# Stochastic Marine Predator Algorithm with Multiple Candidates

Purba Daru Kusuma, Ratna Astuti Nugrahaeni
Computer Engineering, Telkom University, Bandung, Indonesia

***Abstract*—This work proposes a metaheuristic algorithm that modifies the marine predator algorithm (MPA), namely, the stochastic marine predator algorithm with multiple candidates (SMPA-MC). The modification is conducted in several aspects. The proposed algorithm replaces the three fixed equal size iteration phases with linear probability. Unlike the original MPA, in this proposed algorithm, the selection between exploration and exploitation is conducted stochastically during iteration. In the beginning, the exploration-dominant strategy is implemented to increase the exploration probability. Then, during the iteration, the exploration probability decreases linearly. Meanwhile, the exploitation probability increases linearly. The second modification is in the prey's guided movement. Different from the basic MPA, where the prey moves toward the elite with small step size, several candidates are generated with equal inter-candidate distance in this work. Then, the best candidate is chosen to replace the prey's current location. The proposed algorithm is then implemented to solve theoretical mathematic functions and a real-world optimization problem in production planning. The simulation result shows that in the average fitness score parameter, the proposed algorithm is better than MPA, especially in solving multimodal functions. The simulation result also shows that the proposed algorithm creates 9%, 19%, and 30% better total gross profit than particle swarm optimization, marine predator algorithm, and Komodo mlipir algorithm, respectively.**

***Keywords*—*Metaheuristic; marine predator algorithm; stochastic system; production planning***

## I. Introduction

Optimization is a subject that is widely used and studied. Optimization is implemented in many areas, especially in operations research, such as manufacturing [1], logistics [2], transportation [3], education [4], finance [5], and so on. Optimization becomes more important because of its objective nature to maximize productivity or output or minimize resources within certain constraints and limitations. This circumstance often occurs in real-world problems, from the simple one like managing the school bus route to the complex one, such as handling the production process in manufacturing that builds products with many components, such as cars, airplanes, ships, and so on.

In general, optimization methods can be divided into two groups: exact and approximate. The exact methods have an advantage that true or global optimal is guaranteed to find. The problem is that the exact method needs excessive computation resources in solving complex and large dimension problems. On the other hand, approximate methods do not guarantee that global optimal can be found. The objective of approximate methods is to find near-optimal or acceptable optimal while avoiding local optimal [6], especially in multimodal problems. Fortunately, the approximate approach is popular because of its adaptability to computational resource constraints. The metaheuristic algorithm is a well-known and widely used method that uses an approximate approach. In metaheuristic, optimization is achieved during iteration.

Many studies have proposed new metaheuristic algorithms in this last decade. Many algorithms were inspired by nature, especially animals. These algorithms were developed based on animal behavior during foraging, such as grey wolf optimizer (GWO) [7], dragonfly algorithm (DA) [8], whale optimization algorithm (WOA) [9], and so on. Besides foraging, several algorithms were developed by mimicking animal behavior during reproduction, such as Komodo mlipir algorithm (KMA) [10], red deer algorithm (RDA) [11], Cuckoo search algorithm (CSA) [12], butterfly optimization algorithm (BOA) [13], and so on.

Besides proposing a new algorithm, many studies in metaheuristic algorithms were conducted to modify the existing algorithm. These modifications were conducted to improve the algorithm's performance or to make the algorithm more suitable to solve specific problems. Several well-known algorithms that have been widely modified or combined are genetic algorithm (GA), particle swarm optimization (PSO), simulated annealing (SA), tabu search (TS), and so on. Farag et al. [14] improved the binary-real coded genetic algorithm with k-means clustering to solve the unit commitment problem. Deb et al. [15] developed a non-dominated sorting genetic algorithm (NSGA II) derivative of a genetic algorithm to find pareto optimal in solving the multi-objective optimization problem. Sylia et al. [16] hybridized the PSO with proportional fair scheduling (PFS) to solve resource allocation in the orthogonal frequency division multiplexing (OFDM) transmission for future long-term evolution (LTE) and 5G networks.

One shortcoming metaheuristic algorithm is the marine predator algorithm (MPA). This algorithm is a metaphor-inspired algorithm that mimics the behavior of sea predators, such as shark, marlin, and swordfish, during hunting prey [17]. This algorithm is one of the rare metaheuristic algorithms that uses iteration to control the exploration and exploitation. The iteration is divided into three phases. The first phase conducts exploration [17]. The second phase conducts both exploration and exploitation [17]. The third phase conducts exploitation [17]. Exploration is mostly conducted by implementing the Brownian motion, while exploitation is conducted by implementing the Levy movement.

Even though this algorithm is new, it has been used in many optimization studies, such as in task scheduling [18], power system [19], hydrothermal scheduling [20], and so on. Moreover, studies conducted on MPA modification have been found but are still limited. Based on this circumstance, this MPA is still potential to modify. Several studies in metaheuristic, such as KMA [10], also used MPA as a performance comparison.

