融合鲸鱼算法的混合灰狼优化算法

梁昔明 [1], 李 星 [1], 龙 文 [2]

(1. 北京建筑大学 理学院, 北京, 102616)
(2. 贵州财经大学 经济系统仿真贵州省重点实验室, 贵州 贵阳 550025)

摘 要: 针对灰狼优化 (GWO) 算法存在容易陷入局部最优、收敛速度慢、求解精度不高等问题, 提出一种融合鲸鱼算法的混合灰狼优化 (HWGWO) 算法. 首先在鲸鱼算法的螺旋泡网狩猎行为中融入 Levy 飞行并将其整体引入灰狼优化算法; 然后将动态权重和差分进化思想引入灰狼优化算法; 最后利用贪婪选择策略来保留较好的灰狼位置. 选取 23 个测试函数进行数值试验, 结果表明, HWGWO 算法在收敛速度和求解精度上都有所提升. 此外, 利用 HWGWO 算法求解拉伸/压缩弹簧设计问题得到的设计方案更有效.

关键词: 灰狼优化算法; Levy 飞行; 鲸鱼优化算法; 数值试验; 弹簧设计问题

通过对自然界中不同生物群体的行为机制进行模拟, 产生了众多的群体智能优化算法, 其中较为经典的有粒子群优化 (particle swarm optimization, PSO) 算法 [1]、蚁群优化 (ant colony optimization, ACO) 算法 [2] 和萤火虫算法 (firefly algorithm, FA)[3].

2014 年, 澳大利亚学者 Mirjalili 等人受灰狼等级机制及其捕食行为的启发提出了一种新型的群体智能优化算法 – 灰狼优化 (grey wolf optimization, GWO) 算法 [4]. 通过大量的研究发现 GWO 算法存在后期收敛速度慢、求解精度不高以及容易出现早熟收敛等问题. 为了能够提高灰狼优化算法的寻优性能, 部分学者 [5−9] 利用算法间的优势互补, 将 GWO 算法与不同算法进行结合产生了一系列的混合优化算法, 并取到了较好的寻优性能. 鲸鱼优化算法 (whale optimization algorithm, WOA)[10] 与 GWO 算法相似也存在包围狩猎行为, 并且它的螺旋泡网狩猎行为具有较好的局部寻优能力, 它利用随机概率因子选择不同狩猎行为的机制能够有效避免早熟收敛现象的发生. 此外, 鲸鱼优化算法只根据最佳个体来进行位置更新缺乏种群多样性且全局搜索能力不高. 基于上述考虑, 本文提出了一种混合灰狼优化 (HWGWO) 算法, 将 Levy 飞行 [11] 和鲸鱼算法的螺旋泡网狩猎行为融入 GWO 算法, 并利用随机概率因子选择不同的狩猎行为. 为了加快混合灰狼优化算法的收敛速度, 改变 GWO 算法的位置更新公式, 并将惯性权重引入其中, 同时利用贪婪选择策略 [12] 选取较好的灰狼位置, 以进一步提高混合灰狼优化算法的求解精度. 数值试验表明, 混合灰狼优化 (HWGWO) 算法在收敛速度与求解精度上普遍好于对比算法.

收稿日期: 2021-07-02
资助项目: 国家重点研究计划项目 (2016YFC0700601); 贵州省科学技术基金项目 ([2020]1Y012); 贵州省教育厅创新群体项目 (KY[2021]015); 中央支持地方科研创新团队项目 (PXM2013_014210_000173); 北京建筑大学市属高校科研业务费专项资金项目 (X18193); 北京建筑大学研究生创新项目 (PG2021100)

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

1 基本灰狼优化算法

对无约束连续优化问题：

$$\min f(x), x \in R^n \tag{1}$$

若 $x^* \in R^n$ 满足：

$$f(x^*) \le f(x), \forall x \in R^n \tag{2}$$

则称 x^* 为 $f(x)$ 在全空间 R^n 上的全局极小点. 本文选取目标函数值为灰狼个体的适应度值.

GWO 算法是根据灰狼群体的等级机制以及狩猎行为而产生的一种群体智能优化算法.

