An adaptive elitism-based immigration for grey wolf optimization algorithm

Duangjai Jitkongchuen, Pongsak Phaidang and Piyalak Pongtawevirat
College of Innovative Technology and Engineering
Dhurakij Pundit University
Bangkok, Thailand
{duangjai.jit, pongsak.phg, piyalak.pot}@dpu.ac.th

Abstract— **This paper proposed the adaptive elitism-based immigration to improve the grey wolf optimization performance. The concept of elitism-based immigration is to generated immigrants and replaces it to the worst individuals in the population. The elite immigrants in our proposed are mutated before replace to the worst individuals and the parameter to control mutation ratio and elite immigrants ratio are adaptive. The performances have been evaluated by using 7 well-known benchmark functions and compared with the traditional grey wolf optimizer (GWO) algorithm, particle swarm optimization (PSO) and differential evolution (DE) algorithm. The experimental results showed that the proposed algorithm has ability to solving optimization problems.**

Keywords— Metaheuristic algorithm; Grey wolf optimization algorithm; Elitism-base immigrants; Optimization problems

I. INTRODUCTION

An optimization problem can be separated into maximum or minimum problem. There are various technologies have been proposed to resolve this problems. A meta-heuristic method is the method that has been commonly used to solve optimization problems. Meta-heuristic can be separated into three classes (i) evolutionary, (ii) physics-based and (iii) swarm intelligence [1].

Evolutionary algorithm uses mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, and selection. The genetic algorithm (GA) was proposed by Holland [2] is the most popular algorithm in this field. The concept of GA is that the new generation is created by the crossover and mutation from previous generation. Differential evolution algorithm (DE) was proposed by Storn and Price [3] is one of the evolutionary algorithms. DE has operations (crossover and mutation) that are inversed with GA. Gene expression programming was proposed by Ferreira [4] is the algorithm that has the fundamental difference from GA. The individuals in GA, called chromosomes, are linear strings of fixed length while the individuals in GEP are encoded as linear strings of fixed length. Therefore, chromosomes of GEP have different sizes and shapes.

The physics-based algorithm is the second class of meta-heuristic. The algorithm is used in this field such as Gravitational Search Algorithm (GSA) [5], Electromagnetism-Like Algorithm (EM) [6] and Central Force Optimization (CFO) [7]. The solutions in the GSA population are called agents. These the agents interact with each other based on the Newtonian gravity and the laws of motion. The agent with the heavier mass is the best solution. Then the solution in EM algorithm is determined as a charged particle and it is related to the objective function. There is electromagnetic force exists between two particles. The particle with more charge will attract the other and the other one will repulse the anterior. The higher the magnitude of attraction or repulsion is to the better the objective function value. EM algorithm has four phases: initialization, neighborhood search to exploit the local minima, calculation of total force and movement along the direction of the force. And the solutions in the CFO algorithm are called probes. Probes fly along the decision search space under the impression of gravity and the equations of motion is used to change their positions. All probes are trapped in close orbits of big masses with largest gravitational field and probe that is slowly move toward refer to the highest mass or fitness.

The third class of meta-heuristic is the swarm intelligence. The SI algorithms imitate the social behavior of swarms, schools, herds or pack in nature. Particle swarm optimization (PSO) is the one popular algorithm of SI is proposed by Kennedy and Eberhart [8]. This algorithm is inspired by social behavior of bird flock or fish school. The candidate solutions in the PSO algorithm are called particles. Each particle changes its velocity to search for better position in the search space. The one successful SI algorithm is Ant Colony Optimization (ACO). ACO is proposed by Dorigo, Birattari and Stutzle [9] is based on the behavior of ants to finding the shortest paths for food searching. Karaboga and Basturk [10] proposed an artificial bee colony (ABC) algorithm. ABC simulates the foraging behavior of honey bees. There are three components: scout, onlooker and employed bees. Employed bees that assumed to be food source go to find food source around the search space and dance when come back to hive. Scout bees that assumed to be food source has been forsake start to finding a new food source. Onlooker bees select food sources rely on employed bees dancing. Fruit fly optimization algorithm (FOA) is proposed by Pan [11]. This algorithm is applied from the food finding behavior of fruit fly. Fruit fly has talent in vision and osphresis. They used their vision and osphresis to find food and their flock location and go towards

that direction. Yang is proposed the bat algorithm [12]. This algorithm simulates from the echolocation behavior of microbats. Echolocation is a type of sonar that helps bats to find prey and recognize different types of insects even in complete darkness.

