

# Guided imitation optimizer: a metaheuristic combining guided search and imitation search

Purba Daru Kusuma, Meta Kallista

Department of Computer Engineering, Faculty of Electrical Engineering, Telkom University, Bandung, Indonesia

## Article Info

### Article history:

Received Dec 31, 2023

Revised Feb 13, 2024

Accepted Mar 21, 2024

### Keywords:

Exploitation

Exploration

Metaheuristic

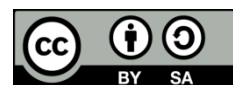
Optimization

Swarm intelligence

## ABSTRACT

This paper proposes a novel metaphor-free metaheuristic, namely the guided imitation optimizer (GIO). This metaheuristic combines the guided search and imitation-based search. There are five guided searches and three imitation-based searches. Meanwhile, there are three references used in this metaheuristic: global finest, a randomly picked solution among the swarm, and a randomized solution within the search space. GIO is then evaluated by using 23 classic functions that consist of seven high dimension unimodal functions (HDUF), six high dimension multimodal functions (HDMF), and ten fixed dimension multimodal functions (FDMF). Through simulation, GIO is superior to golden search optimizer (GSO), grey wolf optimizer (GWO), puzzle optimization algorithm (POA), and coati optimization algorithm (COA) in handling most of these functions. GIO is the first finest in tackling seventeen functions and second finest in tackling six functions. Tight competition occurs between GIO and COA due to the performance of COA which becomes the second finest in handling most of these functions.

This is an open access article under the [CC BY-SA](#) license.



## Corresponding Author:

Purba Daru Kusuma

Department of Computer Engineering, Telkom University

Buahbatu Street, Bandung, Indonesia

Email: purbodaru@telkomuniversity.ac.id

## 1. INTRODUCTION

Optimization work is a subject in applied mathematics that has been extensively used in various areas, including engineering, industry, and finance. For example, power flow optimization is necessary in the operation of the electrical power system [1]. Optimization plays important role in other energy related system, such as electricity distributed system [2] and power system stabilizer [3]. In the manufacturing system, optimization plays a significant role in scheduling the arriving job in the distributed flow-shop system [4] or assembly job-shop system [5]. In supply chain management, vehicle routing problems become a common issue to optimize, for example in the context of capacitated vehicle routing problems where there is a limitation on the fleet capacity [6]. In recent years, many studies on the vehicle routing problem have used the electric vehicle as the use case, for example in its relationship with charging and discharging functions [7]. In the financial sector, portfolio optimization is a crucial issue in arranging the assets held by individuals or institutions to maximize the investment return and minimize the investment risk [8]. Optimization deals with problems with multiple possible solutions.

In finding the finest solution, optimization faces three key issues: decision variables, constraints, and objective function. In some works, constraints are called hard constraints while objectives are called soft constraints. Both constraints and objectives are constructed from decision variables. Constraints or hard constraints are designed to limit the possible solutions and become the boundaries of the solution space. Objectives or soft constraints are some entities that become the focus of optimization. It may be minimization

or maximization. The examples of minimization objectives are minimizing total tardiness [4], total completion time [5], total inventory time [5], and waiting time [9]. On the other hand, maximization-based objectives are often related to revenue [10], profit [11], and service level [6].

In general, optimization can be solved using two approaches: deterministic and stochastic. Deterministic or exact method is powerful in guaranteeing the global optimal solution. But deterministic methods have disadvantages in tackling complex or complicated problems with high dimensions or huge number of decision variables. The deterministic method needs excessive computational resources, so it is difficult to implement in handling various practical optimization problems [12]. Meanwhile, stochastic method is more flexible and feasible to be implemented for handling various and complex problems [12]. Its random search strategy can reduce the computational resource, but with the consequence that the real optimal is not ensured to be found [12].

Metaheuristics is a popular stochastic based optimization method. As a stochastic tool, it does not search for all possible or available solutions. In general, it starts with a full random search inside the search space. Then, the current solution is improved as the iteration goes. Because of its flexibility and feasibility [13], metaheuristic has been widely used in various optimization projects.

The popularity of metaheuristic is followed by the massive development of various metaheuristics. This massive development comes from several reasons. First, there are various stochastic methods that can be explored, modified, and implemented to develop a new metaheuristic. Second, a new metaheuristic can also be built by modifying the existing metaheuristic or combining some existing metaheuristics, such as hybrid pelican Komodo algorithm (HPKA) [14] or guided pelican algorithm (GPA) [15]. Third, as stated in no-free lunch (NFL) theory, there is no one tool that is most suitable to solve all problems [12]. Any optimization method may be superior in handling some problems but mediocre or inferior in handling other problems [12].

In recent decades, nature becomes the main source of inspiration in proposing a new metaheuristic. Many metaheuristics are inspired by the animal behavior during mating or searching for food, such as: grey wolf optimizer (GWO) [16], Komodo mlpir algorithm (KMA) [17], puzzle optimization algorithm (POA) [18], coati optimization algorithm (COA) [19], northern goshawk optimizer (NGO) [20], naked mole-rat algorithm (NMR) [21], marine predator algorithm (MPA) [22], butterfly optimization algorithm (BOA) [23], cheetah optimizer (CO) [24], clouded leopard optimizer (CLO) [25], squirrel search optimizer (SSO) [26], and white shark optimizer (WSO) [27].

Several metaheuristics are also built by mimicking the human or social behavior or game mechanics. Some metaheuristics that are built based on the human or social behavior are paint optimizer (PO) [28], sewing training-based optimizer (STBO) [29], teaching learning-based optimizer (TLBO) [30], driving training-based optimizer (DTBO) [31], chef-based optimization algorithm (CBOA) [32], and election-based optimization algorithm (EBOA) [33]. Several metaheuristics that are built based on the game mechanics are darts game optimizer (DGO) [34], football game-based optimizer (FBGO) [35], and POA [36].

Unfortunately, there exists critiques because of the explosive number of metaheuristics. Many metaphor-based metaheuristics are claimed to have poor or limited novelty and hide behind their metaphor [12]. In their first introduction, these metaheuristics shown their adoption of the metaphor as the novelty [12]. But, through deeper analysis, especially based on the mathematical model used in them, some metaheuristics have high similarity [12]. Studies in proposing new metaheuristics are also often trapped in beating each other rather than focusing on finding or exploring new strategies. Fortunately, not all new metaheuristics are metaphor-based metaheuristics. Some others, such as golden search optimizer (GSO) [37], total interaction algorithm (TIA) [38], and average subtraction-based optimizer (ASBO) [39] are metaphor-free metaheuristics. They use their main strategy for their name.