灰狼个体狩猎行为定义如下：

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \left| \vec{C} \cdot \vec{X}_p(t) - \vec{X}(t) \right| \tag{3}$$

这里 t 是当前迭代次数, $\vec{X}_p(t)$ 和 $\vec{X}(t)$ 分别是猎物和灰狼的当前位置向量, $\vec{A}$ 和 $\vec{C}$ 是系数向量, $\vec{A} \cdot \left| \vec{C} \cdot \vec{X}_p(t) - \vec{X}(t) \right|$ 为包围步长. 其中 $\vec{A}$ 和 $\vec{C}$ 定义如下：

$$\vec{A} = 2\vec{r}_1 \cdot \vec{a} - \vec{a} \tag{4}$$

$$\vec{C} = 2\vec{r}_2 \tag{5}$$

这里 $\vec{r}_1$ 和 $\vec{r}_2$ 是分量在 [0,1] 中取值的 n 维随机向量, $\vec{a}$=$a(1,1,\cdots,1)^T$, a 是随着迭代次数从 2 线性递减到的收敛因子.

GWO 算法按照适应度值由低到高排列, 将适应度值最小的前三只灰狼分别定义为前三等级灰狼记为 α, β 和 δ, 其余灰狼为第四等级灰狼记为 ω. α, β 和 δ 能够识别猎物的位置, 并能够指导 ω 更新各自的位置来完成狩猎, 那么 ω 的位置更新公式用以下数学表达式表示：

$$\vec{X}_1(t) = \vec{X}_\alpha(t) - \vec{A}_1 \cdot \left| \vec{C}_1 \cdot \vec{X}_\alpha(t) - \vec{X}(t) \right| \tag{6}$$

$$\vec{X}_2(t) = \vec{X}_\beta(t) - \vec{A}_2 \cdot \left| \vec{C}_2 \cdot \vec{X}_\beta(t) - \vec{X}(t) \right| \tag{7}$$

$$\vec{X}_3(t) = \vec{X}_\delta(t) - \vec{A}_3 \cdot \left| \vec{C}_3 \cdot \vec{X}_\delta(t) - \vec{X}(t) \right| \tag{8}$$

$$\vec{X}(t+1) = (\vec{X}_1 + \vec{X}_2 + \vec{X}_3)/3 \tag{9}$$

2 混合灰狼优化算法

针对 GWO 算法存在的问题并对其进行改进, 从而得到一种融入鲸鱼算法的混合灰狼优化 (HWGWO) 算法.

2.1 融入螺旋泡网狩猎机制

为了避免 GWO 算法陷入局部最优, 本文不仅融入了 WOA 算法的螺旋泡网狩猎方式, 同时也采用 WOA 算法中的随机概率因子来选取不同的狩猎行为. 此外, WOA 算法也易陷入局部最优且全局搜索能力不高, 因此需要通过 $Levy$ 飞行算法的随机游走性能来提高算法的全局搜索能力以及丰富算法的种群多样性. 具体操作如下：首先, 按照 Mantegna[13] 提出的随机步长公式生成随机向量 $\overrightarrow{Levy}$, 公式如下：

$$\overrightarrow{Levy} = (s_1, s_2, \cdots, s_n), s_i = \frac{\mu}{|\nu|^{1/\lambda}}, i = 1, 2, \cdots, n \tag{10}$$

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

其中参数 λ 取 1.5, μ 和 ν 服从方差为 $\sigma_\mu = [\frac{\Gamma(1+\lambda)*\sin(\pi*\lambda/2)}{\Gamma[(1+\lambda)/2]*\lambda*2^{(\lambda-1)/2}}]^{1/\lambda}$ 和 $\sigma_\nu = 1$ 的正态分布. 接着, 引入随机概率因子 $p \in [0,1]$, 当 $p \geq 0.5$ 时, 灰狼群体根据最优个体的位置 $\vec{X}_\alpha$ 采用混合 $Levy$ 飞行算法的螺旋泡网狩猎行为进行位置更新, 即

$$\vec{X}'(t) = \vec{X}_\alpha(t) + \overrightarrow{Levy} \cdot \left|\vec{X}_\alpha - \vec{X}(t)\right| \cdot e^{bl} \cdot \cos(2\pi l) \tag{11}$$

其中 b 为螺旋方程的常量系数, 通常取 b= 1, l 为 [-1,1] 间的一个随机数; 当 $p < 0.5$ 时, 灰狼个体按照公式 (6)-(9) 进行包围狩猎.

