Run-Catch Optimizer: A New Metaheuristic and Its Application to Address Outsourcing Optimization Problem

Purba Daru Kusuma, *Member IAENG*, Fussy Mentari Dirgantara

***Abstract*—This study designs a new stochastic optimization i.e., metaheuristic technique, namely run-catch optimizer (RCO). RCO provides a distinct mechanism regarding the diversification-intensification strategy. Each member runs two sequential activities in every iteration. The first activity is running and the second one is catching. Each activity generates a seed. In the first activity, a virtual best member moves away from the corresponding member to become the first seed. The second seed is generated along the way between the corresponding member and the first seed in the first activity. If both seeds fail to improve, the member conducts a random search to find a new member. Otherwise, the better seed replaces the corresponding member. Then, RCO is challenged to handle both theoretical and real-world optimization problems. The classic 23 functions represent theoretical problems, while the outsourcing optimization problem represents the practical problem. In these simulations, RCO is confronted with five other algorithms: grey wolf optimizer (GWO), particle swarm optimization (PSO), marine predator algorithm (MPA), Komodo mlipir algorithm (KMA), and pelican optimization algorithm (POA). The result shows that RCO is better than POA, KMA, MPA, GWO and PSO in optimizing 20, 21, 13, 12, and 21 functions consecutively. Meanwhile, RCO is better than PSO, GWO, and KMA, but worse than MPA and POA in optimizing the outsourcing problem.**

***Index Terms*—optimization, metaheuristic, swarm intelligence, outsourcing.**

I. Introduction

METAHEURISTIC is a well-known technique widely used in a lot of optimization works. This algorithm has been applied in many areas. In transportation, metaheuristic algorithm is popular to address various vehicle routing problems, for example the capacitated vehicle routing problem [1], vehicle problem with time windows and split delivery [2], vehicle dispatching problem [3], and so on. In the logistic area, it has been applied in addressing various truck scheduling [4], cross docking system [5], and so on. In production and manufacturing, it has been applied in various flow-shop scheduling [6], job-shop scheduling [7], inventory management [8], and so on. In the education area, it also has been applied in addressing course timetabling [9]. In the energy management system, metaheuristic has been applied to optimize the operation of power system [10]. Its popularity comes from two circumstances. First, the algorithm is flexible in addressing various optimization problems with the given computational resource. Second, there are hundreds of metaheuristic algorithms that be chosen and combined to address any optimization problem.

The flexibility of metaheuristic in addressing various problems comes from its approximate approach. By using the approximate approach, it does not trace all available or possible solutions to find the global optimal solution [11]. It conducts a stochastic search within space. Based on this circumstance, the metaheuristic method does not guarantee in finding the global optimal but put the best effort to find the acceptable solution i.e., quasi-optimal solution [11]. It is different from the exact method that ensures the global optimal solution. However, the exact method is impossible to be used in addressing a complex optimization problem with high dimensions and ample space because it needs excessive computational resources [11]. A metaheuristic generally contains two phases: initialization and iteration. A solution is generated stochastically within the space during the initialization. Then, this solution is improved during the iteration phase. Metaheuristic performs two strategies in improving the solution: intensification and diversification. In intensification, a new solution is generated near the current member. In diversification, a new solution is generated randomly within the space.

To the recent day, there are a vast number of metaheuristic methods available to select for any optimization problems. Many optimizations still use the classic algorithms, such as genetic algorithm (GA), tabu search (TS), harmony search (HS), simulated annealing (SA), and so on. These algorithms are still popular because they have simple mechanics so that they can be easily modified and improved. GA used two methods: crossover and mutation. Crossover is conducted by combining two selected current solutions to produce new members [12]. Mutation is conducted by modifying the selected current solutions [12]. TS implements neighborhood search and uses a list or memory to avoid revisiting the latest members [13]. There are two options to generate a new member in HS. First, a solution can be generated from the

Manuscript received March 21, 2023; revised July 25, 2023. This work was financially supported by Telkom University, Indonesia.

Purba Daru Kusuma is an assistant professor in computer engineering, at Telkom University, Indonesia (e-mail: purbodaru@telkomuniversity.ac.id).

Fussy Mentari Dirgantara is a lecturer in computer engineering, Telkom University, Indonesia (e-mail: fussymentari@telkomuniversity.ac.id)

harmony memory [14]. Second, a solution can be generated randomly from the space [14]. SA is a local search-based algorithm. It generates a new solution near the current solution [15]. If this new solution is better than the current solution, this solution is accepted immediately to replace the current solution [15]. Otherwise, a specific stochastic process is conducted to determine whether this new solution is accepted to replace the current solution [15].

Many new algorithms adopt nature's mechanics, especially animals. Several algorithms mimic the behavior of animals during foraging or finding food sources, such as grey wolf optimizer (GWO) [16], pelican optimization algorithm (POA) [17], marine predator algorithm (MPA) [18], capuchin search algorithm (CSA) [19], artificial bee colony (ABC) [20], cat swarm optimization algorithm (CSOA) [21], and so on. Some algorithms, such as emperor penguin colony (EPO) [22], firefly algorithm (FA) [23], and cuckoo search algorithm (CSA) [24], mimic animal movement. Some techniques, like the red deer algorithm (RDA) [25], are inspired by the mating process of the animal. Moreover, some techniques, such as Komodo mlipir algorithm (KMA), combine foraging and mating processes [26]. Other new algorithms are inspired by the mechanics of the traditional game, such as hide object game optimization (HOGO) [27], darts game optimizer (DGO) [28], shell game optimizer (SGO) [29], and football game optimization (FBGO) [30].

This massive number of metaheuristics comes from two reasons. First, various stochastic approaches can be used to develop a new metaheuristic algorithm. Each method has its advantages and disadvantages. Second, the no-free-lunch theory has stated that there is not an algorithm that is suitable and superior for addressing all kinds of problems [31]. On the other hand, there are various optimization problems in the real world because optimization is applied widely by human beings, from individuals to large-scale organizations. Besides, various types of animal behavior have become the inspiration to develop new metaheuristic. Unfortunately, there are critiques regarding the metaphor-based algorithm because of their lack of novel technique and hiding behind the metaphors [32].

