Received: March 15, 2023. Revised: April 11, 2023.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Extended Stochastic Coati Optimizer

Purba Daru Kusuma[1]* **Ashri Dinimaharawati[1]**

[1]*Computer Engineering, Telkom University, Indonesia*
* Corresponding author's Email: purbodaru@telkomuniversity.ac.id

Abstract: A new metaheuristic can be developed by constructing from scratch, modifying the existing metaheuristics, or hybridizing some metaheuristics. This work presents a new metaheuristic: extended stochastic coati optimizer (ESCO). ESCO is developed by expanding the shortcoming coati optimization algorithm (COA). ESCO expands the number of searches and references used in COA. ESCO also implements a stochastic process for each unit to choose the searches that will perform. It differs from COA, which splits the population into two fixed groups, each performing its strategy. ESCO implements three sequential phases in every iteration. Two options can be chosen in every phase. ESCO has three references in its guided search: the global best unit, a randomly selected unit, and a randomized unit within the search space. In this work, ESCO is challenged to solve 23 classic functions and benchmarked with five shortcoming metaheuristics: guided pelican algorithm (GPA), puzzle optimization algorithm (POA), average subtraction-based optimizer (ASBO), and coati optimization algorithm (COA). The result presents the superiority of ESCO among five shortcoming metaheuristics by outperforming the GPA, POA, GSO, ASBO, and COA in solving 13, 21, 23, 16, and 13 functions, respectively. Through investigation, the multiple search approach is more effective than the single search approach.

Keywords: Metaheuristic, Optimization, Coati optimization algorithm, Stochastic.

1. Introduction

Metaheuristics are popular in various optimization studies, especially in the engineering field. In studies regarding electric vehicle (EV) systems, metaheuristics have been implemented to optimize the routing problem of the plugin hybrid electric vehicles (PHEV) [1], fast charging stations [2], distributed photovoltaic systems [3], and so on. In studies regarding manufacturing systems, metaheuristics have been implemented to optimize the coupling between integrated batch machines and direct shipping trucks [4], reducing risk in the spare part inventory system [5], and so on. In studies regarding power systems, metaheuristics have been implemented to optimize the installation and allocation of the distributed generation units [6], radial distribution system [7], reactive power flow in the large-scale power system [8], and so on.

The popularity of metaheuristics comes from two main reasons: efficiency and flexibility. Metaheuristic is efficient enough. After all, it does not consume excessive computational resources and time because it does not trace all possible solutions within the search space [9]. Metaheuristic deploys a stochastic approach to find the solution. On the other hand, this approach has the consequence that metaheuristic does not guarantee a real optimal solution [9]. On the other hand, the exact method guarantees on finding the optimal solution [10]. Metaheuristic guarantees on giving the best effort to find the quasi-optimal solution within the given time. Metaheuristics are flexible in solving various optimization problems. This flexibility comes from its nature of focusing on the objective and search space and is independent of the problem itself [9]. It performs a trial-and-error approach in every iteration. On the other hand, the exact method is not flexible and feasible to solve various and complex problems.

In recent decades, there has been a huge number of new metaheuristics. Many of them are metaphor-based metaheuristics. Animal behaviors are commonly used as inspiration for the development of

metaheuristics. Example of metaheuristics that mimic animal behavior is white shark optimizer (WSO) [11], reptile search algorithm (RSA) [12], marine predator algorithm (MPA) [13], cheetah optimizer (CO) [14], grey wolf optimizer (GWO) [15], clouded leopard optimizer (CLO) [16], snow leopard optimization algorithm (SLOA) [17], Komodo mlipir algorithm (KMA) [18], northern goshawk optimizer (NGO) [19], butterfly optimization algorithm (BOA) [20], red fox optimization algorithm (RFO) [21], remora optimization algorithm (ROA) [9], and so on. Some metaheuristics imitates the mechanism of traditional game, such as puzzle optimization algorithm (POA) [22], ring toss game-based optimization (RTGBO) [23], and so on. Some metaheuristics utilize term leader for their reference, such as random selected leader-based optimizer (RSLBO) [24], mixed leader-based optimizer (MLBO) [25], and so on.

Fortunately, some shortcomings of metaheuristics do not use a metaphor. Some of these metaheuristics are average subtraction-based optimizer (ASBO) [26], golden search optimizer (GSO) [27], total interaction algorithm (TIA) [28], and three on three optimizers (TOTO) [29]. As metaphor-free metaheuristics, their name represents their novel strategy directly. It differs from metaphor-based metaheuristics, which face critique for hiding the mere novelty of their metaphor [10].

The coati optimization algorithm (COA) is one of the shortcomings of metaheuristics. As its name suggests, COA is a metaphor-based metaheuristic. COA adopts the behavior of coati in nature [30]. As a swarm-based metaheuristic, COA is constructed by a certain number of coatis. There are two behaviors of the coati in this algorithm. The first one is the behavior of coatis during hunting and attacking the iguana as their prey [30]. The second one is the behavior of coatis in escaping from their predators [30]. In its first introduction, COA is tested to solve CEC-2011 as a theoretical optimization test and four engineering design problems as a practical optimization test [30]. This COA is proven to outperform eleven other metaheuristics: WSO, RSA, MPA, tunicate search algorithm (TSA), whale optimization algorithm (WOA), multiverse optimizer (MVO), GWO, teaching learning-based optimizer (TLBO), gravitational search algorithm (GSA), particle swarm optimization (PSO), and genetic algorithm (GA).

There are several notes regarding this shortcoming COA despite its outstanding performance. COA performs segregation of roles in its first phase by splitting its population into two same size groups without considering the quality of each coati. Then, coatis in the first group perform a guided search toward the global best solution, while coatis in the second group perform a guided search toward a randomized solution within the search space. All coatis perform neighborhood searches in the second phase.

As stated in the no-free-lunch theory, no method performs superior in solving all problems [14]. Some methods may perform well in solving some problems, but their performance needs to improve in solving other problems. On the other hand, many new metaheuristics were developed by modifying previous metaheuristics, such as the modified honey badger algorithm (MHBA) [31], guided pelican algorithm (GPA) [32], ensemble grey wolf optimizer (EGWO) [33], random walk grey wolf optimizer (RWGWO) [34], and so on. In their first introduction, their performance is better than the basic ones. Based on this circumstance, the research question in this work is how to improve the performance of COA, especially in the low maximum iteration and low population size circumstances.

This work is aimed to propose a new metaheuristic based on COA. Following this objective, the main contributions of this work are as follows.