2.2 改变包围狩猎机制的位置更新公式

为了提高灰狼优化算法的收敛速度, 受文献 [14] 以及惯性权重 [15] 的启发, 本文设计一种新的位置更新公式来代替公式 (9), 具体的数学表达式如下:

$$\vec{W}_1 = \left|\vec{X}_1\right| \Big/ \left(\left|\vec{X}_1\right| + \left|\vec{X}_2\right| + \left|\vec{X}_3\right|\right) \tag{12}$$

$$\vec{W}_2 = \left|\vec{X}_2\right| \Big/ \left(\left|\vec{X}_1\right| + \left|\vec{X}_2\right| + \left|\vec{X}_3\right|\right) \tag{13}$$

$$\vec{W}_3 = \left|\vec{X}_3\right| \Big/ \left(\left|\vec{X}_1\right| + \left|\vec{X}_2\right| + \left|\vec{X}_3\right|\right) \tag{14}$$

$$\vec{X}'(t) = \vec{W}_1 \cdot (\vec{X}_1 - \vec{X}_\alpha) + \vec{W}_2 \cdot (\vec{X}_2 - \vec{X}_\beta) + \vec{W}_3 \cdot (\vec{X}_3 - \vec{X}_\delta) \tag{15}$$

并且这里

$$\vec{C} = 2\vec{r}_1 \tag{16}$$

其中 $\vec{r}_1$ 取值与公式 (4) 中的 $\vec{r}_1$ 相同.

2.3 贪婪选择策略

为了提高 GWO 算法的求解精度, 比较 $\vec{X}(t)$ 和 $\vec{X}'(t)$ 的适应度值, 保留较小适应度值所对应的位置, 作为本次更新后的位置 $\vec{X}(t+1)$. 具体操作如下: 若 $f(\vec{X}'(t)) < f(\vec{X}(t))$, 则 $\vec{X}(t+1) = \vec{X}'(t)$, 否则, $\vec{X}(t+1) = \vec{X}(t)$. 这里, $f\left(\vec{X}\right)$ 为灰狼的适应度值.

混合灰狼优化 (HWGWO) 算法步骤如下:

步骤 1 设置算法参数: 种群规模 N, 最大迭代次数 Max_iter. 初始化灰狼种群的位置 $\vec{X}_i(0)(i = 1, 2, \cdots, N)$ 令 $t = 0$.

步骤 2 计算灰狼种群 $\vec{X}_i(t)$ 适应度值大小, 选择适应度值最小的三只灰狼 $\vec{X}_\alpha, \vec{X}_\beta, \vec{X}_\delta$.

步骤 3 随机生成概率因子 p, 随机数 l, 设置 $b = 1$, 并生成参数 $\vec{A}$ 和 $\vec{C}$.

步骤 4 按照随机概率因子的大小选择不同的位置更新公式得到 $\vec{X}'_i(t)$.

步骤 5 计算 $\vec{X}'_i(t)$ 的适应度值并与 $\vec{X}_i(t)$ 的适应度值对比, 保留较小适应度值所对应的位置将其记为 $\vec{X}_i(t+1)$; 判断是否满足最大迭代次数或者是否达到指定精度, 若是则输出全局最优解 $\vec{X}^*$, 否则令 $t = t + 1$, 并转入步骤 2.

HWGWO 算法的流程图如图 1 所示.

3 实验结果与分析

为了验证 HWGWO 算法的寻优性能, 将文献 [16] 中的 23 个测试函数作为无约束优化问题的目标函数. 本文设置种群数为 50, 对每个问题独立求解 30 次.

3.1 固定最大迭代次数下的对比

在固定算法最大迭代次数为 1000 次的情况下, 将 HWGWO 算法与 GWO 算法、EGWO

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

算法 [17]、SOGWO 算法 [18]、BGWO 算法 [19] 对 23 个目标函数分别独立进行 30 次寻优试验所得到的目标函数值的最好值、平均值以及标准差统计在表 1 中, 其余四种算法的相关数据都来源于文献 [19].

