Hybrid Cooperative Crayfish and Mountain Gazelle Algorithm for Global Optimization

Boon-Hao Lee
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Selangor, Malaysia
bhao2002@gmail.com

Ying-Loong Lee
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Selangor, Malaysia
leeyingl@utar.edu.my

Kok-Chin Khor
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Selangor, Malaysia
kckhor@utar.edu.my

***Abstract*—The crayfish optimization algorithm (COA) and mountain gazelle optimization algorithm (MGO) have emerged as two powerful metaheuristics for global optimization. It has been shown that these two metaheuristics outperform conventional metaheuristics and are capable of finding near-optimal solutions for most of the benchmark functions. However, in certain benchmark functions, COA and MGO do not perform better than the conventional metaheuristics. To this end, a new Hybrid Cooperative COA-MGO algorithm (HCCMGA) is proposed. In the proposed HCCMGA algorithm, the solution vector is split into two solution subvectors of smaller dimensions, one optimized by COA and the other by MGO. The fitness of each search agent in COA and MGA is evaluated in a cooperative manner, whereby the solution subvector in COA is combined with the best solution subvector from MGO and vice versa. The proposed HCCMGA is evaluated using 13 benchmark functions and is compared with several state-of-the-art metaheuristics. Results show that the proposed HCCMGA outperforms state-of-the-arts optimization algorithms in solution quality, search stability and convergence speed, as the HCCMGA achieves solutions closest to the optimum in eight out of the 13 benchmark functions with low standard deviations and less numbers of iterations.**

Keywords—Crayfish Optimization Algorithm (COA); Mountain Gazelle Optimization (MGO); Global Optimization; Hybrid Cooperation.

I. INTRODUCTION

Many metaheuristic evolutionary optimization algorithms have been designed for global optimization. Notable evolutionary algorithms include Particle Swarm Optimization (PSO) [1], Greywolf Optimization Algorithm (GWO) [2], Ant Colony Optimization (ACO) [3], Ant Lion Optimization (ALO) [4], and Genetic Algorithm (GA) [5]. Some new metaheuristic evolutionary algorithm include Artificial Protozoa Optimizer (APO) [6], Electric Eel Foraging Optimization (EEFO) [7], Greylag Goose Optimization (GGO) [8], Hippopotamus Optimization Algorithm (HO) [9], Whale Optimization Algorithm (WOA) [10] and Sine Cosine Algorithm (SCA) [11]. The main goal of all these metaheuristics is to gain the global optimum solution for any given unconstrained optimization problem. In order to achieve this goal, every metaheuristic has been equipped with two main characteristics, which are the exploration and exploitation.

Exploration is the capability to find the best solution from an optimization problem within the entire search space specified for the problem, while exploitation is the ability to find a better outcome near a good solution [12]. A key design aspect for all metaheuristics is balancing the exploration and exploitation abilities to discover the best global solution effectively for any optimization issues. According to [13], the exploration and exploitation abilities in evolutionary computing could contradict each other, as strengthening the exploration ability may weaken the exploitation ability and vice versa. Therefore, striking an equilibrium between exploration and exploitation is crucial in the design of metaheuristics.

According to the no free lunch theorem [14], the mean performance of a metaheuristic is almost identical to those of other metaheuristics because an algorithm that performs well in solving a set of problems may perform poorly in solving another set of problems. This could be due to varying exploration and exploitation abilities in the design of the metaheuristics. Most of the traditional nature-inspired metaheuristics are developed based on a specific type of natural processes or phenomena, which are used to explore and exploit solutions. Many such metaheuristics are effective and efficient in solving complex optimization problems. However, these metaheuristics, which are often designed solely based on a single specific natural process or phenomenon, are only advantageous for solving certain classes of problems. In view of this, we could combine the inherent nature-inspired mechanisms in these metaheuristics to strengthen both the exploration and exploitation of solutions.

Recently, the crayfish optimization algorithm (COA) [15] and mountain gazelle optimization (MGO) [16] have emerged as powerful metaheuristics for global optimization. Several attempts hybridize these algorithms with other techniques. For example, COA has been hybridized with Self-adaptive Differential Evolution (SaDE) to solve complex optimization problems [17]. Next, chaotic-based MGO (CMGO) has also been proposed as a combination of chaotic maps and MGO to solve optimization problems [18]. However, these algorithms could not perform well in solving certain problems. In this paper, we aim to increase the exploration and exploitation abilities of these metaheuristics by proposing a hybrid cooperative COA-MGO algorithm (HCCMGA). The contributions of our study can be summarized as below:

1) We propose splitting the solution vector into two solution subvectors of smaller dimensions, one optimized by COA and the other by MGO.
2) We adopt a cooperative fitness evaluation approach, whereby each candidate solution subvector obtained by COA will be recombined with the best solution subvector obtained by MGO for COA to determine the best solution subvector. A similar mechanism is applied to find the best solution subvector for MGO.

This work was supported in part by the Universiti Tunku Abdul Rahman Research Fund (UTARRF) under Grant IPSR/RMC/UTARRF/2024-C1/L05.

3) We evaluate the proposed HCCMGA based on stability, solution quality, and convergence speed. We show that HCCMGA outperforms existing metaheuristics in various benchmark test functions.