1) ESCO is the improved version of the shortcoming COA.
2) The improvement is performed by expanding the sequential phase, references in the guided search, number of searches, and shifting the fixed split to a stochastic split in the segregation of roles.
3) The performance of ESCO is evaluated by challenging it to solve 23 classic functions and comparing its performance with five other metaheuristics (GPA, POA, GSO, ASBO, and COA).
4) The hyper-parameters test is performed to evaluate the dominance of the searches in solving optimization problems.

The rest of this paper is structured as follows. The review regarding the COA and the other shortcomings of metaheuristics are performed in section two. A detailed description of the proposed ESCO is presented in section three. This presentation includes the main concept, the difference with COA, and the formalization through pseudocode and mathematical model. The simulation performed to evaluate the ESCO, and its result is presented in section four. The in-depth analysis of the result, finding, and their connection with the theoretical basis is provided in section five. Finally, the conclusion and future research suggestions are summarized in section six.

2. Related works

COA is a shortcoming metaheuristic that adopts the behavior of coati during hunting prey and escaping

Table 1. Comparison among shortcoming metaheuristics

No	Metaheuristic	Metaphor	Number of Phases	Number of Searches	Reference
1	GPA [32]	pelican	2	2	global best unit
2	POA [22]	puzzle	2	2	a randomly selected unit within the population
3	GSO [27]	-	1	1	global best unit, local best unit
4	ASBO [26]	-	3	3	best and worst units within the population
5	COA [30]	coati	2	3	best unit within the population, a randomized unit within the search space
6	MLBO [25]	leader	1	1	a mixture between global best unit and a randomized unit within the search space
7	KMA [18]	komodo	1	4	some best units, the best unit within the population
8	NGO [19]	northern goshawk	2	2	a randomly selected unit within the population
9	BOA [20]	butterfly	1	2	the best unit within the population and two randomly selected units within the population
10	RFO [21]	red fox	2	2	the best unit within the population
11	TIA [28]	-	1	1	all other units within the population
12	TOTO [29]	-	1	6	global best unit, a randomized unit within search space, a randomly unit within the population
13	this work	coati	3	6	global best unit, a randomized unit within search space, a randomly unit within the population

from predators [30]. As an overview, the coati is a mammal in America, such as South America, Central America, Mexico, and the United States. Its dimension ranges from 33 to 69 cm, and its weight ranges from 2 to 8 kg [30]. It means coati is middle size mammal. Coatis eat small animals such as tarantulas, birds, reptiles, and eggs [30]. The green iguana is its favorite. On the other hand, coati faces the bigger animals as its predators [30].

By abstracting its metaphor, the concept of COA can be described as follows. COA is constructed of a certain number of solutions. As a metaheuristic, COA is split into two steps: initialization and iteration. In the initialization, all solutions are generated randomly within the search space [30]. This process follows uniform random with lower and upper boundary constraints in every dimension. The iteration consists of two sequential phases. The first phase is guided search, while the second is random or neighborhood search. In the first phase, segregation of roles is applied by splitting the population into two same size groups. In the first group, a guided search toward the best solution in the population (iguana on the tree) is performed [30]. The second group performs a guided search toward a randomized solution within the search space (iguana on the ground) [30]. In the second phase, a random search within the local search space is performed [30]. This local search space size decreases due to the increase in iteration. The strict acceptance-rejection approach is performed in every phase, which means a new solution cannot replace the current solution, except if the new solution is better than the current solution. Due to this approach, the best solution in the population means the best global solution.

Like COA, many metaphor-based metaheuristics adopt animal behavior while hunting prey as their core strategy. This hunting process can be seen as a guided search because there is a reference (prey) to guide the movement. On the other hand, the term escaping from a predator is often used to represent the random or neighborhood search. Table 1 presents the summarization of some shortcoming metaheuristics. Some of these metaheuristics in Table 1 use metaphor while the others do not.

Based on this review, especially by connecting the mechanics of COA and the mechanics of other metaheuristics summarized in Table 1, improvement of COA is still possible. This improvement and modification can be performed through several approaches, such as expanding the number of sequential phases, increasing the number of searches, enriching the references, modifying the splitting mechanism, and so on. These approaches are then used to develop new metaheuristic based on COA in this work.

3. Model

The ESCO is designed to improve the existing COA. The improvement comes from two terms: extending and stochastic. First, ESCO extends the strategy implemented in COA. Second, ESCO is more stochastic than COA. There are three extensions, according to ESCO. First, ESCO implements three sequential rather than two sequential phases in the COA. Second, ESCO implements six searches rather than only three, as in the COA. Third, ESCO uses three references in its guided searches: the global best unit (iguana on the tree), a randomized unit within the search space (iguana on the ground), and a randomly selected unit.

ESCO implements the segregation of roles differently from COA. In ESCO, the segregation of roles is performed in all phases. It is different from COA, where the segregation of roles is implemented only in the first phase. In ESCO, the segregation of roles is performed stochastically. It is also different from COA, where the segregation of roles is a static process where the first half of the population performs the guided search toward the best global unit. The second half of the population performs the guided search relative to the randomized unit.

The rationale for choosing this strategy is as follows. First, as many shortcoming metaheuristics perform multiple searches, more searches may give a better opportunity for improvement. Second, choosing a randomly selected unit as a reference is common in many shortcoming metaheuristics to improve the exploration capability. Third, stochastic approach in segregating roles is chosen rather than a static split to avoid a monotone search as performed in a static split in COA.

The proposed ESCO consists of three sequential phases. In the first and second phases, each unit performs guided searches. In the third phase, each unit performs a random search. There are two possible searches in every phase. In the first phase, each unit performs the guided search toward the global best unit or the guided search relative to a randomized unit within the search space. In the second phase, each unit performs a guided search relative to a randomly selected unit. However, there are two options in this second phase. The starting point of the guided search may be the corresponding unit or the randomly selected unit. In the third phase, each unit performs a neighborhood search. However, two options can be selected as the local search boundary. The options selected in each phase are performed stochastically based on a threshold. The first option is chosen if a generated random number is below the threshold. Otherwise, the second option is chosen.