Based on this opportunity, this work proposes modifying and improving the basic MPA. As metaheuristic algorithm, the proposed model consists of conceptual model, algorithm in pseudocode, and the mathematical model. In this work, the proposed model is evaluated by implementing this proposed model into the simulation to solve the theoretical mathematical optimization problem and a real-world optimization problem. In this simulation, the convergence and sensitivity of the algorithm are also evaluated.

The contributions of this work are as follows.

*1)* The proposed algorithm replaces the static division of the iteration with the stochastic approach where the opportunity to conduct exploration or exploitation changes during the iteration.

*2)* This work proposes the existence of several candidates during the Brownian motion or Levy movement, where their fitness score is considered to become the prey’s next move.

*3)* This work implements the modified version of MPA to optimize real-world production planning problem.

The remainder of this paper is organized as follows. The mechanism of the basic MPA is discussed in the second section. The model of the proposed algorithm that consists of a conceptual, algorithm, and mathematical model is explained in the third section. The fourth section explains the simulation to evaluate the proposed algorithm's performance. In this work, there are five simulations. The first to fourth simulations are conducted to evaluate the proposed algorithm’s performance in solving 23 well-known optimization functions. The fifth simulation is conducted to evaluate the proposed algorithm’s performance in solving a real-world production optimization problem. The more profound analysis related to the simulation result and findings is discussed in the fifth section. Finally, the conclusion and the future research potential related to this work are summarized in the sixth section.

## II. Related Work

The marine predator algorithm is a metaheuristic algorithm that adopts sea predator behavior or movement during foraging or hunting prey [17]. This algorithm combines Brownian motion and Levy movement. Levy movement is a derivative of random walk movement whose characteristics are closely related to sea predators, such as shark, swordfish, or marline, during searching for prey [21]. In this algorithm, the Levy movement is combined with the Brownian motion to conduct exploration and exploitation. MPA consists of two sets of agents: predators and prey. The adoption of the Levy movement is like the Cuckoo search algorithm (CSA). CSA is developed based on the parasitism behavior of cuckoo birds during finding a nest for their eggs [12]. In CSA, the cuckoo implements the Levy movement only [12].

This algorithm divides exploration and exploitation depending on the iteration. It is very different from many common metaheuristic algorithms, such as particle swarm optimization, genetic algorithm, harmony search (HS), and so on, where the decision of running the exploration or exploitation does not depend on the iteration. MPA focuses on avoiding local optimal in the early phase and improving the solution in the later phase.

Although rare, one example of an algorithm where the exploitation-exploration decision depends on the iteration is simulated annealing (SA). In general, SA focuses on exploitation by conducting neighborhood search in every iteration [22]. Exploration is conducted by accepting a new worse solution based on some probabilistic calculation to avoid local optimal [22]. In SA, the outer loop iterates from the initial high temperature to the final low temperature. When the temperature is high, the new worse solution is easily accepted. During the decrease in temperature, accepting the worse solution becomes more difficult [22]. In the end, a new worse solution is hard to accept. This exploration-to-exploitation approach is like MPA but with different mechanism.

In MPA, the iteration is divided into three phases with the same duration. In the first phase, the process focuses on exploration by implementing the Brownian movement for all prey. The objective is that local optimal should be avoided in the early phase. In the second phase, the population is divided into two equal-size groups. The first group consists of preys that conduct exploration by implementing the Brownian movement. The second group consists of preys that conduct exploitation by implementing the Levy movement. The objective is that the algorithm focuses on improving searching quality. In the third phase, all populations focus on exploitation by adopting the Levy movement. This division is illustrated in Fig. 1.

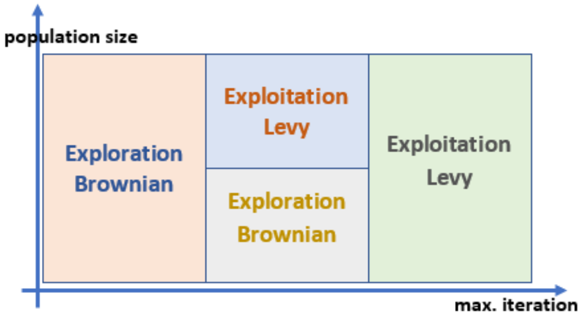

Fig. 1. Static Division in Marine Predator Algorithm.

There are several sequential processes in every iteration. The first process is the prey’s movement which is explained earlier. The second process is updating the predators. The prey replaces its related predator only if its fitness is better than its related predator. The third process is the eddy formation. This eddy formation is another method used in MPA to avoid local optimal. This process is conducted stochastically based on the fish aggregating device affects that range from 0 to 1. There are two possible actions related to this process based on probabilistic calculations. If a certain generated random number is below the fish aggregating devices, the prey randomly conducts a long jump within its local problem space that narrows as iteration goes. Otherwise, this prey will move toward two randomly selected prey at a certain speed.

In the prey's guided movement, whether Brownian movement or Levy flight, only one location is considered, as shown in Fig. 2. This location may be within the path between the prey and the elite or the extended distance from the elite. The determination of this new location depends on the step size, which is determined stochastically depending on the selected movement and the gap between the predator and the prey. The fitness value is not considered.