This work focus on the new meta-heuristic called the grey wolf optimizer. Mirjalili proposed the grey wolf optimizer (GWO) algorithm in [1]. This algorithm is based on the leadership hierarchy and hunting approach of grey wolves. Our algorithm proposes a migration of wolves and the immigrants' ratio is adaptive to improve the exploration and exploitation.

The rest of this paper is organized as follows: Section 2 describe the original grey wolf optimizer algorithm. We describe GWO with immigrants' schemes in Section 3. Section 4 presents the computational results and analysis. Finally, the conclusions are discussed in section 5.

II. GREY WOLF OPTIMIZATION ALGORITHM

The grey wolf optimizer is a new nature-inspired algorithm. The algorithm was inspired by the grey wolf behavior to live in a pack. They have a social dominant hierarchy. The different functions are used to separate their hierarchies into 4 levels.

The first level is called alpha is the leaders in the pack. The alpha is responsible for making decisions about hunting, sleeping place and so on. The beta is the second level is the subordinate wolves. The beta function is to help the alpha in decision making or other pack activities. The delta is the third level. The delta wolves are scouts, sentinels, elders, hunters and caretakers. Scouts are responsible for patrolling the boundaries area and warning the pack when there is danger. Sentinels protect and assure the safety of the pack. Elders who used to be alpha or beta are the expertise wolves. Hunters cooperate with alpha and beta to hunting prey and providing food for the pack. And the caretakers are responsible for take care the weak, ill and injured wolves in the pack. The lowest level is omega. The omega wolves have to follow with the other entire dominant. In some cases the omega is also the babysitters in the pack.

Wolves have skill to memorize the prey position and to encircle them. The leader in the hunt is alpha and the deputy leaders are beta and delta wolf, respectively. To solve optimization problems, the best solution is assumed to be alpha. The beta and delta are similar to the second and the third optimal solutions. The rest of the candidate solutions are assumed to be omega. The leader team, alpha, beta and delta, is guide to hunting prey. The position of alpha, beta and delta are to impact the position of omega wolves.

A. Encircling prey

Grey wolves encircle prey during the hunt. In order to create mathematically model of encircling behavior, we assumed t is the iteration number, $\vec{X}$ is the grey wolf position and $\vec{X}_p$ is the prey position. The equation is as follows

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

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

And the vectors a and c are calculated as follows

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

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

where the value of a is decreased from 2 to 0 over the course of iterations in order to impress exploration and exploitation, $\vec{r}_1$ and $\vec{r}_2$ are random vectors in the range [0, 1]. Wolves are permitted to reach any position by using their random vectors so a grey wolf can update its position inside the space around the prey in any random location. The vector $\vec{C}$ is a random value in the range [0, 2] and it is to provide random values at all times in order to impress exploration. The case of local optima stagnation was helped from this component.

B. Hunting

Grey wolves have skill to memorize the prey position and to encircle them. In the hunt, the alpha is leader while the beta and delta are participant. To simulate the hunting behavior of grey wolves, we assumed that the alpha, beta and delta have better knowledge about the possible location of prey. Then the omega wolves are updated their positions according to the position of their three best solutions. 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 (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.

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

$$\vec{D}_\alpha = \left| \vec{C}_1 \cdot \vec{X}_\alpha - \vec{X} \right| \tag{10}$$

$$\vec{D}_\alpha = \left| \vec{C}_1 \cdot \vec{X}_\alpha - \vec{X} \right| \tag{11}$$

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

The parameter a that controls the tradeoff between exploration and exploitation is updated in the final step. In each iteration, the parameter a is linearly updated to range from 2 to 0 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 original grey wolf optimization is to simulate the grey wolf behavior. Wolves have skill to memorize the prey position and to encircle them. The alpha, beta and delta wolf are leaders to hunting. The grey wolf optimization has ability to solve global search problems but it trapped in the local optimum in some cases.

This paper proposed the elitism-based immigration to improve the grey wolf optimization to reduce local optimum ratio and diffuse value to expand the search space. The principle of elitism-based immigration is generated immigrants and replaces it to the worst individuals in the population. The elite from the previous population are used to conduct the immigrants toward the current environment.