By abstracting the metaphor, various new metaheuristics are built based on swarm intelligence or swarm movement. In these swarm-based metaheuristics, some references are often used to guide the movement, such as global finest, local finest, randomly picked solution, randomized solution within the space, or combination among them. For example, the slime mold algorithm uses global finest and two randomly picked solutions as references [40]. Meanwhile, crossover or imitation strategy that becomes the main strategy in the popular genetic algorithm (GA) [41] becomes less popular. New metaheuristics that adopt crossover or imitation approach are more difficult to find. Coronavirus optimization algorithm (COVIDOA) is an example of new metaheuristic that adopts imitation based search through frameshifting of the parent [42]. Like GA, COVIDOA deploys crossover and mutation in a different way compared with GA [42]. This circumstance makes the development of a new metaheuristic by combining swarm movement and imitation challenging.

Based on these problems and opportunities, this work is carried out to present a new metaheuristic called a guided-imitation optimizer (GIO). GIO is a metaheuristic that combines the guided search and imitation-based search. It consists of five guided searches and three imitation-based searches. It uses three references for its searches: the global finest, a randomly picked solution, and a randomized solution within space. The main scientific contributions to this paper: i) this work presents a novel metaheuristic that combines

multiple guided searches and multiple imitation-based searches, ii) this work exhibits the main concept, strategy, and formulation of the GIO, iii) the effectivity and efficiency of GIO are tested through 23 classic functions that can be split into seven high dimension unimodal functions (HDUF), six high dimension multimodal functions (HDMF), and ten fixed dimension multimodal functions (FDMF), and iv) the performance of the shown GIO is also competed with four new swarm-based metaheuristics.

The remainder of this paper is arranged as follows. Section 2 presents a review of several new studies proposing new metaheuristics. Section 3 presents the concept, detailed description, and formalization of GIO. Section 3 also presents the testing scenario to assess the performance of GIO. Section 4 presents the evaluation result of GIO, that consists of the comparison and data; and performs the in-depth analysis based on the test data, findings, and complexity of GIO. In the end, the conclusion and the future work potential regarding this work are presented in section 5.

## 2. RELATED WORKS

In general, all metaheuristics are stochastic based methods. It means that metaheuristic is identical to the use of random numbers [25]. As a searching method, metaheuristic deploys random search too [25]. Metaheuristics can also be seen as a trial-and-error tool. It means there is no assurance that the next move will produce a better solution than the current solution. Metaheuristic is also known as iterative method [12] where the improvement is performed through iteration.

There are various methods in the searching procedure performed by the metaheuristic. Uniform random search is common in the initialization phase [25]. It means the initial solution can be anywhere within the search space. In different point of view, any location within the search space has equal opportunity to become the initial solution.

During the iteration phase, there are several kinds of searches: imitation or crossover, full random search, neighborhood search, and guided search. Imitation is the backbone of evolution-based metaheuristics, such as GA. Imitation means the copy procedure of the value of some decision variables to produce the next solution. Neighborhood search means a new solution is generated close to the current solution. Neighborhood search can be found in several metaheuristics, such as tabu search (TS) [43], simulated annealing (SA) [44], invasive weed optimizer (IWO) [45], and artificial bee colony (ABC) [46]. A full random search means the new solution is generated within the search space. This method is performed in some metaheuristics, especially when the neighborhood search faces stagnation or in other words, the agent is structing in the local optimal. The guided search means the next solution is generated along the way from the current solution to or away from the reference. The guided search is the backbone strategy in the swarm-based metaheuristic. Particle swarm optimizer (PSO) is an example of the early built swarm-based metaheuristic where each agent performs guided search toward the mixture of the global finest and its local finest [47].

A summary of some new metaheuristics is shown in Table 1. The information in Table 1 includes the number of strategies, the existence of the guided search, the existence of the random/neighborhood search, the existence of imitation-based search, references used in the guided search, the existence of sorting procedure at the beginning of every iteration, and the use of metaphor.

Table 1. Summary of some new metaheuristics

No	Metaheuristic	Number of strategies	Guided search	Random or neighborhood search	Imitation based search	References	Sorting during iteration	Use of metaphor
1	KMA [17]	4	yes	yes	no	Some finest solutions, finest solution	yes	yes
2	POA [18]	2	yes	yes	no	Randomized solution	no	yes
3	POA [36]	2	yes	no	yes	Randomly picked solution	no	yes
4	GSO [37]	1	yes	yes	no	Local finest, global finest	no	no
5	NGO [20]	2	yes	yes	no	Randomly picked solution	no	yes
6	GWO [16]	1	yes	no	no	Some finest solutions	yes	yes
7	MPA [22]	5	yes	yes	no	Local finest	no	yes
8	SMA [40]	3	yes	yes	no	Global finest, two randomly picked solutions	no	yes
9	TIA [38]	1	yes	no	no	All other solutions	no	no
10	ASBO [39]	3	yes	no	no	Finest solution, worst solution	no	no
11	COVIDOA [42]	2	no	yes	yes	All solutions (roulette wheels)	yes	yes
12	EBOA [33]	2	yes	yes	no	Most popular solution	no	yes
13	DTBO [31]	3	yes	yes	no	A randomly picked solution from some finest solutions	yes	yes
14	STBO [29]	3	yes	yes	yes	A randomly picked solution from all better solutions or itself	no	yes
15	CBOA [32]	3	yes	yes	yes	Finest solution, a randomly picked solution from some finest solutions	yes	yes
16	This work	8	yes	no	yes	Global finest, a randomly picked solution, a randomized solution	no	no

Table 1 indicates that most of the news of metaheuristics are metaphor-based metaheuristics. Only a few of them are metaphor-free metaheuristics. Most of these new metaheuristics adopt swarm intelligence which is known as the guided search. Some of these swarm-based metaheuristics are enriched with random search or neighborhood search. Most of these metaheuristics also deploy multiple strategies rather than single strategy only. Unfortunately, metaheuristic that use imitation or crossover as its backbone strategy is very rare. The imitation strategy becomes the additional strategy embedded in the swarm-based metaheuristics.

Based on this review, the opportunity to propose a new metaphor-free metaheuristic is still open. Moreover, this opportunity also comes from an approach in embedding swarm intelligence and imitation strategies. As found in the new metaheuristics, deploying multiple strategies becomes the rational choice so that the proposed metaheuristic is competitive with the existing metaheuristics.

### 3. METHOD

#### 3.1. Proposed model

GIO is constructed by combining guided searches and imitation based searches. In the guided searches, a solution moves relative to a reference. In the imitation-based search, a solution copies some values of the reference. There are three references used in this GIO: the global finest, a randomly picked solution, and a randomized solution within the search space.

There are five guided searches. The first guided search is searching toward the global finest. The second guided search is searching toward the randomly picked solution. The third guided search is searching away from the randomly picked solution. The fourth guided search is searching toward the randomized solution within the space. The fifth guided search is searching away from the randomized solution within the search space. The reason for searching toward and searching away relative to the second and third references is that there is a possibility that these references are better or worse than the related solution. The illustration of these guided searches is shown in Figure 1.

