International Journal on Advanced Science Engineering Information Technology

Vol.13 (2023) No. 5
ISSN: 2088-5334

Adaptive Cone Algorithm

Purba Daru Kusuma [a,*], Meta Kallista [a]

[a] *Computer Engineering, Telkom University, Buah Batu Street, Bandung, 40238, Indonesia*
Corresponding author: [*]purbodaru@telkomuniversity.ac.id

***Abstract*—This study was conducted to promote a new adaptive cone algorithm (ACA) algorithm. ACA is a metaheuristic technique based on swarm intelligence. ACA contains three steps. Each agent moves closer to the global reference in the first step. Then, each agent searches for a better solution around the current solution in the second step. The global reference searches for better solutions around it in the third step. This algorithm is named cone because the local space size declines linearly during the iterative process. ACA introduces a new adaptability model to improve the exploration strategy when a better solution cannot be achieved. It is conducted by enlarging the local solution space. ACA is challenged to find the final solution for theoretical and practical problems. The 23 functions are chosen as theoretical optimization problems. The portfolio optimization problem is selected as the practical problem. ACA is compared with five algorithms: particle swarm optimization (PSO), grey wolf optimizer (GWO), marine predator optimization (MPA), average subtraction-based optimizer (ASBO), and pelican optimization algorithm (POA). The result shows that ACA is competitive in finding the optimal solution for 23 functions and outperforms all sparing algorithms in achieving the highest total capital gain in tackling the portfolio optimization problem. ACA is superior to PSO, GWO, MPA, ASBO, and POA in solving 20, 11, 13, 4, and 21 functions, respectively. In the future, ACA can be implemented in solving various practical optimization problems.**

***Keywords*— Metaheuristic; swarm intelligence; adaptive method; portfolio optimization problem; IDX30 index.**

Manuscript received 1 Sep. 2022; revised 3 Jun. 2021; accepted 6 Jul. 2023. Date of publication 31 Oct. 2023.

I. INTRODUCTION

Optimization is a popular subject that is widely studied. This popularity comes from its nature. Optimization is utilized to find the most appropriate solution within the constraints. It is like the human process, whether personal or institutional. For example, a fund manager is challenged to arrange his clients' limited investments to meet several objectives, such as maximizing the risk-adjusted return [1], preventing too much risk [2], and so on. A factory manager must manage the incoming orders to minimize the total completion time [3], idle energy consumption [4], total actual flow time [5], maximum completion time [6], and so on. A university or school manager must manage the facilities, especially rooms and lecturers, to achieve some objectives, such as minimizing the unserved participants [7], minimizing the overcapacity of classrooms [8], or minimizing idle time [9]. A fleet manager must arrange his vehicles to execute all pickup and delivery orders for some objectives, such as minimizing the cost [10], transshipment [11], number of vehicles [12], travel distance [12], distribution cost [13], logistic cost [14], and so on.

Metaheuristic algorithms have become the popular methods used in many optimization problems. This popularity comes from its flexibility to be implemented in various optimization problems. They also tackle the limited computational resource because it uses an approximate approach [15]. By deploying an approximate approach, it does not trace all available solutions. The consequence is that they cannot ensure the global optimal solution. These algorithms try to find the sub-optimal solution [15]. Today, many algorithms can be chosen to solve an optimization problem.

Despite its popularity, several notes regarding the development or implementation of metaheuristic algorithms exist. First, many latest algorithms used metaphors as a claimed novelty rather than proposed a significant improvement. Second, there is room for adjustment in metaheuristic algorithms. This adjustment improves its performance depending on the problem it tries to overcome. Proper adjustment can improve the result. However, misjudgment may worsen the result. Third, most metaheuristic algorithms are adjusted manually before the optimization begins. Because the adjusted parameters remain static along the iteration process, this algorithm cannot change

the strategy when it faces several circumstances, for example, when it cannot improve its current solution. The following paragraph will explain A more detailed review of these issues. Fourth, engineering problems have become the most popular practical use cases in many studies proposing new metaheuristics. Meanwhile, studies choosing non-engineering problems as their practical optimization problem are hard to find.

Many metaheuristics are built based on the nature mechanism. Most of them mimic the behavior of animals during foraging (searching for food or hunting prey) and mating. This circumstance is inevitable because the nature of the foraging and mating process is like the metaheuristic algorithm. Metaheuristic algorithm needs iteration to improve its solution quality, i.e., finding the near-optimal solution [15]. In the foraging process, animals cannot know the exact food location. So, the animal will go from one place to another to find the food location. In the mating process, the crossover of several individuals in the current generation will produce new descendants that inherit some characteristics of their parents. Besides, these descendants may have new characteristics too. The quality of the new generation may be better than the previous generation. Some algorithms that are built based on the reproduction mechanism are genetic algorithm (GA) [16], evolutionary algorithm (EA) [17], invasive weed optimizer (IWO) [18], and so on. Meanwhile, several examples of algorithms that are built based on the foraging mechanism are particle swarm optimization (PSO) [19], artificial bee colony (ABC) [20], ant colony optimization (ACO) [21], grey wolf optimization (GWO) [22], whale optimization algorithm (WOA) [23], capuchin search algorithm (CSA) [24], marine predator algorithm (MPA) [25] and so on. Red deer algorithm (RDA) [26] and Komodo mlipir algorithm (KMA) [27] are examples of algorithms that combine the foraging and mating. Meanwhile, several algorithms are built based on the mechanics of some traditional games, such as darts game optimization (DGO) [28], football game optimizer (FBGO) [29], and hide object game optimization (HOGO) [30]. Several algorithms adopt other mechanics, such as simulated annealing (SA) [31], tabu search (TS) [32], harmony search (HS) [33], and teaching learning-based algorithm (TLBO) [34]. Ironically, the real mechanics of the algorithm are not presented in its name.

Many algorithms contain some parameters that can be adjusted. These adjusted parameters are important to make the algorithm suitable for solving the problem. These parameters can be found in old-fashioned or brand-new algorithms. The examples are as follows. GA is equipped with the mutation rate to control the exploration strategy [16]. A higher mutation rate makes a higher probability for the individual to leave its current solution [16]. HS is enriched with the pitch adjustment rate (PAR) and harmony memory consideration rate (HMCR) [33]. Higher HMCR increases the probability of the future solution is created from the harmony memory (exploitation) [33]. Contrary, lower HMCR increases the probability of the future solution is generated randomly within the search space (exploration) [33]. In KMA, the number of females, big males, and small males must be determined before the optimization process is conducted. Every type of Komodo dragon plays a unique role and optimization strategy.

Meanwhile, the mlipir rate also must be stated to control the small males' speed [27]. The females have two reproduction options: sexual reproduction and parthenogenesis [27]. In its original form, the probability is equal [27]. But it can be modified as an adjusted parameter. In MPA, the fishing aggregate devices can be adjusted to control the probability of whether the exploration is conducted by generating a new location randomly within the search space or based on two randomly selected preys [25].