The rest of the paper is organized as follows. In Section 2, a brief introduction of COA and MGO is given. In Section 3, the proposed HCCMGA is described in detail, while in Section 4, the experimental study and results are presented and proposed. Lastly, Section 5 concludes final remarks and future directions.

II. Standard COA and Standard MGO

A. Standard Crayfish Optimization Algorithm

COA is a metaheuristic developed by Jia et al. in 2023 [15]. It is a bio-inspired swarm optimization algorithm inspired by the crayfish summer resort, competition, and food-hunting habits. The exploration stage for COA is the summer resort phase, while the foraging and competition stages belong to the exploitation stage of COA. Conventional algorithms such as the genetic algorithm (GA), simulated annealing (SA) algorithm, and grey wolf optimization algorithm (GWO) have been used to compare with the algorithm. The result shows that COA has better optimization effects.

The process of COA starts with the initialization of the population. Eq. (1) shows the initialization of the COA algorithm:

$$X = [X_1, X_2, \dots, X_N] = \begin{bmatrix} X_{1,1} & \dots & X_{1,j} & \dots & X_{1,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ X_{i,1} & \cdots & X_{i,j} & \cdots & X_{i,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ X_{N,1} & \cdots & X_{N,j} & \cdots & X_{N,dim} \end{bmatrix}, \quad (1)$$

where X is the initial position of the population, N is the population size, dim is the number of dimensions/variables in each position, and $X_{i,j}$ is the position of the i-th crayfish in the j-th dimension. Each crayfish is set as a size $1 \times dim$ matrix in the multidimensional optimization problem. Each row in the matrix is a candidate solution to the problem. Meanwhile, for a group of variables $(X_{i,1}, X_{i,2}, \dots, X_{i,dim})$, every variable X_i cannot be positioned out beyond lower and upper boundaries. For initialization, $X_{i,j}$ can be obtained as

$$X_{i,j} = l_j + (u_j - l_j) \times r, \quad (2)$$

where u_j is the superior limit of the jth dimension, l_j is the bottom limit of the j-th dimension, and r is a random value within 0 to 1.

After the candidate solution, X, has been initialized, a temperature needs to be calculated because it can affect the behavior of the crayfish. Eq. (3) will be used to calculate the temperature:

$$t = r \times 15 + 20. \quad (3)$$

Note that the feeding area of a crayfish is between 15°C to 30°C. Besides that, the nourishing portion of crayfish will be influenced by temperature. The nourishing intake of crayfish can be calculated with

$$p = C_1 \times \left(\frac{1}{\sqrt{2\pi}\,\sigma} \, exp\left(-\frac{(t-\mu)^2}{2\sigma^2} \right) \right), \quad (4)$$

where t is the environment's temperature of the located crayfish, while μ is the most appropriate temperature for crayfish, σ and C_1 are the variables that limit the crayfish's intake at different temperatures.

If t is higher than 30°C, the crayfish will enter the summer resort phase, the COA's exploration stage. At this stage, the crayfish will find a tunnel for summer vacation. The tunnel X_s is identified as

$$X_s = (X_G + X_L) / 2, \quad (5)$$

where X_G is the optimal placement acquired so far at the present repetition and X_L is the optimal placement of the present populace.

Before entering the cave, the crayfish will fight for caves randomly. If r is lower than 0.5, the crayfish will straightaway move into the cave for summer vacation since no other crayfish competes for the caves. The following equation will be applied to define the location of the crayfish when it enters the cave.

$$X_{i,j}^{T+1} = X_{i,j}^{T} + C_2 r \left(X_s - X_{i,j}^{T} \right), \quad (6)$$

where T is the present repetition index, and C_2 is the decreasing curve, calculated as follows:

$$C_2 = 2 - (T/T_{max}), \quad (7)$$

where T_{max} is the maximum number of iterations. Crayfish aim to go towards the cave, which is the best solution, and it will enhance the exploitation capability of COA, enabling COA to converge quicker.

When t is higher than 30, and r is higher or equal to 0.5, another crayfish is also intended to enter the cave. They then compete for the cave through the following equation:

$$X_{i,j}^{T+1} = X_{i,j}^{T} - X_{z,j}^{T} + X_s, \quad (8)$$

where z is the random crayfish's individual, computed by

$$z = \text{round}\big(r(N-1)\big) + 1. \quad (9)$$

In (9), the round function rounds the floating-point value to the nearest integer. In this phase, the crayfish will fight with each other to enter a cave and the crayfish X_i will adjust their location depending on the location X_z of another crayfish. Thus, by modifying their position, the search scope of COA is enlarged, and the exploration capability of the COA algorithm is expanded.

When t is less than or equal to 30°C, it is appropriate to feed the crayfish. So, the crayfish will approach the food. The crayfish will compute the food size after discovering it. If the food is too huge, the crayfish will shred it using its claws and eat it using its second and third walking feet. The food position X_f is computed with

$$X_f = X_G, \quad (10)$$

Meanwhile, the food size Q is explained in the following equation:

$$Q = C_3 r\big(f_i / f_f\big), \quad (11)$$

where C_3 is the food factor for the biggest food with a fixed value of 3, f_i is the fitness value of the ith crayfish, and f_f is the fitness value of the food position.