A candidate is generated in every phase. The proposed ESCO adopts strict acceptance-rejection strategy in which a candidate will replace the current unit only if the candidate is better than the current unit. This concept is formalized in Algorithm 1. The last value of the global best unit becomes the final solution. Meanwhile, the detail formalization of each process within the algorithm is presented in Eq. (1) to Eq. (12). Below is the list of annotations used in this model.

c	*unit candidate*
f	*objective function*
r	*floating point random number between 0 and 1*
r_1	*first phase threshold*
r_2	*second phase threshold*
r_3	*third phase threshold*
t	*iteration*
t_{max}	*maximum iteration*
U	*uniform random*
x_i	*corresponding unit*
x_s	*randomly selected unit*
x_u	*upper boundary*
x_l	*lower boundary*
x_b	*global best unit*
X	*set of units/units*
x_{ll}	*local lower boundary*
x_{lu}	*local upper boundary*
x_{gr}	*randomized unit within the search space*

There are two processes performed in the initialization phase. The first process is generating initial unit randomly within the search space as formalized in Eq. (1). The second process is updating the global best unit as formalized in Eq. (2).

$$x_i = x_l + r(x_u - x_l) \tag{1}$$

$$x_b{}' = \begin{cases} x_i, f(x_i) < f(x_b) \\ x_b, else \end{cases} \tag{2}$$

The guided searches in the first phase are formalized by using Eq. (3) to Eq. (5). Eq. (3) states that a reference is generated within the search space. Eq. (4) formalizes the guided search toward the global best unit. Eq. (5) formalizes the guided search relative to the randomized unit within the search space. Meanwhile, Eq. (6) formalizes the updating process of the corresponding unit.

$$x_{gr} = x_l + r(x_u - x_l) \tag{3}$$

$$c = x_i + r(x_b - 2x_i) \tag{4}$$

Algorithm 1: extended stochastic coati optimizer

begin
  for i=1: $n(X)$
    initialize x_i using Eq. (1)
    find x_{best} using Eq. (2)
  end for
  for t=1: t_{max}
    for i=1: $n(X)$
      //step 1
      determine x_{gr} using Eq. (3)
      if $U(0,1) < r_1$ **then**
        generate c using Eq. (4)
      else
        generate c using Eq. (5)
      update x_i using Eq. (6)
      update x_b using Eq. (2)
      //step 2
      determine x_s using Eq. (7)
      if $U(0,1) < r_2$ **then**
        determine c using Eq. (8)
      else
        determine c using Eq. (9)
      update x_i using Eq. (6)
      update x_b using Eq. (2)
      //step 3
      determine x_{lu} using Eq. (10)
      determine x_{ll} using Eq. (11)
      determine c using Eq. (12)
      update x_i using Eq. (6)
      update x_b using Eq. (2)
    end for
  end for
end

$$c = \begin{cases} x_i + r\left(x_{gr} - 2x_i\right), f\left(x_{gr}\right) < f(x_i) \\ x_i + r\left(x_i - 2x_{gr}\right), else \end{cases} \quad (5)$$

$$x_i' = \begin{cases} c, f(c) < f(x_i) \\ x_i, else \end{cases} \quad (6)$$

The guided search in the second phase is formalized by using Eq. (7) to Eq. (9). Eq. (7) formalizes the randomly selected unit among the population. As the uniform random is used, all units have equal opportunity to choose. Eq. (8) formalizes the guided search of the corresponding unit relative to the randomly selected unit. On the other hand, Eq. (9) formalizes the guided search of the randomly selected unit relative to the corresponding unit.

$$x_s = U(X) \quad (7)$$

$$c = \begin{cases} x_i + U(0,1).(x_s - 2x_i), f(x_s) < f(x_i) \\ x_i + U(0,1).(x_i - 2x_s), else \end{cases} \quad (8)$$

$$c = \begin{cases} x_s + U(0,1).(x_s - 2x_i), f(x_s) < f(x_i) \\ x_s + U(0,1).(x_i - 2x_s), else \end{cases} \quad (9)$$

The random search in the third phase is formalized using Eqs. (10) to Eq. (12). As a neighborhood search, a candidate is randomly generated around the corresponding unit. Eq. (10) formalizes the local lower boundary calculation while Eq. (11) formalizes the local upper boundary calculation. Eq. (12) formalizes the random search due to the use of both local boundaries.

$$x_{ll} = \begin{cases} \frac{x_l}{t}, U(0,1) < r_3 \\ x_l\left(1 - \frac{t}{t_{max}}\right), else \end{cases} \quad (10)$$

$$x_{lu} = \begin{cases} \frac{x_u}{t}, U(0,1) < r_3 \\ x_u\left(1 - \frac{t}{t_{max}}\right), else \end{cases} \quad (11)$$

$$c = x_i + \left(1 - 2U(0,1)\right).U(x_{ll}, x_{lu}) \quad (12)$$

4. Simulation and result

This section presents the performance evaluation of the proposed ESCO in solving a theoretical optimization problem, benchmarking with other metaheuristics, and the hyper-parameters. The set of 23 classic functions is chosen as the problem.

In this test, ESCO is benchmarked with five shortcoming metaheuristics: GPA [32], POA [22], GSO [27], ASBO [26], and COA [30]. The rationale for choosing these five metaheuristics is as follows. All these five competitors are the shortcoming metaheuristics due their first introduction is no older than 2021. GSO is chosen due its distinct characteristic by implementing non-strict acceptance approach. Meanwhile, COA is chosen because ESCO is the improvement of COA. It is essential to evaluate the improved version with the original version.

Several parameters are set as follows. The population size is 5. The maximum iteration is 25. In ESCO, all ratios are 0.5, representing the balance between the first and second choices. In GPA, the number of candidates is 5.

The first sub-test in the theoretical optimization problem test is solving the high-dimension unimodal functions. As unimodal functions, these functions contain only one optimal unit. The main challenge is finding this optimal unit as fast as possible. In this

Table 2. Fitness score comparison in solving high dimension unimodal functions.