Most of metaheuristic algorithms use static strategy along the iteration process. There is no change in the strategy, for example, shifting from the exploration dominant strategy to the exploitation dominant strategy, or vice versa, when the current strategy fails to improve. The adjusted parameters remain unchanged until the iteration ends. Most algorithms only implement an acceptance-rejection strategy for the new solution. If this new solution is better than the current one, it replaces it. In HOGO, if the new solution is better than the current solution, then this new solution replaces the current solution [30]. Otherwise, the current solution does not change, or a new solution that is generated within the local solution space replaces the current solution. In ABC, if the scout bee leaves the current food source (solution), if there is not any improvement for several trials [20]. An example of algorithms that implement adaptability is KMA. In KMA, the swarm size increases gradually to the maximum size when there is no improvement and decreases gradually to the minimum size when the improvement occurs [27]. However, increasing the population size makes the computational process increase too.

Many metaheuristic algorithms are locked in the beating competition rather than focus on promoting new mechanisms. This circumstance occurs, especially in many shortcoming algorithms. The 23 functions have become the standard tools. An algorithm is claimed to be better than or superior to the previous algorithms if it can beat the sparing algorithms in as many functions as possible. In its first appearance, HOGO is benchmarked with GA, PSO, gravitational search algorithm (GSA), TLBO, GWO, GOA, emperor penguin optimizer (EPO), and spotted hyena optimizer (SHO) [30]. RDA is compared with GA, SA, PSO, ABC, imperialist competitive algorithm (ICA), and firefly algorithm (FA) [26]. The hybrid leader-based optimization (HLBO) is compared with hunger game search (HGS), slime mold algorithm (SMA), MPA, tunicate search algorithm (TSA), WOA, GWO, TLBO, GSA, PSO, and GA [35].

Engineering problems have become the favorite practical optimization problems in the first release of a new or modified metaheuristic. For example, optimization problems in mechanical engineering, such as welded beam design problems, speed reducer design problem, tension/compression spring design problem, and vessel design problem have been used for practical use cases in the first introduction of POA [36], coati optimization algorithm (COA) [37], and chameleon swarm algorithm (CSA) [38]. The optimal power flow system was used in the introduction of the modified honey badger algorithm (MHBA) [39]. Feature selection problem has been used as a use case in the first introduction of the flower pollination algorithm (FPA) [40]. Four problems in civil engineering (52-bar planar truss structure, 120-bar dome truss structure, 3-bay 15-story frame, and 3-bay 24-story frame) have been used as use cases in the

first introduction of stochastic paint optimizer (SPO) [41]. The economic load dispatch problem also has been used for use cases in some studies proposing metaheuristics, such as cheetah optimizer (CO) [42]. Unfortunately, the financial sector optimization problem is not popular for use case studies proposing new metaheuristics.

Based on this problem, this study introduces a new adaptive cone algorithm (ACA) algorithm. The term cone is chosen due to its mechanics of narrowing the local solution space width linearly as the iteration goes. This reduction strategy represents shifting the exploration dominant strategy to the exploitation dominant strategy as the iteration goes. This approach can be found in several algorithms, such as MPA [25] and HLBO [35]. But ACA transforms the strategy differently. The term adaptive represents the motivation of authors to develop new algorithms that can shift the strategy when the current strategy fails to improve the solution's quality. In ACA, the exploration dominant strategy is chosen when the algorithm fails to improve the current solution.

Based on this explanation, below are the contributions of this work.

- ACA is a new adaptive algorithm to the circumstances it faces so that it can shift the strategy based on the situation.
- ACA contains minimum adjusted parameters to avoid misjudgment.
- This work uses an optimization problem in the financial sector where this sector is rare to be used as a practical case in studies conducting new metaheuristic algorithms.

The remainder of this paper is structured as follows. The research methodology is presented in the second section. This second section consists of the proposed model and the simulation scenario. Then, the discussion regarding the simulation result and its in-depth analysis is carried out in the third section. The conclusion and future search potential regarding this work are presented in section four.

II. Material and Method

A. Proposed Model

This subsection presents the model. It consists of three parts: concept, algorithm, and mathematical model. The concept presents the mechanics of ACA and the reasoning behind its mechanics. The algorithm becomes the formalization of the algorithm that describes the sequential process within ACA. More detailed formalization of ACA is presented in the mathematical model.

ACA consists of three steps. These steps are carried out sequentially. The first step is swarm movement closer to the global reference. The second step is randomly searching for each agent within its local solution space. The third step is the random search of the global reference within its local solution space. The global reference is the highest quality solution among the population and iteration so far.

The reasoning behind this strategy is as follows. The first step is designed to improve the agent's current solution by using global reference as guidance. This first step is designed to track the possibility of a better solution between the current solution and the global reference. Moreover, in the first step, the agent has a chance to surpass the global reference to improve the global reference. The second step is designed to allow the agent to find other solutions independent of the global reference. The objective is to avoid the local optimal trap if moving closer to the current global reference means moving closer to the local optimal. The third step is designed to find a possibility for the global reference to improve its quality by searching locations around it.

The adaptability of ACA is presented in the local solution space width. The local solution space width declines gradually during the iteration. It represents the transformation of the exploration-dominant strategy to the exploitation-dominant strategy. But when the algorithm fails to improve the global reference, then the local solution space width will be reset to the initial width. The reasoning behind this strategy is as follows. When the algorithm fails to find a better solution, it can be assumed that the algorithm is locked in the local optimal. So, the best strategy regarding this circumstance is conducting the exploration dominant strategy. In ACA, the exploration dominant strategy can be conducted by maximizing the local solution space size.

This concept is then translated into an algorithm and mathematical model. This algorithm is presented in algorithm 1 in pseudocode, and the flowchart in Fig. 1. Some annotations used in the algorithm and mathematical model can be seen below.

b_l	lower bound
b_u	upper bound
c_1	first step candidate
c_2	second step candidate
c_3	third step candidate
x	Agent
X	set of agents
x_{ref}	global reference
x_{pref}	previous global reference
x_{tar}	target
f	fitness function
w_{cur}	current width factor
t	iteration
t_{max}	maximum iteration
T_{im}	improvement threshold

Below is the explanation of algorithm 1. Line 3 to line 5 represent the initialization process. Line 7 to line 18 represent the iteration process. The global reference becomes the final solution. Line 10 to line 15 represent the three sequential steps in every iteration. The process is started with the initialization. In the initialization, the initial location for all agents is set randomly. This process follows uniform distribution within the solution space as seen in (1).

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

The iteration is conducted after the initialization. The previous global best is set at the beginning of every iteration. This previous global reference is used to determine whether there is improvement at the end of every iteration. This process is formalized by using (2).