There are three imitation based searches. The first one is imitating some values of the global finest. The second one is imitating some values of the randomly picked solution. The third one is imitating some values of the randomized solution within the space. The imitation-based searches are performed stochastically where the probability of the imitation-based search increases as the iteration increases too. This imitation-based search is illustrated in Figure 2.



Figure 1. The five guided searches

1	3	6	7	3	6	9	2	4	current solution
5	7	8	2	4	5	5	3	1	reference
5	3	6	7	4	5	9	3	4	new solution

Figure 2. Imitation based search

Each search generates an offspring. It means there are eight off-springs for a related solution in every iteration. The finest offspring is then chosen from among these eight ones for possible replacement of the current value of the related solution. The population is constructed as  $X = \{x_1, x_2, x_3, \dots, x_n\}$  where  $x$  is a solution,  $X$  is a set of solutions, and  $n$  is the swarm size. Each solution also consists of multiple entities so that  $x_i$  can be shown as  $\{x_{i,1}, x_{i,2}, x_{i,3}, \dots, x_{i,m}\}$  where  $i$  is the solution index and  $m$  is dimension. The formalization of GIO is shown in algorithm 1. This algorithm can be split into two parts: initialization and iteration. The initialization is shown from line 2 to line 5 while the iteration is shown from line 6 to line 18.

**Algorithm 1: guided imitation optimizer**

```

1  Begin
2    for i=1 to n
3      initialize  $x_i$  using (1)
4      update  $x_{best}$  using (2)
5    end for
6    for t=1 to  $t_{max}$ 
7      select  $x_{ref1}$  using (3)
8      generate  $x_{ref2}$  using (4)
9      for i=1 to n
10       for j=1 to m
11         generate  $r_1$  using (5)
12         generate candidates using (6) to (13)
13       end for
14       select  $c_{best,i}$  using (14)
15       update  $x_i$  using (15)
16       update  $x_{best}$  using (2)
17     end for
18   end for
19 end

```

During initialization, the initial solution is distributed randomly within the space. This procedure is formulated by using (1). Meanwhile, the global finest is updated every time a new solution is initialized. The updating procedure of the global finest is formulated by using (2). In (1),  $x_{i,j}$  is solution  $i$  in dimension  $j$ ,  $r_2$  is the real random number from 0 to 1,  $x_{l,j}$  is the lower border of dimension  $j$ , and  $x_{u,j}$  is the upper border of dimension  $j$ . In (2),  $x_{best}$  is the global finest and  $f$  is the objective function. The related solution alters the current value of the global finest only if the related solution is better than the global finest one.

$$x_{i,j} = x_{l,j} + r_2(x_{u,j} - x_{l,j}) \quad (1)$$

$$x_{best}' = \begin{cases} x_i, & f(x_i) < f(x_{best}) \\ x_{best}, & else \end{cases} \quad (2)$$

The iteration procedure runs from the first iteration to the maximum iteration. In the beginning, two references are selected. This selection is formulated by using (3) and (4).  $x_{ref1}$  is the first reference while  $x_{ref2}$  is the second reference.

$$x_{ref1} = U(X) \quad (3)$$

$$x_{ref2,j} = x_{ref2,j} + r_2(x_{u,j} - x_{l,j}) \quad (4)$$

After selecting the references, the next procedure is to perform the eight searches. The five guided searches are formulated using (6) to (10). On the other hand, the three imitation based searches are formulated using (11) to (13). In (6) to (13),  $c$  represents the offspring.

$$r_1 = U(0,1) \quad (5)$$

$$c_{1,i,j} = x_{i,j} + r_2(x_{best,j} - r_3x_{i,j}) \quad (6)$$

$$c_{2,i,j} = x_{i,j} + r_2(x_{ref1,j} - r_3x_{i,j}) \quad (7)$$

$$c_{3,i,j} = x_{i,j} + r_2(x_{i,j} - r_3x_{ref1,j}) \quad (8)$$

$$c_{4,i,j} = x_{i,j} + r_2(x_{ref2,j} - r_3x_{i,j}) \quad (9)$$

$$c_{5,i,j} = x_{i,j} + r_2(x_{i,j} - r_3x_{ref2,j}) \quad (10)$$

$$c_{6,i,j} = \begin{cases} x_{best,j}, & r_1 < \frac{t}{t_{max}} \\ x_{i,j}, & else \end{cases} \quad (11)$$

$$c_{7,i,j} = \begin{cases} x_{ref1,j}, & r_1 < \frac{t}{t_{max}} \\ x_{i,j}, & else \end{cases} \quad (12)$$

$$c_{8,i,j} = \begin{cases} x_{ref2,j}, r_1 < \frac{t}{t_{max}} \\ x_{i,j}, else \end{cases} \quad (13)$$

After all, offsprings are generated, there are three sequential procedures. The first step is selecting the finest offspring. This procedure is formulated by using (14). The second procedure is updating the related solution with the selected offspring. This procedure is formulated by using (15). The finest offspring alters the current value of the related solution only if the improvement takes place. The third procedure is updating the global finest. This procedure is formulated by using (2). After the iteration procedure is completed, the global finest becomes the end solution.

$$c_{best,i} = c_{k,i} \in C_i, \min(f(c_{k,i})) \quad (14)$$

$$x_i' = \begin{cases} c_{best,i}, f(c_i) < f(x_i) \\ x_i, else \end{cases} \quad (15)$$

### 3.2. Simulation setup

The first assessment is performed to evaluate the performance of proposed GIO in handling theoretical problems. The 23 benchmark functions are chosen as the theoretical problem. These 23 functions can be clustered into three groups: HDUF, HDMF, and FDMF. In this assessment, GIO is compared with four new metaheuristics: GWO, GSO, POA, and COA. There are five parameters observed in this test: mean, standard deviation, minimum score, maximum score, and the mean rank. In this first test, the swarm size is set to 5 while the maximum iteration is set 30. It means that all these metaheuristics must find an acceptable solution in the low swarm size and low maximum iteration circumstances. The second assessment is to evaluate the hyperparameters. GIO has only two adjusted parameters: the swarm size and maximum iteration. Based on this circumstance, the hyperparameter test consists of two parts. The first part is evaluating the relationship between swarm size and the average fitness score. The second part is evaluating the relationship between the maximum iteration and the average fitness score. In general, the simulation setup of this assessment is presented in Figure 3.

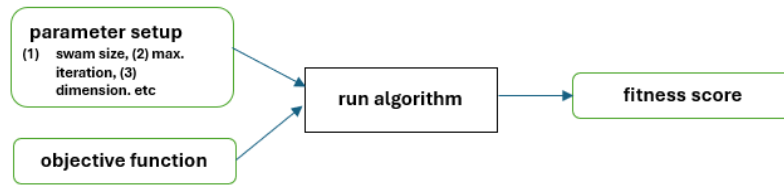


Figure 3. Simulation setup

## 4. RESULTS AND DISCUSSION

### 4.1. Simulation data

The first group of the classic functions is HDUF. This group consists of seven functions. The unimodal function is a function that consists of a single optimal solution which is the global optimal solution. In general, the global optimal solution of this function is easy to find. The main challenge is finding this global optimal solution as fast as possible. In this test, the dimensions are set to 30. The data is shown in Table 2.