F	Paramater	GPA [32]	POA [22]	GSO [27]	ASBO [26]	COA [30]	ESCO
1	mean	2.0003×10^{2}	3.6364	2.5476×10^{4}	0.0124	0.0424	0.0000
	st dev	6.7222×10^{1}	1.3906×10^{1}	8.6026×10^{3}	0.0076	0.0681	0.0000
	min	7.6136×10^{1}	0.0000	1.1069×10^{4}	0.0023	0.0037	0.0000
	max	3.3011×10^{2}	6.4000×10^{1}	4.5043×10^{4}	0.0306	0.3077	0.0000
	mean rank	5	4	6	2	3	1
2	mean	5.9243×10^{17}	0.0000	2.9481×10^{33}	0.0000	0.0000	0.0000
	st dev	2.7788×10^{18}	0.0000	1.1671×10^{34}	0.0000	0.0000	0.0000
	min	0.0000	0.0000	3.9329×10^{22}	0.0000	0.0000	0.0000
	max	2.0003×10^{19}	0.0000	5.4553×10^{34}	0.0000	0.0000	0.0000
	mean rank	5	1	6	1	1	1
3	mean	3.2715×10^{3}	3.2228×10^{4}	4.0693×10^{4}	3.6309×10^{2}	4.3760×10^{2}	0.0097
	st dev	1.0851×10^{3}	2.9756×10^{4}	2.1833×10^{4}	6.5978×10^{2}	8.8119×10^{2}	0.0349
	min	1.2071×10^{3}	0.0000	1.2621×10^{4}	2.0772×10^{1}	1.9532	0.0000
	max	5.4733×10^{3}	1.1119×10^{5}	9.7657×10^{4}	2.8712×10^{3}	4.1855×10^{3}	0.1645
	mean rank	4	5	6	2	3	1
4	mean	1.8999×10^{1}	3.5818×10^{1}	5.6858×10^{1}	0.2120	1.0377	0.0006
	st dev	6.7488	2.9754×10^{1}	8.6944	0.0829	0.4585	0.0009
	min	1.1093×10^{1}	0.0000	4.1105×10^{1}	0.1012	0.2776	0.0000
	max	4.2141×10^{1}	8.8000×10^{1}	6.8931×10^{1}	0.4024	1.8408	0.0043
	mean rank	4	5	6	2	3	1
5	mean	9.2654×10^{3}	8.2305×10^{5}	4.9669×10^{7}	2.4081×10^{1}	2.4775×10^{1}	2.3951×10^{1}
	st dev	6.9375×10^{3}	3.6957×10^{6}	1.8904×10^{7}	0.1239	1.6774	0.0262
	min	2.1812×10^{3}	2.4000×10^{1}	1.6667×10^{7}	2.3786×10^{1}	2.3971×10^{1}	2.3854×10^{1}
	max	3.1916×10^{4}	1.7361×10^{7}	9.9967×10^{7}	2.4339×10^{1}	3.2077×10^{1}	2.3982×10^{1}
	mean rank	4	5	6	2	3	1
6	mean	1.6244×10^{2}	6.9909×10^{1}	2.4017×10^{4}	3.7859	5.1718	4.8053
	st dev	6.5430×10^{1}	2.9976×10^{2}	6.3396×10^{3}	0.5651	0.4684	0.4337
	min	7.9222×10^{1}	6.0000	1.2186×10^{4}	2.0291	4.0283	3.8911
	max	3.2928×10^{2}	1.4120×10^{3}	3.9807×10^{4}	4.4553	6.0239	5.6261
	mean rank	5	4	6	1	3	2
7	mean	0.2462	0.2189	2.2771×10^{1}	0.0527	0.0358	0.0088
	st dev	0.1271	0.6327	1.1237×10^{1}	0.0339	0.0208	0.0059
	min	0.0836	0.0040	6.5735	0.0125	0.0018	0.0000
	max	0.5279	3.0286	5.2111×10^{1}	0.1312	0.0897	0.0233
	mean rank	5	4	6	3	2	1

challenge, exploitation capability plays a significant role. This group has seven functions (F1 to F7): Sphere, Schwefel 2.22, Schwefel 1.2, Schwefel 2.21, Rosenbrock, Step, and Quartic. Most of them have large search spaces. Quartic is the only function in this group whose search space is narrow. In this work, the dimension of these functions is 25. The result is presented in Table 2. The result, which precision is less than 10^{-4}, is rounded to 0.0000.

Table 2 indicates the superiority of ESCO in solving high-dimension unimodal functions. ESCO is in the first rank in solving six functions (Sphere, Schwefel 2.22, Schwefel 1.2, Schwefel 2.21, Rosenbrock, and Quartic) and in the second rank in solving one function (Step). Meanwhile, four metaheuristics achieve the same result in the first rank in solving Schwefel 2.22. These metaheuristics are POA, ASBO, COA, and ESCO. This result indicates that ESCO performs well in solving problems with narrow search space to large search space.

The second sub-test in solving theoretical problems is the high-dimension multimodal functions. These functions have multiple optimal. One optimal is the global optimal, representing the destination. The other optimal is the local optimal. The main challenge is finding the area where the global optimal exists and avoiding being entrapped in the local optimal. There are six functions in these groups (F8 to F13). These functions are Schwefel, Rastrigin, Ackley, Griewank, and Penalized 2. Among them, Rastrigin represents a problem with a narrow search space, while Schwefel and Griewank represent problems with a large search space. In this work, the dimension in this group is 25. The result is presented in Table 3.

Table 3. Fitness score comparison in solving high dimension multimodal functions

F	Parameter	GPA [32]	POA [22]	GSO [27]	ASBO [26]	COA [30]	ESCO
8	mean	-5.5483×10^{3}	-2.2324×10^{3}	-2.6046×10^{3}	-3.0055×10^{3}	-3.4020×10^{3}	-3.3724×10^{3}
	st dev	8.5996×10^{2}	4.4147×10^{2}	6.2191×10^{2}	6.1490×10^{2}	4.2619×10^{2}	4.3061×10^{2}
	min	-6.8697×10^{3}	-3.1517×10^{3}	-3.6837×10^{3}	-4.7884×10^{3}	-4.2537×10^{3}	-4.2086×10^{3}
	max	-3.8963×10^{3}	-1.4180×10^{3}	-1.2388×10^{3}	-2.0362×10^{3}	-2.6287×10^{3}	-2.6239×10^{3}
	mean rank	1	6	5	4	2	3
9	mean	1.1965×10^{2}	1.5437	2.2540×10^{2}	8.7277	3.6085	0.0000
	st dev	2.4053×10^{1}	4.5913	4.2901×10^{1}	1.9805	1.1264×10^{1}	0.0000
	min	6.7213×10^{1}	0.0000	1.4568×10^{2}	4.0002	0.0027	0.0000
	max	1.7450×10^{2}	1.9175×10^{1}	2.9682×10^{2}	1.4001×10^{1}	5.2404×10^{1}	0.0000
	mean rank	5	2	6	4	3	1
10	mean	5.3988	5.5532	1.8732×10^{1}	2.5481	0.0430	0.0000
	stdev	0.8562	7.2591	0.7345	0.3645	0.0348	0.0001
	min	3.9091	0.0000	1.6595×10^{1}	2.0085	0.0088	0.0000
	max	6.7558	1.6729×10^{1}	1.9817×10^{1}	3.4214	0.1308	0.0006
	mean rank	4	5	6	3	2	1
11	mean	2.3189	0.8503	2.3085×10^{2}	0.3167	0.1481	0.0000
	st dev	0.4336	2.0821	6.9652×10^{1}	0.1512	0.2431	0.0000
	min	1.3487	0.0000	8.1073×10^{1}	0.0948	0.0010	0.0000
	max	3.4639	8.4397	3.3599×10^{2}	0.5993	0.7658	0.0000
	mean rank	5	4	6	3	2	1
12	mean	1.5973×10^{1}	1.8112	5.9599×10^{7}	0.0916	0.7146	0.8379
	st dev	7.6043	0.2402	4.6406×10^{7}	0.1935	0.2273	0.1560
	min	4.5788	1.7600	1.0076×10^{6}	0.0078	0.3883	0.4525
	max	3.2081×10^{1}	2.8865	1.5818×10^{8}	0.7299	1.1477	1.1132
	mean rank	5	4	6	1	2	3
13	mean	7.6457×1^{1}	1.2984×10^{6}	1.6530×10^{8}	8.3868	3.1785	3.0718
	st dev	9.8608×10^{1}	5.1387×10^{6}	9.3905×10^{7}	0.9470	0.2635	0.1189
	min	8.1068	3.0144	2.1460×10^{7}	6.9080	2.4778	2.7266
	max	4.2394×10^{2}	2.3426×10^{7}	3.6046×10^{8}	1.0174×10^{1}	3.6744	3.2059
	mean rank	4	5	6	3	2	1