The first step of elitism-based immigration in proposed algorithm is to set the ratio of the number of elitism-based immigrants as shown in Eq. (13) where r_{ei} is the ratio of the number of elitism-based immigrants and NP is population size.

$$im = r_{ei} * NP \tag{13}$$

Afterwards, the immigrants are generated from the elite of the previous generation then compared the random value in the range [-1, 1] with mutation probability (P_m). If the random value is less than the mutation probability, then add the random value to the immigrants and replace it to the worst individuals in the population.

$$X_{worst} = X_{im} \pm rand, rand < P_m \tag{14}$$

The next step, the average fitness value of the immigrants is compared with the average fitness value of the population. If the average fitness value of the immigrants is more than the average fitness value of the population, then the mutation probability is increased to distribute value.

$$P_m^{new} = \begin{cases} P_m + C_1, & avg(f_{im}) > avg(f_{pop}) \\ P_m - C_1, & otherwise \end{cases} \tag{15}$$

where C_1 is random value in the range [0, 1].

When the algorithm is trapped in the local optimum, the ratio of the number of elitism-based immigrants (r_{ei}) is increased for more immigrants which more mutation opportunity.

$$r_{ei}^{new} = r_{ei} + (r_{ei} * C_2) \tag{16}$$

where C_2 is random value in the range [0, 1].

IV. EXPERIMENTAL RESULTS

The performances of the proposed algorithm have been evaluated by using seven commonly-used benchmark functions [14]. The benchmark functions details are shown in Table I where N is dimension of the functions, S is the boundary of search space, and f_{min} is the optimum.

```
Initialize the grey wolf population X_i (i=1, 2, ..., n)
Initialize a, A and C
Initialize P_m, C_1, C_2, r_ei
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)
   Update current wolf's position by Eq. (5)

    perform elitism-based immigration
   if (rand < P_m)
      generate r_ei * NP immigrants by mutating
      replace the worst individuals with the immigrants
      if (avg(f_im) > avg(f_pop)) P_m is increased by Eq. (15)
      else P_m is decreased by Eq. (15)
   end if

   Update a, A and C
   Calculate the fitness of all search agents
   Update X_α, X_β, X_δ
   if (is trapped local optimum) r_ei is increased by Eq. (16)
end while
```

Fig. 1. Pseudo code of an adaptive elitism-based immigrants for GWO

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

The experimental results showed that the proposed algorithm performs better than the other algorithms for the functions f_1, f_2 and f_7. The DE only could solve the optimum for functions f_4 and f_5. Both the proposed algorithm and DE were successful to find the minimum of zero for function f_6. Then the proposed algorithm was able to provide competitive results as well for function f_3.

V. CONCLUSIONS

The traditional grey wolf optimization algorithm has three main steps of hunting, searching for prey, encircling prey and attacking prey. In some cases, GWO has local optimum problems so this paper proposed the adaptive elitism-based immigration to improve the grey wolf optimization performance. The concept of elitism-based immigration is to

generated immigrants and replaces it to the worst individuals in the population. The elitism-based immigration is used to reduce local optimum ratio and diffuse 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 has ability to solving optimization problems.

TABLE I. 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

TABLE II. THE EXPERIMENTAL RESULTS

Functions	Proposed	GWO	PSO	DE
f_1	7.07E-29	6.59E-28	1.36E-03	8.20E-14
f_2	1.94E-26	7.18E-17	4.21E-02	1.50E-09
f_3	7.25E-04	3.29E-06	70.1256	6.80E-11
f_4	1.31E-10	5.61E-07	1.0865	0
f_5	24.7213	26.81258	96.7183	0
f_6	0	0.816579	1.02E-04	0
f_7	8.91E-05	0.002213	1.23E-01	4.63E-03

REFERENCES

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

[2] J. H. Holland, "Genetic algorithm," Sci Am, 1992, pp. 66–72.

[3] R. Storn and K. Price, "Differeiitial evolution – a simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol. 11, 1997, pp. 341–359.

[4] C. Ferreira, "Gene expression programming: a new adaptive algorithm for solving problems," Complex Systems, vol. 13, 2001, pp. 87–129.

[5] E. Rashedi, H. Nezamabadi-pour and S. Saryazdi, "GSA: a gravitational search algorithm," Information Sciences, vol. 179, 2009, pp. 2232–2248.

[6] C. Zhang, X. Li, L. Gao and Q. Wu, "An improved electromagnetism-like mechanism algorithm for constrained optimization," Expert Systems with Applications, vol. 40, 2013, pp. 5621–5634.

[7] R. A. Formato, "Central force optimization: a new metaheuristic with applications in applied electromagnetics," Electromagnetics Research, 2007, pp. 425–491.

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

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

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

[11] W. T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," Knowledge-Based Systems, vol. 26, 2012, pp. 69–74.

[12] X. S. Yang, "A new metaheuristic bat-inspired algorithm," Springer, 2010, pp. 65–74.

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

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