The data depicted in Table 2 indicates that GIO performs superior in handling HDUFs. GIO creates the finest data of all seven HDUFs. This superior data comes with several notes. GIO becomes the sole winner in handling five functions (F3, F4, F5, F6, and F7). Meanwhile, the finest data is also achieved by POA in handling F1. Similar circumstances also occur in handling F2 where the global optimal solution is also achieved by GWO, POA, and COA.

The second group of the classic functions is HDMFs. Like in the first group, the dimension of these functions is from 2 to unlimited. In this test, the dimensions are also set to 30. This group consists of six functions. As multimodal functions, these functions have multiple optimal solutions. One optimal solution is the global optimal solutions while the other is the local optimal solutions. Based on this circumstance, the main challenge is avoiding the local optimal entrapment. The data is shown in Table 3. The data in Table 3 indicates that GIO is still superior in handling HDMFs. Among these six functions, GIO is on the first rank in handling five functions (F8, F10, F11, F12, and F13) and on the second rank in handling one function (F9).

Table 2. Comparison data for the HDUF

F	Parameter	GWO [16]	GSO [37]	POA [36]	COA [19]	GIO
1	mean	$3.2089 \times 10^3$	$3.1240 \times 10^4$	0.0000	0.0034	0.0000
	std. dev.	$4.1376 \times 10^3$	$1.1202 \times 10^4$	0.0000	0.0049	0.0000
	min	2.8285	$1.7295 \times 10^4$	0.0000	0.0000	0.0000
	max	$1.5946 \times 10^4$	$5.9520 \times 10^4$	0.0000	0.0211	0.0000
	mean rank	4	5	1	3	1
2	mean	0.0000	$1.2892 \times 10^{42}$	0.0000	0.0000	0.0000
	std. dev.	0.0000	$6.0250 \times 10^{42}$	0.0000	0.0000	0.0000
	min	0.0000	0.0000	0.0000	0.0000	0.0000
	max	0.0000	$2.8264 \times 10^{43}$	0.0000	0.0000	0.0000
	mean rank	1	5	1	1	1
3	mean	$6.0285 \times 10^4$	$6.3340 \times 10^4$	$2.4339 \times 10^4$	$2.7205 \times 10^2$	3.3244
	std. dev.	$1.2422 \times 10^5$	$3.4919 \times 10^4$	$2.6376 \times 10^4$	$3.5221 \times 10^2$	8.1483
	min	$6.7799 \times 10^1$	$3.1491 \times 10^4$	0.0000	5.2564	0.0011
	max	$4.5622 \times 10^5$	$1.8301 \times 10^5$	$9.8432 \times 10^4$	$1.5264 \times 10^3$	$3.5129 \times 10^1$
	mean rank	4	5	3	2	1
4	mean	$4.5655 \times 10^1$	$5.3414 \times 10^1$	$3.3454 \times 10^1$	0.3116	0.0054
	std. dev.	$3.2771 \times 10^1$	5.6680	$2.4767 \times 10^1$	0.1806	0.0029
	min	2.4248	$4.2597 \times 10^1$	0.0000	0.0942	0.0014
	max	$1.0000 \times 10^2$	$6.5398 \times 10^1$	$7.8000 \times 10^1$	0.8747	0.0116
	mean rank	4	5	3	2	1
5	mean	$9.4773 \times 10^6$	$6.6570 \times 10^7$	$1.3437 \times 10^3$	$2.9043 \times 10^1$	$2.8866 \times 10^1$
	std. dev.	$1.9134 \times 10^7$	$2.8428 \times 10^7$	$5.5833 \times 10^3$	0.1239	0.0732
	min	$2.1957 \times 10^2$	$2.9106 \times 10^7$	$2.9000 \times 10^1$	$2.8757 \times 10^1$	$2.8661 \times 10^1$
	max	$8.8474 \times 10^7$	$1.3585 \times 10^8$	$2.5637 \times 10^4$	$2.9379 \times 10^1$	$2.8960 \times 10^1$
	mean rank	4	5	3	2	1
6	mean	$2.4327 \times 10^3$	$3.2144 \times 10^4$	$7.4202 \times 10^1$	6.3854	4.4338
	std. dev.	$3.0096 \times 10^3$	$6.3433 \times 10^3$	$3.0681 \times 10^2$	0.3284	0.5114
	min	7.0013	$1.7136 \times 10^4$	7.2500	5.7104	3.4171
	max	$9.9282 \times 10^3$	$4.2928 \times 10^4$	$1.4132 \times 10^3$	6.8506	5.4172
	mean rank	4	5	3	2	1
7	mean	5.6672	$2.7459 \times 10^1$	0.0616	0.0259	0.0083
	std. dev.	$1.3042 \times 10^1$	$1.3265 \times 10^1$	0.0926	0.0187	0.0050
	min	0.0481	8.6964	0.0000	0.0010	0.0012
	max	$5.4423 \times 10^1$	$6.0294 \times 10^1$	0.3680	0.0838	0.0200
	mean rank	4	5	3	2	1

