Grey Wolf Optimization Algorithm with Invasion-based Migration Operation

Duangjai Jitkongchuen, Pongsak Phaidang, Piyalak Pongtawevirat
Faculty of Information Technology
Dhurakij Pundit University
Bangkok, Thailand
duangjai.jit@dpu.ac.th, pongsak.phg@dpu.ac.th, piyalak.pot@dpu.ac.th

***Abstract*—This paper proposed a solution to improve the grey wolf optimizer performance with integrate the invasion-based migration operation. The traditional grey wolf optimizer algorithm have three main steps of hunting, searching for prey, encircling prey and attacking prey whereas the wolves have only one pack. The wolves in our proposed algorithm have more pack and have migrated between them. The invasion-based migration operation is used when the algorithm is trapped in the local optimum. The results are evaluated by a comparative with the traditional grey wolf optimizer (GWO) algorithm, particle swarm optimization (PSO) and differential evolution (DE) algorithm on 11 well-known benchmark functions. The experimental results showed that the proposed algorithm is capable of efficiently to solving complex optimization problems.**

Keywords*—Metaheuristic algorithm; Grey wolf optimizer algorithm; Invasion-based migration

I. INTRODUCTION

Metaheuristics have been widely used to solve complex combinatorial optimization problems. The approach of metaheuristics can be divided into two classes (1) single solution based and (2) population based [1]. In generally, the search process starts with one candidate solution. The single candidate solution is enhanced over the course of iterations while the population based starts the search process with a random initial population and this population is enhanced over the course of iterations. Features of the search process have two phases, exploration and exploitation. Exploration means to create various solutions so as to explore the search space while exploitation means to concentrate the search in a local search space.

The swarm intelligence is the one subset of metaheuristics. This algorithm concentrates to study the group behaviors that result from the local collaboration of the individuals with each other and with their environment. Some of the popular swarm intelligence is particle swarm optimization (PSO). The original PSO is proposed by Kennedy and Eberhart [2]. It simulates social behavior of the movement of organisms in a bird flock or fish school. The process of PSO optimizes a problem by having a population of candidate solutions, called particles, and trying to improve candidate solutions by moving these particles around in the search-space. Another swarm intelligence algorithm is ant colony optimization (ACO), proposed by Dorigo, Birattari and Stutzle [3]. This algorithm simulates the social behavior of ants to finding the shortest path between the nest and a source of food.

Karaboga and Basturk [4] proposed an artificial bee colony (ABC) algorithm for numeric function optimization. ABC simulates the intelligent foraging behavior of a honeybee swarm. This algorithm has three types of bees that are scout, onlooker and employed bees. Scout bees explore around the search space whereas employed and onlooker bees select food sources on the experience of them and improve their positions. Firefly algorithm was proposed in [5], is one of swarm intelligence. It simulates from social behavior of fireflies. Each firefly individual has flashing light to communicate among fireflies. The firefly that has brighter light has attracted the other less bright fireflies to moving towards it. Gandomi and Alavi [6] presented krill herd (KH) algorithm. The KH algorithm is based on the simulation of the herding behavior of krill individuals. The distances of each individual krill from food and from density of the herd are determined as the objective function for the krill movement.

This paper focused on the new metaheuristic called the grey wolf optimizer. The grey wolf optimizer algorithm (GWO) was proposed in [1]. The concept of GWO algorithm is to imitate the grey wolf behavior to live in a pack. Our algorithm proposed an alternative solution to improve the GWO performance with integrate the migration operation. The migration causes the information exchange within population and generates new candidate individuals.

The rest of the paper is organized as follows: the next section explains some backgrounds about the original grey wolf optimizer algorithm. In Section 3, the proposed algorithm is presented. The computational results from selected benchmark functions show in Section 4. Finally, the conclusions will be discussed in section 5.

II. GREY WOLF OPTIMIZER ALGORITHM

The grey wolf optimizer algorithm (GWO) was proposed in [1] is one of the latest bio-inspired algorithm. The main concept of this algorithm is to simulate the grey wolf behavior to live in a pack. They have a serious social dominant hierarchy. The top level is the leaders, called alpha. The alpha is responsible for making decisions in the pack. The persistence of the wolf pack is based on alpha's decision. The

978-1-5090-0806-3/16/$31.00 copyright 2016 IEEE
ICIS 2016, June 26-29, 2016, Okayama, Japan

second level is the subordinate wolves, called beta. The operation of beta is to help the alpha in decision making or other activities. The third level is the subordinate wolves, called delta. The members in this category consist of scouts, sentinels, elders, hunters and caretakers. Scouts are liable for observation the boundaries of region and warning the pack in case of any danger. Sentinels protect and guarantee the safety of the pack. Elders are the expertise wolves who used to be alpha or beta. Hunters help the alphas and betas when hunting prey and providing food for the pack and the caretakers are responsible for caring for the weak, ill, and wounded wolves in the pack. The lowest level is omega. The omega wolves have to comply with all the other dominant wolves. In some cases the omega is also the babysitters in the pack.