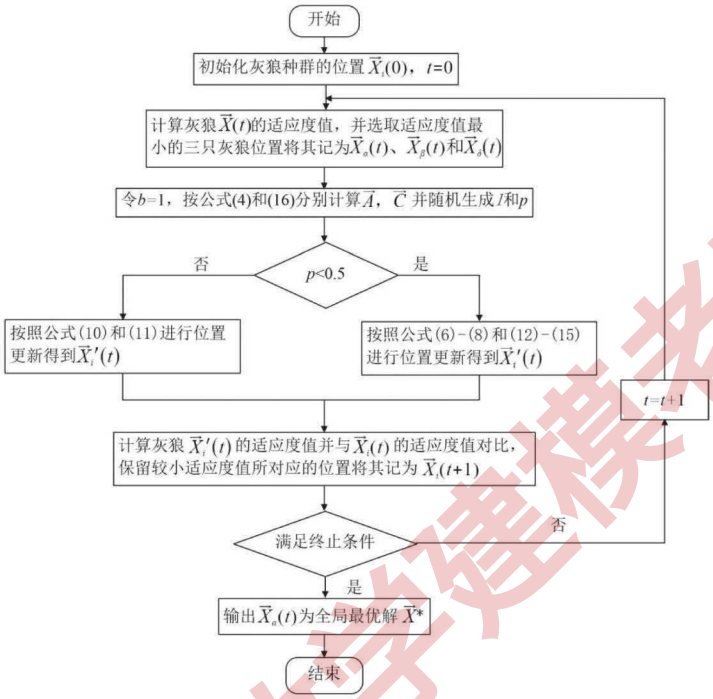

图 1 HWGWO 算法流程图

从表 1 可以看出, 对于 18 个目标函数 $f_1 \sim f_{13}, f_{16} \sim f_{20}$, HWGWO 算法所得目标函数值比其它四种算法所得的结果好; 对于目标函数 f_{14} 和 f_{15}, HWGWO 算法所得目标函数值仅次于 BGWO 算法所得的结果; 对于目标函数 $f_{21} \sim f_{23}$, HWGWO 算法所得目标函数值比其余四种算法所得的结果差. 因此, 在固定最大迭代次数为 1000 次的情况下, HWGWO 算法的寻优性能在 GWO 算法上得到了较大的提高且与其它改进算法相比具有一定的优势.

3.2 指定目标函数值精度下的对比

为了进一步比较 HWGWO 算法和 GWO 算法的寻优性能, 设置两种算法的最大迭代次数为 1000 次, 各算法分别对每个目标函数独立求解 30 次, 统计两种算法在 30 次独立求解中目标函数值达到 10^{-6} 精度所需要的最多迭代次数、最少迭代次数、平均迭代次数以及成功率, 相应的统计结果如表 2 所示.

从表 2 可以看出, 对于 15 个目标函数 $f_1 \sim f_4, f_6, f_7, f_9 \sim f_{13}, f_{16}, f_{18}, f_{21}, f_{22}$, HWGWO 算法所需要的各类迭代次数比 GWO 算法少且成功率普遍提高; 对于其余 8 个目标函数, HWGWO 算法和 GWO 算法在 1000 次迭代中均没有到达指定的目标函数值精度. 因此, 在 1000 次迭代中, HWGWO 算法到达指定目标函数值精度的成功率普遍比 GWO 算法高, 并且所需要的迭代次数普遍比 GWO 算法少, 说明了 HWGWO 算法达到同一指定目标函数值精度所需的计算量普遍比 GWO 算法小.