Table 3. Comparison data for the HDMF

F	Parameter	GWO [16]	GSO [37]	POA [36]	COA [19]	GIO
8	mean	$-6.0766 \times 10^1$	$-2.9805 \times 10^3$	$-6.3115 \times 10^2$	$-4.0885 \times 10^3$	$-5.8179 \times 10^3$
	std. dev.	$2.2428 \times 10^2$	$6.3864 \times 10^2$	$1.3026 \times 10^2$	$4.8512 \times 10^2$	$6.0766 \times 10^2$
	min	$-5.4289 \times 10^2$	$-4.4572 \times 10^3$	$-9.3975 \times 10^2$	$-5.4440 \times 10^3$	$-6.9978 \times 10^3$
	max	$2.8163 \times 10^2$	$-2.1147 \times 10^3$	$-3.9920 \times 10^2$	$-3.2294 \times 10^3$	$-4.9424 \times 10^3$
	mean rank	5	3	4	2	1
9	mean	$3.7674 \times 10^1$	$2.7457 \times 10^2$	$3.0911 \times 10^1$	0.0618	3.4019
	std. dev.	$1.9629 \times 10^1$	$3.2482 \times 10^1$	$1.0947 \times 10^2$	0.2164	8.9217
	min	0.6036	$2.2088 \times 10^2$	0.0000	0.0000	0.0000
	max	$7.0684 \times 10^1$	$3.2992 \times 10^2$	$4.8402 \times 10^2$	1.0287	$3.0961 \times 10^1$
	mean rank	4	5	3	1	2
10	mean	6.2863	$1.8751 \times 10^1$	1.3986	0.0091	0.0002
	std. dev.	4.8255	0.8060	4.5718	0.0053	0.0001
	min	0.5179	$1.7112 \times 10^1$	0.0000	0.0017	0.0000
	max	$1.4997 \times 10^1$	$2.0393 \times 10^1$	$1.7457 \times 10^1$	0.0248	0.0004
	mean rank	4	5	3	2	1
11	mean	$3.1236 \times 10^1$	$2.8642 \times 10^2$	0.1101	0.0866	0.0009
	std. dev.	$3.3773 \times 10^1$	$8.4099 \times 10^1$	0.3314	0.1658	0.0030
	min	1.2017	$1.4478 \times 10^2$	0.0000	0.0000	0.0000
	max	$9.1159 \times 10^1$	$5.2613 \times 10^2$	1.2722	0.5131	0.0103
	mean rank	4	5	3	2	1
12	mean	$2.8236 \times 10^7$	$9.9010 \times 10^7$	1.6623	0.6528	0.5520
	std. dev.	$6.9235 \times 10^7$	$7.7622 \times 10^7$	0.0000	0.1658	0.1731
	min	1.5188	$1.0083 \times 10^7$	1.6623	0.3944	0.3153
	max	$2.5260 \times 10^8$	$2.9460 \times 10^8$	1.6623	1.0936	0.9362
	mean rank	4	5	3	2	1
13	mean	$3.7150 \times 10^7$	$2.2172 \times 10^8$	$4.4869 \times 10^6$	3.1173	2.5002
	std. dev.	$7.8634 \times 10^7$	$1.1072 \times 10^8$	$2.1045 \times 10^7$	0.0469	0.2506
	min	3.4814	$3.8948 \times 10^7$	3.1400	3.0455	2.0435
	max	$2.6464 \times 10^8$	$4.3676 \times 10^8$	$9.8712 \times 10^7$	3.2184	2.9439
	mean rank	4	5	3	2	1

The third group of the classic functions is the FDMFs. Like the second group, functions in this group consist of multiple optimal solutions. This group consists of ten functions. Although the dimension of these functions is low, the global optimal solution is not easier to find because of the terrain of these functions. The data is shown in Table 4.

Table 4. Comparison data for the FDMF

F	Parameter	GWO [16]	GSO [37]	POA [36]	COA [19]	GIO
14	mean	4.5702x10 <sup>1</sup>	8.2354	9.5935	5.8988	3.1483
	std. dev.	1.0783x10 <sup>2</sup>	4.7194	4.2878	4.3178	2.7454
	min	1.2670x10 <sup>1</sup>	1.0100	2.9821	0.9984	0.9980
	max	4.2197x10 <sup>2</sup>	1.8305x10 <sup>1</sup>	1.2671x10 <sup>1</sup>	1.8304x10 <sup>1</sup>	1.1719x10 <sup>1</sup>
	mean rank	5	3	4	2	1
15	mean	0.1560	0.0313	0.1276	0.0030	0.0019
	std. dev.	0.0548	0.0310	0.03754	0.0046	0.0048
	min	0.0902	0.0020	0.0472	0.0004	0.0003
	max	0.3725	0.1001	0.1484	0.0204	0.0236
	mean rank	5	3	4	2	1
16	mean	-0.0235	-0.8406	0.0000	-1.0313	-1.0292
	std. dev.	0.1642	0.5590	0.0000	0.0006	0.0046
	min	-0.7234	-1.0316	0.0000	-1.0316	-1.0316
	max	0.1806	1.5223	0.0000	-1.0292	-1.0141
	mean rank	4	3	5	1	2
17	mean	5.8934x10 <sup>1</sup>	1.6557	2.9992x10 <sup>1</sup>	0.3989	0.4017
	std. dev.	2.7375x10 <sup>1</sup>	4.5941	2.2146x10 <sup>1</sup>	0.0035	0.0071
	min	2.8207x10 <sup>1</sup>	0.3981	3.6711	0.3981	0.3981
	max	1.9584x10 <sup>2</sup>	2.1938x10 <sup>1</sup>	5.5602x10 <sup>1</sup>	0.4160	0.4319
	mean rank	5	3	4	1	2
18	mean	1.1992x10 <sup>3</sup>	1.9070x10 <sup>1</sup>	6.0000x10 <sup>2</sup>	8.6918	3.4265
	std. dev.	2.6291x10 <sup>3</sup>	2.5948x10 <sup>1</sup>	0.0000	1.4952x10 <sup>1</sup>	1.8067
	min	5.5605x10 <sup>2</sup>	3.0000	6.0000x10 <sup>2</sup>	3.0000	3.0000
	max	1.2667x10 <sup>4</sup>	8.4844x10 <sup>1</sup>	6.0000x10 <sup>2</sup>	6.9836x10 <sup>1</sup>	1.1506x10 <sup>1</sup>
	mean rank	5	3	4	2	1
19	mean	-0.0019	-0.0197	-1.9173	-0.0495	-0.0495
	std. dev.	0.0072	0.0175	0.0000	0.0000	0.0000
	min	-0.0331	-0.0495	-1.9173	-0.0495	-0.0495
	max	0.0000	0.0000	-1.9173	-0.0495	-0.0495
	mean rank	5	4	1	2	2
20	mean	-0.0051	-2.4889	-0.0042	-3.1339	-3.2318
	std. dev.	0.0000	0.5391	0.0020	0.1427	0.0575
	min	-0.0051	-3.1234	-0.0051	-3.3078	-3.3054
	max	-0.0051	-1.3707	0.0000	-2.6505	-3.1048
	mean rank	4	3	5	2	1
21	mean	-0.2731	-1.3593	-0.2739	-5.4403	-5.5778
	std. dev.	0.0000	0.7581	0.0020	2.4723	2.4451
	min	-0.2731	-3.1535	-0.2789	-9.7219	-9.7287
	max	-0.2731	-0.4236	-0.2731	-2.4787	-1.3548
	mean rank	5	3	4	2	1
22	mean	-0.2936	-2.5151	-0.2951	-5.8511	-5.5088
	std. dev.	0.0000	2.5708	0.0037	2.5955	2.6288
	min	-0.2936	-1.0326x10 <sup>1</sup>	-0.3043	-1.0093x10 <sup>1</sup>	-1.0106x10 <sup>1</sup>
	max	-0.2936	-0.5184	-0.2936	-2.2968	-1.5911
	mean rank	5	3	4	1	2
23	mean	-0.3217	-2.2718	-0.3226	-6.0851	-4.2484
	std. dev.	0.0000	0.8416	0.0040	2.4817	2.1794
	min	-0.3217	-3.7867	-0.3403	-1.0093x10 <sup>1</sup>	-9.7136
	max	-0.3217	-0.8051	-0.3217	-2.5770	-2.0862
	mean rank	5	3	4	1	2

The data in Table 4 indicates that GIO is superior in handling the FDMFs. But its superiority is not so high as in the first and second groups of functions. GIO is the finest only in handling five functions (F14, F15, F18, F20, and F21). Meanwhile, GIO becomes the second finest at handling five functions (F16, F17, F19, F22, and F23). On the other hand, COA is the finest at handling four functions (F16, F17, F22, and F23).

