 种算法在 Glass 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	215.74	235.5	255.38	12.471
GA	278.37	282.32	286.77	4.139
ACO	269.72	273.46	280.08	3.585
PSO	270.57	275.71	283.52	4.557
HBMO	245.73	247.71	249.54	2.438
IGSO	215.1349	219.4834	223.4760	—
IDE	213.20	213.23	213.31	0.028
BOA	428.7888	505.3418	599.5460	36.7987
IBOA	**210.4307**	**218.4789**	**241.2814**	**9.0920**

表 5 9 种算法在 Cancer 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	2999.19	3251.21	3251.59	251.140
GA	2999.32	3249.46	3427.43	229.734
ACO	2970.49	3046.06	3242.01	90.500
PSO	2973.50	3050.04	3318.88	110.801
HBMO	2989.94	3112.42	3210.78	103.471
IDE	2984.07	2984.07	2984.07	0
IBA	2964.3869	2967.8246	2970.7369	3.1502
BOA	3307.4227	3558.3359	3756.6774	113.1327
IBOA	**2.9644e+3**	**2.9644e+3**	**2.9644e+3**	**4.6545e–13**

表 6 10 种算法在 CMC 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	5842.20	5893.60	5934.40	47.160
GA	5705.63	5756.60	5812.65	50.369
ACO	5701.92	5819.13	5912.43	45.634
PSO	5700.99	5820.96	5923.25	46.960
HBMO	5699.27	5713.98	5725.35	12.690
IDE	5541.64	5541.64	5541.65	0.001
IBA	5693.7239	5693.7512	5693.7983	0.0032
BPFPA	5693.7239	5693.7283	5693.7246	0.0016
BOA	6607.5554	7160.9892	7840.6507	324.3583
IBOA	**5693.7239**	**5693.7284**	**5693.7244**	**0.0013**

表 7 8 种算法在 Survival 数据集上聚类结果比较

算法	Best	Average	Worst	Std
K-means	2976.9441	2983.3164	2988.4278	4.8661
PSO	2964.3871	3140.8281	4728.7901	543.0715
ABC	2982.8207	3046.3477	3190.5028	48.8843
BA	2970.7369	3125.0756	3179.4228	149.2531
IBA	2566.9889	2567.0314	2567.8248	0.1765
BPFPA	2566.9889	2567.0307	2567.8248	0.1869
BOA	2602.0468	2946.7588	3083.7954	119.7539
IBOA	**2566.9889**	**2567.0724**	**2567.8248**	**0.2572**

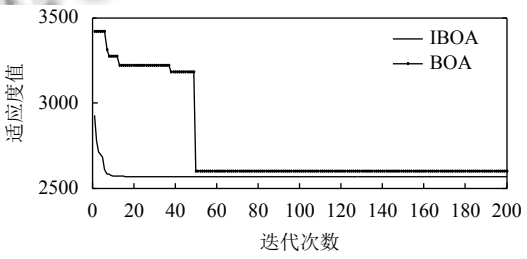

图 3 IBOA 与 BOA 求解 Survival 数据集函数收敛曲线图

4 结论

本文将精英策略的思想重新定义蝴蝶优化算法的局部搜索迭代公式且遗传算法相结合提出了一种改进的蝴蝶优化聚类算法, 通过求解 1 个人工数据集和 5 个 UCI 数据库中不同规模的数据, 统计分析结果表明 IBOA 算法能够避免陷入局部最优, 具有较快的收

敛速度和较强的鲁棒性, 能够有效解决聚类问题且与其他聚类算法相比具有一定优势.