One classic problem in metaheuristic algorithms is the adjusted parameters. In general, adjusted parameters are needed to generate a better member. One indispensable adjusted parameter is the maximum iteration. This parameter is essential because all metaheuristic methods rely on iteration process in improving their member. The acceptable member can be found in some problems in the low iteration. On the other hand, high iteration is needed to address a complex problem with high dimensions and ample space. The swarm size is also the indispensable adjusted parameter in the swarm-based metaheuristic algorithm. These multiple members or members that work parallelly in every iteration are needed to diversify the member. Implementing a swarm-based approach aims to speed up the convergence and avoid the local optimal trap. In general, the higher value of these parameters tends to improve the member.

Many metaheuristic algorithms use several adjusted parameters besides swarm size and the maximum iteration. In KMA, the proportion of Komodo types can be adjusted to prioritize the selected strategy [26]. Meanwhile, the mlipir rate can be adjusted to speed up the small male movement [26]. In MPA, the fishing aggregate devices can be adjusted to prioritize the random search selection between choosing from the space or randomly selected from the two current preys [18]. Although GWO does not declare the adjusted parameters, the existence of three leaders becomes the adjusted parameters [16]. The number of leaders can be adjusted from three to another number if it does not surpass the swarm size. The mutation rate can be adjusted in GA to control the diversification [12]. In PSO, the movement weights can be adjusted to prioritize the movement between the current, global, and local best solutions [33]. In TS, the tabu list size can be adjusted to store the number of latest solutions that are forbidden to reuse [13]. The algorithm will produce a high-quality solution when these parameters are correctly adjusted. On the other hand, the wrong adjustment will end in a poor solution.

The second problem is that many metaheuristic algorithms focus on improving the current solution, not the global best solution. The example is as follows. Improvements in GA are conducted by cross-overing the selected current member [12]. In some cases, these mechanics produce a sensible solution. It means that the new solution may be better than the inferior parent but worse than the superior parent. The improvement of the highest quality solution may come from the mutation. However, mutation is not conducted for all solutions. In PSO, the improvement is conducted based on the proportion of the global and local best members [33]. In KMA, the improvement of the high-quality solutions is conducted for the big males by avoiding other big males whose quality is worse [26]. The females' and small males' action does not improve the highest quality member. In MPA, the improvement of the high-quality member is not conducted in all phases of iteration [18]. This circumstance improves the best solution, not fast.

Based on these problems, a new metaheuristic technique, namely run-catch optimizer (RCO), is developed in this work. This algorithm is based on swam intelligence. This technique contains a certain number of members that act autonomously and use the global best member as collective knowledge. The algorithm contains two actions in every iteration: run and catch. In this work, the effort to improve the global best member is conducted in every iteration for all members. It differs from many algorithms where the algorithm focuses on improving the solution's quality, not the global best solution. In this work, the developed algorithm is challenged to overcome both theoretical and real-world optimization problems.

Several contributions regarding this work are as follows.

- In this developed algorithm, the global best member is improved in every iteration by all members.
- The developed algorithm is a metaphor-free algorithm that does not hide behind any nature mechanics.
- This work implements the developed algorithm to optimize the outsourcing problem where this process is important and common in production systems due to the limitation of production capacity.

The rest of this paper is arranged as follows. The second section depicts the developed algorithm's model, which contains three parts: concept, algorithm, and mathematical model. The third section depicts the simulation to assess the performance of the developed algorithm. Then, the fourth

section discusses the findings and deeper analysis. Finally, the fifth section concludes the work and explores the future research potential.

II. Proposed Model

In this section, the model of RCO is exhibited. This presentation is divided into three issues. The first issue is the conceptual model. This issue explains the mechanics of the algorithm, especially the diversification and intensification strategies. It also depicts the reasoning behind this strategy. The second issue is the algorithm. This issue explains the formal structure of the algorithm. This algorithm is explained in pseudocode. The third part is the mathematical model. It explains the detailed formulation and equation in every procedure.

The conceptual model of RCO is as follows. As a swarm-based algorithm, the system contains several members. These members represent a set of solutions. Each member moves within space autonomously to find the global optimal solution. The global best member is a member whose fitness is the best. This global best member is updated every time the member moves to a new member. The last value of the global best member becomes the final member. This also can be seen as a swarm intelligence that adopts animal foraging mechanisms.

The algorithm contains two activities: run and catch. These activities become the reason for the naming of this algorithm. These activities are conducted sequentially for every member in every iteration. Each activity generates a seed. In the first activity, the global best member runs away from the member's current location to find a new location. This new location becomes the first seed. In the second activity, the member tries to catch the first seed by moving randomly within the space between the member's current seed and the first seed. Then, there are three possible new locations for the members. First, if the member's current location is still equal to or better than the first and second seeds, then the member will move randomly within the space. Otherwise, the member will move to the seed whose fitness score is better. The illustration of these activities is depicted in Fig. 1.

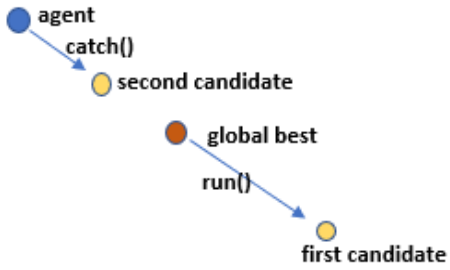

Fig. 1. Run and catch mechanism

The reasoning behind this concept is as follows. In general, the global best member is better than the related member. It means there is a possibility of improvement by moving the global best member to a new location away from the related member. So, the run activity can be seen as an effort to improve the global best member. Then, the second activity is conducted to search for the possibility of improvement within the space between the member's current member and the new location generated in the first activity. So, the catch activity can be seen as an effort to improve the current member.

Generally, the first seed tends to be better than the second seed. It usually occurs when this directed movement is in the appropriate direction, but the global optimal has not yet been reached. However, when the first seed jumps over the global optimal, the second seed may be better than the first seed. Based on this circumstance, the better seed should be chosen as the new member.

This run and catch activities may fail to improve the current member. There are two possibilities related to this circumstance. First, the global optimal has been reached. Second, there is more than one optimal member, for example, in multimodal problems. In this context, the system may be trapped in the local optimal. This problem is overcome by conducting a random search within space.

The diversification-intensification strategy of RCO can be explained as follows. The random search within space represents the diversification strategy. Meanwhile, the directed movement for both run and catch can be either diversification or intensification. The movement can represent diversification when the current member is far from the global best member. Meanwhile, when the current member is near the global best member, the movement can be seen as an intensification.