Table 3 indicates that ESCO is still superior in solving high-dimension multimodal functions. Among these six functions, ESCO is in the first rank in solving four functions (Rastrigin, Ackley, Griewank, and Penalized 2) and in the third rank in solving two functions (Schwefel and Penalized). Due to its result, like in the first group, ESCO can tackle problems with extra-large search space.

The third sub-test in this first test is solving the fixed dimension multimodal functions. The search space is also narrow. Although the dimension in these functions is minimal and the search space is narrow, more is needed to solve them. There are ten functions in this group (F14 to F23). In some functions, the terrain is flat, and the area where the optimal global exits are very narrow. In some other functions, the terrain is very wavy. It makes it very difficult to find the area of the optimal global unit. On the other hand, the unit is easily thrown away from the area of the global optimal. These functions are Shekel Foxholes, Kowalik, Six Hump Camel, Branin, Goldstein-Price, Hartman 3, Hartman 6, Shekel 5, Shekel 7, and Shekel 10. The result is presented in Table 4.

Table 4 indicates that ESCO is still influential in solving fixed-dimension multimodal functions. Among these ten functions, ESCO is in the first rank in solving two functions (Kowalik and Hartman 3), in the second rank in solving two functions (Branin and Goldstein-Price), in the third rank in solving three functions (Six Hump Camel, Hartman 6, and Shekel 10), and in the fourth rank in solving three functions (Shekel Foxholes, Shekel 5, and Shekel 7).

The comparison result indicating the superiority of the ESCO compared to other metaheuristics is presented in Table 5. Table 5 strengthens the superiority of ESCO among other metaheuristics. ESCO is significantly superior compared to POA and GSO. Meanwhile, ESCO is also still superior compared to GPA, ASBO, and COA. The superiority of ESCO is significant compared to all these five competitors in solving high dimension functions, whether they are unimodal functions or multimodal ones. ESCO is absolute superior to GSO in solving the fixed dimension multimodal functions. On the other hand, ESCO is still superior to POA and ASBO, and inferior to GPA and COA in the third group of

Table 4. Fitness score comparison in solving fixed dimension multimodal functions

F	Paramater	GPA [32]	POA [22]	GSO [27]	ASBO [26]	COA [30]	ESCO
14	mean	1.7969	9.7338	$3.0007x10^{1}$	4.5799	4.7257	6.0382
	st dev	1.0624	4.2440	$8.4975x10^{1}$	2.6687	3.9108	3.7220
	min	0.9980	1.0291	0.9980	1.9920	0.9980	1.0023
	max	4.9505	$1.2670x10^{1}$	$4.1895x10^{2}$	$1.0763x10^{1}$	$1.3619x10^{1}$	$1.2671x10^{1}$
	mean rank	1	5	6	2	3	4
15	mean	0.0057	0.1071	0.0384	0.1123	0.0053	0.0036
	st dev	0.0077	0.0549	0.0380	0.0403	0.0085	0.0062
	min	0.0007	0.0023	0.0013	0.0257	0.0004	0.0004
	max	0.0204	0.1484	0.1170	0.1484	0.0338	0.0226
	mean rank	3	5	4	6	2	1
16	mean	-1.0315	-0.4391	-0.9048	-0.0387	-1.0311	-1.0300
	st dev	0.0001	0.4548	0.2622	0.0903	0.0008	0.0025
	min	-1.0316	-0.9216	-1.0316	-0.2956	-1.0316	-1.0316
	max	-1.0313	0.0000	-0.2477	0.0000	-1.0285	-1.0201
	mean rank	1	5	4	6	2	3
17	mean	0.3981	2.5935	0.6136	1.1446	0.4060	0.4042
	st dev	0.0000	3.0319	0.8370	1.2706	0.0339	0.0101
	min	0.3981	0.4438	0.3981	0.6438	0.3981	0.3981
	max	0.3983	$1.2729x10^{1}$	4.3170	6.1148	0.5578	0.4458
	mean rank	1	6	4	5	3	2
18	mean	3.0012	$3.9242x10^{1}$	$1.3103x10^{1}$	$2.8000x10^{1}$	$1.1356x10^{1}$	3.0668
	st dev	0.0011	$5.0483x10^{1}$	$2.0943x10^{1}$	$8.0917x10^{1}$	$2.0474x10^{1}$	0.0968
	min	3.0001	3.0000	3.0001	3.0000	3.0000	3.0001
	max	3.0043	$1.7139x10^{2}$	$9.3723x10^{1}$	$2.7800x10^{2}$	$8.4157x10^{1}$	3.3429
	mean rank	1	6	4	5	3	2
19	mean	-0.0495	-0.0495	-0.0147	-0.0495	-0.0495	-0.0495
	st dev	0.0000	0.0000	0.0132	0.0000	0.0000	0.0000
	min	-0.0495	-0.0495	-0.0495	-0.0495	-0.0495	-0.0495
	max	-0.0495	-0.0495	0.0000	-0.0495	-0.0495	-0.0495
	mean rank	1	1	6	1	1	1
20	mean	-3.2873	-1.1154	-2.0987	-0.5785	-3.1067	-2.9316
	st dev	0.0664	0.5684	0.6779	0.4211	0.0846	0.2761
	min	-3.3222	-2.7233	-3.1465	-1.6231	-3.2627	-3.1995
	max	-3.1208	-0.1895	-0.9088	-0.0849	-2.9434	-2.2183
	mean rank	1	5	4	6	2	3
21	mean	-6.3070	-0.4190	-2.4484	-3.9940	-5.7729	-3.9766
	st dev	3.2695	0.0993	1.9795	3.8261	2.3804	0.6626
	min	$-1.0148x10^{1}$	-0.6601	-9.3624	$-1.0153x10^{1}$	-8.8823	-4.8331
	max	-2.6235	-0.3172	-0.5020	-0.4965	-2.3876	-2.8793
	mean rank	1	6	5	3	2	4
22	mean	-7.3651	-0.4701	-1.9404	-4.2655	-4.5937	-3.9079
	st dev	3.2304	0.1947	0.9546	3.3589	1.8577	1.3811
	min	$-1.0384x10^{1}$	-1.0086	-4.1265	$-1.0403x10^{1}$	-8.9320	-8.7333
	max	-2.7460	-0.2936	-0.5520	-0.9100	-2.3489	-2.0361
	mean rank	1	6	5	3	2	4
23	mean	-6.1046	-0.6132	-2.3495	-2.4003	-5.3194	-3.7469
	st dev	3.6205	0.2298	1.5521	1.5470	2.3417	0.6773
	min	$-1.0520x10^{1}$	-1.2409	-7.8994	-5.1285	-9.8285	-4.8052
	max	-2.4904	-0.3774	-0.7323	-0.5556	-2.4063	-2.0363
	mean rank	1	6	5	4	2	3