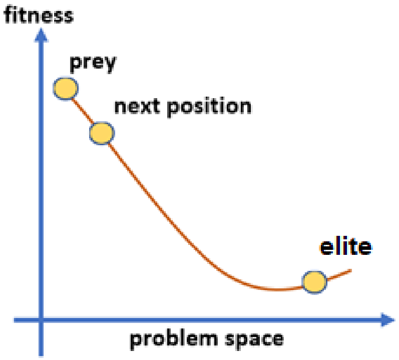

Fig. 2. Guided Movement in Marine Predator Algorithm.

Moving toward the best solution is common in many metaheuristic algorithms. In particle swarm optimization (PSO), each agent moves toward local best and global best with a certain proportion [23]. In Komodo mlipir algorithm (KMA), female mates with the highest quality big male to produce two offspring [10]. The first offspring is close to the female, and the second one is close to the highest quality big male. Then, offspring whose fitness is better becomes the replacement. Meanwhile, the small male moves toward the big male. In the red deer algorithm, this idea is implemented during the fighting between the male commander and stag and the mating of the commander and harem [11].

Even though the MPA is proven as a competitive algorithm, there are several questions or review due to this algorithm. First, is there any possible method to conduct exploration-to-exploitation approach despite this fixed division during the iteration? Second, is there any method to improve the movement of the prey rather than the small step size?

There are several possible modifications due to this basic MPA mechanism. The first is eliminating the fixed size division of the iteration while the concept of exploration dominant in the early iteration and the exploitation dominant in the later iteration is still adopted. The second is to create several new location candidates for prey during the guided movement. Their fitness score is considered so that the prey moves to a more promising location.

## III. Proposed Model

In this section, the proposed model will be discussed in detail. The model consists of a conceptual and mathematical model. The conceptual model explains the concept and the difference between the proposed algorithm and the original MPA, especially in the exploration and exploitation division and the improvement of the guided movement. The mathematical model consists of the main algorithm of SMPA-MC and the mathematical formulae following the algorithm.

Like MPA, this proposed algorithm consists of two sets of agents: preys and predators. Both preys and predators have equal population sizes. The relation between prey and predator is one-to-one. After the prey moves, then their fitness score is evaluated. If the prey's fitness score is better than the predator's fitness score, then the predator moves to the prey's location.

As a metaheuristic algorithm, SMPA-MC consists of two parts: initialization and iteration. In the initialization, the initial prey' and predators' location is generated randomly within the problem space using a uniform distribution.

As a derivative version of MPA, the iteration affects the exploration and exploitation division in this proposed algorithm. Unlike MPA, where this division is divided into fixed three phases, this division is conducted based on a stochastic approach in this proposed algorithm. In the beginning, the probability of exploration is high. Contrary, the probability of exploitation is low. During the iteration, the probability of exploration declines linearly while the probability of exploitation climbs up linearly. At the end of the iteration, the probability of exploitation is high, while the probability of exploration is low. This mechanism is illustrated in Fig. 3.

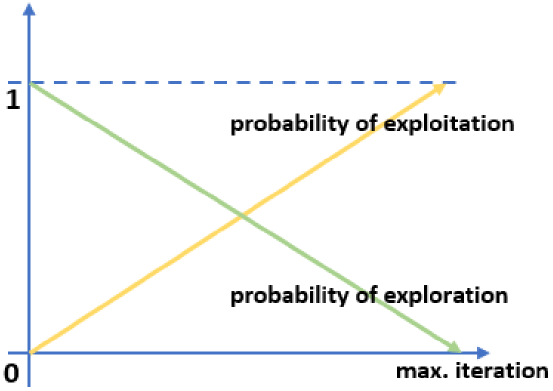

Fig. 3. Stochastic-based Exploitation and Exploration.

The guided movement of the proposed algorithm is also different from the MPA. This proposed algorithm generates multiple candidates between the prey and the elite. The inter-candidate distance is equal. One candidate whose fitness is the best among these candidates is then chosen as the best candidate. This mechanism is adopted from KMA, especially in the mating process between the highest quality male and the female [10]. The difference is that in KMA, sexual reproduction only produces two offspring. In this proposed algorithm, the guided movement generates multiple candidates. The best new generation or candidate becomes the replacement. Then, this best candidate location becomes the prey's next location. This concept is illustrated in Fig. 4.

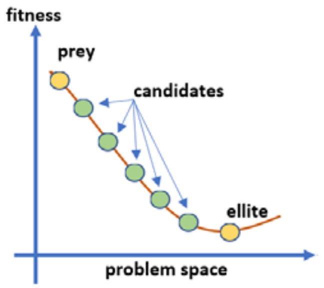

Fig. 4. Guided Movement with Multiple Candidates / Alternatives.

