蜣螂优化算法（DBO）的复现及改进

摘要

本文介绍并复现了蜣螂优化算法（DBO）并对其进行改进。DBO 受蜣螂滚球、跳舞、觅食、偷窃和繁殖行为的启发，将种群按行为划分为滚球、繁殖、觅食和偷窃四个子种群，并分别采用不同的更新规则进行搜索。改进的蜣螂优化算法（IDBO）在 DBO 的基础上，加入了自适应参数调整、遗传算法混合和局部搜索增强等技术，以提高算法的性能。实验结果表明，IDBO 在 CEC2005 测试集上取得了比 DBO 更好的优化结果，并具有更快的收敛速度。

关键词：蜣螂优化算法；改进蜣螂优化算法

1 引言

优化问题长期以来一直是研究的焦点，大量复杂的优化问题（例如 NP 完全问题）特别难以使用传统的数学规划技术（例如共轭梯度和拟牛顿法）来解决。在这方面，大量的群体智能（SI）优化算法被引入，具有易于实现、自学习能力和框架简单的优点。具体来说，SI 系统可以被视为一个群体，其中每个个体表示整个搜索空间中的候选解决方案。另外，SI 系统的特点是个体交互促进智能行为的出现。

例如，一种著名的基于群体的技术，即粒子群优化（PSO）技术，以其收敛速度快、参数少和求解精度满意的优点而受到广泛的研究关注 [2]。此外，蚁群优化（ACO）算法也与 PSO 方法一样成为著名的基于 SI 的技术 [4]。这些基于 SI 的算法主要模仿自然界生物（例如鱼、昆虫和鸟类）的社会行为 [3]。例如， [6] 中提出了灰狼优化器（GWO）算法，该算法模拟了灰狼的领导层次结构（包括 alpha、beta、delta 和 omega）和狩猎行为。还有，鲸鱼优化算法（WOA） [5]、Harris hawks 优化器（HHO）算法 [1] 等等。

本文不仅介绍了一种新的群体智能算法——蜣螂优化算法 (DBO)，还提出了改进的蜣螂优化算法 (IDBO)。

2 算法灵感来源

蜣螂俗称屎壳郎，是自然界常见的昆虫，通常承担分解者的角色。根据蜣螂的滚球、跳舞、觅食、偷窃和繁殖行为，提出了蜣螂优化器。

蜣螂有一个有趣的习惯，它们会把粪便滚成球，为了避免抢夺，需要快速移动并尽量走直线，故蜣螂利用天体线索 (特别是太阳、月亮和偏振光) 来导航，如果完全没有光源 (也就是在完全黑暗的环境中)，蜣螂的就不再走直线，而是弯曲的，有时甚至略圆，有很多因素 (如

风、地面不平) 都会导致蜣螂偏离原来的方向。当蜣螂在滚球的过程中遇到障碍物时，通常会爬到粪球上面“跳舞”(包括一系列的旋转和停顿)，决定它们的运动方向。

从蜣螂的习性中观察发现，其获取粪球主要有以下两个目的：1) 用来产卵和养育下一代；2) 作为食物。蜣螂会把粪球埋起来，雌性蜣螂会在粪球里产卵，粪球不仅是蜣螂幼虫的发育场所，也是必需的食物。所以，粪球对蜣螂的生存起着不可替代的作用 [7]。

3 算法框架

3.1 滚球行为

3.1.1 无障碍模式

当前进道路通畅时，蜣螂会利用太阳光进行导航，假设光源强度会影响蜣螂的位置，则位置更新如下：

$$\begin{aligned} x_i(t+1) &= x_i(t) + \alpha \times k \times x_i(t-1) + b \times \Delta x, \\ \Delta x &= |x_i(t) - X^w| \end{aligned} \tag{1}$$

其中，t 表示当前迭代次数，x_i 表示种群中第 i 只蜣螂在第 t 次迭代位置。α 是值为 -1 或 1 的自然系数，1 表示无偏差，-1 表示偏离原方向，$k \in (0, 0.2]$ 表示一个常值，代表偏转系数，b 是 $(0, 1)$ 之间的一个常值。X^w 表示当前种群中的最差位置，ΔX 用于模拟光强的变化。