Before the further explanation, there are several annotations used in RCO. These annotations are as follows. Meanwhile, the algorithm of RCO is depicted in algorithm 1.

b_l, b_u	lower boundary, upper boundary
c_1	first seed
c_2	second seed
d	dimension of the problems
f	fitness function
x	member
X	set of members
x_{best}	global best members
t	iteration
t_{max}	maximum iteration
U	uniform random

The explanation of algorithm 1 is as follows. Line 1 states that the best member becomes the final member. Lines 5 and 6 depict the initialization step. Lines 11 to 14 depict the iteration step. Line 9 states that the iteration is conducted from the first iteration until maximum iteration is reached. Line 10 states that all members are updated in every iteration. Lines 11 and 12 depict the seed's generation. Line 14 states that the global best member is updated after every member is updated.

The third part is the mathematical model. The mathematical model of RCO is very simple. It consists of only five equations.

$$x = U(b_l, b_u) \tag{1}$$

$$x'_{best} = \begin{cases} x, f(x) < f(x_{best}) \\ x_{best}, else \end{cases} \tag{2}$$

algorithm 1: RCO algorithm

output: x_{best}
begin
//initialization
  for all x **do**
    set initial member using (1)
    update x_{best} using (2)
  end
  //iteration
  for t=1 to t_{max} **do**
    for all x **do**
      generate c_1 using (3)
      generate c_2 using (4)
      set new member for x using (5)
      update x_{best} using (2)
    end for
  end for
end

$$c_1 = x_{best} + U(0,1).(x_{best} - x) \quad (3)$$

$$c_2 = x + U(0,1).(c_1 - x) \quad (4)$$

$$x' = \begin{cases} c_1, f(c_1) < f(c_2) \wedge f(c_1) < f(x) \\ c_2, f(c_2) < f(c_1) \wedge f(c_2) < f(x) \\ x, f(x) \leq f(c_1) \wedge f(x) \leq f(c_2) \end{cases} \quad (5)$$

The explanation of these equations is as follows. Equation (1) is used to generate the initial member. It states that this initial member is generated randomly within the space. Then, (2) is used for the global best updating process. It states that the new member replaces the global best member only if it is better than the global best member. Equation (3) states that the first seed is generated randomly within the space between the global best member and its improvement point based on the current member. Equation (4) states that the second seed is generated randomly between the current member and the first seed. Equation (5) states that there are three possible new members for the replacement of the current member: the first seed, the second seed, and a randomized member within the space. This decision is taken based on the quality of the first and second seeds, compared to the current member.

III. Simulation and Result

RCO is then utilized in the simulation so that its performance can be assessed. This work carries out four simulations. In the first simulation, RCO is challenged to overcome the theoretical optimization problem. In the second simulation, the convergence of RCO is assessed. In the third simulation, the relation between the population size and the performance of RCO is assessed. In the fourth simulation, RCO is challenged to optimize a real-case problem.

The first simulation is carried out to assess the performance of RCO in optimizing the theoretical problem. This work uses the well-known 23 functions. The twenty-three functions can be classified into three groups. The first group contains seven high dimension unimodal functions. The second group contains six high dimension multimodal functions. The third group contains ten fixed dimension multimodal functions. The detailed description related to these functions is exhibited in Table 1. Functions 1 to 7 are the high dimension unimodal functions. Functions 8 to 13 are high dimension multimodal functions. Functions 14 to 23 are the fixed dimension multimodal functions. In this work, the dimension for high dimension functions is set to 30, which represents a high dimension problem.

TABLE I
FUNCTIONS

No	Function	Dim	Space	Target
1	Sphere	30	[-100, 100]	0
2	Schwefel 2.22	30	[-100, 100]	0
3	Schwefel 1.2	30	[-100, 100]	0
4	Schwefel 2.21	30	[-100, 100]	0
5	Rosenbrock	30	[-30, 30]	0
6	Step	30	[-100, 100]	0
7	Quartic	30	[-1.28, 1.28]	0
8	Schwefel	30	[-500, 500]	-418.9 x *dim*
9	Ratsrigin	30	[-5.12, 5.12]	0
10	Ackley	30	[-32, 32]	0
11	Griewank	30	[-600, 600]	0
12	Penalized	30	[-50, 50]	0
13	Penalized 2	30	[-50, 50]	0
14	Shekel Foxholes	2	[-65, 65]	1
15	Kowalik	4	[-5, 5]	0.0003
16	Six Hump Camel	2	[-5, 5]	-1.0316
17	Branin	2	[-5, 5]	0.398
18	Goldstein-Price	2	[-2, 2]	3
19	Hartman 3	3	[1, 3]	-3.86
20	Hartman 6	6	[0, 1]	-3.32
21	Shekel 5	4	[0, 10]	-10.1532
22	Shekel 7	4	[0, 10]	-10.4028
23	Shekel 10	4	[0, 10]	-10.5363

These 23 functions represent various kinds of optimization problems. The unimodal function is a function that has only one optimal solution. On the other hand, the multimodal function is a function that has more than one optimal solution. One member is the global optimal, while the others are the local optimal. The high dimension function is a function where dimension varies from only two dimensions to a hundred or thousand dimensions. On the other hand, the fixed dimension function is a function in that dimension that is static and cannot be modified. The space of these functions varies from narrow, such as in Quartic Rastrigin, Hartman 3 and Hartman 6, to very wide, such as Schwefel and Griewank.

TABLE II
PARAMETER SETTING

Parameter	Value
swarm size	20
maximum iteration	100
weights (PSO)	0.1
fishing aggregate devices (MPA)	0.5
big male proportion (KMA)	0.4
number of females (KMA)	1
mlipir rate (KMA)	0.2

This simulation confronts RCO with five metaheuristics: PSO, GWO, MPA, KMA, and POA. The reason for choosing these algorithms is as follows. These five algorithms are also swarm-based metaheuristics that use foraging mechanisms. PSO represents the early version of swarm intelligence. MPA and GWO represent the new techniques used in many studies related to optimization. Meanwhile, KMA and POA represent new metaheuristics, but studies that used these techniques are still rare to find. The adjusted parameters. PSO and MPA represent algorithms with few adjusted parameters. KMA

represents an algorithm with many adjusted parameters. GWO and POA represent algorithms without any explicit adjusted parameters. The parameter setting related to this simulation is exhibited in Table 2. The simulation result is depicted in Table 3 for the average fitness score and Table 4 for the standard deviation. The maximum iteration setting represents the low iteration.