Different mechanism also occurs during the eddy formation. During this process, there are two possible actions. The first is that the prey moves randomly within its local problem space. In the beginning, the local problem space is wide. During the iteration, this local problem space decreases linearly too. It also reflects the transition from exploration to exploitation during the iteration. The second action is that the prey moves to a location in the middle of the prey's current location, and the other prey's location is selected randomly. Although the mechanism during eddy formation is different, the motivation is the same.

This conceptual model is then interpreted into sequential steps. These sequential steps are as follows.

- Step 1: generate initial preys and predators.
- Step 2: set initial iteration.
- Step 3: generate several candidates based on the movement that is chosen stochastically.
- Step 4: select the best candidate to replace the current prey.
- Step 5: update the related predator.
- Step 6: implement eddy formation.
- Step 7: if the maximum iteration has not been reached, go to step 3. Otherwise, iteration stops.
- Step 8: select the best predator to become the final solution.

This conceptual model is then transformed into a mathematical model. The mathematical model consists of two parts: main algorithm and formulae. The main algorithm is shown in algorithm 1. Meanwhile, several annotations used in the mathematical model are as follows.

| | |
|---|---|
| $b_l$ | lower bound |
| $b_u$ | upper bound |
| $c$ | candidate |
| $c_{best}$ | best candidate |
| $C$ | set of candidates |
| $f$ | fitness |
| $fad$ | fish aggregating devices |
| $P$ | population |
| $r$ | predator |
| $R$ | set of predators |
| $t$ | time / iteration |
| $t_h$ | time threshold |
| $t_{max}$ | maximum iteration |
| $y$ | prey |
| $Y$ | set of preys |

**algorithm 1: SMPA-MC main algorithm**

```
//initialization
for $i$ = 1 to $n(P)$
  generate ($y_i$)
  generate ($r_i$)
end
//iteration
for $t$ = 1 to $t_{max}$
  generate ($t_h$)
  for $i$ = 1 to $n(P)$
    if $U(0, 1) < t_h$ then
      for $j$ = 1 to $n(C)$
        generate-guided-exploitation ($c_j, y_i, r_i$)
      end
      $c_{best}$ = select best candidate ($C$)
    else
      for $j$ = 1 to $n(C)$
        generate-guided-exploration ($c_j, y_i, r_i$)
      end
      $c_{best}$ = select best candidate ($C$)
    end if
    $y_i = c_{best}$
    $r_i$ = update ($y_i, r_i$)
    if $U(0,1) < fad$ then
      $y_i$ = limited random move ($y_i, t_h, l_i$)
    else
      $y_{sel} = U(Y)$
      $y_i$ = half move ($y_i, y_{sel}$)
    end if
  end
end
$s_{final}$ = find best ($R$)
```

All predators' and preys' initial location is generated in the initialization process. This initial location is generated randomly within the problem space. This process is formalized by using (1) and (2). Equation (1) generates the prey's initial location, while (2) is used to generate the predator's initial location. All predators and prey are distributed randomly within the problem space.

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

$$r = U(b_l, b_u) \tag{2}$$

The iteration process runs after the initialization process ends. At the beginning of every iteration, a time threshold is calculated. This threshold determines whether this iteration is conducted for guided exploitation or guided exploration. The time threshold is calculated by using (3).

$$t_h = \frac{t}{t_{max}} \tag{3}$$

A random number is then generated, and it follows uniform distribution as shown in algorithm 1. If this random number is less than the time threshold, guided exploitation is conducted. Otherwise, guided exploration is conducted. In both guided exploration and exploitation, several candidates are generated. Then, the best candidate is selected among these candidates. After the best candidate is selected, this candidate replaces the related prey. This process is formalized by using (4) to (6). Equation (4) is used for the guided exploitation. Equation (5) is used for the guided exploration. Equation (6) formalizes the best candidate selection.

$$c_j = r + \frac{j}{n(C)}(r - y) \tag{4}$$

$$c_j = y + \frac{j}{n(C)}(r - y) \tag{5}$$

$$c_{best} = c \in C \wedge \min(f(c)) \tag{6}$$

The predator location is then evaluated after a prey moves to its new location. If this prey's fitness score is better than the predator's fitness score, then this prey becomes the new predator. This process is formalized by using (7).

$$r' = \begin{cases} y, f(y) < f(r) \\ r, else \end{cases} \tag{7}$$

The last process in every iteration is applying the eddy formation. There are two possible actions in this process. The selection determined stochastically end depends on the fish aggregating devices value. This process can be limited to random movement or half movement. A prey will move randomly within its local problem space in the limited random movement. On the other hand, in the half movement, a prey will move to the middle between its current location and other prey selected randomly. The limited random movement is formalized using (8), while the half movement is formalized using (9).

$$y' = y + (2U(0,1) - 1)(1 - t_h)(b_u - b_l) \tag{8}$$

$$y' = y + \frac{y_{sel} - y}{2} \tag{9}$$

The complexity of this algorithm, as it is presented in big O notation, is O($t_{max}$.$n(P)$.$n(C)$). This presentation means that the complexity is the multiplication between the maximum iteration, population size, and the number of candidates.