Grey wolves have an ability to memorize the positions of prey and to encircle them. The alpha performs the leader in the hunt. In order to simulate the hunting behavior of the grey wolves to the mathematically model, the best solution is assumed to be alpha (α). The beta (β) and delta (δ) is similar to the second and the third optimal solutions, respectively. The rest of the candidate solutions are assumed to be omega (ω). The hunting is guided by alpha, beta and delta while the omega wolves should update their positions by considering the positions of these three best solutions.

A. Encircling prey

Grey wolves encircle prey during the hunt. In order to mathematically model encircling behavior, the following equations are used.

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{1}$$

$$\vec{D} = \left| \vec{C} \cdot \vec{X}_p(t) - \vec{X}(t) \right| \tag{2}$$

where t is the iteration number, $\vec{X}$ is the grey wolf position and $\vec{X}_p$ is the prey position. The vectors $\vec{A}$ and $\vec{C}$ are calculated as follows

$$\vec{A} = 2a \cdot \vec{r}_1 - a \tag{3}$$

$$C = 2\vec{r}_2 \tag{4}$$

where $\vec{r}_1$ and $\vec{r}_2$ are random vectors in the range [0, 1] and the value of a is decreased from 2 to 0 over the course of iterations. The vector $\vec{C}$ is a random value in the range [0, 2]. This vector is used to provide random weights to define attractiveness of prey.

B. Hunting

In order to mathematically simulate the hunting behavior of grey wolves, the alpha, beta and delta are assumed to have better knowledge about the possible location of prey. The first three best solutions obtained so far and force the other search agents (including the omegas) to update their positions according to the position of the best search agents. The wolves' positions are updated as follow:

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{5}$$

where $\vec{X}_1, \vec{X}_2, \vec{X}_3$ are determined as in Eq. (6)-(8), respectively.

$$\vec{X}_1 = \left| \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha \right| \tag{6}$$

$$\vec{X}_2 = \left| \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta \right| \tag{7}$$

$$\vec{X}_3 = \left| \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \right| \tag{8}$$

where $\vec{X}_\alpha, \vec{X}_\beta, \vec{X}_\delta$ are the first three best solutions at a given iteration t, $\vec{A}_1, \vec{A}_2, \vec{A}_3$ are determined as in Eq. (3), and $\vec{D}_\alpha, \vec{D}_\beta, \vec{D}_\delta$ are determined as in Eq. (9)-(11), respectively.

$$\vec{D}_\alpha = \left| \vec{C}_1 \cdot \vec{X}_\alpha - \vec{X} \right| \tag{9}$$

$$\vec{D}_\beta = \left| \vec{C}_2 \cdot \vec{X}_\beta - \vec{X} \right| \tag{10}$$

$$\vec{D}_\delta = \left| \vec{C}_3 \cdot \vec{X}_\delta - \vec{X} \right| \tag{11}$$

where $\vec{C}_1, \vec{C}_2, \vec{C}_3$ are determined as in Eq. (4).

The final process is the updating of the parameter a. The parameter a that controls the tradeoff between exploration and exploitation is linearly updated to range from 2 to 0 in each iteration as shown in Eq. (12)

$$a = 2 - t\frac{2}{MaxIter} \tag{12}$$

where t is the iteration number and MaxIter is the total number of iteration.

III. THE PROPOSED ALGORITHM

The traditional grey wolf is to simulate the grey wolf behavior with only one pack therefore the proposed algorithm shows that how to simulate the grey wolf with more than one pack. The migration strategy is included in this proposed algorithm to reduce local optimum ratio and distribute value to expand the search space.