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

表 1 固定最大迭代次数下算法所得目标函数值的数据试验结果

目标函数	算法	最好值	平均值	标准差	目标函数	算法	最好值	平均值	标准差
f_1	GWO	5.6503e-73	3.6380e-70	7.4449e-70	f_{13}	GWO	1.6371e-05	3.2789e-01	1.7235e-01
	EGWO	9.7148e-75	5.6482e-72	8.4739e-72		EGWO	1.5260e-05	2.5167e-01	1.6174e-01
	SOGWO	1.2857e-78	6.9686e-72	3.1590e-72		SOGWO	2.5493e-05	3.1958e-01	1.9517e-01
	BGWO	6.4384e-106	8.1282e-102	1.6995e-101		BGWO	8.4454e-04	3.4227e-01	1.7744e-01
	HWGWO	0	0	0		HWGWO	3.7114e-11	1.9800e-02	2.9300e-02
f_2	GWO	7.1480e-42	4.6790e-41	3.5967e-41	f_{14}	GWO	9.9800e-01	2.8291e+00	3.2818e+00
	EGWO	1.1501e-42	7.3998e-42	9.2394e-42		EGWO	9.980e-01	4.0651e+00	4.1790e+00
	SOGWO	6.5340e-42	6.5893e-41	7.1263e-41		SOGWO	9.980e-01	2.4401e+00	2.7071e+00
	BGWO	1.8773e-59	5.3200e-58	7.8500e-58		BGWO	9.980e-01	9.9800e-01	7.4590e-10
	HWGWO	0	0	0		HWGWO	9.980e-01	1.9218e+00	1.4164e+00
f_3	GWO	1.6402e-26	1.2726e-18	4.0690e-18	f_{15}	GWO	3.0749e-04	5.7168e-03	8.9861e-03
	EGWO	2.0604e-25	5.8968e-21	1.3634e-20		EGWO	3.0749e-04	4.3492e-03	8.1457e-03
	SOGWO	1.8847e-23	2.6758e-19	8.9610e-19		SOGWO	3.0749e-04	3.6594e-03	7.5980e-03
	BGWO	3.0107e-32	5.8140e-27	1.6716e-26		BGWO	3.0749e-04	3.1282e-04	2.2643e-05
	HWGWO	0	0	0		HWGWO	3.0749e-04	3.8141e-04	1.2236e-04
f_4	GWO	1.8553e-18	2.4693e-17	3.8076e-17	f_{16}	GWO	-1.0316	-1.0316	3.9579e-09
	EGWO	2.9711e-19	7.7112e-18	1.0116e-17		EGWO	-1.0316	-1.0316	2.9097e-09
	SOGWO	3.7856e-19	1.8102e-17	3.1788e-17		SOGWO	-1.0316	-1.0316	4.4186e-09
	BGWO	7.8981e-31	3.1842e-28	8.1645e-28		BGWO	-1.0316	-1.0316	6.5843e-16
	HWGWO	0	0	0		HWGWO	-1.0316	-1.0316	5.8044e-16
f_5	GWO	2.5101e+01	2.6384e+01	7.1383e-01	f_{17}	GWO	0.39789	0.39789	1.0820e-07
	EGWO	2.5128e+01	2.6452e+01	8.9877e-01		EGWO	0.39789	0.39789	1.5072e-07
	SOGWO	2.5103e+01	2.6303e+01	7.5245e-01		SOGWO	0.39789	0.39789	9.6657e-08
	BGWO	2.5277e+01	2.6257e+01	5.6234e-01		BGWO	0.39789	0.39789	1.5645e-08
	HWGWO	2.3625e+01	2.4333e+01	4.2920e-01		HWGWO	0.39789	0.39789	0
f_6	GWO	7.8189e-06	4.1782e-01	3.2364e-01	f_{18}	GWO	3	3	1.9350e-06
	EGWO	7.2512e-06	3.3356e-01	2.6612e-01		EGWO	3	3	2.9064e-06
	SOGWO	8.0385e-06	3.1870e-01	2.8472e-01		SOGWO	3	3	5.6784e-06
	BGWO	2.5586e-04	4.3764e-04	1.0629e-04		BGWO	3	3	1.7627e-06
	HWGWO	4.3129e-12	3.1782e-06	1.1622e-05		HWGWO	3	3	2.0955e-15
f_7	GWO	1.0023e-04	4.5029e-04	2.9404e-04	f_{19}	GWO	-3.8628	-3.8620	2.3822e-03
	EGWO	1.2518e-04	5.4262e-04	3.8437e-04		EGWO	-3.8628	-3.8618	2.5328e-03
	SOGWO	1.7050e-04	5.3704e-04	2.9614e-04		SOGWO	-3.8628	-3.8618	2.3543e-03
	BGWO	3.3507e-05	3.4264e-04	2.3371e-04		BGWO	-3.8628	-3.8623	1.5919e-03
	HWGWO	5.6877e-07	4.3498e-05	3.4777e-05		HWGWO	-3.8628	-3.8628	2.5844e-15
f_8	GWO	-7.6392e+03	-6.3347e+03	6.9474e+02	f_{20}	GWO	-3.3220	-3.2458	9.5716e-02
	EGWO	-7.7825e+03	-6.3262e+03	6.8624e+02		EGWO	-3.3220	-3.2796	6.9474e-02
	SOGWO	-8.2349e+03	-6.6120e+03	1.0989e+03		SOGWO	-3.3220	-3.2546	7.8799e-02
	BGWO	-7.4253e+03	-5.5666e+03	1.1819e+03		BGWO	-3.3220	-3.2897	5.4300e-02
	HWGWO	-1.2569e+04	-1.2467e+04	2.5256e+02		HWGWO	-3.3220	-3.2941	5.0600e-02
f_9	GWO	0	3.8475e-01	1.5029e+00	f_{21}	GWO	-1.0153e+01	-9.4762e+00	1.7547e+00
	EGWO	0	1.6023e-01	8.7764e-01		EGWO	-1.0153e+01	-8.8041e+00	2.2750e+00
	SOGWO	0	4.2972e-01	1.6976e+00		SOGWO	-1.0153e+01	-9.8146e+00	1.2876e+00
	BGWO	0	0	0		BGWO	-1.0153e+01	-9.9218e+00	1.2854e+00
	HWGWO	0	0	0		HWGWO	-1.0153e+01	-7.2643e+00	2.5262e+00
f_{10}	GWO	7.9936e-15	1.4152e-14	2.4567e-15	f_{22}	GWO	-1.0403e+01	-1.0048e+01	1.3484e+00
	EGWO	7.9936e-15	1.2849e-14	2.7174e-15		EGWO	-1.0403e+01	-1.0403e+01	7.4234e-04
	SOGWO	7.9936e-15	1.3086e-14	2.5861e-15		SOGWO	-1.0403e+01	-1.0403e+01	2.5033e-04
	BGWO	4.4409e-15	7.6383e-15	1.0840e-15		BGWO	-1.0403e+01	-1.0403e+01	2.0949e-04
	HWGWO	8.8818e-16	8.8818e-16	0		HWGWO	-1.0403e+01	-7.0135e+00	2.8308e+00
f_{11}	GWO	0	1.2261e-03	3.9761e-03	f_{23}	GWO	1.0536e+01	-1.0536e+01	1.4470e-04
	EGWO	0	1.4503e-03	5.7501e-03		EGWO	-1.0536e+01	-1.0085e+01	1.7518e+00
	SOGWO	0	2.1280e-03	5.3652e-03		SOGWO	-1.0536e+01	-1.0179e+01	1.3600e+00
	BGWO	0	0	0		BGWO	-1.0533e+01	-1.0523e+01	6.2558e-03
	HWGWO	0	0	0		HWGWO	-1.0536e+01	-8.1106e+00	3.2631e+00
f_{12}	GWO	6.4349e-03	2.6870e-02	1.5172e-02					
	EGWO	1.5087e-06	2.3286e-02	1.5524e-02					
	SOGWO	1.4471e-06	2.3969e-02	1.0955e-02					
	BGWO	3.5517e-05	1.8374e-03	2.9960e-03					
	HWGWO	5.6744e-13	7.1878e-07	1.9964e-06					