## IV. Simulation and Result

The proposed algorithm is then implemented into a simulation to observe its performance. There are five simulations conducted in this work. The first simulation is conducted to evaluate its performance in solving mathematical problems. The second simulation is conducted to evaluate its performance in achieving the convergence condition. The third simulation is conducted to evaluate the proposed algorithm's performance related to the fishing aggregate devices. The fourth simulation is conducted to evaluate the performance related to the number of candidates. The fifth simulation is conducted to evaluate its performance in solving a real-world problem.

In this simulation, the proposed SMPA-MC algorithm is compared with several algorithms: PSO, HS, hide object game optimizer (HOGO), KMA, and MPA. The reason for choosing these algorithms as a comparison is that these algorithms use distinct exploration-exploitation mechanisms. PSO and HS represent the well-known old-fashioned algorithm. HOGO and KMA represent the shortcoming algorithms that hybridize many common methods. MPA is chosen to observe the performance improvement due to modifying its basic form.

PSO is a well-known algorithm that is developed based on swarm intelligence. In PSO, each agent moves to a new location depending on the weighted cumulative method among its current location, its local best, and the global best [23]. The global best represents the collective intelligence shared among agents, and it is updated every time a new local best is found [23].

HS represents the non-population-based metaheuristic algorithm. Moreover, this algorithm is the simplest one among other algorithms. The exploration-exploitation decision is conducted based on the stochastic approach [24]. A new solution can be generated from the harmony memory (exploitation) or anywhere else within the problem space (exploration) based on the harmony memory considering rate (HMCR) [24].

HOGO represents a game-based algorithm. It mimics the behavior of the old hide-object game. This algorithm is also a population-based algorithm that consists of a set of agents. The agent's movement depends on the global best, the global worst, and the randomly selected agent through a weighted cumulative method [25]. An agent tends to move toward the global best and avoid the global worst [25].

KMA represents a hybrid metaheuristic algorithm. It combines swarm intelligence and an evolution system. The males conduct the PSO-like movement by moving toward the better big males [10]. On the other hand, the evolution system is conducted by the female by mating with the highest quality big male to generate better offspring [10]. Meanwhile, exploration is conducted by parthenogenesis or asexual reproduction [10].

MPA represents an algorithm where the current iteration affects the decision to conduct exploration or exploitation. In KMA, HOGO, HS, and PSO, the iteration does not affect the decision. It also represents a population-based algorithm where an agent consists of two engaged agents: prey and predator.

There are several common parameters used in the simulation. These parameters are shown in Table I. The value of population size and maximum iteration represents the moderate computation process. The weights in PSO represent the balance movement.

TABLE I. Parameters Default Value

| Parameters | Default Value |
|---|---|
| population size | 20 |
| maximum iteration (except HS) | 200 |
| maximum iteration (HS) | 4000 |
| fishing aggregate devices (MPA and SMPA-MC) | 0.2 |
| current speed weight (PSO) | 0.5 |
| social weight | 0.5 |
| cognitive weight | 0.5 |
| number of candidates (SMPA-MC) | 10 |

In the first simulation, the proposed algorithm is implemented to solve or find the global optimal of the given functions. There are 23 functions to be solved. Seven functions are unimodal functions. Six functions are multimodal functions. Ten functions are fixed dimension multimodal functions. The seven unimodal functions include Sphere, Schwefel 2.22, Schwefel 1.2, Schwefel 2.21, Rosenbrock, Step, and Quartic. The multimodal functions include Schwefel, Rastrigin, Ackley, Griewank, Penalized, and Penalized-2. The fixed dimension multimodal functions include Foxholes, Kowalik, Six Hump Camel, Branin, Goldstein-Price, Hartman 3, Hartman 6, Shekel

5, Shekel 7, and Shekel 10. The detail of the functions, which consists of formulae, dimension, problem space, and global optimal, is shown in Table II. Meanwhile, the result is shown in Table III.

Table III shows that in general, the proposed algorithm performs well and meets the metaheuristic criteria in finding the near-optimal solution and avoiding the local optimal trap. Moreover, the proposed algorithm can find the true optimal solution in solving five multimodal functions: Shekel Foxholes, Kowalik, Six Hump Camel, Branin, and Goldstein-Price. Unfortunately, its performance is not so good in solving Hartman 3 function.

Compared to other algorithms, the proposed model is competitive enough. Its performance is superior in solving 10 functions. Meanwhile, HOGO has become the most challenging algorithm due to its outstanding performance in solving 9 functions. Compared with MPA, the proposed SMPA-MC is better at solving 13 functions. Most of these are multimodal functions, especially the fixed dimension multimodal functions with narrow problem space. The proposed algorithm also outperforms at least three algorithms in solving 22 functions.

The second simulation is conducted to evaluate the performance of the proposed algorithm in achieving the convergence situation. This simulation is conducted by solving the 23 benchmark functions.  There are three maximum iterations in this simulation: 50, 100, and 150. The result is shown in Table IV.