3.1.2 有障碍模式

当蜣螂遇到障碍物时，会跳“导航舞”确定方向，如图 1所示。作者用切线函数模仿跳舞行为，从而获得新方向，如图 2所示。定义域仅考虑 $[0, \pi]$ 之间，且当 θ 值为 0，$\frac{\pi}{2}$，π 时，不考虑位置更新。位置更新公式如下：

$$x_i(t+1) = x_i(t) + \tan(\theta)|x_i(t) - x_i(t-1)| \tag{2}$$

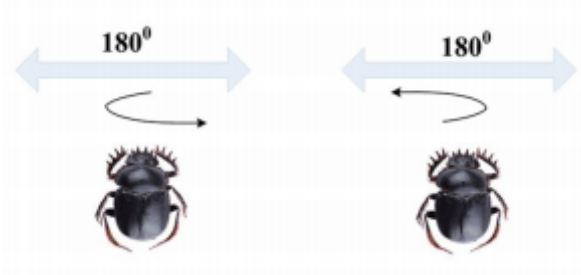

图 1. 蜣螂舞蹈行为

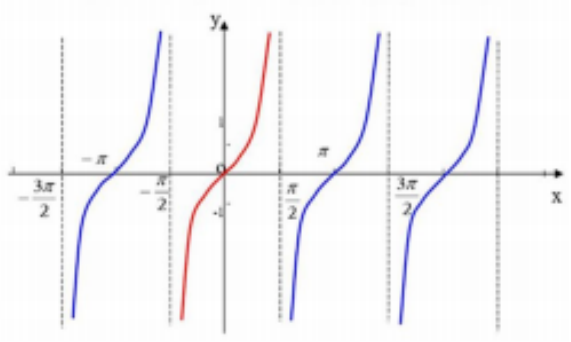

图 2. 正切函数图像

3.2 繁殖行为

自然界中，蜣螂会找一个安全隐蔽的地方产卵。作者提出一种边界选择策略来模拟产卵区域，也就是将上下限用如下公式进行更新：

$$\begin{aligned} Lb^* &= \max(X^* \times (1-R), Lb), \\ Ub^* &= \min(X^* \times (1+R), Ub) \end{aligned} \tag{3}$$

其中，X^* 表示局部最优，$R = 1 - \frac{t}{T}$，t 为当前迭代次数，T 为最大迭代次数，Lb 和 Ub 分别表示上界和下界。随后，个体位置将更新为：

$$B_i(t+1) = X^* + b_1 \times (B_i(t) - Lb^*) + b_2 \times (B_i(t) - Ub^*) \tag{4}$$

b1、b2 为两个独立的大小为 1*D 范围为 (0,1) 的随机向量。

3.3 觅食行为

一些小的蜣螂会在较优的范围内觅食，我们需要建立最佳的觅食区域以模仿在自然界中的觅食行为，觅食区域定义如下：

$$\begin{aligned} Lb^b &= \max(X^b \times (1-R), Lb), \\ Ub^b &= \min(X^b \times (1+R), Ub). \end{aligned} \tag{5}$$

其中，X^b 表示全局最优，$R = 1 - \frac{t}{T}$，t 为当前迭代次数，T 为最大迭代次数，Lb 和 Ub 分别表示上界和下界。随后，小蜣螂位置将更新为：

$$x_i(t+1) = x_i(t) + C_1 \times (x_i(t) - L_b^b) + C_2 \times (x_i(t) - U_b^b) \tag{6}$$

其中，C_1 为服从正态分布的随机数，即 $C_1 \sim \mathcal{N}(0,1)$，$C_2$ 为 $1 \times D$ 维的 $(0,1)$ 间的随机向量。

3.4 偷窃行为

在种群中，会有一些蜣螂去偷别人的食物，这在自然界也是及其常见的，其中偷窃蜣螂的位置更新为：

$$x_i^{t+1} = x_{ibest}^t + S \cdot g \cdot (|x_i^t - x_{gbest}^t| + |x_i^t - x_{ibest}^t|) \tag{7}$$

其中，S 表示一个常数值，g 表示大小为 1*D 的随机向量，服从正态分布。

3.5 种群划分

文章将种群划分为四个子种群，具体子种群数由实际问题决定，这四个子种群对应上述四种行为：滚球行为、繁殖行为、觅食行为和偷窃行为。最后，整体蜣螂优化算法的伪代码如下。

Algorithm 1 The framework of the DBO algorithm

Require: The maximum iterations $T_{\max}$, the size of the particle's population N.