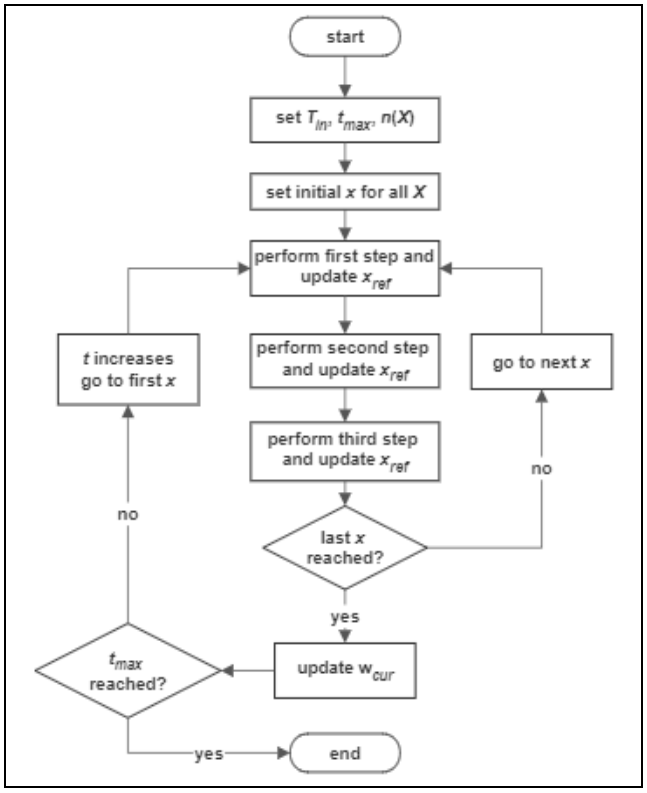

Fig. 1 Flowchart of adaptive cone algorithm

algorithm 1: Adaptive Cone Algorithm

output: x_{ref}
begin
  for all x **in** X **do**
    set initial location for x using (1)
  end for
  $w_{cur} \leftarrow 1$
  for $t = 1$ **to** t_{max} **do**
    set x_{prev} using (2)
    for all x **in** X **do**
      conduct first step using (3) to (5)
      update x_{ref} using (6)
      conduct second step using (7) and (8)
      update x_{ref} using (6)
      conduct third step using (9)
      update x_{ref} using (10)
    end for
    update w_{cur} using (11)
  end for
end

$$x_{pbest} = x_{best} \tag{2}$$

The first step is the step closer to the global best. This process is conducted by every agent and formalized by using (3) to (5). Equation (3) is used to determine the target. This target surpasses the global reference. Equation (4) states that the first step candidate is determined randomly between the current solution and the target. Equation (5) states that the first step candidate replaces the current solution if it is better than the current solution. Then, the global reference is updated by using (6).

$$x_{tar} = x + 2(x_{best} - x) \tag{3}$$

$$c_1 = x + U(0,1)(x_{tar} - x) \tag{4}$$

$$x' = \begin{cases} c_1, f(c_1) < f(x) \\ x, else \end{cases} \tag{5}$$

$$x_{best} = \begin{cases} x, f(x) < f(x_{best}) \\ x_{best} \end{cases} \tag{6}$$

The second step is the random search of an agent within its local solution space. It is formalized by using (7) and (8). Equation (7) is used to determine the second step candidate. Equation (8) is used to determine whether the second step candidate will replace the agent's current solution. Then, the global reference is updated for the second time by using (6).

$$c_2 = x + (2U - 1)\left(1 - \frac{w_{cur}}{t_{max}}\right)(b_u - b_l) \tag{7}$$

$$x' = \begin{cases} c_2, f(c_2) < f(x) \\ x, else \end{cases} \tag{8}$$

The third step is the random search for the global reference within its local solution space. It is formalized by using (9) and (10). Equation (9) is used to determine the third step candidate. Equation (10) states the global reference moves to the new solution only if the new solution is better than the current solution.

$$c_3 = x_{best} + (2U - 1)\left(1 - \frac{w_{cur}}{t_{max}}\right)(b_u - b_l) \tag{9}$$

$$x_{best}{}' = \begin{cases} c_3, f(c_3) < f(x_{best}) \\ x_{best}, else \end{cases} \tag{10}$$

After all agents have conducted these three steps, the next process is to determine the current width factor. This process is formalized by using (11).

$$w_{cur}{}' = \begin{cases} w_{cur} + 1, f(x_{best}) < T_{in}.f(x_{pbest}) \\ 1, else \end{cases} \tag{11}$$

The complexity of ACA is presented as $O(3t_{max}.n(X))$. This presentation shows that the complexity of ACA is linear to the population size or maximum iteration. Letter 3 represents the number of steps conducted by every agent in every iteration.

B. Simulation Scenario

In this work, ACA is challenged to find the optimal solution for both theoretical and practical optimization problems through simulation. The 23 functions represent theoretical optimization problems. Meanwhile, portfolio optimization problem represents the practical optimization problem.

The 23 functions are well-known optimization problems. They have been used in a lot of studies conducting metaheuristic algorithm, such as MPA [25], KMA [27], DGO [28], HOGO [30], average subtraction-based algorithm (ASBO) [43], pelican optimization algorithm (POA) [36], and so on. These functions represent both unimodal and multimodal problems. Unimodal function has one optimal solution [44]. On the other hand, multimodal function has more than one optimal solution [44]. One solution becomes the global optimal solution while the others become the local optimal [44]. In general, multimodal functions are more complex than unimodal functions. Although the unimodal function is simpler than the multimodal one, there is not any guarantee that an algorithm can find the optimal solution in the given iteration. The algorithm still can fail to reach the optimal solution at the end of iteration may because the solution space is too large, or the dimension is too high.