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

表 2 指定目标函数值精度下算法所需迭代次数的试验结果

目标函数	算法	最多迭代次数	最少迭代次数	平均迭代次数	成功率
f_1	GWO ($HWGWO$)	94 (42)	85 (30)	90.0333 (35.1000)	100% (100%)
f_2	GWO ($HWGWO$)	121 (60)	113 (41)	117.4667 (50.9667)	100% (100%)
f_3	GWO ($HWGWO$)	335 (71)	239 (42)	281.5667 (54.3667)	100% (100%)
f_4	GWO ($HWGWO$)	229 (64)	192 (47)	210.4333 (55.1333)	100% (100%)
f_5	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_6	GWO ($HWGWO$)	1000 (1000)	1000 (456)	1000 (704.0333)	0% (80%)
f_7	GWO ($HWGWO$)	1000 (1000)	1000 (509)	1000 (952.3000)	0% (13.33%)
f_8	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_9	GWO ($HWGWO$)	1000 (44)	118 (29)	203.4000 (35.9333)	93.33% (100%)
f_{10}	GWO ($HWGWO$)	132 (59)	118 (46)	125.0333 (51.4000)	100% (100%)
f_{11}	GWO ($HWGWO$)	1000 (42)	88 (29)	245.6667 (35.7333)	83.33% (100%)
f_{12}	GWO ($HWGWO$)	1000 (1000)	1000 (325)	1000 (549.6667)	0% (90%)
f_{13}	GWO ($HWGWO$)	1000 (1000)	1000 (545)	1000 (908.3000)	0% (33.33%)
f_{14}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_{15}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_{16}	GWO ($HWGWO$)	1000 (1000)	32 (9)	936.8000 (901.2000)	6.67% (10%)
f_{17}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_{18}	GWO ($HWGWO$)	1000 (29)	58 (13)	694.6667 (21.6333)	50% (100%)
f_{19}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_{20}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)
f_{21}	GWO ($HWGWO$)	1000 (1000)	1000 (43)	1000 (599.7667)	0% (43.33%)
f_{22}	GWO ($HWGWO$)	1000 (1000)	1000 (60)	1000 (968.6667)	0% (3.33%)
f_{23}	GWO ($HWGWO$)	1000 (1000)	1000 (1000)	1000 (1000)	0% (0%)

4 应用

为了验证 HWGWO 算法在实际生活中的可行性以及有效性, 将 HWGWO 算法应用到经典的拉伸/压缩弹簧设计工程问题中. 如图 2 所示, 拉伸/压缩弹簧设计是通过调节导线直径 (d)、平均线圈直径 (D) 和有效线圈数 (P) 三个变量使得拉伸/压缩弹簧在满足四个约束条件下的重量最小, 令 $\vec{X}=[x_1x_2x_3]=[dDP]$, 用数学模型可描述如下:

$$\begin{cases} \min f(\vec{X})=(x_3+2)x_2x_1^2, \\ s.t.g_1(\vec{X})=1-\frac{x_2^3x_3}{71785x_1^4}\le 0, \\ \quad g_2(\vec{X})=\frac{4x_2^2-x_1x_2}{12566(x_2x_1^3-x_1^4)}+\frac{1}{5108x_1^2}\le 0, \\ \quad g_3(\vec{X})=1-\frac{140.45x_1}{x_2^2x_3}\le 0, \\ \quad g_4(\vec{X})=\frac{x_1+x_2}{1.5}-1\le 0, \\ 0.05\le x_1\le 2, 0.25\le x_2\le 1.30, 2.0\le x_3\le 15. \end{cases}$$