The judgment of the food size depends on the biggest food size. If Q is larger than $(C_3 + 1)/2$, it implies that the food

size is too big. Thus, the crayfish will shred the food by using its first claw foot. The position of the food is defined as

$$X_f = \exp\left(-\frac{1}{Q}\right) \times X_f. \quad (12)$$

After the food is torn and becomes tinier, the second and third feet of the crayfish will grab the food and place it into the mouth alternately. The equation below, which includes the sine and cosine methods, will be used to replicate the alternating process:

$$X_{i,j}^{T+1} = X_{i,j}^{T} + X_f p(cos(2\pi r) - \sin(2\pi r)). \quad (13)$$

When $Q \leq (C_3 + 1)/2$, it implies that the food size is small and the crayfish will just approach the food and eat it straight away with the following mathematical function:

$$X_{i,j}^{T+1} = \left(X_{i,j}^{T} - X_f\right)p + prX_{i,j}^{T}, \quad (14)$$

In short, the crayfish will use various feeding approaches depending on the food size Q, and the position of the food X_{food} is the optimal solution. If the food size Q is suitable for the crayfish to eat, it will straightaway go towards the food. However, if the food size Q is huge, then there is still a notable gap between the best fitness value and the crayfish. Thus, X_f should be decreased and it should become nearer to the food. Besides that, the uncertainty of the food intake will be controlled to enhance the COA algorithm. Therefore, throughout the foraging stage, COA will aim to move towards the optimum solution and enhance the exploitation capability of the algorithm.

B. Standard Mountain Gazelle Optimization

MGO is a novel metaheuristics algorithm by Abdollahzadeh et al. in 2022 [16]. This algorithm is motivated by gazelles' hierarchical and social life. This algorithm uses the four key causes in the social living of mountain gazelles to perform optimization operations. These four factors include migration to search for food, bachelor male herds, maternity herds, and solitary and territorial males.

In this algorithm, every gazelle (X_i) will become a member of either bachelor male herds, maternity herds, or territorial solitary males. A brand-new gazelle will be originated from either one of these three herds. The mature male gazelle in the herd area is the best solution for the MGO algorithm. Besides that, around one-third of the searched populace in the whole populace is expected to have the lowest price since in the male bachelor group, the gazelles are too youthful and not yet mature enough to oversee a territory.

Additionally, maternity gazelle herds are the alternative solutions available to the populace. The forceful gazelles that have superior solutions will be retained during each iteration. Meanwhile, other gazelles added to the overall populace that have lower prices are viewed as elderly and not healthy gazelles. These gazelles will be eliminated from the overall populace.

When the male gazelle becomes mature and powerful enough, they will make a solid territory. The distance between each territory is great enough. The young male gazelles will aim to gain an uncontrolled land, while the adult male gazelles will try to protect their environment. The following equation will be used to create the adult male region:

$$A = m_g - \left|\left(i_1 B - i_2 X(t)\right) \times F\right| \times Cof_r, \quad (15)$$

where m_g is the location vector of the optimal universal solution for the adult males, i_1 and i_2 are the random integers between 1 to 2, B is the coefficient vector of the young males, calculated using (16), F is computed using (17), and Cof_r is the randomly chosen coefficient vector revised in each iteration computed using (18):

$$B = X_a \times \lfloor r_1 \rfloor + M_p \times \lceil r_2 \rceil, a = \left\{\left\lceil\frac{N}{3}\right\rceil \cdots N\right\}, \quad (16)$$

$$F = N_1(D) \times \exp\left(2 - T \times \left(\frac{2}{T_{max}}\right)\right), \quad (17)$$

$$Cof_i = \begin{cases} (a+1) + r_{3,} \\ a \times N_2(D), \\ r_4(D), \\ N_3(D) \times N_4(D)^2 \times \cos\left((2r_4) \times N_3(D)\right), \end{cases} \quad (18)$$

where X_a is the random result for young gazelle in the interval of a, M_p is the mean number of search agents $\left\lceil\frac{N}{3}\right\rceil$ that were selected randomly, N is the overall number of gazelles, r_1 and r_2 are the random numbers within 0 to 1, N_1 is the random value from the normal distribution, exp() is the exponential function, T_{max} is the maximum number of iterations, T is the present repetition index, a is computed using (19), r_3, r_4 and r are the random between 0 and 1, N_2, N_3 and N_4 are the random values in the normal space and the problem dimensions.

$$a = -1 + T \times \left(\frac{-1}{T_{max}}\right), \quad (19)$$

where T_{max} is the maximum number of repetitions, and T is the repetition index.

Maternity herds play a significant part in the wheel of life of mountain gazelles since they can produce solid males. This action can be formulated using

$$M = \left(B + Cof_{2,r}\right) + \left(i_3 \times m_g - ri_4 \times X_r\right) \times Cof_{3,r}, \quad (20)$$

where B is the impact factor vector for young males, computed using (16), $Cof_{2,r}$ is the randomly chosen coefficient vectors that are computed using (18), i_3 and i_4 are the random numbers between 1 and 2, m_g is the best adult male universal result in the present iteration, and X_r is the gazelle's vector position selected randomly based on the population.