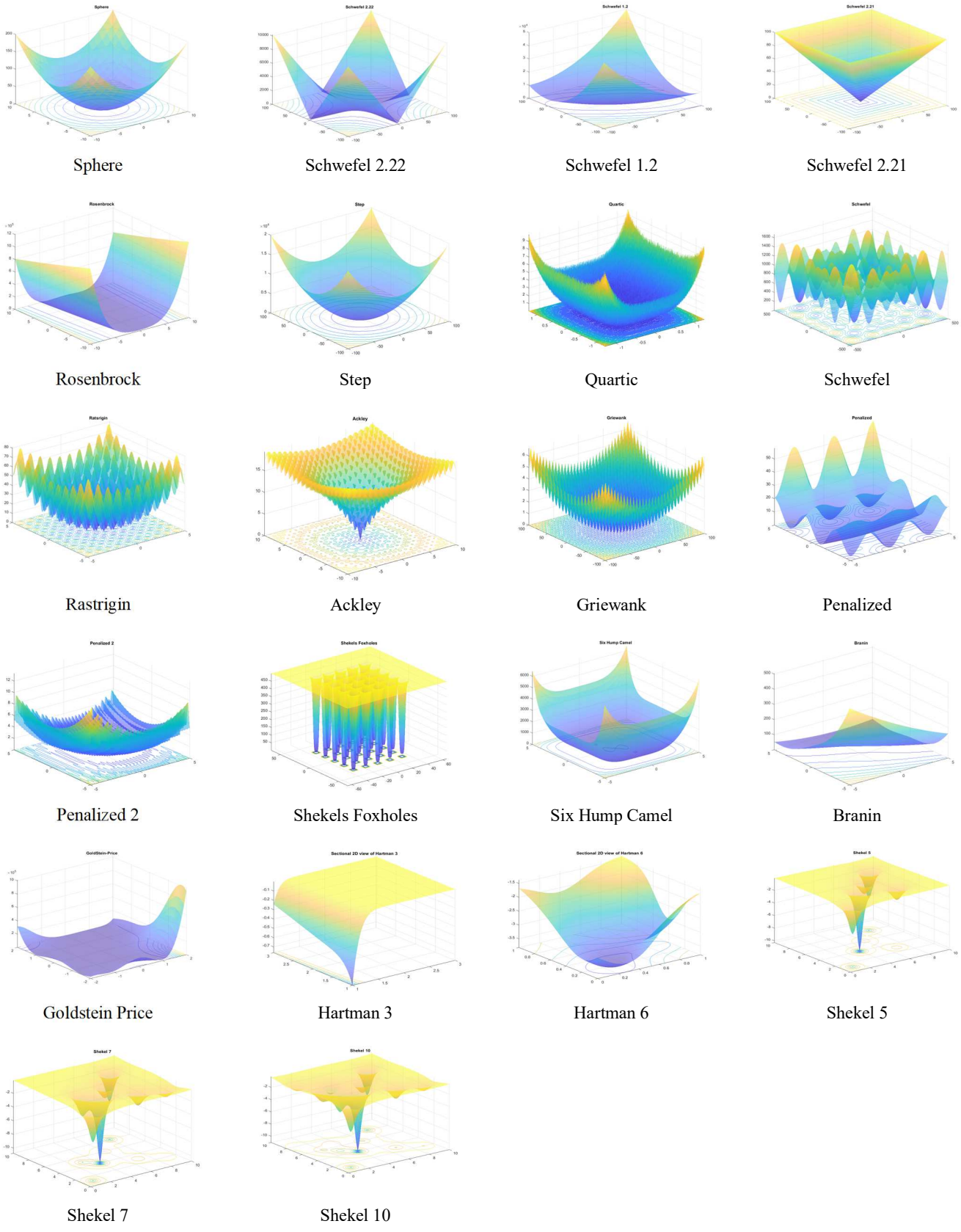

Fig. 2 Illustration of Benchmark Functions

The 23 functions can be clustered into three groups: high-dimension unimodal, high dimension multimodal, and fixed-dimension multimodal functions. The first group consists of functions 1 to 7. The second group consists of functions 8 to 13. The third group consists of functions 14 to 23. The detailed specification of every function is shown in Table 1. Moreover, the visualization of the sectional two-dimensional benchmark function can be seen in Fig. 2. The number of

dimensions for the high dimension functions is 20. Kowalik function is excluded from this visualization because it needs four parameters.

TABLE I
23 FUNCTIONS

No	Function	Dim	Solution space
f1	Sphere	20	[-100, 100]
f2	Schwefel 2.22	20	[-100, 100]
f3	Schwefel 1.2	20	[-100, 100]
f4	Schwefel 2.21	20	[-100, 100]
f5	Rosenbrock	20	[-30, 30]
f6	Step	20	[-100, 100]
f7	Quartic	20	[-1.28, 1.28]
f8	Schwefel	20	[-500, 500]
f9	Ratsrigin	20	[-5.12, 5.12]
f10	Ackley	20	[-32, 32]
f11	Griewank	20	[-600, 600]
f12	Penalized	20	[-50, 50]
f13	Penalized 2	20	[-50, 50]
f14	Shekel Foxholes	2	[-65, 65]
f15	Kowalik	4	[-5, 5]
f16	Six Hump Camel	2	[-5, 5]
f17	Branin	2	[-5, 5]
f18	Goldstein-Price	2	[-2, 2]
f19	Hartman 3	3	[1, 3]
f20	Hartman 6	6	[0, 1]
f21	Shekel 5	4	[0, 10]
f22	Shekel 7	4	[0, 10]
f23	Shekel 10	4	[0, 10]

In this simulation, ACA is competed with PSO, GWO, MPA, ASBO, and POA. All these algorithms are non-adaptive algorithms. PSO was chosen because this algorithm is the root of swarm intelligence. PSO also became the sparing algorithm in the latest studies in metaheuristic algorithm. GWO and MPA are the relatively new algorithms that are popular and have been used in many optimization studies. ASBO and POA are the new metaheuristic algorithms but have not been popular yet.

The parameter setting in the first simulation is as follows. The population size is 20. The maximum iteration is 200. In ACA, the improvement threshold is 0.5. In PSO, the weights are 0.1. In MPA, the fish aggregating device is 0.5.

This work chooses the portfolio optimization problem as the practical problem. This problem was chosen because metaheuristic studies that use problems in the financial sector are rare. In general, many studies that propose new metaheuristic algorithms choose engineering problem as the practical problem.

A portfolio can be defined as a set of assets or investment held by an individual or institution. These assets can be cash, gold, foreign currency, bond, stocks, and so on. In a broader perspective, these assets can also be fixed assets, such as land, house, commercial building, and so on. The asset must be productive, and its value must grow within a certain period. The risk of some assets may be high while the others may be low. Some assets, such as stocks, may produce revenue in a certain period while the others, such as gold, do not produce revenue. The productive assets may become the source of passive income for their holders or owners. On the other hand, non-productive assets can be used as hedge tools.

Individuals or institutions often diversify their assets into several types that can be seen as a portfolio. The objective is to maximize revenue or minimize risk [45]. By diversifying the assets, some assets can become anchors when the values of other assets fall. This portfolio can be a mix of certain assets, such as cash, gold, and stocks. On the other hand, a portfolio may consist of stocks only, but these stocks come from several companies.

Using stock, an investor can take a profit in two ways. The first way is from the dividend. Dividend is a fraction of annual net profit that is shared to the stockholders [46]. This dividend is decided in the general meeting of the shareholders that is usually conducted annually. This dividend is based on the company's audited financial statements. The second way is from capital gain. Capital gain is the increase of the stock's price. In some circumstances, an investor of public listed company prefers the capital gain to the dividend [47]. The investor can buy a stock and then hold it for a certain period. After that, the investor can sell this stock.