The result in Table 3 shows that RCO is a good metaheuristic algorithm. In general, RCO can find the quasi-optimal solution for the functions. RCO also can avoid the local optimal trap in addressing multimodal functions. Moreover, it found the global optimal in addressing four functions: Schwefel 2.22, six hump camel, Branin, and Goldstein-Price. It is also competitive compared with the sparing algorithms. RCO outperforms PSO, GWO, MPA, KMA, and POA in addressing 21, 12, 13, 21, and 20 functions respectively. Based on this result, GWO becomes the most challenging algorithm to beat. The number of functions beaten by RCO in every group for every algorithm is depicted in Table 5.

Table 5 shows that RCO is superior among sparing metaheuristics in addressing fixed dimension multimodal functions. Its superiority occurs for all sparing algorithms. Meanwhile, RCO is superior to PSO, KMA, and POA but inferior to GWO and MPA in addressing high dimension unimodal functions. This circumstance also occurs in addressing the high dimension multimodal functions.

TABLE III
SIMULATION RESULT (AVERAGE FITNESS SCORE)

Function	PSO	GWO	MPA	KMA	POA	RCO	Better Than
1	$6.227x10^{3}$	$1.136x10^{-2}$	$5.812x10^{2}$	$6.404x10^{3}$	$3.167x10^{4}$	$3.743x10^{-2}$	PSO, MPA, KMA, POA
2	0	0	0	$6.975x10^{22}$	0	0	KMA
3	$1.773x10^{4}$	$3.137x10^{-1}$	$2.806x10^{3}$	$2.336x10^{4}$	$5.803x10^{4}$	$8.426x10^{3}$	PSO, KMA, POA
4	$2.716x10^{1}$	$2.126x10^{-3}$	2.201	$3.276x10^{1}$	$6.451x10^{1}$	$3.057x10^{1}$	KMA, POA
5	$1.091x10^{6}$	$4.761x10^{2}$	$2.695x10^{2}$	$3.776x10^{6}$	$6.321x10^{7}$	$8.083x10^{2}$	PSO, KMA, POA
6	$5.068x10^{3}$	7.259	$6.578x10^{2}$	$5.544x10^{3}$	$3.187x10^{4}$	$5.170x10^{-2}$	PSO, GWO, MPA, KMA, POA
7	$8.924x10^{-1}$	$3.653x10^{-2}$	$1.293x10^{-1}$	1.823	$3.032x10^{1}$	$2.425x10^{-1}$	PSO, KMA, POA
8	$-2.608x10^{3}$	$6.763x10^{-2}$	$-3.328x10^{3}$	$-8.138x10^{3}$	$-3.438x10^{3}$	$-7.707x10^{3}$	PSO, GWO, MPA, POA
9	$2.316x10^{2}$	$1.245x10^{-2}$	$1.229x10^{2}$	$2.116x10^{2}$	$3.257x10^{2}$	$1.274x10^{2}$	PSO, KMA, POA
10	$1.259x10^{1}$	$2.139x10^{-3}$	6.845	$1.373x10^{1}$	$1.932x10^{1}$	9.441	PSO, KMA, POA
11	$5.548x10^{1}$	$4.356x10^{-3}$	6.632	$6.494x10^{1}$	$2.906x10^{2}$	$1.684x10^{-1}$	PSO, MPA, KMA, POA
12	$8.473x10^{4}$	1.663	6.505	$3.799x10^{5}$	$7.583x10^{7}$	9.349	PSO, KMA, POA
13	$1.640x10^{6}$	$5.719x10^{-2}$	$4.779x10^{2}$	$6.548x10^{6}$	$1.999x10^{8}$	$2.249x10^{1}$	PSO, MPA, KMA, POA
14	4.837	$1.267x10^{1}$	3.198	9.718	1.771	1.045	PSO, GWO, MPA, KMA, POA
15	$3.086x10^{-2}$	$1.484x10^{-1}$	$4.591x10^{-3}$	$1.548x10^{-2}$	$2.102x10^{-3}$	$6.301x10^{-3}$	PSO, GWO, KMA
16	-1.029	$7.932x10^{-18}$	-1.026	$-9.786x10^{-1}$	-1.029	-1.032	PSO, GWO, MPA, KMA, POA
17	$9.032x10^{-1}$	$5.560x10^{1}$	$7.095x10^{-1}$	$8.713x10^{-1}$	$4.042x10^{-1}$	$3.981x10^{-1}$	PSO, GWO, MPA, KMA, POA
18	4.283	$6.000x10^{2}$	4.647	3.127	3.095	3.000	PSO, GWO, MPA, KMA, POA
19	$-2.163x10^{-3}$	$-7.520x10^{-4}$	-3.733	$-9.534x10^{-1}$	$-4.954x10^{-2}$	$-4.524x10^{-2}$	PSO, GWO, KMA
20	-2.562	$-5.089x10^{-3}$	-2.015	-2.797	-2.968	-3.275	PSO, GWO, MPA, KMA, POA
21	-4.952	$-2.731x10^{-1}$	-1.791	-5.281	-3.169	-5.713	PSO, GWO, MPA, KMA, POA
22	-4.007	$-2.936x10^{-1}$	-1.730	-4.474	-4.020	-6.683	PSO, GWO, MPA, KMA, POA
23	-4.268	$-3.217x10^{-1}$	-2.064	-4.772	-3.741	-4.386	PSO, GWO, MPA, POA

TABLE IV
SIMULATION RESULT (STANDARD DEVIATION)