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

这是一个约束优化问题, 先利用外点罚函数法将其转换为一系列无约束优化问题, 再利用 HWGWO 算法求解各无约束优化问题, 以获得拉伸/压缩弹簧设计的最优方案.

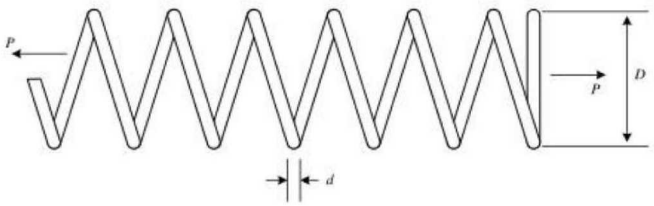

图 2 拉伸/压弹簧设计问题示意图

对于拉伸/压缩弹簧设计的优化问题, HWGWO 算法的参数设置和文献 [20] 中 ISCA 算法相同, 即设置种群规模为 5 和最大迭代次数为 100 次. 表 3 为 HWGWO 算法与其它算法所得到的拉伸/压缩弹簧重量的最小值以及它们所对应的三个变量值, 其它算法的数据来自于文献 [20].

表 3 HWGWO 算法和其余算法求解拉/压弹簧设计问题的实验结果

算法	变量			重量
	d	D	P	
WOA [10]	0.051207	0.345215	12.54854	0.0126763
GWO [4]	0.05169	0.356737	11.28885	0.012666
EEGWO [21]	0.051673	0.35634	11.3113	0.012665
SCA [22]	0.05078	0.334779	12.72269	0.0127097
ISCA [20]	0.0520217	0.364768	10.8323	0.012667
HWGWO	0.0530	0.3902	9.5542	0.01264014

从表 3 可以看出, HWGWO 算法所得到的拉伸/压缩弹簧重量的最小值比其它 5 种算法所得到的结果小, 表明了对于拉伸/压缩弹簧设计问题, 利用 HWGWO 算法所得的方案比其它 5 种算法所得的方案更有效.

5 结论

针对灰狼优化算法在实际寻优中求解精度低、收敛速度慢以及存在早熟收敛现象的问题, 提出融入鲸鱼优化算法的混合灰狼优化 (HWGWO) 算法. 灰狼群体利用随机概率因子选择进行包围狩猎行为或者选择引入 Levy 飞行算法的螺旋泡网狩猎行为, 动态的调整了群体的全局搜索和局部搜索能力; 引入惯性权重以及差分进化思想, 改变灰狼包围狩猎的位置更新公式, 以加快群体的寻优并避免群体陷入局部最优; 利用贪婪选择策略保留较好的灰狼个体, 提高了算法的求解精度. 通过对 23 个目标函数的数值试验, 将 HWGWO 算法与相关算法对比, 试验结果表明, HWGWO 算法在求解精度、计算量和收敛速度上普遍好于 GWO 算法; 与 EGWO 算法、SOGWO 算法和 BGWO 算法的寻优性能相比, HWGWO 算法具有一定的优势. HWGWO 算法在求解拉伸/压缩弹簧设计问题并与 5 种算法的对比中, HWGWO

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

算法得到的设计方案更有效.

参考文献

[1] Akbari R, Ziarati K. A rank based particle swarm optimization algorithm with dynamic adaptation[J]. Journal of Computational and Applied Mathematics, 2011, 235(8): 2694-2714.

[2] Seckiner S U, Eroglu Y, Emrullah M, et al. Ant colony optimization for continuous functions by using novel pheromone updating[J]. Applied Mathematics and Computation, 2013, 219(9): 4163-4175.

[3] Yang X S, Hosseini S S S, Gandomi AH. Firefly Algorithm for solving non-convex economic dispatch problems with valve loading effect[J]. Applied Soft Computing Journal, 2011, 12(3): 1180-1186.

[4] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69: 46-61.