TABLE II
LIST OF IDX30 COMPANIES

No	Code	Current Price	6 Month Capital Gain
1	ADRO	3,140	1,460
2	ANTM	2,470	140
3	ASII	7,225	1,250
4	BBCA	7,525	125
5	BBNI	9,200	2,250
6	BBRI	4,450	280
7	BBTN	1,690	-60
8	BMRI	8,100	925
9	BRPT	820	-165
10	BUKA	286	-284
11	CPIN	4,900	-800
12	EMTK	1,850	-45
13	EXCL	2,640	-360
14	ICBP	8,475	-425
15	INCO	7,700	2,950
16	INDF	6,425	-75
17	INKP	7,675	-150
18	KLBF	1,610	5
19	MDKA	5,150	1,500
20	MIKA	2,890	530
21	PGAS	1,780	240
22	PTBA	4,400	1,740
23	SMGR	6,775	-1,425
24	TBIG	2,760	-320
25	TINS	1,770	185
26	TLKM	4,260	120
27	TOWR	955	-235
28	UNTR	30,300	8,475
29	UNVR	4,740	110
30	WSKT	525	-199

In the second simulation, the portfolio optimization problem focuses on the stocks. The IDX30 index refers to these stocks. All companies in the IDX30 index are listed on the Indonesia stock exchange (IDX). The IDX30 index consists of 30 companies with the highest capitalization and liquidity. This index is often used by the investor on the Indonesian stock exchange as the reference, besides the other indexes, such as LQ45, Kompas 100, BUMN20, and so on. This index is updated periodically. Some companies may be pushed out of the index and replaced by other companies due to their financial performance or market activity. The detailed description of the IDX30 members is shown in Table 2.

Below is the explanation of Table 2. The company code is a unique code the Indonesian stock exchange gives as the regulator. This code is used to differentiate a company from

other companies. The current price is the current price of the stock. This price is presented as rupiah per share. The 6-month capital gain is the increasing value of the current price compared with the price six months ago. Shorter period is used to measure the short-term performance while longer period is used to measure the long-term performance. The positive value means that the stock price increases so that investors can get profit if they sell this stock. On the contrary, the negative value means that the stock price decreases, so the investor may lose their investment if they sell this stock at the current price. In certain circumstances, the investors still sell these stocks to avoid more loss if they still hold them.

The objective of this portfolio optimization is to maximize the total capital gain. Meanwhile, the constraints are as follows. First, the investor must hold all these 30 stocks. In every stock, the number of shares ranges from 50 lots to 200 lots. A lot means 100 shares. The maximum invested capital is three billion rupiah. It means that the investment must not surpass this maximum invested capital. This problem represents the high dimension unimodal problem. These 30 stocks represent the 30 dimensions.

This second simulation benchmarks ACA with PSO, GWO, MPA, ASBO, and POA. The population size is 20. The maximum iteration is 200. This setting is applied to all algorithms.

III. Results and Discussion

A. Simulation Result

This subsection presents the simulation result. The first result is obtained from the simulation conducted for the theoretical optimization problem. The second result is obtained from the simulation conducted for the practical optimization problem. The first result is presented in Table 3 and Table 4, while the second result is presented in Table 5. Table 3 presents the average fitness score of every algorithm in solving the 23 functions. Then, the more profound comparison between ACA and the sparing algorithms is presented in Table 4.

TABLE III
Result on 23 benchmark functions

F.	PSO	GWO	MPA	ASBO	POA	ACA	Better Than
1	3.039×10^{3}	8.424×10^{-17}	1.694×10^{2}	3.580×10^{-50}	1.412×10^{4}	5.602×10^{1}	PSO, MPA, POA
2	0	0	0	0	0	0	-
3	8.836×10^{3}	1.610×10^{-13}	6.528×10^{2}	3.186×10^{-10}	2.161×10^{4}	3.836×10^{3}	PSO, POA
4	2.138×10^{1}	6.310×10^{-8}	4.747×10^{-1}	5.604×10^{-20}	4.959×10^{1}	2.352×10^{1}	POA
5	7.551×10^{5}	1.900×10^{1}	2.4959×10^{1}	1.851×10^{1}	1.692×10^{7}	8.788×10^{2}	PSO, POA
6	2.374×10^{3}	4.750	1.664×10^{2}	1.031×10^{-3}	1.295×10^{4}	3.204×10^{1}	PSO, MPA, POA
7	2.209×10^{-1}	4.690×10^{-2}	3.244×10^{-2}	5.519×10^{-3}	5.863	6.409×10^{-1}	PSO, POA
8	-2.291×10^{3}	1.516×10^{-12}	-2.907×10^{3}	-3.672×10^{3}	-2.959×10^{3}	-5.403×10^{3}	PSO, GWO, MPA, ASBO, POA
9	1.443×10^{2}	0	6.606×10^{1}	1.571	1.931×10^{2}	7.925×10^{1}	PSO, POA
10	1.126×10^{1}	2.951×10^{-11}	4.562	1.349	1.841×10^{1}	1.583×10^{1}	POA
11	2.737×10^{1}	0	2.668	4.380×10^{-2}	1.471×10^{2}	1.312	PSO, MPA, POA
12	1.191×10^{4}	1.907	3.516	4.713×10^{-4}	1.965×10^{7}	7.677	PSO, POA
13	4.489×10^{5}	3.140	1.630×10^{1}	4.217	6.649×10^{7}	2.813×10^{1}	PSO, POA
14	4.423	1.267×10^{1}	3.776	9.980×10^{-1}	1.370	1.234	PSO, GWO, MPA, POA
15	2.687×10^{-2}	1.484×10^{-1}	2.854×10^{-3}	6.403×10^{-2}	2.757×10^{-3}	7.815×10^{-4}	PSO, GWO, MPA, ASBO, POA
16	-1.031	7.456×10^{-27}	-1.028	-2.469×10^{-2}	-1.031	-1.032	PSO, GWO, MPA, ASBO, POA
17	6.529×10^{-1}	5.561×10^{1}	5.854×10^{-1}	6.438×10^{-1}	3.999×10^{-1}	3.981×10^{-1}	PSO, GWO, MPA, ASBO, POA
18	4.500	6.000×10^{2}	3.301	3.000	3.031	3.000	PSO, GWO, MPA, POA
19	-3.674×10^{-3}	-7.009×10^{-4}	-3.845	-4.954×10^{-2}	-4.954×10^{-2}	-4.954×10^{-2}	PSO, GWO
20	-2.512	-5.089×10^{-3}	-2.087	-1.373	-3.036	-3.273	PSO, GWO, MPA, ASBO, POA
21	-4.856	-2.731×10^{-1}	-2.201	-9.186	-4.535	-7.631	PSO, GWO, MPA, POA
22	-3.098	-2.936×10^{-1}	-2.249	-1.015×10^{1}	-4.655	-7.594	PSO, GWO, MPA, POA
23	-4.762	-3.218×10^{-1}	-2.508	-9.249	-4.368	-7.722	PSO, GWO, MPA, POA

The result shows that the performance of ACA is good and competitive. It can find the acceptable solution in all functions. Moreover, ACA can find the global optimal solution of three functions: Schwefel 2.22, Six Hump Camel, and Goldstein Price. ACA also outperforms all sparing algorithms in solving five functions: Schwefel, Kowalik, Hartman 6, Branin, and Six Hump Camel.