Function	PSO	GWO	MPA	KMA	POA	RCO
1	$1.890x10^{3}$	$4.216x10^{-2}$	$3.049x10^{2}$	$1.949x10^{3}$	$4.722x10^{3}$	$3.373x10^{-2}$
2	0	0	0	$2.507x10^{23}$	0	0
3	$3.621x10^{3}$	$9.224x10^{-1}$	$1.387x10^{3}$	$2.591x10^{4}$	$1.376x10^{4}$	$6.959x10^{3}$
4	5.549	$3.984x10^{-3}$	1.528	5.596	4.667	8.567
5	$4.200x10^{5}$	$1.785x10^{3}$	$3.322x10^{2}$	$2.582x10^{6}$	$2.426x10^{7}$	$1.021x10^{-3}$
6	$1.520x10^{3}$	$1.677x10^{-2}$	$3.058x10^{2}$	$1.749x10^{3}$	$4.497x10^{3}$	$1.510x10^{-1}$
7	$4.757x10^{-1}$	$2.127x10^{-2}$	$5.068x10^{-2}$	$7.460x10^{-1}$	9.782	$1.084x10^{-1}$
8	$4.164x10^{2}$	$1.987x10^{-1}$	$3.398x10^{2}$	$1.313x10^{3}$	$3.512x10^{2}$	$6.780x10^{2}$
9	$3.151x10^{1}$	$5.114x10^{-2}$	$3.089x10^{1}$	$2.637x10^{1}$	$2.954x10^{1}$	$2.891x10^{1}$
10	1.222	$3.727x10^{-3}$	$9.511x10^{-1}$	$9.479x10^{-1}$	$6.923x10^{-1}$	3.996
11	$1.451x10^{1}$	$1.253x10^{-2}$	2.461	$1.840x10^{1}$	$3.276x10^{1}$	$2.364x10^{-1}$
12	$1.858x10^{5}$	$1.601x10^{-2}$	3.349	$4.912x10^{5}$	$6.105x10^{7}$	3.977
13	$1.342x10^{6}$	$2.275x10^{3}$	$1.013x10^{3}$	$4.986x10^{6}$	$8.221x10^{7}$	$1.298x10^{1}$
14	2.955	$3.669x10^{-15}$	1.744	7.566	1.531	$2.169x10^{-1}$
15	$3.532x10^{-2}$	$8.420x10^{-8}$	$2.365x10^{-3}$	$1.499x10^{-2}$	$3.248x10^{-4}$	$8.850x10^{-3}$
16	$7.028x10^{-3}$	$3.073x10^{-17}$	$5.657x10^{-3}$	$1.658x10^{-1}$	$1.420x10^{-3}$	$2.273x10^{-16}$
17	$8.954x10^{-1}$	$1.877x10^{-5}$	$2.274x10^{-1}$	1.084	$7.197x10^{-3}$	$1.138x10^{-16}$
18	3.980	$1.406x10^{-4}$	1.535	$2.609x10^{-1}$	$1.274x10^{-1}$	0
19	$2.911x10^{-3}$	$1.869x10^{-3}$	$1.399x10^{-1}$	1.229	$1.433x10^{-17}$	$1.265x10^{-2}$
20	$3.863x10^{-1}$	$8.978x10^{-19}$	$3.067x10^{-1}$	$4.980x10^{-1}$	$1.098x10^{-1}$	$5.992x10^{-2}$
21	3.239	$5.733x10^{-17}$	$7.169x10^{-1}$	2.898	$8.700x10^{-1}$	3.529
22	2.836	0	$6.497x10^{-1}$	2.677	1.617	3.563
23	2.333	0	$8.039x10^{-1}$	2.256	1.759	3.057

The second simulation is taken to assess the convergence of RCO, especially in the low iteration. RCO is still challenged in this simulation to overcome the 23 functions. Meanwhile, there are three maximum iteration values in this simulation: 25, 50, and 75. The result is depicted in Table 6.

TABLE V
NUMBER OF FUNCTIONS BEATEN IN EVERY GROUP

Algorithm	Number of Functions		
	In 1st Group	In 2nd Group	In 3rd Group
PSO	5	6	10
GWO	1	1	10
MPA	2	3	8
KMA	7	5	9
POA	6	6	8

TABLE VI
CONVERGENCE SIMULATION RESULT

Function	Average Fitness Score		
	t_{max} = 25	t_{max} = 50	t_{max} = 75
1	$7.985x10^{2}$	$2.641x10^{1}$	$8.891x10^{-1}$
2	0	0	0
3	$2.033x10^{4}$	$1.343x10^{4}$	$8.820x10^{3}$
4	$4.749x10^{1}$	$4.041x10^{1}$	$3.643x10^{1}$
5	$1.994x10^{5}$	$8.531x10^{3}$	$5.283x10^{2}$
6	$9.235x10^{2}$	$1.351x10^{1}$	$9.325x10^{-1}$
7	1.126	$5.386x10^{-1}$	$2.976x10^{-1}$
8	$-7.316x10^{3}$	$-7.581x10^{3}$	$-7.665x10^{3}$
9	$1.511x10^{2}$	$1.229x10^{2}$	$1.408x10^{2}$
10	$1.247x10^{1}$	9.989	$1.186x10^{1}$
11	9.982	1.202	$3.434x10^{-1}$
12	$8.767x10^{3}$	$2.043x10^{1}$	$1.083x10^{1}$
13	$1.098x10^{5}$	$6.388x10^{1}$	$3.627x10^{1}$
14	5.164	1.5804	1.369
15	$9.221x10^{-3}$	$3.447x10^{-3}$	$8.036x10^{-3}$
16	-1.032	-1.032	-1.032
17	$3.981x10^{-1}$	$3.981x10^{-1}$	$3.981x10^{-1}$
18	3.000	3.000	3.000
19	$-4.954x10^{-2}$	$-4.758x10^{-2}$	$4.695x10^{-2}$
20	-3.259	-3.245	-3.282
21	-4.884	-5.676	-5.280
22	-3.701	-4.572	-4.131
23	-5.211	-4.596	-4.728

Table 6 shows that RCO performs well in the convergence aspect. There are 14 functions that the stable result is achieved in the very low iteration. Most of them are multimodal functions. Meanwhile, there are five unimodal functions that the convergence has not been reached when the maximum iteration is set at 75.

The third simulation is carried out to assess the relation between the population and the performance of RCO. Its performance is measured based on the average fitness score. Like in the first and second simulations, the 23 functions are used as the problem. In this simulation, there are three values of population size which are 5, 15, and 15. All these values represent very low population size because they are still below the standard value in the first simulation. Meanwhile, the maximum iteration is set to 100. The result is exhibited in Table 7.

Table 7 indicates the variety of responses due to the increase of population size. In general, in the first group of functions, the increase of swarm size improves the performance, except in Schwefel 2.22. In Schwefel 2.22, the global optimal solution was achieved when the population size was still 5. In the second group of functions, the significant improvement is achieved only in three functions (Griewank, Penalized, and Penalized 2). In the third function, significant improvement is achieved only in Shekel Foxholes. Meanwhile, in three functions (Six Hump Camel, Branin, and Goldstein Price), there is not any significant improvement because the global optimal solution has been achieved.