The proposed algorithm starts with the traditional grey wolf which has more than one pack. When the algorithm is trapped in local optimum, the migration strategy is deployed. The migration strategy is applied from [7]. Biological invasions can be divided into three phases. In the first phase, a propagule which a group of individuals concerned in the invasion is formed in its native habitat. Secondly, the propagule moves to a new range where a new subpopulation forms. In the final phase, the spread of the propagule in the new habitat is determined by the competition process of native individual. This process acts as a filter that is passed only by individuals that own the characteristics needed to survive novel selection pressures. The invasion relies on two key components: the propagule size (the number of individuals composing a propagule) and the propagule number (the rate at which propagules are introduced per unit of time).

As discussed, the grey wolf starts the migration operation by chosen one pack that has best fitness value, called best pack. The other packs are immigrated to the best pack. In each pack (except the best pack), the number of wolves that have migrated should unequal. Grey wolves that have the fitness value better than the average fitness value of their pack would migrate to best pack. The immigration wolf are selected as follows

$$M_i = \left\{ X_i^P \,,\, f\left(X_i^P\right) \succ \frac{1}{NP}\sum_{k=1}^{NP} f\left(X_k^P\right) \right\} \tag{13}$$

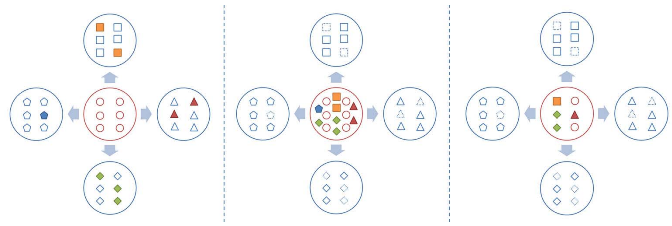
where M_i is the immigration wolves, P is pack, NP is population size, $f\left(X_i^P\right)$ is the fitness value associated to the individual X_i^P and $\succ$ is a binary relation stating that the left member is fitter than the right member.

After immigration operation, the best pack would have more individual wolves so the selection operation is performed to reduced wolf amounts of best pack. In the selection operation, the fitness value of each individual is evaluated and sorted them by ascending (minimize function). The selected individual is chosen from the sequence of fitness value. Then the other packs would randomly generate new individual for replace to the emigration wolf.

Fig. 1. Pseudo code of the GWO with invasion-based migration operation

```
Initialize the grey wolf population X_i (i = 1, 2,..., n)
Initialize a, A and C
Calculate the fitness of each search agent
X_α = the best search agent
X_β = the second best search agent
X_δ = the third best search agent
while (t < Max number of iterations)
    for ( X_i in each pack)
        Update current wolf's position by Eq. (5)
        Update a, A and C
        Calculate the fitness of all search agents
        Update X_α, X_β, X_δ
    end for
    if (is trapped local optimum)
        Evaluate the average fitness of each pack
        for ( X_i in each pack and is not best pack)
            Select i individuals that have better fitness than average fitness of pack
            Insert i selected individuals into a migration ( M_i )
        end for
        Insert a migration ( M_i ) to best pack
        Select new individuals of best pack by evaluate fitness value
        Random new individuals for emigration
    end if
end while
```

Fig. 2 The three phases of the migration model [7]



IV. EXPERIMENTAL RESULTS

The performances of the proposed algorithm have been evaluated by using eleven commonly-used benchmark functions [8]. The control parameters used in the experiments are shown in Table I and the details of benchmark functions are shown in Table II where N is dimension of the functions, S is the boundary of search space, and f_{min} is the optimum. The functions f_1 to f_7 are continuous unimodal functions and the functions f_8 and f_{11} are multimodal functions.

TABLE I. THE CONTROL PARAMETERS USED IN THE EXPERIMENTS

Parameters	Values
Population size (NP)	100
Number of pack	5
Maximum number of iterations	
Function f_1, f_6, f_{10}	1500
Function f_2, f_{11}	2000
Function f_7	3000
Function f_3, f_4, f_9	5000
Function f_8	9000
Function f_5	20000

The proposed algorithm was evaluated performances with the grey wolf optimizer (GWO) algorithm, particle swarm optimization (PSO) algorithm and differential evolution (DE) algorithm. The experimental results are shown in Table III. The results of GWO, PSO and DE were taken from the results reported in [1].

The experimental results showed that the proposed algorithm performs better than the other algorithms for the functions f_1, f_2, f_7 and f_{10}. For the functions f_6 and f_{11}, both the proposed algorithm and DE were successful to find the optimum solution. For the function f_9, only the proposed algorithm achieved the minimum of zero and for the functions f_4 and f_5, only the DE could solve the optimum. Then the proposed algorithm was able to furnish competitive results as well for functions f_3 and f_8.