functions. Contrary, GPA is superior in this third group by achieving the first rank in solving nine out of ten functions.

The second test is the hyperparameter test. This test focuses on the adjusted parameters in ESCO that do not affect the computational resource. In general,

Table 5. Group based superiority of ESCO

Group	Number of Functions Where ESCO is Better				
	GPA [32]	POA [22]	GSO [27]	ASBO [26]	COA
1	7	6	7	5	6
2	5	6	6	5	4
3	1	9	10	6	3
Total	13	21	23	16	13

Table 6. Relation between r_1 and the average fitness score

F	Average Fitness Score		Which r_1 is Better?
	r_1 = 0.1	r_1 = 0.9	
1	0.0057	0.0000	high
2	0.0000	0.0000	none
3	9.4903×10^1	0.0000	high
4	0.2160	0.0000	high
5	2.4209×10^1	2.3944×10^1	none
6	4.9641	4.9850	none
7	0.0216	0.0076	high
8	-3.2280×10^3	-3.1975×10^3	none
9	0.1213	0.0000	high
10	0.0182	0.0000	high
11	0.0121	0.0000	high
12	0.7535	0.7083	none
13	3.1339	3.0960	none
14	5.0813	4.9205	none
15	0.0025	0.0022	none
16	-1.0292	-1.0293	none
17	0.4018	0.4019	none
18	3.4362	4.2966	none
19	-0.0495	-0.0495	none
20	-2.9251	-3.0081	none
21	-4.3738	-4.2138	none
22	-3.6406	-4.1395	none
23	-3.9246	-3.8671	none

the population size and maximum iteration improve the quality of the final unit but with the consequence of the increase of computational resources. On the other hand, due to some previous work, the increase of these two parameters does not guarantee improvement due to the characteristics of the problem. In this second test, three sub-tests are performed to evaluate the relation between the three ratios in ESCO and the average fitness score. Moreover, this test evaluates which strategy is more dominant in every phase of solving the problem. The result is presented in Table 6, Table 7, and Table 8.

Table 6 indicates different circumstances regarding the value of r_1 to the average fitness score. In the first group of functions, there are four functions where the average fitness score is improved when the r_1 is high, and there are three functions with a less significant difference. In the second group, there are four functions where the average fitness score is improved when r_1 is high, and there are two functions with a less significant difference. In the third group, all functions meet no difference in the average fitness score, whether r_1 is low or high.

Table 7. Relation between r_2 and the average fitness score

F	Average Fitness Score		Which r_2 is Better?
	r_2 = 0.1	r_2 = 0.9	
1	0.0000	0.0000	none
2	0.0000	0.0000	none
3	0.0285	0.0033	high
4	0.0006	0.0000	high
5	2.3942×10^1	2.3954×10^1	none
6	4.9843	4.8414	none
7	0.0114	0.0079	none
8	-3.3225×10^3	-3.3567×10^3	none
9	0.0000	0.0000	none
10	0.0001	0.0000	none
11	0.0000	0.0000	none
12	0.7378	0.7273	none
13	3.0884	3.0597	none
14	3.9395	4.7153	none
15	0.0041	0.0027	none
16	-1.0300	-1.0304	none
17	0.4036	0.4084	none
18	3.1300	3.0431	none
19	-0.0495	-0.0495	none
20	-3.0531	-2.9787	none
21	-3.8775	-4.2917	none
22	-3.8991	-3.6995	none
23	-4.0728	-3.7992	none

Table 7 indicates that the different values of r_2 provide a less significant effect in improving the average fitness score. Only two functions from the first group in which the average fitness score is significantly improved when r_2 is high. Otherwise, there is no difference between whether r_2 is low or high.

Table 8 indicates that the different values of r_3 provide a less significant effect in improving the average fitness score. Only two functions from the first group and one function in the third group in which the average fitness score is significantly improved when r_3 is high. Otherwise, there is no difference between whether r_3 is low or high.

5. Discussion

This section discusses the in-depth analysis regarding the result, findings, and the linkage with the theoretical background. This section is divided into five parts. The first part is analysis regarding the comparison between the proposed ESCO with five metaheuristics. The second part is an analysis of the hyperparameter tests. The third part is an analysis of

Table 8. Relation between r_3 and the average fitness score

F	Average Fitness Score		Which r_3 is Better?
	r_3 = 0.1	r_3 = 0.9	
1	0.0000	0.0000	none
2	0.0000	0.0000	none
3	0.0303	0.0135	high
4	0.0003	0.0003	high
5	2.3943×10^{1}	2.3951×10^{1}	none
6	4.8133	4.9208	none
7	0.0085	0.0143	none
8	-3.0982×10^{3}	-3.6965×10^{3}	none
9	0.0000	0.0000	none
10	0.0000	0.0000	none
11	0.0000	0.0000	none
12	0.8299	0.5889	none
13	3.0728	3.0937	none
14	5.2817	4.3584	none
15	0.0032	0.0015	high
16	-1.0262	-1.0309	none
17	0.4070	0.4037	none
18	3.0669	4.2604	none
19	-0.0495	-0.0495	none
20	-2.9931	-2.9702	none
21	-2.8400	-4.5794	none
22	-3.1617	-4.6331	none
23	-3.2393	-4.2544	none

the algorithm's complexity. The fourth part is the analysis regarding the limitations of the proposed ESCO and its linkage with future development.