TABLE VII
RELATION BETWEEN SWARM SIZE AND AVERAGE FITNESS SCORE

Function	Average Fitness Score		
	$n(X)$ = 5	$n(X)$ = 10	$n(X)$ = 15
1	$2.711x10^{3}$	$2.284x10^{1}$	$9.516x10^{-1}$
2	0.000	0.000	0.000
3	$3.085x10^{4}$	$1.585x10^{4}$	$1.123x10^{4}$
4	$6.071x10^{1}$	$4.773x10^{1}$	$3.739x10^{1}$
5	$8.936x10^{5}$	$8.649x10^{3}$	$7.047x10^{2}$
6	$1.976x10^{3}$	$2.305x10^{1}$	$5.036x10^{-1}$
7	1.022	$5.285x10^{-1}$	$2.796x10^{-1}$
8	$-6.444x10^{3}$	$-7.486x10^{3}$	$-7.353x10^{3}$
9	$2.087x10^{2}$	$1.505x10^{2}$	$1.347x10^{2}$
10	$1.671x10^{1}$	$1.210x10^{1}$	9.932
11	$2.467x10^{1}$	1.221	$2.120x10^{-1}$
12	$1.190x10^{5}$	$2.093x10^{1}$	$1.181x10^{1}$
13	$1.804x10^{6}$	$6.575x10^{1}$	$3.523x10^{1}$
14	7.128	2.156	1.329
15	$1.487x10^{-2}$	$6.737x10^{-3}$	$6.340x10^{-3}$
16	-1.032	-1.032	-1.032
17	$3.981x10^{-1}$	$3.981x10^{-1}$	$3.981x10^{-1}$
18	5.250	3.000	3.000
19	$-3.898x10^{-2}$	$-4.309x10^{-2}$	$-4.954x10^{-2}$
20	-3.285	-3.273	-3.254
21	-5.977	-5.563	-6.065
22	-4.199	-5.340	-5.339
23	-4.017	-4.179	-4.961

The fourth simulation is used to assess the performance of RCO in optimizing the real-case problem. RCO is challenged to optimize the outsourcing problem. The outsourcing problem is chosen because this outsourcing strategy is essential and expected in supply chain management, especially in the production system. Outsourcing is important because, in general, every manufacturer's production capacity, skilled workers, and technology resource are limited [34]. Its limitation comes from two circumstances. First, a manufacturer is not able to produce all the product items it offers. This circumstance usually occurs in a manufacturer that sells multiple products. Second, its production capacity is limited, so the ordered quantity surpasses the production capacity. Moreover, in the global supply chain era, outsourcing around the globe has become more strategic [35].

In this work, the scenario is a single product and multiple vendors. This manufacturer is a sock manufacturer. A manufacturer outsources only one product. On the other hand, there are several vendors that can produce this product for the manufacturer. Each vendor has its production capacity. Moreover, each vendor has its product price that can be seen from the manufacturer's perspective as the outsourcing price. Every outsourcing order must be in the economic order quantity (EOQ).

The parameters set regarding this outsourcing optimization problem are as follows. The system contains ten vendors. Five vendors are prominent, while five others are small vendors. The order quantity constraint for the big vendor is from 1,000 to 2,000 dozen for each vendor. The order quantity constraint for the small vendor is from 100 to 500 dozen. The outsourcing cost for the big vendor ranges from 108,000 to 120,000 rupiah per dozen. On the other hand, the outsourcing cost for the small vendor ranges from 96,000

to 108,000 rupiah per dozen. The small vendors can offer lower outsourcing costs because they have not paid the value-added tax, and their labor cost is lower than the big ones. On the other hand, the prominent vendors offer a bigger production capacity. Although the outsourcing cost of the prominent vendors is higher than the small vendors, the manufacturer still needs the prominent vendors because the total outsourcing quantity is much higher than the total production capacity of the small vendors. In this simulation, the total outsourcing quantity is 8,000 dozen. The objective is to minimize the total outsourcing cost, which is obtained by accumulating the outsourcing cost of all vendors.

This simulation confronts the RCO with five algorithms: PSO, MPA, GWO, KMA, and POA. The swarm size is set at 10, and the maximum iteration is set at 50. This setting represents the low swarm and low iteration. The result is exhibited in Table 8.

TABLE VIII
OUTSOURCING SIMULATION RESULT

Algorithm	Total Cost
PSO	Rp 891,993,454
GWO	Rp 938,451,310
MPA	Rp 881,247,272
KMA	Rp 884,176,421
POA	Rp 878,230,000
RCO	Rp 882,671,250

Table 8 shows that RCO is competitive enough among other algorithms to optimize this outsourcing problem. The performance of RCO is better than PSO, GWO, and KMA. Meanwhile, its performance is worse than MPA and POA. Table 7 also shows that the performance gap among these algorithms is tight. RCO is 1.0%, 5.9% and 0.2% better than PSO GWO and KMA. On the other hand, RCO is 0.2% and 0.5% worse than MPA and POA.

IV. DISCUSSION

In this section, a comprehensive analysis, and the findings regarding the simulation result will be discussed. This discussion is also divided into three issues: the general performance in addressing the 23 functions, the convergence, and the performance in addressing the outsourcing optimization problem.

In general, RCO is proven as a good and competitive algorithm for addressing the 23 functions. It can find an acceptable solution for all functions. Most of these solutions are quasi-optimal ones, while four of them are global optimal ones. This achievement occurs in all function groups: the high dimension unimodal functions, high dimension multimodal functions, and fixed dimension multimodal functions. It means that RCO has met the general requirements for any metaheuristic where the acceptable solution should be found when the iteration ends, and the local optimal trap should be avoided. The result also shows that its performance is good in all space sizes.

The result also shows that RCO is a competitive metaheuristic benchmarked with the five sparing algorithms: PSO GWO, MPA, KMA, and POA in addressing the 23 functions. RCO is significantly superior to PSO, KMA, and POA. Meanwhile, MPA and GWO are difficult to beat. RCO is superior to MPA and GWO in addressing fixed dimension functions. Meanwhile, RCO is inferior to MPA and GWO in addressing high dimension functions.

Based on the result related to the convergence aspect, RCO is proven as a metaheuristic that can achieve convergence in the low iteration. It needs a low computational process to address the optimization problem. This performance is achieved primarily by addressing the multimodal functions. It also means that RCO can step away from the local optimal trap and find the area where the global optimal member exists in the early iteration.

Based on the third simulation result, there are variety regarding the response of algorithm regarding the increase of population size. In many unimodal functions, the increase of population size improves performance significantly. On the other hand, in many multimodal functions, the increase of population size does not improve performance significantly.

In regard to the simulation result on addressing the outsourcing optimization problem, RCO is also proven competitive among the sparing algorithms. It is better than three metaheuristics but worse than two metaheuristics. But the performance gap among the algorithms is narrow.