When the male gazelles become adults, they will be more likely to make a new colony and grab control over female gazelles. Meanwhile, the youthful males will fight with others over the control and domain of the female gazelles. The following equation is used to express these gazelles' actions mathematically.

$$H = (X(T) - D) + \left(i_5 m_g - i_6 B\right) \times Cof_r, \quad (21)$$

where $X(T)$ is the gazelle's location vector in the present repetition, i_5 and i_6 are the random integers between 1 and 2, m_g is the male gazelle's location vector, which indicates the optimal result. D is computed using

$$D = \left(|X(T)| + \left|m_g\right|\right) \times (2r_6 - 1), \quad (22)$$

where $X(T)$ is the gazelle's location vector in the present repetition, m_g is the male gazelle's location vector, which indicates the optimal solution, and r_6 is the random integer within 0 to 1.

Gazelles always find food supplies and will go a long way to get and migrate. The following equation formulates the gazelles' behavior mathematically:

$$S = (u - l) \times r_7 + l, \quad (23)$$

where u is the upper limits, l is the lower limits, r_7 is the random value between 0 and 1.

These four techniques are implemented in all gazelles to generate a new era of gazelles. A new generation will be included in the whole populace, and every era is equivalent to a new reproduction. Furthermore, each gazelle will be sorted according to the ascending order during each era. The best gazelles with positive solutions and excellent quality will be kept in the overall population. On the other hand, weak or old gazelles will be eliminated from the overall populace. A mature male is the best gazelle that possesses the area.

III. Hybrid Cooperative COA-MGO Algorithm

In the proposed HCCMGA, the *dim*-dimensional solution space is divided into two lower-dimensional solution subspaces, one optimized by COA and the other by MGO. Here, we introduce a splitting factor $S = m{:}n$, where m and n are the dimensions of the solution subspace of COA and that of MGO, respectively. To understand the splitting factor, we use the following example: Assuming a 10-dimensional solution space, we can split it into an 8-dimensional solution subspace and a 2-dimensional solution subspace. If the 8-dimensional solution subspace is to be optimized by COA whereas the other by MGO, this indicates $S = 8{:}2$ where $m = 8$ and $n = 2$, as shown in Fig. 1. Based on this subspace setting, the solution vector of each crayfish in COA consists of 8 variables whereas that of each mountain gazelle in MGO consists of 2 variables. The splitting factor provides flexibility in determining the degree of dominance between COA and MGO in the optimization process. With the example of $S = 8{:}2$, COA contributes more to the optimization process than MGO. In other words, the optimization process is dominantly affected by the optimization mechanism of COA. To exert equal dominance between COA and MGO, $m = n$ can be set for the splitting factor.

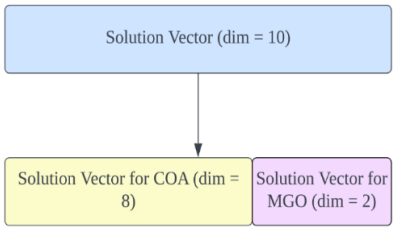

Fig. 1. Example of splitting the solution vector into two different solution subvectors.

After splitting the solution space between COA and MGO, the candidate solutions obtained by COA and MGO can be assessed through a cooperative fitness evaluation approach similar to [19]. We first denote P_M and P_C as the MGO and COA populations, respectively. Next, we refer $P_M.\hat{\mathbf{y}}$ and $P_C.\hat{\mathbf{y}}$ to as the best candidate solutions found by MGO and COA, respectively. Next, we define the following functions:

$$Q_{COA}(P_C.\mathbf{X}_i) = (P_M.\hat{\mathbf{y}}, P_C.\mathbf{X}_i), \quad (24)$$

$$Q_{MGO}(P_M.\mathbf{X}_i) = (P_M.\mathbf{X}_i, P_C.\hat{\mathbf{y}}), \quad (25)$$

where $P_C.\mathbf{X}_i$ is the candidate solution of the i-th crayfish and $P_M.\mathbf{X}_i$ is the candidate solution of the i-th mountain gazelle. In (24), $Q_{COA}(P_C.\mathbf{X}_i)$ returns the *dim*-dimensional solution vector where the candidate solution of the i-th crayfish is merged with the best solution found by MGO. For example, when $dim = 10$ and $S = 8{:}2$, $Q_{COA}(P_C.\mathbf{X}_i)$ returns a 10-dimensional solution vector consisting of the 8-dimensional $P_C.\mathbf{X}_i$ and 2-dimensional $P_M.\hat{\mathbf{y}}$. Similarly, $Q_{MGO}(P_M.\mathbf{X}_i)$ in (25) returns the *dim*-dimensional solution vector where the candidate solution of the i-th mountain gazelle is combined with the best solution found by COA.

Using (24) and (25), the fitness of each search agent in COA and MGO populations can be evaluated. This exhibits a form of cooperation between COA and MGO, whereby COA determines the best solution in its solution subspace, given the best solution information from MGO; whereas MGO identifies its best solution in its subspace, given the best solution information obtained by COA. Such cooperative information exchanges can significantly improve the search diversity of both algorithms while retaining their respective optimization mechanisms.