The scenario of the second assessments for both parts is as follows. In the first part, there are two values for swarm size: 10 and 30. In the second part, there are also two values for the maximum iteration: 40 and 80. In the first part, the maximum iteration is set 30 while in the second part, the swarm size is set 5, columns 2 and 3. The data of the first part and the second part is shown in Table 5, columns 4 and 5.



Table 5. Data of the relation between adjusted parameters and average fitness score

Function	Average Fitness Score			
	$n(X) = 10$	$n(X) = 30$	$t_{max} = 40$	$t_{max} = 80$
1	0.0000	0.0000	0.0000	0.0000
2	0.0000	0.0000	0.0000	0.0000
3	1.6306	0.0865	0.1581	0.0000
4	0.0021	0.0003	0.0001	0.0000
5	$2.8800 \times 10^1$	$2.8715 \times 10^1$	$2.8853 \times 10^1$	$2.8879 \times 10^1$
6	3.4946	1.9709	4.4951	4.3506
7	0.0050	0.0020	0.0074	0.0025
8	$-7.1948 \times 10^3$	$-8.5142 \times 10^3$	$-6.3097 \times 10^3$	$-7.0289 \times 10^3$
9	4.0756	4.7956	1.4231	0.0000
10	0.0000	0.0000	0.0000	0.0000
11	0.0033	0.0018	0.0000	0.0000
12	0.3525	0.1316	0.5792	0.5766
13	1.9694	1.2402	2.5028	2.4100
14	1.9416	1.0024	2.0888	1.2201
15	0.0007	0.0006	0.0013	0.0016
16	-1.0316	-1.0316	-1.0292	-1.0311
17	0.3986	0.3981	0.3987	0.3989
18	3.0030	3.0005	3.0104	3.0020
19	-0.0495	-0.0495	-0.0495	-0.0495
20	-3.264	-3.3026	-3.2089	-3.2342
21	-7.1882	-8.7324	-6.1033	-6.0115
22	-6.9203	-8.9980	-5.9724	-6.9457
23	-6.4704	-8.9659	-4.9636	-6.7135

The data in columns 2 and 3 in Table 5 indicates that there are two circumstances regarding the relation between the increase of swarm size and the average fitness score. The first one is the increase of swarm size improves the average fitness score significantly. The list of functions that are in the first group or in the second group is shown in Table 6. The data in column 4 and 4 in Table 5 also indicates that there are two circumstances regarding the relation between the increase of maximum iteration and the average fitness score. The first one is that the increase in maximum iteration improves the average fitness score significantly. The list of functions that are in the first group or in the second group is also shown in Table 6.

Table 6. Group based list regarding the relation between adjusted variables and average fitness score

Group	Swarm size to Average Fitness Score		Max. Iteration to Average Fitness Score	
	1 <sup>st</sup> Circumstance	2 <sup>nd</sup> Circumstance	1 <sup>st</sup> Circumstance	2 <sup>nd</sup> Circumstance
1	F4, F7	F1 - F3, F5, F6	F3, F7	F1, F2, F4 - F6
2	F12	F8 - F11, F13	F9	F8, F10 - F13
3	-	F14 - F23	-	F14 - F23

The data in Table 6 indicates that there is not any significant improvement of the average fitness score due to the increasing of swarm size in most functions. There are only two functions in the first group, and one function in the second group, where its average fitness core is improved significantly. Fortunately, in some functions, the second circumstance occurs because the global optimal solution has been found or the end solution is near the global optimal solution. These functions are (F1, F2, F10, F11, F12, F14, F15, F16, F17, F20, F21, F22, F23). The data in Table 6 also indicates that there are only three functions whose average fitness score is improved by increasing the maximum iteration from 40 to 80. It means that convergence has been achieved in the low maximum iteration for most functions. This convergence occurs whether the end solution is the global optimal solution, near to the global optimal solution, or still far from the global optimal solution.

#### 4.2. Discussion

In this section, the discussion focuses on the improvement of the proposed GIO in relation to the recent development of metaheuristics. Different from several recent studies that still use old metaheuristics like GA and PSO, the discussion in this paper compares the performance of GIO only with recent metaheuristics. Moreover, the discussion also includes the technical comparison between GIO and its contenders, limitations, computational complexity, and the potential for further study.

The main finding of this paper is through evaluation, GIO is proven superior to GWO, GSO, POA, and COA in handling all groups of functions (HDUF, HDMF, and FDMF). The superiority of GIO mostly comes from the high dimension functions. Meanwhile, its superiority is not so high when handling the fixed dimension multimodal functions. Since the test using unimodal functions is designed to evaluate the

exploitation capability while multimodal functions are designed to evaluate the exploration capability [31], this means GIO is excellence in both exploration and exploitation.

This consideration is explained through the hyperparameter evaluation that has been performed and the data are shown in Table 6. In general, it is shown that an acceptable solution has been achieved in the low swarm size and the low maximum iteration. It means that the increase of swarm size or maximum iteration from low to high does not improve the solution significantly although the computation cost increases linearly. In some functions, the end solution is the global optimal solution or near the global optimal solution. Unfortunately, in some functions, the end solution is still far from the global optimal solution.

The superiority of GIO to GWO, GSO, POA, and COA in most of functions in 23 classic functions can be seen as the superiority of multiple strategies to limited strategies. As seen in Table 1, GWO and GSO deploy only a single strategy. Meanwhile, POA deploys only two strategies where the first strategy is guided search, and the second strategy is imitation-based search [34].

This superiority also means the necessity in variety of the references. GIO uses three references: global finest, a randomly picked solution, and a randomized solution. GWO uses single reference: resultant of some finest solutions [16]. GSO uses two references: global finest and local finest [37]. POA uses a reference: a randomized selected solution [36]. COA uses two references: the finest solution in every iteration and a randomized solution within the search space [19].

The complexity analysis in the iteration phase of GIO is shown as  $O(8n.m.t_{max})$  where  $n$  is the swarm size,  $m$  is dimension or decision variable, and  $t_{max}$  is the maximum iteration. This means the computational complexity of GIO is proportional to the swarm size, decision variables, or the maximum iteration. Term 8 in this big-O notation is applied due to there are eight searches in every iteration for every agent. As the dimension or decision variables cannot be adjusted, user should focus on the swarm size and maximum iteration as the hyperparameter setup. The main consideration is whether adjusting the swarm size or maximum iteration may improve the performance.

Although GIO is proven in producing superior performance, it still has limitations. The first limitation is this superior performance comes from multiple seeds where each seed represents a distinct search method. In other words, GIO performs massive multiple searches although only single finest seed replaces the current solution. It is different from many other metaheuristics that employ only one search method or few search methods. The second limitation of this work is that there are a lot of use cases that can be used to investigate GIO in a more comprehensive manner. These use cases can be theoretical or practical. But there is no single paper that can accommodate all these use cases. The practical use cases can also be found in broader fields of engineering with various characteristics, such as whether they are numerical or combinatorial problems.