TABLE IV
Head-to-head comparison

Algorithm	Number of Functions that is Beaten			
	1st Group	2nd Group	3rd Group	Total
PSO	5	5	10	20
GWO	-	1	10	11
MPA	2	2	9	13
ASBO	-	-	4	4
POA	6	6	9	21

Table 4 also shows that ACA is competitive enough compared with competitors. It is superior relative to PSO and POA. It is also competitive enough compared with GWO and MPA. Meanwhile, ASBO has become the most difficult algorithm to beat. In general, it is shown that ACA is very competitive in solving functions in the third group. Meanwhile, ACA is less competitive in solving functions in the first and second groups.

TABLE V
Result on portfolio optimization problem

No	Code	Total Capital Gain
1	ACA	419,933,901
2	POA	396,797,161
3	ASBO	300,048,716
4	MPA	305,671,120
5	GWO	141,087,696
6	PSO	306,923,548

Table 5 shows that ACA is the best algorithm for the portfolio optimization problem. It outperforms all sparing algorithms. Compared with other algorithms, its total capital gain is 6%, 40%, 37%, 197%, and 37% higher than POA, ASBO, MPA, ASBO, and POA. This result proves that ACA is competitive not only in solving the theoretical optimization problem, but also in solving the practical optimization problem.

B. Discussion

More proper analysis is discussed in this subsection based on the results and findings. This discussion also becomes the bridge between the literature and the simulation result. There are some findings related to this simulation result.

First, ACA is a good metaheuristic algorithm. ACA has met the main objective in developing metaheuristic algorithm by finding the near optimal solution within the given iteration. This circumstance occurs in both the 23 functions and the portfolio optimization problem. In solving the 23 functions, ACA is very competitive in solving the fixed dimension multimodal functions. Meanwhile, it is less competitive in solving the high dimension unimodal functions and high dimension multimodal functions.

Second, the result strengthens the no-free-lunch theory. It has been said that performance or effectiveness of an algorithm depends on the problem it tries to solve [48]. The superiority of ACA in solving in the third group and on the other hand, its inferiority in solving functions in the first and second groups is in line with this statement.

This second finding is also strengthened with the results in Table 4 and 5. There is an extreme difference between both tables. ASBO is very superior in solving the 23 functions. Meanwhile, ASBO loses its superiority in solving the portfolio optimization problem. Moreover, the performance of ASBO is worse than POA, PSO, and MPA in solving the portfolio optimization problem. On the other hand, although POA becomes the worst algorithm in solving the 23 functions, it is much superior in solving the portfolio optimization problem than ASBO, MPA, GWO, and PSO. The total capital gain created by POA is a little bit lower than ACA. The performance of PSO in solving portfolio optimization problems is very competitive although its performance is solving the 23 functions is poor.

Third, solving the integer-based optimization problem may be different from the floating point-based optimization problem. The 23 benchmark functions represent the floating point-based optimization problem. Meanwhile, the portfolio optimization problem represents the integer-based optimization problem because the stock share is presented in integers and cannot be split or fractioned. Many latest metaheuristic algorithms use 23 benchmark functions to measure its performance. The result is that many algorithms try to create more precise solutions because better performance can only be achieved by increasing its precision. The floating point-based optimization problem is commonly found in many engineering areas, especially in mechanical engineering. Ironically, optimization problems in engineering area may not need precision as high as in the 23 benchmark functions. Meanwhile, many practical optimization problems can be modeled in the integer-based problem or in other world the integer programming. These problems are commonly found in operational research studies, such as in the manufacturing process. Moreover, in the operational research, the objective is relatively simple, such as calculating production cost, total travel distance, total tardiness, total profit, and so on.

Fourth, any studies introducing new metaheuristic algorithms should also implement ACA in solving various practical optimization problems. This finding is the continuation of the third finding. Although the 23 functions represent the variety in the optimization problem, in the end, the algorithm should be utilized in the practical purpose. Conducting only the 23 benchmark functions as the evaluation tool may end with misjudgment. On the other hand, there are various problems with specific circumstances in the real world.

Fifth, random search strategies are still important, especially in solving the integer-based problem. Among all the algorithms that are implemented in the simulation, only ACA and POA implement random search [36]. This random search becomes the key factor in achieving high total capital gain. ACA conducts random searches in the second and third steps. On the other hand, POA implements random search when determining the swarm target [36]. MPA implements partial random search during the eddy formation [25]. On the other hand, PSO, GWO, and ASBO do not implement random searches. But random searches may become a disadvantage in solving the floating point-based problem.

This work also presents the opportunity to enrich any designed metaheuristic with the adaptive capability. In general, this adaptability is designed to overcome the stagnation issue and there are many ways to perform it. But the main principle lies in performing diversification. In ACA, the local search space is reset like in the first iteration so that the random search traces a wider area. In ABC, the related bee or agent performs full random search [20]. In KMA, adaptive approach is performed by increasing the population size [27]. Meanwhile, many more methods can be chosen, such as changing the direction of the swarm, modifying the speed or step size, and many more.

IV. CONCLUSION

This study has demonstrated the effort in creating new metaheuristic algorithms, namely adaptive cone algorithm. This algorithm is proven effective in solving both theoretical problems and practical ones. ACA can find the global optimal solution by solving three functions: Schwefel 2.22, Six Hump Camel, and Goldstein Price. ACA also outperforms all sparing algorithms in finding the optimal solution of Schwefel, Kowalik, Six Hump Camel, Branin, and Hartman 6. ACA is superior in solving the fixed dimension multimodal functions but less competitive in solving the high dimensional unimodal functions and high dimension multimodal functions. ACA is very superior in solving the portfolio optimization problem by creating the highest total capital gain. Compared with other algorithms, its total capital gain is 6%, 40%, 37%, 197%, and 37% higher than POA, ASBO, MPA, ASBO, and POA.

This work can become the baseline for further research in metaheuristic algorithm. First, many studies are still needed to evaluate the performance of the adaptive cone algorithm by implementing this algorithm to solve various practical

optimization problems, from the engineering optimization problem to the combinatorial optimization problem. Second, studies conducted to modify this algorithm with other algorithms or methods are also challenging.

ACKNOWLEDGMENT

Telkom University, Indonesia, provided the fund for this work.

REFERENCES

[1] F. Luan, W. Zhang, and Y. Liu, "Robust international portfolio optimization with worst-case mean-LPM," *Math Probl Eng*, vol. 2022, pp. 1–10, Feb. 2022, doi: 10.1155/2022/5072487.

[2] M. Escobar-Anel, M. Wahl, and R. Zagst, "Portfolio optimization with wealth-dependent risk constraints," *Scand Actuar J*, vol. 2022, no. 3, pp. 244–268, Mar. 2022, doi: 10.1080/03461238.2021.1962962.