Overall, the simulation result strengthens the no-free-lunch theory that states that there is no algorithm suitable to address all kinds of problems. The performance or quality of a technique is highly related to the problem it tries to address. This circumstance can be seen from the result in the theoretical optimization problem and real-world or practical optimization problem. In general, RCO is superior in addressing the fixed dimension multimodal functions. Its superiority is applied to all sparing algorithms. However, RCO is inferior to GWO and MPA in addressing high dimension functions. The opposite result occurs in the outsourcing optimization problem. POA outperforms RCO in the outsourcing optimization problem and becomes the best metaheuristic in addressing this problem, although POA is inferior in addressing the 23 functions. This circumstance also comes from the different characteristics between the 23 functions and the outsourcing optimization problem. In the 23 functions, the solution is formatted in a high precision floating point. Contrary, in the optimization work on outsourcing problem, the solution is presented in the integer. The opposite result also occurs in GWO. GWO becomes the most challenging metaheuristic to beat in addressing the 23 functions, but its performance is the worst among all metaheuristics in addressing the outsourcing optimization problem.

The computational complexity of RCO can be explained as follows. This complexity can be split into two parts: the initialization and the iteration. During the initialization, the complexity can be presented as $O(n(X).d)$. In the initialization, there are two loops where the loop for whole population is the outer loop while the loop for whole dimensions is the inner loop. Meanwhile, during the iteration, the complexity can be presented as $O(t_{max}.n(X).d)$. In the iteration, there are three loops. Loop until maximum iteration is the outer loop, loop for whole population is the middle loop, and loop for whole dimensions is the inner loop.

There are several considerations regarding the designed RCO. Despite its general outstanding performance relative to its competitors, RCO is still inferior to GWO and MPA in solving high dimension problems. This weakness can be used

as a baseline for future development and improvement. Second, many latest metaheuristics are developed by deploying multiple strategies, such as osprey optimization algorithm (OOA) [36], extended stochastic coati optimization (ESCO) [37], average subtraction-based optimization (ASBO) [38], adaptive balance optimization (ABO) [39], clouded leopard optimization (CLO) [40], and so on. This approach is designed to overcome the weaknesses of a strategy with the strengths of other strategies. Furthermore, there are various practical optimization problems, and their complexity increases as time goes on. These recent and upcoming problems can be used as use-cases for RCO to prove its superiority.

V. Conclusion

This work has presented that the run-catch optimizer (RCO) becomes a good and competitive metaheuristic, although RCO does not have adjusted parameters except the swarm size and the maximum iteration, it meets the general requirements of an acceptable and competitive metaheuristic. First, it can find the acceptable member or solution within the iteration constraint. Second, it can avoid the local optimal trap. The assessment result shows that RCO is competitive in addressing the 23 functions and the outsourcing optimization problem. RCO also can achieve convergence in the constraint of the low iteration in many functions. Most of them are multimodal functions. RCO outperforms PSO, GWO, MPA, KMA, and POA in addressing 21, 12, 13, 21, and 20 functions respectively in addressing the 23 functions. Meanwhile, RCO is better than PSO, GWO, and KMA, but worse than MPA and POA in addressing the outsourcing optimization problem.

This work has exhibited that the designed technique is promising to become the baseline for future improvement and development. RCO should be utilized in many other optimization problems to assess its performance more comprehensively. Future studies can be conducted by combining this technique with other techniques to combine both advantages. Meanwhile, future improvement should not be conducted by adding more adjusted parameters. Moreover, it is also suggested that the improvement can be conducted to make RCO become more adaptive, especially in speeding up the convergence once the area of the global optimal member is found or to make RCO can find the area of the global optimal member exists earlier.

References

[1] Q. Qiao, F. Tao, H. Wu, X. Yu, and M. Zhang, "Optimization of a capacitated vehicle routing problem for sustainable municipal solid waste collection management using the PSO-TS algorithm", *International Journal of Environment Research and Public Health*, vol. 17, ID: 2163, pp. 1-22, 2020.

[2] O. S. Olaniyi, A. K. James, A. A. Ibrahim, and A. F. Makanjuola, "On the application of a modified genetic algorithm for solving vehicle routing problems with time windows and split delivery", *IAENG International Journal of Applied Mathematics*, vol. 52, no. 1, pp. 101-109, 2022.

[3] B. Lv, Y. Tian, and B. Liu, "Airport taxi dispatching based on VISSIM and multi-objective programming model", *American Journal of Computer Sciences and Applications*, vol. 3, no. 23, pp. 1-13, 2020.

[4] J. Yang, B. Yue, F. Feng, J. Shi, H. Zong, J. Ma, L. Shangguan, and S. Li, "Concrete vehicle scheduling based on immune genetic algorithm", *Mathematical Problems in Engineering*, vol. 2022, ID. 4100049, pp. 1-15, 2022.

[5] T. W. Liao, "Integrated inbound vehicle routing and scheduling under a fixed outbound schedule at a multi-door cross-dock terminal", *IEEE Transactions on Intelligent Transportation Systems*, pp. 1-13, 2021, online first.

[6] P. D. Kusuma and A. S. Albana, "A parallel permutation flow-shop scheduling model by using a two-step evolutionary algorithm to minimize intermediate storage with tolerable maximum completion time", *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 6, pp. 464-475, 2021.

[7] X. Luo, Q. Qian, and Y. F. Fu, "Improved genetic algorithm for solving flexible job shop scheduling problem", *Procedia Computer Science*, vol. 166, pp. 480-485, 2020.

[8] S. J. Sajadi and A. Ahmadi, "An integrated optimization model and metaheuristics for assrotnemnt planning, shelf space allocation, and inventory management of perishable products: a real application", *PLoS ONE*, vol. 13, no. 3, ID. e0264186, pp. 1-30, 2022.

[9] J. Xu, "Improved genetic algorithm to solve the scheduling problem of college English course", *Complexity*, vol. 2021, ID. 7252719, pp. 1-11, 2021.

[10] J. Qian and G. Chen, "Improved multi-goal particle swarm optimization algorithm and multi-output BP network for optimal operation of power system", *IAENG International Journal of Applied Mathematics*, vol. 52, no. 3, pp. 576-588, 2022.

[11] H. R. Moshtaghi, A. T. Eshlaghy, and M. R. Motadel, "A comprehensive review on meta-heuristic algorithms and their classification with novel approach", *Journal of Applied Research on Industrial Engineering*, vol. 8, no. 1, pp. 63-89, 2021.