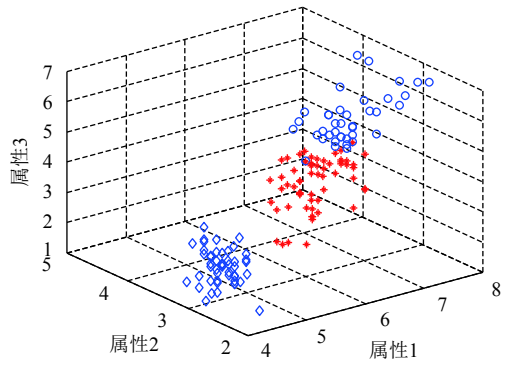

图 4 IBOA 求解 Iris 数据集的聚类效果图

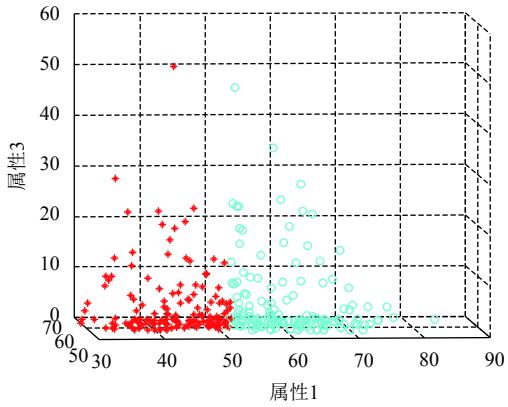

图 5 IBOA 求解 Survival 数据集的聚类效果图

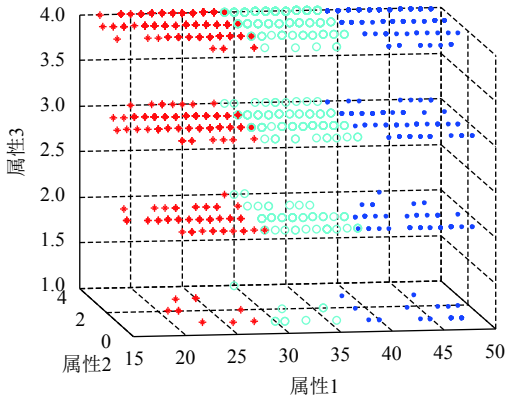

图 6 IBOA 求解 CMC 数据集的聚类效果图

参考文献

1 罗可, 李莲, 周博翔. 一种蜜蜂交配优化聚类算法. 电子学报, 2014, 42(12): 2435–2440. [doi: 10.3969/j.issn.0372-2112.2014.12.015]

2 杜明煜, 雷秀娟. 一种改进的求解聚类问题的萤火虫群优化算法. 陕西师范大学学报 (自然科学版), 2014, 42(3): 20–23.

3 王勇臻, 陈燕, 张金松. 一种改进的求解聚类问题的差分进化算法. 计算机应用研究, 2016, 33(9): 2630–2633. [doi: 10.3969/j.issn.1001-3695.2016.09.014]

4 杨辉华, 王克, 李灵巧, 等. 基于自适应布谷鸟搜索算法的 K-means 聚类算法及其应用. 计算机应用, 2016, 36(8): 2066–2070. [doi: 10.11772/j.issn.1001-9081.2016.08.2066]

5 沈小云, 衣俊艳. 面向聚类分析的自适应弹性网络算法研究. 计算机工程与应用, 2017, 53(9): 175–183. [doi: 10.3778/j.issn.1002-8331.1611-0316]

6 Wang R, Zhou YQ, Qiao SL, *et al*. Flower pollination algorithm with bee pollinator for cluster analysis. Information Processing Letters, 2016, 116(1): 1–14. [doi: 10.1016/j.ipl.2015.08.007]

7 熊珍, 傅秀芬. 求解聚类问题的异构蝙蝠算法. 计算机工程与设计, 2017, 38(3): 677–681, 728.

8 张新明, 姜云, 刘尚旺, 等. 灰狼与郊狼混合优化算法及其聚类优化. 自动化学报. https://doi.org/10.16383/j.aas.c190617. [2020-03-26].

9 刘志鹏, 胡亚琦, 张卫卫. 自适应细菌觅食的 FCM 聚类优化算法研究. 现代电子技术, 2020, 43(6): 144–148.

10 武森, 何慧霞, 范岩岩. 拓展差异度的高维数据聚类算法. 计算机工程与应用 https://www.cnki.net/KCMS/detail/11.2127.tp.20200323.1607.004.html. [2020-03-24].

11 李珍萍, 田宇璇, 卜晓奇, 等. 无人仓系统订单分批问题及 K-max 聚类算法. 计算机集成制造系统. http://kns.cnki.net/kcms/detail/11.5946.TP.20200320.1423.010.html. [2020-03-20].

12 Arora S, Singh S. Butterfly optimization algorithm: A novel approach for global optimization. Soft Computing, 2018, 23(3): 715–734.

13 Wang GG, Deb S, Cui ZH. Monarch butterfly optimization. Neural Computing and Applications, 2019, 31(7): 1995–2014. [doi: 10.1007/s00521-015-1923-y]

14 Arora S, Singh S. Butterfly algorithm with Lèvy Flights for global optimization. Proceedings of 2015 International Conference on Signal Processing, Computing and Control (ISPCC). Waknaghat, India. 2015. 220–224.