[3] H. Xuan, H. Zhang, and B. Li, "An improved discrete artificial bee colony algorithm for flexible flowshop scheduling with step deteriorating jobs and sequence-dependent setup times," *Math Probl Eng*, vol. 2019, pp. 1–13, Dec. 2019, doi: 10.1155/2019/8520503.

[4] K. Geng, C. Ye, L. Cao, and L. Liu, "Multi-objective reentrant hybrid flowshop scheduling with machines turning on and off control strategy using improved multi-verse optimizer algorithm," *Math Probl Eng*, vol. 2019, pp. 1–18, Jun. 2019, doi: 10.1155/2019/2573873.

[5] D. Kurniawan, A. C. Raja, S. Suprayogi, and A. H. Halim, "A flow shop batch scheduling and operator assignment model with time-changing effects of learning and forgetting to minimize total actual flow time," *Journal of Industrial Engineering and Management*, vol. 13, no. 3, p. 546, Nov. 2020, doi: 10.3926/jiem.3153.

[6] I. Ribas and R. Companys, "A computational evaluation of constructive heuristics for the parallel blocking flow shop problem with sequence-dependent setup times," *International Journal of Industrial Engineering Computations*, vol. 12, no. 3, pp. 321–328, 2021, doi: 10.5267/j.ijiec.2021.1.004.

[7] P. D. Kusuma and A. S. Albana, "University course timetabling model in joint courses program to minimize the number of unserved requests," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 10, 2021, doi: 10.14569/IJACSA.2021.0121014.

[8] F. Luan, W. Zhang, and Y. Liu, "Robust international portfolio optimization with worst-case mean-LPM," *Math Probl Eng*, vol. 2022, pp. 1–10, Feb. 2022, doi: 10.1155/2022/5072487.

[9] M. Escobar-Anel, M. Wahl, and R. Zagst, "Portfolio optimization with wealth-dependent risk constraints," *Scand Actuar J*, vol. 2022, no. 3, pp. 244–268, Mar. 2022, doi: 10.1080/03461238.2021.1962962.

[10] H. Xuan, H. Zhang, and B. Li, "An improved discrete artificial bee colony algorithm for flexible flowshop scheduling with step deteriorating jobs and sequence-dependent setup times," *Math Probl Eng*, vol. 2019, pp. 1–13, Dec. 2019, doi: 10.1155/2019/8520503.

[11] K. Geng, C. Ye, L. Cao, and L. Liu, "Multi-objective reentrant hybrid flowshop scheduling with machines turning on and off control strategy using improved multi-verse optimizer algorithm," *Math Probl Eng*, vol. 2019, pp. 1–18, Jun. 2019, doi: 10.1155/2019/2573873.

[12] D. Kurniawan, A. C. Raja, S. Suprayogi, and A. H. Halim, "A flow shop batch scheduling and operator assignment model with time-changing effects of learning and forgetting to minimize total actual flow time," *Journal of Industrial Engineering and Management*, vol. 13, no. 3, p. 546, Nov. 2020, doi: 10.3926/jiem.3153.

[13] I. Ribas and R. Companys, "A computational evaluation of constructive heuristics for the parallel blocking flow shop problem with sequence-dependent setup times," *International Journal of Industrial Engineering Computations*, vol. 12, no. 3, pp. 321–328, 2021, doi: 10.5267/j.ijiec.2021.1.004.

[14] P. D. Kusuma and A. S. Albana, "University course timetabling model in joint courses program to minimize the number of unserved requests," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 10, 2021, doi: 10.14569/IJACSA.2021.0121014.

[15] M. Mokhtari, M. Vaziri Sarashk, M. Asadpour, N. Saeidi, and O. Boyer, "Developing a model for the university course timetabling problem: a case study," *Complexity*, vol. 2021, pp. 1–12, Dec. 2021, doi: 10.1155/2021/9940866.

[16] I. Balan, "A new genetic approach for course timetabling problem," *Journal of Applied Computer Science & Mathematics*, vol. 15, no. 1, pp. 9–14, 2021, doi: 10.4316/JACSM.202101001.

[17] H. Hernández-Pérez and J.-J. Salazar-González, "A branch-and-cut algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem," *Eur J Oper Res*, vol. 297, no. 2, pp. 467–483, Mar. 2022, doi: 10.1016/j.ejor.2021.05.040.

[18] D. Wolfinger and J.-J. Salazar-González, "The pickup and delivery problem with split loads and transshipments: a branch-and-cut solution approach," *Eur J Oper Res*, vol. 289, no. 2, pp. 470–484, Mar. 2021, doi: 10.1016/j.ejor.2020.07.032.

[19] S. Fazi, J. C. Fransoo, T. van Woensel, and J.-X. Dong, "A variant of the split vehicle routing problem with simultaneous deliveries and pickups for inland container shipping in dry-port based systems," *Transp Res E Logist Transp Rev*, vol. 142, p. 102057, Oct. 2020, doi: 10.1016/j.tre.2020.102057.

[20] X. Yuan, Q. Zhang, and J. Zeng, "Modeling and solution of vehicle routing problem with grey time windows and multiobjective constraints," *J Adv Transp*, vol. 2021, pp. 1–12, Mar. 2021, doi: 10.1155/2021/6665539.

[21] Y. Wang, L. Ran, X. Guan, and Y. Zou, "Multi-depot pickup and delivery problem with resource sharing," *J Adv Transp*, vol. 2021, pp. 1–22, Jun. 2021, doi: 10.1155/2021/5182989.

[22] J. Swan *et al.*, "Metaheuristics 'in the large,'" *Eur J Oper Res*, vol. 297, no. 2, pp. 393–406, Mar. 2022, doi: 10.1016/j.ejor.2021.05.042.

[23] J. Xu, "Improved genetic algorithm to solve the scheduling problem of college English courses," *Complexity*, vol. 2021, pp. 1–11, Jun. 2021, doi: 10.1155/2021/7252719.

[24] A. Slowik and H. Kwasnicka, "Evolutionary algorithms and their applications to engineering problems," *Neural Comput Appl*, vol. 32, no. 16, pp. 12363–12379, Aug. 2020, doi: 10.1007/s00521-020-04832-8.

[25] A. Ibrahim, F. Anayi, M. Packianather, and O. A. Alomari, "New hybrid invasive weed optimization and machine learning approach for fault detection," *Energies (Basel)*, vol. 15, no. 4, p. 1488, Feb. 2022, doi: 10.3390/en15041488.

[26] D. Freitas, L. G. Lopes, and F. Morgado-Dias, "Particle swarm optimisation: a historical review up to the current developments," *Entropy*, vol. 22, no. 3, p. 362, Mar. 2020, doi: 10.3390/e22030362.

[27] Y. Celik, "An enhanced artificial bee colony algorithm based on fitness weighted search strategy," *Automatika*, vol. 62, no. 3–4, pp. 300–310, Oct. 2021, doi: 10.1080/00051144.2021.1938477.