[12] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future", *Multimedia Tools and Applications*, vol. 80, pp. 8091-8126, 2021.

[13] H. Bentsen, A. Hoff, and L. M. Hvattum, "Exponential extrapolation memory for tabu search", *EURO Journal on Computational Optimization*, vol. 10, ID. 100028, pp. 1-21, 2022.

[14] M. Dubey, V. Kumar, M. Kaur and T.-P. Dao, "A systematic review on harmony search algorithm: theory, literature, and applications", *Mathematical Problems in Engineering*, vol. 2021, ID. 5594267, pp. 1-22 2021.

[15] M. Almarashi, W. Deabes, H. H. Amin, and A.-R. Hedar, "Simulated annealing with exploratory sensing for global optimization", *Algorithms*, vol. 13, no. 9, pp. 1-26, 2020.

[16] Y. Hou, H. Gao, Z. Wang, and C. Du, "Improved grey wolf optimization algorithm and application", *Sensors*, vol. 22, ID: 3810, pp. 1-19, 2022.

[17] P. Trojovsky and M. Dehghani, "Pelican optimization algorithm: a novel nature-inspired algorithm for engineering applications", *Sensors*, vol. 22, ID: 855, pp. 1-34, 2022.

[18] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic", *Expert Systems* with Applications, vol. 152, ID: 113377, 2020.

[19] M. Braik, A. Sheta, and H. El-Hiary, "A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm", *Neural Computing and Applications*, vol. 33, pp. 2515-2547, 2021.

[20] S. Xiao, W. Wang, H. Wang, and Z. Huang, "A new multi-objective artificial bee colony algorithm based on reference point and opposition", *International Journal of Bio-inspired Computation*, vol. 19, no. 1, pp. 18-28, 2022.

[21] A. M. Ahmed, T. A. Rashid, and S. A. M. Saeed, "Cat swarm optimization algorithm: a survey and performance evaluation", *Computational Intelligence and Neuroscience*, vol. 2020, ID. 4854895, pp. 1-20, 2020.

[22] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Emperor penguins colony: a new metaheuristic algorithm for optimization", *Evolutionary Intelligence*, vol. 12, pp. 211-226, 2019.

[23] Y. Li Y. Zhao, Y. Shang, and J. Liu, "An improved firefly algorithm with dynamic self-adaptive adjustment", *PLoS ONE*, vol. 16, no. 10 ID. e0255951, pp. 1-24, 2021.

[24] K. Sharma, S. Singh, and R. Doriya, "Optimized cuckoo search algorithm using tournament selection function for robot path planning", *International Journal of Advanced Robotic Systems*, vol. 18, no. 3, pp. 1-11, 2021.

[25] A. M. Fathollahi-Fard, M. Hajjaghaei-Keshteli, and R. Tavakkoli-Moghaddam, "Red deer algorithm (RDA): a new nature-inspired metaheuristic", *Soft Computing*, vol. 24, pp. 14637-14665, 2020.

[26] Suyanto, A. A. Ariyanto, and A. F. Ariyanto, "Komodo mlipir algorithm", *Applied Soft Computing*, vol. 114, pp. 1-17, 2022.

[27] M. Dehghani, Z. Montazeri, S. Saremi, A. Dehghani, O. P. Malik, K. Al-Haddad, and J. M. Guerrero, "HOGO: hide objects game optimization", *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 4, pp. 216-225, 2020.

[28] M. Dehghani, Z. Montazeri, H. Givi, J. M. Guerrero, and G. Dhiman, "Darts game optimizer", *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 286-294, 2020.

[29] M. Dehghani, Z. Montazeri, O. P. Malik, H. Givi, and J. M. Guerrero, "Shell game optimization: a novel game-based algorithm", *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 3, pp. 246-255, 2020.

[30] M. Dehghani, M. Mardaneh, J. S. Guerrero O. P. 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.

[31] S. P. Adam, S. A. N. Alexandropoulos, P. M. Pardalos, and M. N. Vrahatis, "No free lunch theorem: a review", *Approximation and Optimization, Springer Optimization and Its Applications*, vol. 145, pp. 57-82, 2019.

[32] 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. Garcia-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.

[33] D. Freitas, L. G. Lopes, and F. Morgado-Dias, "Particle swarm optimization: a historical review up to the current developments", *Entropy*, vol. 22, pp. 1-36, 2020.

[34] M. Alkahtani, "Mathematical modelling of inventory and process outsourcing for optimization of supply chain management" Mathematics, vol. 10, ID. 1142, pp. 1-27, 2022.

[35] Y. Uygun, N. Gotsadze, F. Schupp, L. Gzirishvili, and B. S. T. Nana, "A holistic model for understanding the dynamics of outsourcing", *International Journal of Production Research*, vol. 61, no. 4, pp. 1202-1232, 2023.

[36] M. Dehghani and P. Trojovsky, "Osprey optimization algorithm: a new bio-inspired metaheuristic algorithm for solving engineering optimization problems", *Frontiers in Mechanical Engineering*, vol. 8, ID. 1126450, pp. 1-43, 2023.

[37] P. D. Kusuma and A. Dinimaharawati, "Extended stochastic coati optimizer", *International Journal of Intelligent Engineering and Systems*, vol. 16, no. 3, pp. 482-494, 2023.

[38] 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.

[39] P. D. Kusuma and A. Dinimaharawati, "Adaptive balance optimizer: a new adaptive metaheuristic and its application in solving optimization problem in finance", *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 4, pp. 130-142, 2023.

[40] E. Trojovska and M. Dehghani, "Clouded leopard optimization: a new nature-inspired optimization algorithm", *IEEE Access*, vol. 10, pp. 102876-102906, 2022.

Purba Daru Kusuma is an assistant professor in computer engineering in Telkom University, Indonesia. He received his bachelor's and master's degrees in electrical engineering from Bandung Institute of Technology, Indonesia. He received his doctoral degree in computer science from Gadjah Mada University, Indonesia. His research interests are in artificial intelligence, machine learning, and operational research. He currently becomes a member of IAENG.

Fussy Mentari Dirgantara is a lecturer in computer engineering in Telkom University, Indonesia. She received her bachelor's degree in electrical engineering from Telkom University, Indonesia. Then, she received her master's degree in electrical engineering from Bandung Institute of Technology, Indonesia. Her research interests are in control systems, internet of things, and intelligent system.