Ensure: Optimal position X^b and its fitness value f_b.

Initialize the particle's population $i \leftarrow 1, 2, \ldots, N$ and define its relevant parameters
while ($t \leq T_{\max}$) **do**
for $i \leftarrow 1$ to N **do**
  if $i ==$ ball-rolling dung beetle **then**
    $\delta = \text{rand}(1)$;
    if $\delta < 0.9$ **then**
      Select α value by Algorithm 1
      Update the ball-rolling dung beetle's position by using (1);
    else
      Update the ball-rolling dung beetle's position by using (2);
    end if
  end if
  if $i ==$ brood ball **then**
    Update the brood ball's position by using Algorithm 3;
  end if
  if $i ==$ small dung beetle **then**
    Update the small dung beetle's position by using (6);
  end if
  if $i ==$ thief **then**
    Update the position of the thief by using (7);
  end if
end for
if the newly generated position is better than before **then**
  Update it;
end if
$t = t + 1$;
end while
return X^b and its fitness value f_b.

4 复现改进细节

4.1 与 DBO 的对比

在复现的基础上，对算法做了如下改进，称改进后的算法为 IDBO：

- **自适应参数调整**。将太阳光引导系数根据迭代次数动态调整，调整公式为 $1-(t/T)^2$。
- **多策略混合**。与遗传算法的交叉混合，交叉算子为选择多点交叉或单点交叉。
- **局部搜索增强**。采用简单的梯度下降来细化搜索区域。

4.2 创新点

该算法的创新点主要体现在：

- 多个子种群不同的更新规则可以维护局部和全局搜索的平衡。
- 搜索区域有动态变化的特点可以促进有效搜索。
- 结合遗传算法，扩大了种群多样性。

5 实验结果分析

将蜣螂优化算法（DBO）和改进的蜣螂优化算法（IDBO）在 CEC2005 测试集 [8] 的 23 个测试函数上测试的对比实验结果如图 3所示。结果显示其中 13 个函数的结果 IDBO 明显比 DBO 好，其余 10 个函数 IDBO 和 DBO 不相上下，但是 IDBO 的收敛速度普遍比 DBO 好。