TABLE II. BENCHMARK FUNCTIONS

| No | Function | Model | Dimension | Problem Space | Global Opt. |
|---|---|---|---|---|---|
| 1 | Sphere | $\sum_{i=1}^{d} x_i^2$ | 10 | [-100, 100] | 0 |
| 2 | Schwefel 2.22 | $\sum_{i=1}^{d}\lvert x_i\rvert + \prod_{i=1}^{d}\lvert x_i\rvert$ | 10 | [-100, 100] | 0 |
| 3 | Schwefel 1.2 | $\sum_{i=1}^{d}\left(\sum_{j=1}^{i} x_j\right)^2$ | 10 | [-100, 100] | 0 |
| 4 | Schwefel 2.21 | $\max\{\lvert x_i\rvert, 1 \le i \le d\}$ | 10 | [-100, 100] | 0 |
| 5 | Rosenbrock | $\sum_{i=1}^{d-1}(100(x_{i+1} + x_i^2)^2 + (x_i - 1)^2)$ | 10 | [-30, 30] | 0 |
| 6 | Step | $\sum_{i=1}^{d-1}(x_i + 0.5)^2$ | 10 | [-100, 100] | 0 |
| 7 | Quartic | $\sum_{i=1}^{d} i\,x_i^4 + random\ [0,1]$ | 10 | [-1.28, 1.28] | 0 |
| 8 | Schwefel | $\sum_{i=1}^{d} -x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$ | 10 | [-500, 500] | -4189.8 |
| 9 | Ratsrigin | $10d + \sum_{i=1}^{d}\left(x_i^2 - 10cos(2\pi x_i)\right)$ | 10 | [-5.12, 5.12] | 0 |
| 10 | Ackley | $-20 \cdot exp\left(-0.2 \cdot \sqrt{\frac{1}{d}\sum_{i=1}^{d} {x_i}^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos 2\pi x_i\right) + 20 + exp(1)$ | 10 | [-32, 32] | 0 |
| 11 | Griewank | $\frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} cos\left(\frac{x_i}{\sqrt{i}}\right)+1$ | 10 | [-600, 600] | 0 |
| 12 | Penalized | $\frac{\pi}{d}\left\{10\sin(\pi y_1) + \sum_{i=1}^{d-1}\left((y_i - 1)^2\left(1 + 10\,sin^2(\pi y_{i+1})\right)\right) + (y_d - 1)^2\right\} + \sum_{i=1}^{d} u(x_i, 10,100,4)$ | 10 | [-50, 50] | 0 |
| 13 | Penalized 2 | $0.1\left\{sin^2(3\pi x_1) + \sum_{i=1}^{d-1}\left((x_i - 1)^2\left(1 + sin^2(3\pi x_{i+1})\right)\right) + (x_d - 1)^2\left(1 + sin^2(2\pi x_d)\right)\right\} + \sum_{i=1}^{d} u(x_i, 5,100,4)$ | 10 | [-50, 50] | 0 |
| 14 | Shekel Foxholes | $\left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$ | 2 | [-65, 65] | 1 |
| 15 | Kowalik | $\sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right)^2$ | 4 | [-5, 5] | 0.0003 |
| 16 | Six Hump Camel | $4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$ | 2 | [-5, 5] | -1.0316 |
| 17 | Branin | $\left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$ | 2 | [-5, 5] | 0.398 |
| 18 | Goldstein-Price | $\left(1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right).\left(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right)$ | 2 | [-2, 2] | 3 |
| 19 | Hartman 3 | $-\sum_{i=1}^{4}\left(c_i exp\left(-\sum_{j=1}^{d}\left(a_{ij}(x_j - p_{ij})^2\right)\right)\right)$ | 3 | [1, 3] | -3.86 |
| 20 | Hartman 6 | $-\sum_{i=1}^{4}\left(c_i exp\left(-\sum_{j=1}^{d}\left(a_{ij}(x_j - p_{ij})^2\right)\right)\right)$ | 6 | [0, 1] | -3.32 |
| 21 | Shekel 5 | $-\sum_{i=1}^{5}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$ | 4 | [0, 10] | -10.1532 |
| 22 | Shekel 7 | $-\sum_{i=1}^{7}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$ | 4 | [0, 10] | -10.4028 |
| 23 | Shekel 10 | $-\sum_{i=1}^{10}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$ | 4 | [0, 10] | -10.5363 |

TABLE III. SIMULATION RESULT (MEANS)