Table 4 indicates the superiority of the proposed ESCO over five metaheuristics. ESCO outperforms GPA, POA, GSO, ASBO, and COA in solving 13, 21, 23, 16, and 13 functions. ESCO is absolute superior to GSO, almost absolute superior to POA, significant superior to ASBO, and still superior to GPA and COA. ESCO is absolute superior to GSO since ESCO outperforms GSO in all 23 functions.

By splitting GSO and POA in the first group and GPA, ASBO, and COA in the second group, there are different approach between the first group and the second group. The strategy implemented in metaheuristics in the first group is more limited compared to the metaheuristics in the second group. GSO implements only single guided search that combines the global best unit and local best unit into single reference for single search [27]. GSO does not implement random search dedicatedly [27]. Moreover, GSO does not implement strict acceptance strategy that can be used to avoid the worsening circumstance [27]. On the other hand, POA depends only on a randomly selected unit within the population as a reference for its guided search and imitation based search [22]. Although POA implements strict acceptance approach, it still cannot compete with other metaheuristics.

Implementing strict acceptance approach is proven effective to avoid the worsening circumstance. In GPA, POA, COA and ASBO, there is a guided search where the unit moves toward the reference only if the reference is better than the unit. Otherwise, the unit will avoid the reference. This selective movement can also be found in other shortcoming metaheuristics, such as KMA [18], TIA [28], NGO [19], and so on.

Multiple search strategy is proven better than single search strategy. ESCO, COA, ASBO, GPA, and POA perform multiple search strategies. Meanwhile, GSO performs a single search. In the multiple search strategy implemented in ESCO, COA, and ASBO, at least two searches are performed: guided and random or neighbourhood searches.

The result in Table 6 indicates that the guided search toward the global best unit is better than the guided search relative to a randomized unit within the search space. This option becomes more critical in solving high-dimension problems, whether unimodal or multimodal functions. Meanwhile, this choice is insignificant in solving fixed-dimension multimodal functions because there is no difference in selecting the first and second options.

The result in Table 7 indicates that the difference between choosing the corresponding unit or the reference as the starting point is unimportant in the guided search relative to a randomly selected unit. The result is usually similar. Although choosing one approach between these two options is unnecessary, the guided search relative to a randomly selected unit is still essential, as previously discussed.

The result in Table 8 indicates that the way the local search space is reduced is unimportant. There is no difference in average fitness score whether the local search space declines linearly or negatively exponentially. However, as previously mentioned, neighbourhood or random search with declining local search space is essential.

The complexity of the proposed ESCO, especially in the iteration phase, is presented as $O(3n(X).t_{max})$. The letter 3 represents the three steps implemented for the entire population in every iteration. It means that the complexity of ESCO is linearly proportional to the population size or maximum iteration. This complexity is commonly found in many metaheuristics and is simple enough. This simplicity comes because there is no sorting at the beginning of the iteration.

This work, especially this proposed ESCO, still has limitations despite presenting superior performance. First, ESCO is tested by using 23 classic functions as theoretical problems. Meanwhile, other functions can be used as theoretical problems,

such as IEEE CEC 2011 [30], IEEE CEC 2017 [16], etc. ESCO has yet to be tested to solve practical optimization problems, whether numerical or combinatorial. Numerical optimization problems have a different challenge from combinatorial optimization problems. Combinatorial optimization problems can be seen as a task to arrange a set of blocks so that it does not need a precise floating-point number like in numerical optimization problems. In some numerical optimization problems, the decision variables are presented in integers. Second, a metaheuristic cannot accommodate too many approaches, so many approaches have not been accommodated in this proposed ESCO, such as eliminating the worst units, such as in GSO, invasive weed optimizer (IWO), or non-dominated sorting genetic algorithm (NSGA II).

6. Conclusion

A new metaheuristic developed by modifying and improving the shortcoming coati optimization algorithm (COA) has been presented. This proposed ESCO has been tested to solve 23 classic functions, and the result presents the superiority of ESCO among five shortcoming metaheuristics. The ESCO is better than GPA, POA, GSO, ASBO, and COA in solving 13, 21, 23, 16, and 13 functions, respectively. The global best unit is more effective through investigation than a randomized unit within the search space or the local best unit, especially in solving big dimension problems. It is also shown that both the guided search and random search are essential, and they are better performed separately rather than combined into a single movement.

Future works can be performed in many ways. The proposed ESCO can be challenged to solve various problems, efficient problems, whether they are numerical problems or combinatorial problems. Modifying or improving the existing COA and proposed ESCO is also available.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, Kusuma; methodology, Kusuma; software, Kusuma; validation, Kusuma; formal analysis, Kusuma and Dinimaharawati; data curation, Kusuma; writing-original paper draft, Kusuma; writing-review and editing, Dinimaharawati; visualization, Kusuma; supervision, Dinimaharawati; funding acquisition, Kusuma.

Acknowledgments

This work is funded and supported by Telkom University, Indonesia.

References

[1] X. Li, X. Shi, Y. Zhao, H. Liang, and Y. Dong, “SVND Enhanced Metaheuristic for Plug-In Hybrid Electric Vehicle Routing Problem”, *Applied Sciences*, Vol. 10, No. 2, ID. 441, pp. 1-23, 2020.

[2] P. Antarasee, S. Premrudeepreechacharn, A. Siritaratiwat, and S. Khunkitti, “Optimal Design of Electric Vehicle Fast-Charging Station’s Structure Using Metaheuristic Algorithms”, *Sustainability*, Vol. 15, No. 1, ID. 771, pp. 1-22, 2023.

[3] K. Yenchamchalit, Y. Kongjeen, P. Prabpal, and K. Bhumkittipich, “Optimal Placement of Distributed Photovoltaic Systems and Electric Vehicle Charging Stations Using Metaheuristic Optimization Techniques”, *Symmetry*, Vol. 13, No. 12, ID. 2378, pp. 1-15, 2021.

[4] Y. J. Kim and B. S. Kim, “Population-Based Meta-Heuristic Algorithms for Integrated Batch Manufacturing and Delivery Scheduling Problem”, *Mathematics*, Vol. 10, No. 21, ID. 4127, pp. 1-22, 2022.