Algorithm 1 summarizes the pseudocode of the proposed HCCMGA. Firstly, a population of N search agents is created, with $N_1 = \left\lceil \frac{mN}{m+n} \right\rceil$ agents serving as the MGO population P_M whereas $N - N_1$ agents serving as the COA population P_C. It

Algorithm 1: Pseudocode for Hybrid COA-MGO (HCCMGO) algorithm

Inputs: Population size *N*, iteration number *T*, splitting factor *S*
Outputs: Best position and fitness potential
1. Randomly initialize n-dimensional $P_M.\mathbf{X}_i$ for all $i = 1, \ldots, N_1 = \left\lceil \frac{mN}{m+n} \right\rceil$ and m-dimensional $P_C.\mathbf{X}_j$ for all $j = 1, \ldots, N_2 = N - N_1$
2. Calculate algorithm's fitness levels for the population

3. **while** $T < T_{max}$
   % Crayfish Optimization Algorithm (COA) %
4. Define temperature t in (3)
5. **If** $t > 30$
6. Define cave X_s according to (5)
7. **If** $r < 0.5$
8. Crayfish conduct the summer resort phase in (6)
9. **Else**
10. Crayfish compete for caves through (8)
11. **End**
12. **Else**
13. Obtain food intake p with (4) and food size Q with (11)
14. **If** $Q > 2$
15. Crayfish shreds food by (12)
16. Crayfish forages according to (13)
17. **Else**
18. Crayfish forages according to (14)
19. **End**
20. **End**
21. Update fitness value for COA.
    % Mountain Gazelle Optimization (MGO) %
22. **for** (each Gazelle(X_i)) **do**
23. Compute A using (15)
24. Compute M using (20)
25. Compute H using (21)
26. Compute S using (23)
27. Compute the fitness values of *A, M, H,* and *S*
28. Include the gazelle to the habitat.
29. **End for**
30. Arrange the overall populace in ascending order.
31. Update $best_g$
32. Store the Best Gazelles in the MGO population.
33. Update the fitness value for MGO.
34. Update fitness value by comparing COA and MGO value
35. Exchange the solution from COA and MGO by choosing the certain portions of the best solution from COA and exchange it with the certain portions of the worst solution from MGO and vice versa.
36. $T = T + 1$
37. **End while**
38. Return *best Fitness*

TABLE I. BENCHMARK FUNCTIONS

Function	*Dim*	Range	F_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10, 10]	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30	[-100, 100]	0
$F_4(x) = max\{\lvert x_i \rvert, 1 \le i \le n\}$	30	[-100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30, 30]	0
$F_6(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	[-100, 100]	0
$F_7(x) = \sum_{i=1}^{n} i \times x_i^4 + random[0, 1)$	30	[-1.28, 1.28]	0
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	[-500, 500]	-418.9829 x *dim*
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12, 5.12]	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e\right)$	30	[-32, 32]	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600, 600]	0
$F_{12}(x) = \frac{\pi}{n}\{10 \sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} +$ $\sum_{i=1}^{n} u(x_i, 10, 100, 4)$, where $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i - a^m), & x_i \end{cases}$	30	[-50, 50]	0
$F_{13}(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]) +$ $\sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[-50, 50]	0

is noteworthy that the size of the populations is determined by the splitting factor. Then, the fitness value of each population is calculated. Next, the COA and MGO algorithm is run sequentially for each iteration to find and sort the global optimal solution. After that, the optimal global solutions from COA and MGO are compared to determine the best solution. Next, we introduce solution exchange between P_M and P_C, where the best solution subvector from the MGO population replaces some candidate solution subvectors with the lowest fitness in the COA population and vice versa. This solution exchange provides a low degree of search diversity to both the COA and MGO optimization mechanisms and information regarding the best solutions obtained from other dimensions. After the solution has been exchanged, HCCMGA is run again in another iteration until it reaches the maximum iteration.

In each iteration, the proposed HCCMGA requires the computation of $N_1 N_2 = \left\lceil \frac{mN}{m+n} \right\rceil \left(N - \left\lceil \frac{mN}{m+n} \right\rceil \right)$ fitness evaluations. Thus, the computational complexity of one iteration in the proposed algorithm is of $O(N^2)$. As such, the asymptotic computational complexity of the proposed HCCMGA can be obtained as $O(T_{max}\, N^2)$. Compared to COA and MGO, the computational complexity of the proposed HCCMGA is higher because both COA and MGO incurs an asymptotic computational complexity of $O(T_{max} N)$ for each iteration. Nevertheless, in the next section, we show that the proposed HCCMGA can achieve convergence quicker than the state-of-the-art in many standard benchmark test functions, especially those with multimodal natures. In such a case, the computational complexity required to achieve convergence to near-optimal solutions could be lower than or equivalent to that of MGO and COA.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

Five metaheuristics have been chosen for the simulation to be compared with the proposed HCCMGA using thirteen standard benchmark test functions. The five metaheuristics selected are COA, MGO, GWO, Whale Optimization Algorithm (WOA) [17] and Sine Cosine Algorithm (SCA) [18], which are the current state-of-the-art. For fair performance comparisons, N = 40, $T_{\max}$ = 500, and dim = 30 are set for all the simulations. Table I lists the 13 benchmark functions [15], the function name, their search space ranges, their dimensions and their minimum values. From these 13 benchmark functions, F1 - F7 are unimodal functions (with a single global optimum), while F8 - F13 are multimodal functions (with a single global optimum and many local optima). Unimodal functions assess the performance of a metaheuristic in optimizing functions with only a single optimum, whereas multimodal functions assess the performance of a metaheuristic in optimizing functions with many local optima that could cause premature convergence and trap the algorithm in local optima. A splitting factor of 15:15 has been chosen for the HCCMGA to run this simulation, which means that out of 30-dimensional solution space, a 15-dimensional solution subspace is assigned for COA, and the other 15-dimensional solution subspace is assigned for MGO.