| **Function** | **PSO** | **HS** | **KMA** | **HOGO** | **MPA** | **Proposed Model** | **Better than** |
|---|---|---|---|---|---|---|---|
| Sphere | 329.99 | 505.1458 | 507.3948 | **4.4651e-11** | 0.1662 | 0.1857 | PSO, HS, KMA |
| Schwefel 2.22 | 3.0223e-18 | **0** | 0.0021 | 1.1295e-7 | **0** | **0** | PSO, KMA, HOGO |
| Schwefel 1.2 | 1,410.4218 | 1,203.6542 | 1,677.9713 | **0.1747** | 0.8628 | 13.5649 | PSO, HS, KMA |
| Schwefel 2.21 | 14.2562 | 16.0000 | 14.4289 | **0.0008** | 0.2600 | 0.8221 | PSO, HS, KMA |
| Rosenbrock | 39,750.9494 | 48,923.1329 | 58,963.9907 | **8.6153** | 10.5113 | 21.5729 | PSO, HS, KMA |
| Step | 199.4486 | 327.0324 | 457.9096 | **0.0120** | 2.0221 | 0.0776 | PSO, HS, KMA, MPA |
| Quartic | 0.1062 | 0.0901 | 0.3252 | 0.1152 | **0.0030** | 0.0046 | PSO, HS, KMA, HOGO |
| Schwefel | **-3,342.1276** | -3,328.4096 | -3,172.5575 | -2,758.9118 | -2,031.1057 | -3,212.5081 | KMA, HOGO, MPA |
| Ratsrigin | 31.9483 | 24.1875 | 37.1191 | 14.6673 | **0.1469** | 7.1187 | PSO, HS, KMA, HOGO |
| Ackley | 7.6684 | 9.0253 | 9.4044 | **0.6494** | 0.3033 | 1.1982 | PSO, HS, KMA |
| Griewank | 3.9131 | 5.4470 | 5.9587 | **0.0559** | 0.1842 | 0.3709 | PSO, HS, KMA |
| Penalized | 15.2125 | 17.6490 | 12.2738 | **0.0175** | 0.7669 | 0.5726 | PSO, HS, KMA, MPA |
| Penalized 2 | 17,158.3907 | 7,766.3057 | 6,452.3411 | **0.0864** | 2.7432 | 0.1720 | PSO, HS, KMA, MPA |
| Shekel Foxholes | 10.1559 | **0.9980** | 7.1567 | 5.9722 | 5.1568 | **0.9980** | PSO, HS, KMA, HOGO |
| Kowalik | 0.0102 | 0.0017 | 0.0111 | 0.0047 | 0.0027 | **0.0004** | PSO, HS, KMA, HOGO |
| Six Hump Camel | **-1.0316** | -1.0308 | -1.0278 | -1.0313 | -1.0280 | **-1.0316** | PSO, HS, KMA, HOGO |
| Branin | **0.3980** | 0.3984 | 0.4167 | 0.4086 | 0.8053 | **0.3980** | PSO, HS, KMA, HOGO |
| Goldstein-Price | **3.0000** | **3.0000** | 3.9038 | 3.0195 | 4.1111 | **3.0000** | PSO, HS, KMA, HOGO |
| Hartman 3 | -3.4724 | -0.0495 | -0.7724 | -0.0495 | **-3.7843** | -0.0495 | - |
| Hartman 6 | -3.2106 | -3.2723 | -3.0073 | -3.2157 | -2.0356 | **-3.3221** | PSO, HS, KMA, HOGO |
| Shekel 5 | -6.4334 | -5.8036 | -7.9029 | -4.6978 | -2.0375 | **-10.1310** | PSO, HS, KMA, HOGO |
| Shekel 7 | -5.7752 | -7.4658 | -8.0417 | -5.9069 | -2.4365 | **-10.3892** | PSO, HS, KMA, HOGO |
| Shekel 10 | -4.8883 | -5.0071 | -6.0913 | -6.1175 | -2.2369 | **-10.5257** | PSO, HS, KMA, HOGO |

Table IV shows that in general, the convergence performance of the proposed algorithm is good. It achieves convergence in the early iteration while solving all fixed dimension multimodal functions. Besides, it also achieves convergence in the early iteration in solving the Schwefel 2.22 and Shekel Foxholes functions. Otherwise, it needs a higher maximum iteration to achieve convergence.

The third simulation is conducted to observe the sensitivity of the fishing aggregate devices related to the proposed algorithm's performance. The fishing aggregate devices are chosen due to its role in determining the exploration mechanism. In this simulation, there are three values of the fishing aggregate devices: 0.25, 0.5, and 0.75. These values represent the low, moderate, and high fishing aggregate devices. The result is shown in Table V.

Table V shows that the sensitivity of the fishing aggregate devices is various depend on the problem to solve. The increase of the fishing aggregate devices worsens the proposed algorithm's performance in solving the most of unimodal functions, except Schwefel 2.22. On the other hand, the fishing aggregate devices do not affect the proposed algorithm's performance in solving most of the multimodal functions.

The fourth simulation is conducted to evaluate the sensitivity of the number of candidates related to the proposed algorithm's performance. In this simulation, there are three values of the number of candidates: 5, 10, and 15. These values represent the low, moderate, and high number of candidates. The result is shown in Table VI.

Table VI shows that in general, the number of candidates has positive relation the proposed algorithm's performance. The increase of the number of candidates tends to improve the performance. This circumstance occurs in all functions: unimodal functions and multimodal functions. In the beginning, the improvement is significant. But, after the algorithm reaches its peak performance, the improvement is less significant. In some functions, such as Schwefel 2.22 and Goldstein-Price, the peak performance is achieved in the small number of candidates.