[5] K. Baghizadeh, N. Ebadi, D. Zimon, and L. Jum’a, “Using Four Metaheuristic Algorithms to Reduce Supplier Disruption Risk in a Mathematical Inventory Model for Supplying Spare Parts”, *Mathematics*, Vol. 11, No. 1, ID. 42, pp. 1-19, 2023.

[6] Z. M. Anjum, D. M. Said, M. Y. Hassan, Z. H. Leghari, and G. Sahar, “Parallel Operated Hybrid Arithmetic-Salp Swarm Optimizer for Optimal Allocation of Multiple Distributed Generation Units in Distribution Networks”, *Plos One*, Vol. 17, No. 4, ID. e0264958, pp. 1-38, 2022.

[7] T. Lei, S. Riaz, H. Raziq, M. Batool, F. Pan, and J. Wang, “A Comparison of Metaheuristic Techniques for Solving Optimal Sitting and Sizing Problems of Capacitor Banks to Reduce the Power Loss in Radial Distribution System”, *Complexity*, Vol. 2022, ID. 4547212, pp. 1-14, 2022.

[8] T. L. Duong, M. Q. Duong, V. D. Phan, and T. T. Nguyen, “Optimal Reactive Power Flow for Large-Scale Power Systems Using an Effective Metaheuristic Algorithm”, *Journal of Electrical and Computer Engineering*, Vol. 2020, ID. 6382507, pp. 1-11, 2020.

[9] H. Jia, X. Peng, and C. Lang, "Remora Optimization Algorithm", *Expert Systems with Applications*, Vol. 185, ID. 115665, pp. 1-25, 2021.

[10] J. Swan, S. Adriaensen, A. E. I. Brownlee, K. Hammond, C. G. Johnson, A. Kheiri, F. Krawiec, J. J. Merelo, L. L. Minku, E. Ozcan, G. L. Pappa, P. G. Sanchez, K. Sorensen, S. Vob, M. Wagner, and D. R. White, "Metaheuristics in the Large", *European Journal of Operational Research*, Vol. 297, pp. 393-406, 2022.

[11] M. Braik, A. Hammouri, J. Atwan, M. A. A. Betar, and M. A. Awadallah, "White Shark Optimizer: A Novel Bio-Inspired Meta-Heuristic Algorithm for Global Optimization Problems", *Knowledge-Based Systems*, Vol. 243, ID. 108457, pp. 1-29, 2022.

[12] L. Abualigah, M. A. Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, "Reptile Search Algorithm (RSA): A Nature-Inspired Meta-Heuristic Optimizer", *Expert Systems with Applications*, Vol. 191, ID. 116158, pp. 1-33, 2022.

[13] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine Predators Algorithm: A Nature-inspired Metaheuristic", *Expert System with Applications*, Vol. 152, ID: 113377, 2020.

[14] M. A. Akbari, M. Zare, R. A. Abarghooee, S. Mirjalili, and M. Deriche, "The Cheetah Optimizer: A Nature-inspired Metaheuristic Algorithm for Large-scale Optimization Problems", *Scientific Reports*, Vol. 12, ID. 10953, pp. 1-20, 2022.

[15] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer", *Advances in Engineering Software*, Vol. 69, pp. 46–61, 2014.

[16] E. Trojovska and M. Dehghani, "Clouded Leopard Optimization: A New Nature-Inspired Optimization Algorithm", *IEEE Access*, Vol. 10, pp. 102876-102906, 2022.

[17] P. Coufal, S. Hubalovsky, M. Hubalovska, and Z. Balogh, "Snow Leopard Optimization Algorithm: A New Nature-Based Optimization Algorithm for Solving Optimization Problems", *Mathematics*, Vol. 9, ID. 2832, pp. 1-26, 2021.

[18] Suyanto, A. A. Ariyanto, and A. F. Ariyanto, "Komodo Mlipir Algorithm", *Applied Soft Computing*, Vol. 114, ID: 108043, 2022.

[19] 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.

[20] S. Arora and S. Singh, "Butterfly Optimization Algorithm: A Novel Approach for Global Optimization", *Soft Computing*, Vol. 23, No. 3, pp. 715–734, 2019.

[21] D. Polap and M. Wozniak, "Red Fox Optimization Algorithm", *Expert Systems with Applications*, Vol. 166, ID. 114107, pp. 1-21, 2021.

[22] F. A. Zeidabadi and M. Dehghani, "POA: Puzzle Optimization Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 1, pp. 273–281, 2022, doi: 10.22266/ijies2022.0228.25.

[23] S. A. Doumari, H. Givi, M. Dehghani, and O. P. Malik, "Ring Toss Game-Based Optimization Algorithm for Solving Various Optimization Problems", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 545–554, 2021, doi: 10.22266/ijies2021.0630.46.

[24] F. Zeidabadi, S. Doumari, M. Dehghani, and O. P. Malik, "MLBO: Mixed Leader Based Optimizer for Solving Optimization Problems", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 472–479, 2021, doi: 10.22266/ijies2021.0831.41.

[25] F. A. Zeidabadi, M. Dehghani, and O. P. Malik, "RSLBO: Random Selected Leader Based Optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 5, pp. 529–538, 2021, doi: 10.22266/ijies2021.1031.46.

[26] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "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, ID: e910, pp. 1-29, 2022.

[27] 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.

[28] 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, 2023, doi: 10.22266/ijies2023.0228.20.

[29] P. D. Kusuma and A. Dinimaharawati, "Three on Three Optimizer: A New Metaheuristic with Three Guided Searches and Three Random Searches", *International Journal of Advanced*

Science and Applications, Vol. 14, No. 1, pp. 420-429, 2023.

[30] M. Dehghani, Z. Montazeri, E. Trojovska, and P. Trojovsky, "Coati Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems", *Knowledge-Based Systems*, Vol. 259, ID. 110011, pp. 1-43, 2023.

[31] S. A. Yasear and H. M. A. Ghanimi, "A Modified Honey Badger Algorithm for Solving Optimal Power Flow Optimization Problem", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 4, pp. 142–155, 2022, doi: 10.22266/ijies2022.0831.14.

[32] 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.

[33] X. Yu and X. Wu, "Ensemble Grey Wolf Optimizer and Its Application for Image Segmentation", *Expert Systems with Applications*, Vol. 209, ID. 118267, pp. 1-17, 2022.

[34] Preeti and K. Deep, "A Random Walk Grey Wolf Optimizer Based on Dispersion Factor for Feature Selection on Chronic Disease Prediction", *Expert Systems with Applications*, Vol. 206, ID. 117864, pp. 1-17, 2022.