The experimental statistical results for each of the functions are presented in Table III, while the convergence

TABLE II. MINIMIZATION RESULTS OF BENCHMARK FUNCTIONS

F	HCCMGA	COA	MGO	GWO	WOA	SCA
F1	**0±0**	**0±0**	$2.43E^{-77} \pm 1.26E^{-77}$	$8.08E^{-31} \pm 9.98E^{-31}$	$2.77E^{-79} \pm 1.35E^{-78}$	$1.11E^{1} \pm 3.19E^{1}$
F2	**0±0**	**0±0**	$1.88E^{-45} \pm 7.28E^{-45}$	$1.51E^{-18} \pm 1.09E^{-18}$	$5.23E^{-52} \pm 5.88E^{-52}$	$9.5E^{-3} \pm 1.15E^{-2}$
F3	**0±0**	**0±0**	$2.87E^{-10} \pm 1.16E^{-9}$	$1.01E^{-6} \pm 2.70E^{-6}$	$3.62E^{4} \pm 1.21E^{4}$	$6.5E^{3} \pm 4.61E^{3}$
F4	$8.82E^{-1} \pm 1.74$	**0±0**	$3.36E^{-27} \pm 1.21E^{-26}$	$1.1E^{-7} \pm 1.08E^{-7}$	$4.28E^{1} \pm 2.5E^{1}$	$3.31E^{1} \pm 1.13E^{1}$
F5	$8.61 \pm 1.34E^{1}$	$2.64E^{1} \pm 8.2E^{-1}$	$\mathbf{4.54E^{-21} \pm 1.76E^{-20}}$	$2.68E^{1} \pm 6.71E^{-1}$	$2.77E^{1} \pm 4.2E^{-1}$	$7.27E^{3} \pm 1.53E^{4}$
F6	$3.54E^{-6} \pm 9.68E^{-6}$	$2.92E^{-1} \pm 2.32E^{-1}$	$\mathbf{3.62E^{-9} \pm 1.96E^{-8}}$	$5.27E^{-1} \pm 3.48E^{-1}$	$1.5E^{-1} \pm 1E^{-1}$	$1.58E^{1} \pm 2.93E^{1}$
F7	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$	$\mathbf{4.9E^{-1} \pm 2.78E^{-1}}$
F8	$\mathbf{-1.26E^{4}} \pm 3.15E^{-5}$	$-8.65E^{3} \pm 7.25E^{2}$	$\mathbf{-1.26E^{4} \pm 8.57E^{-10}}$	$-6.06E^{3} \pm 1.2E^{3}$	$-1.09E^{4} \pm 1.65E^{3}$	$-3.78E^{3} \pm 2.44E^{2}$
F9	**0±0**	**0±0**	**0±0**	1.83 ± 2.26	0 ± 0	$3.31E^{1} \pm 4.15E^{1}$
F10	$2.43E^{-15} \pm 2.22E^{-15}$	$\mathbf{8.88E^{-16} \pm 0}$	$1.24E^{-15} \pm 1.08E^{-15}$	$6.07E^{-14} \pm 7.75E^{-15}$	$3.61E^{-15} \pm 2.26E^{-15}$	$1.25E^{1} \pm 9.36$
F11	**0±0**	**0±0**	**0±0**	$3.63E^{-2} \pm 8.28E^{-2}$	$1E^{-2} \pm 3E^{-2}$	$7.7E^{-1} \pm 3.5E^{-1}$
F12	$\mathbf{1.57E^{-32} \pm 5.57E^{-48}}$	$6.33E^{-3} \pm 5.94E^{-3}$	$2.78E^{-28} \pm 8.35E^{-28}$	$3.74E^{-2} \pm 1.71E^{-2}$	$1E^{-1} \pm 4.7E^{-1}$	$1.89E^{4} \pm 1E^{5}$
F13	$\mathbf{1.35E^{-32} \pm 5.57E^{-48}}$	$2.13 \pm 2.57E^{-1}$	$1.36E^{-32} \pm 3.13E^{-34}$	$5.29E^{-1} \pm 1.69E^{-1}$	$2.6E^{-1} \pm 1.3E^{-1}$	$8.06E^{5} \pm 2.79E^{5}$

speed for each of the algorithms based on each function from F1 to F13 is presented in Fig. 3. The statistical values are presented as *mean±standard deviation,* and they are based on the average values from 30 runs. Overall, from this table, HCCMGA has the best solution quality by achieving the minimum mean values in nine out of the thirteen benchmark functions, while COA reaches minimum mean values in eight out of the 13 benchmark functions. MGO has six functions that reach the minimum mean values out of the 13 benchmark test functions. Meanwhile, GWO, WOA, and SCA only have one function that reaches the minimum mean values out of the 13 benchmark test functions.

For unimodal test functions (i.e., F1 – F7), HCCMGA and COA reach the best optimum mean values in F1, F2, and F3. These two algorithms have also achieved the lowest standard deviations, implying a high level of stability. For F4, only COA achieves the optimum mean value. Meanwhile, for F5, HCCMGA places second behind MGO, yielding better mean and standard deviation values. Next, for F6, MGO attains the lowest mean value, followed by HCCMGA. Lastly,

all algorithms obtain similar mean and standard deviation values for F7. In sum, HCCMGA is comparable with COA for unimodal functions, with the latter being slightly better.