[28] S. Liang, T. Jiao, W. Du, and S. Qu, "An improved ant colony optimization algorithm based on context for tourism route planning," *PLoS One*, vol. 16, no. 9, p. e0257317, Sep. 2021, doi: 10.1371/journal.pone.0257317.

[29] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[30] N. Rana, M. S. A. Latiff, S. M. Abdulhamid, and H. Chiroma, "Whale optimization algorithm: a systematic review of contemporary applications, modifications and developments," *Neural Comput Appl*, vol. 32, no. 20, pp. 16245–16277, Oct. 2020, doi: 10.1007/s00521-020-04849-z.

[31] M. Braik, A. Sheta, and H. Al-Hiary, "A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm," *Neural Comput Appl*, vol. 33, no. 7, pp. 2515–2547, Apr. 2021, doi: 10.1007/s00521-020-05145-6.

[32] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic," *Expert Syst Appl*, vol. 152, p. 113377, Aug. 2020, doi: 10.1016/j.eswa.2020.113377.

[33] A. M. Fathollahi-Fard, M. Hajiaghaei-Keshteli, and R. Tavakkoli-Moghaddam, "Red deer algorithm (RDA): a new nature-inspired meta-heuristic," *Soft comput*, vol. 24, no. 19, pp. 14637–14665, Oct. 2020, doi: 10.1007/s00500-020-04812-z.

[34] S. Suyanto, A. A. Ariyanto, and A. F. Ariyanto, "Komodo mlipir algorithm," *Appl Soft Comput*, vol. 114, pp. 1–17, Jan. 2022, doi: 10.1016/j.asoc.2021.108043.

[35] M. Dehghani, Z. Montazeri, H. Givi, J. Guerrero, and G. Dhiman, "Darts game optimizer: a new optimization technique based on darts game," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 286–294, Oct. 2020, doi: 10.22266/ijies2020.1031.26.

[36] M. Dehghani, M. Mardaneh, J. Guerrero, O. Malik, and V. Kumar, "Football game based optimization: an application to solve energy commitment problem," *International Journal of Intelligent*

Engineering and Systems, vol. 13, no. 5, pp. 514–523, Oct. 2020, doi: 10.22266/ijies2020.1031.45.

[37] M. Dehghani *et al.*, "HOGO: hide objects game optimization," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 4, pp. 216–225, Aug. 2020, doi: 10.22266/ijies2020.0831.19.

[38] M. Almarashi, W. Deabes, H. H. Amin, and A.-R. Hedar, "Simulated annealing with exploratory sensing for global optimization," *Algorithms*, vol. 13, no. 9, p. 230, Sep. 2020, doi: 10.3390/a13090230.

[39] F. H. Awad, A. Al-kubaisi, and M. Mahmood, "Large-scale timetabling problems with adaptive tabu search," *Journal of Intelligent Systems*, vol. 31, no. 1, pp. 168–176, Jan. 2022, doi: 10.1515/jisys-2022-0003.

[40] M. Dubey, V. Kumar, M. Kaur, and T.-P. Dao, "A systematic review on harmony search algorithm: theory, literature, and applications," *Math Probl Eng*, vol. 2021, pp. 1–22, Apr. 2021, doi: 10.1155/2021/5594267.

[41] X. Mi, Z. Liao, S. Li, and Q. Gu, "Adaptive teaching–learning-based optimization with experience learning to identify photovoltaic cell parameters," *Energy Reports*, vol. 7, pp. 4114–4125, Nov. 2021, doi: 10.1016/j.egyr.2021.06.097.

[42] M. Dehghani and P. Trojovský, "Hybrid leader based optimization: a new stochastic optimization algorithm for solving optimization applications," *Sci Rep*, vol. 12, no. 1, pp. 1–16, Dec. 2022, doi: 10.1038/s41598-022-09514-0.

[43] P. Trojovský and M. Dehghani, "Pelican optimization algorithm: a novel nature-inspired algorithm for engineering applications," *Sensors*, vol. 22, no. 3, pp. 1–34, Jan. 2022, doi: 10.3390/s22030855.

[44] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowl Based Syst*, vol. 259, p. 110011, Jan. 2023, doi: 10.1016/j.knosys.2022.110011.

[45] M. S. Braik, "Chameleon swarm algorithm: a bio-inspired optimizer for solving engineering design problems," *Expert Syst Appl*, vol. 174, p. 114685, Jul. 2021, doi: 10.1016/j.eswa.2021.114685.

[46] S. A. Yasear and H. 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, Aug. 2022, doi: 10.22266/ijies2022.0831.14.

[47] M. I. A. Latiffi, M. R. Yaakub, and I. S. Ahmad, "Flower pollination algorithm for feature selection in tweets sentiment analysis," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 5, pp. 429–436, 2022, doi: 10.14569/IJACSA.2022.0130551.

[48] A. Kaveh, S. Talatahari, and N. Khodadadi, "Stochastic paint optimizer: theory and application in civil engineering," *Eng Comput*, vol. 38, no. 3, pp. 1921–1952, Jun. 2022, doi: 10.1007/s00366-020-01179-5.

[49] M. A. Akbari, M. Zare, R. Azizipanah-abarghooee, S. Mirjalili, and M. Deriche, "The cheetah optimizer: a nature-inspired metaheuristic algorithm for large-scale optimization problems," *Sci Rep*, vol. 12, no. 1, p. 10953, Jun. 2022, doi: 10.1038/s41598-022-14338-z.

[50] M. Dehghani, Š. Hubálovský, and P. Trojovský, "A new optimization algorithm based on average and subtraction of the best and worst members of the population for solving various optimization problems," *PeerJ Comput Sci*, vol. 8, pp. 1–40, Mar. 2022, doi: 10.7717/peerj-cs.910.

[51] K. Hussain, M. N. Mohd Salleh, S. Cheng, and R. Naseem, "Common benchmark functions for metaheuristic evaluation: a review," *JOIV: International Journal on Informatics Visualization*, vol. 1, no. 4–2, p. 218, Nov. 2017, doi: 10.30630/joiv.1.4-2.65.

[52] L. Chin, E. Chendra, and A. Sukmana, "Analysis of portfolio optimization with lot of stocks amount constraint: case study index LQ45," *IOP Conf Ser Mater Sci Eng*, vol. 300, p. 012004, Jan. 2018, doi: 10.1088/1757-899X/300/1/012004.

[53] N. Prakash and Yogesh L, "Market reaction to dividend announcements during pandemic: an event study," *Vision: The Journal of Business Perspective*, p. 097226292110662, Dec. 2021, doi: 10.1177/09722629211066288.

[54] Silvia and N. Toni, "The effect of profitability and capital structure against company value with dividend policy as moderator variable in consumption companies registered on the 2014-2018 IDX," *Research, Society and Development*, vol. 9, no. 11, p. e019119260, Oct. 2020, doi: 10.33448/rsd-v9i11.9260.

[55] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, Apr. 1997, doi: 10.1109/4235.585893.