[5] Long W, Cai S H, Jiao J J, Xu M, Wu T B, A new hybrid algorithm based on grey wolf optimizer and cuckoo search for parameter extraction of solar photovoltaic models[J]. Energy Conversion and Management, 2020, 203: 112243.

[6] Tawhid M A, Ibrahim A M. A hybridization of grey wolf optimizer and differential evolution for solving nonlinear systems[J]. Evolving Systems: An Interdisciplinary Journal for Advanced Science and Technology, 2020, 11(11): 65-87.

[7] Konstantinov S V, Khamidova U K, Sofronova E A. A Novel Hybrid Method of Global Optimization Based on the Grey Wolf Optimizer and the Bees Algorithm[J]. Procedia Computer Science, 2019, 150: 471-477.

[8] Long W, Jiao J J, Liang X M, Tang M Z, An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization[J]. Engineering Applications of Artificial Intelligence, 2018, 68: 63-80.

[9] Zhang X, Kang Q, Cheng J, et al. A novel Hybrid algorithm based on biogeography-based optimization and grey wolf optimizer[J]. Applied Soft Computing, 2018, 67: 197-214.

[10] Mirjalili S, Lewis A. The Whale Optimization Algorithm[J]. Advances in Engineering Software, 2016, 95: 51-67.

[11] Reible Danny, Mohanty Sanat. A Levy flight-random walk model for bioturbation[J]. Environmental toxicology and chemistry, 2002, 21(4): 875-881.

[12] Storn R, Price K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces [J]. Journal of Global Optimization, 1997, 11: 341-359.

[13] Mantegna. Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes [J]. Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics, 1994, 49(5): 4677-4683.

[14] 逯苗, 何登旭, 曲良东. 非线性参数的精英学习灰狼优化算法 [J]. 广西师范大学学报 (自然科学版), 2021, 5(24): 1-12.

[15] 王秋萍, 王梦娜, 王晓峰. 改进收敛因子和比例权重的灰狼优化算法 [J]. 计算机工程与应用, 2019, 55(21): 60-65+98.

[16] Bansal J C, Singh S. A better exploration strategy in Grey Wolf Optimizer[J]. Journal of Ambient Intelligence and Humanized Computing, 2020, 12(1): 1099-1118

[17] Komijani H, Masoumnezhad M, Zanjireh M M, Mir M. Robust Hybrid Fractional Order Proportional Derivative Sliding Mode Controller for Robot Manipulator Based on Extended Grey Wolf Optimizer[J]. Robotics, 2020, 38(4): 605-616

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

[18] Dhargupta S, Ghosh M, Mirjalili S, Sarkar R. Selective Opposition based Grey Wolf Optimization[J]. Expert Systems With Applications, 2020, 151: 113389

[19] Fan Q S, Huang H S, Li Y T, Han Z G, et al. Beetle antenna strategy based grey wolf optimization[J]. Expert Systems With Applications, 2021, 165: 113882.

[20] Long W, Wu T B, Liang X M, Xu S J. Solving high-dimensional global optimization problems using an improved sine cosine algorithm[J]. Expert Systems with Applications, 2019, 123: 108-126.

[21] Long W, Jiao J, Liang X, Tang M. An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization[J]. Engineering Applications of Artificial Intelligence, 2018, 68: 63-80.

[22] Mirjalili S. SCA: A sine cosine algorithm for solving optimization problems[J]. Knowledge-based Systems, 2016, 96, 120-133.

Hybrid Grey Wolf Optimization Algorithm Fused with Whale Algorithm

LIANG Xi-ming[1], LI Xing[1], LONG Wen[2]

(1. School of Science, Beijing University of Civil Engineering and Architecture Beijing 102616, China)
(2. Key Laboratory of Economics System Simulation, Guizhou University of Finance and Economics, Guiyang 550025, China)

Abstract: Aiming at the problems of grey wolf optimization (GWO) algorithm that are easy to fall into local optimality, slow convergence speed, and low solution accuracy, a hybrid grey wolf optimization (HWGWO) algorithm fused with Levy flight and whale algorithm is proposed. First, the spiral bubble net hunting behavior of the whale algorithm is incorporated; then the dynamic weight and differential evolution ideas are introduced into the grey wolf optimization algorithm; finally, the greedy selection strategy is used to retain a better grey wolf position. 23 test functions are selected for numerical experiments, and the results show that the HWGWO algorithm has improved the convergence speed and solution accuracy. In addition, the design scheme obtained by using the HWGWO algorithm to solve the tension/compression spring design problem is more effective.

Keywords: grey wolf optimization algorithm; levy flight algorithm; whale optimization algorithm; numerical experiment; tension/compression spring design problem

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net