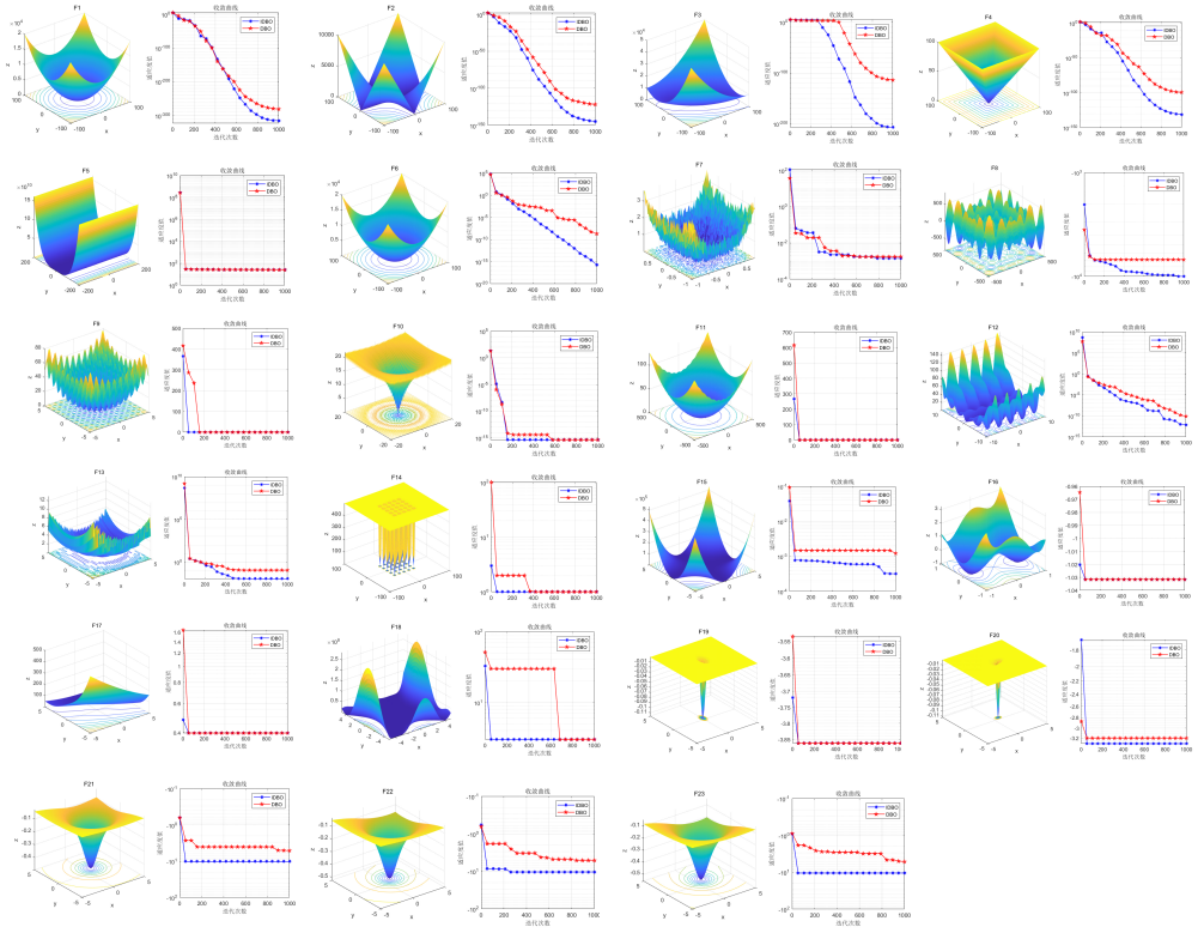

图 3. CEC2005 测试集实验结果示意

6 总结与展望

实验的不足之处在于没有在实际问题中测试 IDBO 的效果与 DBO 的差距，后续可能的研究方向:1) 考虑不同子种群的关系是否可以重叠和交叉，论文中子种群的数量是固定的，是否可以考虑在迭代过程中子种群数量变动。2) 将蜣螂优化算法的思路和优点，如子种群及动

态搜索范围引入别的算法测试是否能改进别的算法。3) 结合现实中的实际问题进一步探索算法的有效性。

参考文献

[1] Ali Asghar Heidari, Seyedali Mirjalili, Hossam Faris, Ibrahim Aljarah, Majdi Mafarja, and Huiling Chen. Harris hawks optimization: Algorithm and applications. *Future generation computer systems*, 97:849–872, 2019.

[2] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. ieee, 1995.

[3] Maodong Li, Guanghui Xu, Bo Fu, and Xilin Zhao. Whale optimization algorithm based on dynamic pinhole imaging and adaptive strategy. *The Journal of Supercomputing*, pages 1–31, 2022.

[4] Jianhua Liu, Jianguo Yang, Huaping Liu, Xingjun Tian, and Meng Gao. An improved ant colony algorithm for robot path planning. *Soft computing*, 21:5829–5839, 2017.

[5] Seyedali Mirjalili and Andrew Lewis. The whale optimization algorithm. *Advances in engineering software*, 95:51–67, 2016.

[6] SMSM Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer adv eng softw 69: 46–61. *ed*, 2014.

[7] Jiankai Xue and Bo Shen. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *The Journal of Supercomputing*, 79(7):7305–7336, 2023.

[8] Xin Yao, Yong Liu, and Guangming Lin. Evolutionary programming made faster. *IEEE Transactions on Evolutionary computation*, 3(2):82–102, 1999.