Future development can be performed in various ways. First, both GIO and COA outperform the other three metaheuristic. It will be interesting to hybridize these two metaheuristics to make more powerful and comprehensive metaheuristic. Second, the imitation-based search should be more promoted as it is now less popular than the swarm movement. It is because GA is still implemented in many optimization studies due to its flexibility and openness to improvement. Third, the use of multiple strategies and multiple references will become more popular in the future. Fourth, random search is still important to be implemented in every iteration to overcome the local optimal problem. Future studies can also be taken by employing GIO to solve various optimization problems. There are also various sets of standard assessments, such as CEC sets in which the problems are more complex due to the existence of equality and inequality boundaries.

## 5. CONCLUSION

This work has built and shown a new metaphor-free metaheuristic called GIO. As its name implies, GIO deploys both guided search which is the backbone of swarm-based metaheuristic and imitation-based search which is the backbone of evolution-based metaheuristic. This approach is transformed into eight searches (five guided searches and three imitation-based searches). Through simulation, GIO outperforms GWO, GSO, POA, and COA in handling most unimodal and multimodal functions. It indicates that GIO has excellent exploration and exploitation capabilities. Through hyperparameter test, GIO can find an acceptable solution in the low swarm size and low maximum iteration circumstances. In the future, the construction of new metaheuristic is still available. This development can be performed by hybridizing GIO and COA. Moreover, upcoming studies can be carried out by implementing GIO to handle various optimization problems in a broader area.

## ACKNOWLEDGEMENTS

The funding for publication of this paper is provided by Telkom University, Indonesia.




## REFERENCES

- [1] J. A. O. -Toro, O. D. Montoya, and L. F. G. -Norena, "Recursive convex approximations for optimal power flow solution in direct current networks," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 6, pp. 5674–5682, Dec. 2022, doi: 10.11591/ijece.v12i6.pp5674-5682.
- [2] M. A. Kamarposhti, S. M. M. Khormandichali, and A. A. A. Solymán, "Locating and sizing of capacitor banks and multiple DGs in distribution system to improve reliability indexes and reduce loss using ABC algorithm," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 2, pp. 559–568, 2021, doi: 10.11591/eei.v10i2.2641.
- [3] M. Zadehbagheri, T. Sutikno, M. J. Kiani, and M. Yousefi, "Designing a power system stabilizer using a hybrid algorithm by genetics and bacteria for the multi-machine power system," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 3, pp. 1318–1331, 2023, doi: 10.11591/eei.v12i3.4704.
- [4] S. Yang, J. Wang, and Z. Xu, "Real-time scheduling for distributed permutation flowshops with dynamic job arrivals using deep reinforcement learning," *Advanced Engineering Informatics*, vol. 54, 2022, doi: 10.1016/j.aei.2022.101776.
- [5] L. Cheng, Q. Tang, L. Zhang, and Z. Li, "Inventory and total completion time minimization for assembly job-shop scheduling considering material integrity and assembly sequential constraint," *Journal of Manufacturing Systems*, vol. 65, pp. 660–672, 2022, doi: 10.1016/j.jmsy.2022.10.013.
- [6] G. Srivastava and A. Singh, "Two evolutionary approaches with objective-specific variation operators for vehicle routing problem with time windows and quality of service objectives," *Applied Soft Computing*, vol. 134, 2023, doi: 10.1016/j.asoc.2022.109964.
- [7] Y. J. Kim and B. D. Chung, "Energy consumption optimization for the electric vehicle routing problem with state-of-charge-dependent discharging rates," *Journal of Cleaner Production*, vol. 385, 2023, doi: 10.1016/j.jclepro.2022.135703.
- [8] Y. Wang and T. Aste, "Dynamic portfolio optimization with inverse covariance clustering," *Expert Systems with Applications*, vol. 213, 2023, doi: 10.1016/j.eswa.2022.118739.
- [9] Y. Lei, G. Lu, H. Zhang, B. He, and J. Fang, "Optimizing total passenger waiting time in an urban rail network: A passenger flow guidance strategy based on a multi-agent simulation approach," *Simulation Modelling Practice and Theory*, vol. 117, 2022, doi: 10.1016/j.simpat.2022.102510.
- [10] A. Meriläinen, P. Puranen, A. Kosonen, and J. Ahola, "Optimization of rooftop photovoltaic installations to maximize revenue in Finland based on customer class load profiles and simulated generation," *Solar Energy*, vol. 240, pp. 422–434, 2022, doi: 10.1016/j.solener.2022.05.057.
- [11] Z. Liu, B. Li, J. Wang, and Y. Qiao, "A method of value model convergence and profit optimization for crossover services," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 10459–10473, 2022, doi: 10.1016/j.jksuci.2022.11.002.
- [12] J. Swan *et al.*, "Metaheuristics 'in the large,'" *European Journal of Operational Research*, vol. 297, no. 2, pp. 393–406, 2022, doi: 10.1016/j.ejor.2021.05.042.
- [13] M. S. Braik, "Chameleon swarm algorithm: a bio-inspired optimizer for solving engineering design problems," *Expert Systems with Applications*, vol. 174, 2021, doi: 10.1016/j.eswa.2021.114685.
- [14] P. D. Kusuma and A. Dinimaharawati, "Hybrid pelican komodo algorithm," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 6, 2022, doi: 10.14569/ijacsa.2022.0130607.
- [15] P. D. Kusuma and A. L. Prasasti, "Guided pelican algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 15, no. 6, pp. 179–190, 2022, doi: 10.22266/ijies2022.1231.18.
- [16] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.
- [17] S. Suyanto, A. A. Ariyanto, and A. F. Ariyanto, "Komodo mlipir algorithm," *Applied Soft Computing*, vol. 114, 2022, doi: 10.1016/j.asoc.2021.108043.
- [18] P. Trojovský and M. Dehghani, "Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications," *Sensors*, vol. 22, no. 3, Jan. 2022, doi: 10.3390/s22030855.
- [19] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 259, 2023, doi: 10.1016/j.knsys.2022.110011.
- [20] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "Northern goshawk optimization: A new swarm-based algorithm for solving optimization problems," *IEEE Access*, vol. 9, pp. 162059–162080, 2021, doi: 10.1109/ACCESS.2021.3133286.
- [21] R. Salgotra and U. Singh, "The naked mole-rat algorithm," *Neural Computing and Applications*, vol. 31, no. 12, pp. 8837–8857, 2019, doi: 10.1007/s00521-019-04464-7.
- [22] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic," *Expert Systems with Applications*, vol. 152, 2020, doi: 10.1016/j.eswa.2020.113377.
- [23] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," *Soft Computing*, vol. 23, no. 3, pp. 715–734, 2018, doi: 10.1007/s00500-018-3102-4.
- [24] M. A. Akbari, M. Zare, R. A. -Abarghoee, S. Mirjalili, and M. Deriche, "The cheetah optimizer: a nature-inspired metaheuristic algorithm for large-scale optimization problems," *Scientific Reports*, vol. 12, no. 1, Jun. 2022, doi: 10.1038/s41598-022-14338-z.
- [25] E. Trojovska and M. Dehghani, "Clouded leopard optimization: a new nature-inspired optimization algorithm," *IEEE Access*, vol. 10, pp. 102876–102906, 2022, doi: 10.1109/ACCESS.2022.3208700.
- [26] M. Suman, V. P. Sakthivel, and P. D. Sathya, "Squirrel search optimizer: Nature inspired metaheuristic strategy for solving disparate economic dispatch problems," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 111–121, 2020, doi: 10.22266/ijies2020.1031.11.
- [27] M. Braik, A. Hammouri, J. Atwan, M. A. A. -Betar, and M. A. Awadallah, "White shark optimizer: a novel bio-inspired metaheuristic algorithm for global optimization problems," *Knowledge-Based Systems*, vol. 243, 2022, doi: 10.1016/j.knsys.2022.108457.
- [28] A. Kaveh, S. Talatahari, and N. Khodadadi, "Stochastic paint optimizer: theory and application in civil engineering," *Engineering with Computers*, vol. 38, no. 3, pp. 1921–1952, 2022, doi: 10.1007/s00366-020-01179-5.
- [29] M. Dehghani, E. Trojovská, and T. Zušćák, "A new human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training," *Scientific Reports*, vol. 12, no. 1, Oct. 2022, doi: 10.1038/s41598-022-22458-9.
- [30] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011, doi: 10.1016/j.cad.2010.12.015.
- [31] M. Dehghani, E. Trojovská, and P. Trojovský, "A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process," *Scientific Reports*, vol. 12, no. 1, Jun. 2022, doi: 10.1038/s41598-022-14225-7.
- [32] E. Trojovská and M. Dehghani, "A new human-based metaheuristic optimization method based on mimicking cooking training," *Scientific Reports*, vol. 12, no. 1, Sep. 2022, doi: 10.1038/s41598-022-19313-2.