For multimodal test functions (i.e., F8 – F13), HCCMGA achieves the lowest mean value in five of the six test functions, while COA and MGO have only excelled in three. In F8, HCCMGA and MGO reach the global average minimum mean value. However, MGO has slightly better stability than HCCMGA because the former achieves a lower standard deviation. Besides that, HCCMGA, COA, and MGO achieve the global minimum for F9 and F11 with the lowest standard deviations. Furthermore, for F10, COA achieves the lowest mean and standard deviation values. For F12 and F13, HCCMGA is the best performer, followed by MGO.

Fig. 2 shows the convergence graphs of the different metaheuristics for F1 – F13. We can see that the proposed HCCMGA converges well in most of the test functions, except F4 and F5. For F1, COA converges faster than HCCMGA, as COA yields the minimum fitness value at around 200 iterations while HCCMGA obtains the minimum fitness value at around 250 iterations. This trend can also be found for functions F2 and F3. For function F4, COA performs better than other algorithms as it is the only algorithm that can find the global minimum in less than 500 iterations. For function F5, MGO converges faster than other algorithms to the minimum fitness value. Meanwhile, for F6, HCCMGA and MGO have similar convergence curves in the first 250 iterations. After that, MGO performs better since it can converge into lower result. For F7, we can see that the convergence of all algorithms is similar since they can find a similar minimum mean value. For F8, both HCCMGA and MGO exhibit the best convergence curves. For F9, COA and HCCMGA outperform MGO in convergence speed. Next, for F10, MGO converges to the solution after 200 iterations, while COA and HCCMGA converge to the optimal solution before 50 iterations, with HCCMGA being slightly faster. Furthermore, for F11, HCCMGA shows the highest convergence speed, followed by COA. For F12 and F13, HCCMGA outperforms other algorithms in convergence speed, followed by MGO.

In short, from the statistical values and convergence speed of each algorithm over the 13 benchmark test functions, we can see that HCCMGA is stable among all algorithms since it can reach the best minimum mean fitness value in the majority of the functions with the lowest standard deviations and it also converges quickly in most of the benchmark test functions within around 250 iterations. Although in some functions, HCCMGA does not achieve the best result among all algorithms, it still places in the top three among all the algorithms in those functions. Thus, we can conclude that the proposed HCCMGA algorithm can perform well with high stability over the unimodal benchmark test functions from F1 to F7, and the multimodal benchmark test functions from F8 to F13.