TABLE II. THE BENCHMARK FUNCTIONS

Functions	N	S	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
$f_3(x)=\sum_{i=1}^{N}(\sum_{j=1}^{i} x_j)^2$	30	[-100, 100]	0
$f_4(x)=max_i\{\lvert x_i\rvert, 1\le i\le N\}$	30	[-100, 100]	0
$f_5(x)=\sum_{i=1}^{N-1}\left[100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right]$	30	[-30, 30]	0
$f_6(x)=\sum_{i=1}^{N}(\lfloor x_i+0.5\rfloor)^2$	30	[-100, 100]	0
$f_7(x)=\sum_{i=1}^{N} ix_i^4+random\,[0,1)$	30	[-1.28, 1.28]	0
$f_8(x)=\sum_{i=1}^{N}-x_i\,sin(\sqrt{\lvert x_i\rvert})$	30	[-500, 500]	-12,569.5
$f_9(x)=\sum_{i=1}^{N}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30	[-5.12, 5.12]	0
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}\right)-\exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos 2\pi x_i\right)+20+e$	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

TABLE III. THE EXPERIMENTAL RESULTS

Functions	Proposed	GWO	PSO	DE
f_1	1.68E-30	6.59E-28	1.36E-03	8.20E-14
f_2	4.41E-28	7.18E-17	4.21E-02	1.50E-09
f_3	3.98E-04	3.29E-06	70.1256	6.80E-11
f_4	3.54E-16	5.61E-07	1.0865	0
f_5	28.04153	26.81258	96.7183	0
f_6	0	0.816579	1.02E-04	0
f_7	1.01E-06	0.002213	1.23E-01	4.63E-03
f_8	-5478.54	-6123.1	-4841.29	-11,080.1
f_9	0	0.310521	46.7042	69.20
f_{10}	2.18E-14	1.06E-13	0.276015	9.72E-08
f_{11}	0	0.04485	0.009215	0

Table IV showed the performances of the proposed algorithm have been compared with the invasion-based migration model for distributed evolutionary algorithms (IM-dEA) [7]. The experimental results show that the proposed algorithm performs better than the IM-dEA all five benchmark functions. According all results, the proposed algorithm was able to provide competitive results among all compared algorithm.

TABLE IV. THE RESULTS WITH OTHER MIGRATION ALGORITHM

Functions	Proposed	IM-dEA
f_5	28.04153	6.73E+02
f_8	-5478.54	-1.53E+05
f_9	0	1.41E+03
f_{10}	2.18E-14	1.36E-02
f_{11}	0	1.69E+00

V. CONCLUSIONS

This paper proposed an alternative solution to improve the grey wolf optimizer performance with integrate the invasion-based migration operation. The traditional grey wolf optimizer algorithm have three main steps of hunting, searching for prey, encircling prey and attacking prey whereas the wolves have only one pack. The wolves in our proposed algorithm have more pack and have migrated between them. The migration operation is used when the algorithm is trapped in the local optimum to reduce local optimum ratio and distribute value to expand the search space. The performance of the proposed algorithm was compared with GWO, PSO and DE algorithms. The results showed that the proposed algorithm is capable of efficiently to solving optimization problems.

REFERENCES

[1] S. Mirjalili, S. M. Mirjalili and A. Lewis, "Grey wolf optimizer," Advances in Engineering Software, vol. 69, pp. 46-61, 2014.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings IEEE International Conference, pp. 1942-1948, 1995.

[3] M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," IEEE Computational Intelligence Magazine, pp. 28-39, 2006.

[4] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," Journal of Global Optimization, vol. 39, pp. 459-471, 2007.

[5] X. S. Yang, "Firefly algorithms for multimodal optimization," in Proceedings of the 5th International Conference on Stochastic Algorithms: Foundations and Applications, pp. 169–178, 2009.

[6] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," Commun Nonlinear Sci Numer Simulat, vol. 17, pp. 4831-4845, 2012.

[7] I. De Falco, A. Della Cioppa, D. Maisto, U. Scafuri and E. Tarantino, "Biological invasion-inspired migration in distributed evolutionary algorithms," Information Sciences, vol. 207, pp. 50-65, 2012.

[8] X. Yao, Y. Liu and L. Gao, "Evolutionary programming made faster," IEEE Transactions on Evolutionary Computation, vol. 3, pp. 82-102, 1999.