- [33] P. Trojovský and M. Dehghani, "A new optimization algorithm based on mimicking the voting process for leader selection," *PeerJ Computer Science*, vol. 8, pp. 1–40, 2022, doi: 10.7717/peerj-cs.976.
- [34] M. Dehghani, Z. Montazeri, H. Givi, J. M. Guerrero, and G. Dhiman, "Darts game optimizer: A new optimization technique based on darts game," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 286–294, 2020, doi: 10.22266/ijies2020.1031.26.
- [35] M. Dehghani, M. Mardaneh, J. Guerrero, O. Malik, and V. Kumar, "Football game based optimization: An application to solve energy commitment problem," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 514–523, 2020, doi: 10.22266/ijies2020.1031.45.
- [36] F. A. Zeidabadi and M. Dehghani, "POA: puzzle optimization algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 15, no. 1, pp. 273–281, Feb. 2022, doi: 10.22266/ijies2022.0228.25.
- [37] M. Noroozi, H. Mohammadi, E. Efatinasab, A. Lashgari, M. Eslami, and B. Khan, "Golden search optimization algorithm," *IEEE Access*, vol. 10, pp. 37515–37532, 2022, doi: 10.1109/ACCESS.2022.3162853.
- [38] P. D. Kusuma and A. Novianty, "Total interaction algorithm: A metaheuristic in which each agent interacts with all other agents," *International Journal of Intelligent Engineering and Systems*, vol. 16, no. 1, pp. 224–234, Feb. 2023, doi: 10.22266/ijies2023.0228.20.
- [39] M. Dehghani, Š. Hubálovský, and P. Trojovský, "A new optimization algorithm based on average and subtraction of the best and worst members of the population for solving various optimization problems," *PeerJ Computer science*, vol. 8, pp. e910–e910, Mar. 2022, doi: 10.7717/peerj-cs.910.
- [40] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020, doi: 10.1016/j.future.2020.03.055.
- [41] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091–8126, 2021, doi: 10.1007/s11042-020-10139-6.
- [42] A. M. Khalid, K. M. Hosny, and S. Mirjalili, "COVIDOA: A novel evolutionary optimization algorithm based on coronavirus disease replication lifecycle," *Neural Computing & Applications*, vol. 34, no. 24, pp. 22465–22492, 2022, doi: 10.1007/s00521-022-07639-x.
- [43] R. M. Olya, S. A. Shayannia, and M. M. Movahedi, "Designing a multiobjective human resource scheduling model using the tabu search algorithm," *Discrete Dynamics in Nature and Society*, vol. 2022, pp. 1–16, Jun. 2022, doi: 10.1155/2022/5223535.
- [44] D. Cheng, "Water allocation optimization and environmental planning with simulated annealing algorithms," *Mathematical Problems in Engineering*, vol. 2022, pp. 1–11, May 2022, doi: 10.1155/2022/2281856.
- [45] A. Ibrahim, F. Anayi, M. Packianather, and O. A. Alomari, "New hybrid invasive weed optimization and machine learning approach for fault detection," *Energies*, vol. 15, no. 4, Feb. 2022, doi: 10.3390/en15041488.
- [46] C. Ganguli, S. K. Shandilya, M. Nehrey, and M. Havryliuk, "Adaptive artificial bee colony algorithm for nature-inspired cyber defense," *Systems*, vol. 11, no. 1, Jan. 2023, doi: 10.3390/systems11010027.
- [47] A. G. Gad, "Particle swarm optimization algorithm and its applications: A systematic review," *Archives of Computational Methods in Engineering*, vol. 29, no. 5, pp. 2531–2561, Aug. 2022, doi: 10.1007/s11831-021-09694-4.

## BIOGRAPHIES OF AUTHORS



**Purba Daru Kusuma**    received his bachelor, and master's degrees in electrical engineering from Bandung Institute of Technology, Indonesia, in 2002 and 2009 respectively. He received his doctoral degree in computer science from Gadjah Mada University, Indonesia, in 2017. Currently, he is an assistant professor in Telkom University, Indonesia. His research area is artificial intelligence, machine learning, and operational research. He can be contacted at email: [purbodaru@telkomuniversity.ac.id](mailto:purbodaru@telkomuniversity.ac.id).



**Meta Kallista**    received her bachelor's degree in mathematics from Brawijaya University, Indonesia, in 2010. She received her master's and doctoral' degree in mathematics from Bandung Institute of Technology, in 2013 and 2019 respectively. Currently, she is an assistant professor in Telkom University, Indonesia. Her major fields of study are mathematical modelling, biomathematics, and fluids dynamics. She can be contacted at email: [metakallista@telkomuniversity.ac.id](mailto:metakallista@telkomuniversity.ac.id).