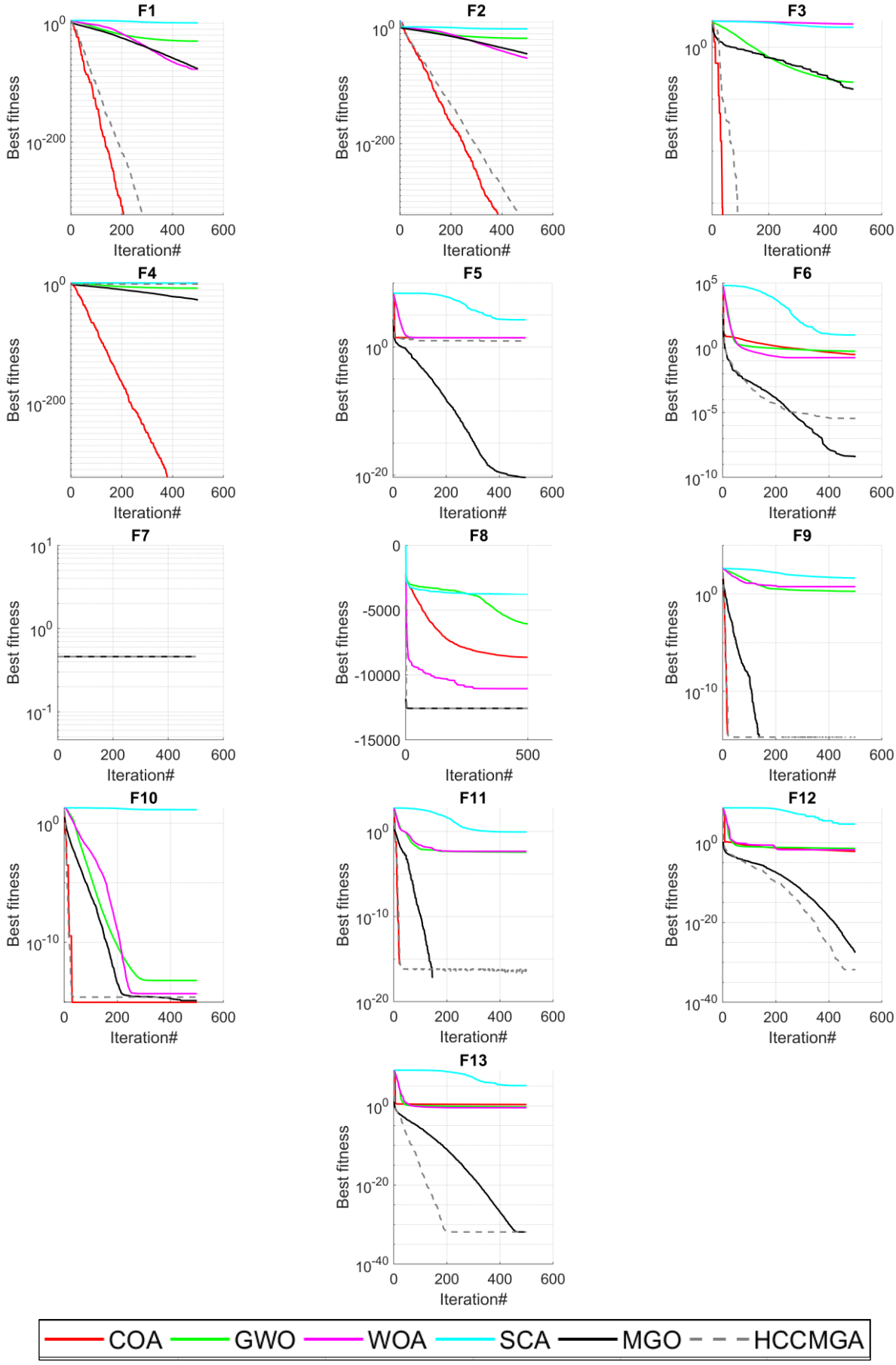

Fig. 2. Convergence performance of each algorithm for functions F1 – F13.

V. Conclusion

In this paper, we hybridize the COA and MGO algorithms. The main idea is to exchange the solution between these two algorithms. The performance of the hybrid algorithm is compared with the standard COA, MGO, GWO, WOA and SCA algorithms based on the thirteen benchmark functions. The experimental results show that the hybrid algorithm achieved the best mean and standard deviation values in most benchmark functions. Besides that, the hybrid algorithm has also outperformed other algorithms based on the convergence speed. However, there are several limitations in this study, Firstly, the HCCMGA algorithm that has been developed has not been tested in real-world applications. Secondly, this algorithm cannot directly apply to discrete or constrained optimization. So, in future work, we shall try to apply the algorithm to solve some real-life engineering problems, such as feature selection, image processing, 3D UAV positioning scheme and others.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Icnn95-international Conference on Neural Networks*, 2002.

[2] J. Clerk Maxwell, *A Treatise on Electricity and Magnetism*, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[4] S. Mirjalili, "The Ant Lion Optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, May 2015.

[5] D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning," 1989.

[6] X. Wang, Václav Snášel, Seyedali Mirjalili, J.-S. Pan, L. Kong, and H. A. Shehadeh, "Artificial Protozoa Optimizer (APO): A novel bio-inspired metaheuristic algorithm for engineering optimization," *Knowledge-based systems*, pp. 111737–111737, Apr. 2024.

[7] W. Zhao *et al.*, "Electric eel foraging optimization: A new bio-inspired optimizer for engineering applications," *Expert systems with applications*, vol. 238, pp. 122200–122200, Mar. 2024.

[8] E.-S. M. El-kenawy, Nima Khodadadi, Seyedali Mirjalili, A. A. Abdelhamid, M. M. Eid, and A. Ibrahim, "Greylag Goose Optimization: Nature-inspired optimization algorithm," *Expert Systems with Applications*, vol. 238, pp. 122147–122147, Mar. 2024.

[9] Mohammad Hussein Amiri, Nastaran Mehrabi Hashjin, Mohsen Montazeri, Seyedali Mirjalili, and Nima Khodadadi, "Hippopotamus optimization algorithm: a novel nature-inspired optimization algorithm," *Scientific Reports*, vol. 14, no. 1, Feb. 2024.

[10] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, May 2016.

[11] S. Mirjalili, "SCA: A Sine Cosine Algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, Mar. 2016.

[12] S. Mirjalili and S. Z. M. Hashim, "A new hybrid PSOGSA algorithm for function optimization," in International Conference on Computer & Information Application, 2012.

[13] A.E. Eiben and C.A. Schippers, "On evolutionary exploration and exploitation," *Fundamenta Informaticate*, vol. 35, no. 1-4, pp. 35-50, 1998.

[14] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," in *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, April 1997.

[15] H. Jia, H. Rao, C. Wen, and Seyedali Mirjalili, "Crayfish optimization algorithm," *Artificial intelligence review*, vol. 56, no. S2, pp. 1919–1979, Sep. 2023.

[16] B. Abdollahzadeh, F. S. Gharehchopogh, N. Khodadadi, and S. Mirjalili, "Mountain Gazelle Optimizer: A new Nature-inspired Metaheuristic Algorithm for Global Optimization Problems," *Advances in Engineering Software*, vol. 174, p. 103282, Dec. 2022.

[17] H. N. Fakhouri, Abdelraouf Ishtaiwi, Sharif Naser Makhadmeh, Mohammed Azmi Al-Betar, and Mohannad Alkhalaileh, "Novel Hybrid Crayfish Optimization Algorithm and Self-Adaptive Differential Evolution for Solving Complex Optimization Problems," *Symmetry*, vol. 16, no. 7, pp. 927–927, Jul. 2024.

[18] Priteesha Sarangi and P. Mohapatra, "Chaotic-Based Mountain Gazelle Optimizer for Solving Optimization Problems," *International Journal of Computational Intelligence Systems*, vol. 17, no. 1, May 2024.

[19] F. van den Bergh and A. P. Engelbrecht, "A Cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, Jun. 2004.