The fifth simulation is conducted to evaluate the performance of the proposed algorithm in solving the real-world optimization problem. An algorithm test using a real-world optimization problem is needed to prove that the algorithm is good theoretically and practically. In this simulation, the proposed algorithm is implemented to optimize the production planning process in a manufacturing company.

TABLE IV. CONVERGENCE TEST RESULT

| Function | Average Fitness Score | | |
|---|---|---|---|
| | $t_{max} = 50$ | $t_{max} = 100$ | $t_{max} = 150$ |
| Sphere | 46.4677 | 3.2242 | 0.7576 |
| Schwefel 2.22 | 0.0020 | 0 | 0 |
| Schwefel 1.2 | 148.1581 | 57.4689 | 21.4743 |
| Schwefel 2.21 | 4.8131 | 2.5311 | 1.3052 |
| Rosenbrock | 652.8542 | 131.7495 | 54.1151 |
| Step | 24.0461 | 2.0388 | 0.2480 |
| Quartic | 0.0145 | 0.0077 | 0.0051 |
| Schwefel | -2,639.4568 | -2,975.1690 | -3,115.1478 |
| Ratsrigin | 9.5166 | 10.0503 | 7.2051 |
| Ackley | 4.4461 | 2.5475 | 1.7403 |
| Griewank | 1.3237 | 0.7802 | 0.5090 |
| Penalized | 2.6256 | 0.9555 | 0.5157 |
| Penalized 2 | 9.6735 | 1.6905 | 0.2997 |
| Shekel Foxholes | 1.0821 | 0.9980 | 0.9980 |
| Kowalik | 0.0008 | 0.0004 | 0.0004 |
| Six Hump Camel | -1.0316 | -1.0316 | -1.0316 |
| Branin | 0.3981 | 0.3981 | 0.3981 |
| Goldstein-Price | 3.0000 | 3.0000 | 3.0000 |
| Hartman 3 | -0.0495 | -0.0495 | -0.0495 |
| Hartman 6 | -3.2780 | -3.3199 | -3.3217 |
| Shekel 5 | -9.6085 | -10.0459 | -10.0976 |
| Shekel 7 | -10.1142 | -10.1338 | -10.1073 |
| Shekel 10 | -10.0074 | -10.4680 | -10.5128 |

TABLE V. RELATION BETWEEN FISHING AGGREGATE DEVICES AND FITNESS SCORE

| Function | Average Fitness Score | | |
|---|---|---|---|
| | *fad* = 0.25 | *fad* = 0.5 | *fad* = 0.75 |
| Sphere | 4.3085 | 7.4816 | 13.1763 |
| Schwefel 2.22 | 0 | 0 | 0 |
| Schwefel 1.2 | 40.6408 | 58.1769 | 124.8039 |
| Schwefel 2.21 | 2.4441 | 2.9302 | 3.5865 |
| Rosenbrock | 131.9147 | 105.0508 | 305.1955 |
| Step | 1.5093 | 5.1901 | 10.8524 |
| Quartic | 0.0077 | 0.0134 | 0.0195 |
| Schwefel | -2,955.7188 | -3,184.1091 | -3,358.1613 |
| Ratsrigin | 10.4572 | 15.6704 | 19.3374 |
| Ackley | 2.6340 | 2.8596 | 3.2112 |
| Griewank | 0.8104 | 0.8855 | 1.1047 |
| Penalized | 1.1464 | 0.5891 | 1.3944 |
| Penalized 2 | 1.1628 | 1.7262 | 3.2667 |
| Shekel Foxholes | 0.9981 | 0.9980 | 0.9980 |
| Kowalik | 0.0004 | 0.0010 | 0.0009 |
| Six Hump Camel | -1.0316 | -1.0316 | -1.0316 |
| Branin | 0.3981 | 0.3981 | 0.3981 |
| Goldstein-Price | 3.0000 | 3.0000 | 3.0002 |
| Hartman 3 | -0.0495 | -0.0495 | -0.0495 |
| Hartman 6 | -3.3202 | -3.3209 | -3.3211 |
| Shekel 5 | -10.0811 | -10.0605 | -10.0372 |
| Shekel 7 | -10.3100 | -10.3351 | -10.2997 |
| Shekel 10 | -10.4786 | -10.4635 | -10.4343 |

TABLE VI. RELATION BETWEEN NUMBER OF CANDIDATES AND FITNESS SCORE

| Function | Average Fitness Score | | |
|---|---|---|---|
| | $n(C) = 5$ | $n(C) = 10$ | $n(C) = 15$ |
| Sphere | 10.2168 | 4.2298 | 1.5692 |
| Schwefel 2.22 | 0 | 0 | 0 |
| Schwefel 1.2 | 63.8637 | 54.9011 | 41.1862 |
| Schwefel 2.21 | 3.7031 | 2.5755 | 2.4947 |
| Rosenbrock | 173.8898 | 167.1175 | 98.5991 |
| Step | 4.2628 | 2.3263 | 1.2434 |
| Quartic | 0.0111 | 0.0017 | 0.0068 |
| Schwefel | -2,584.2545 | -3,054.2677 | -3,016.2328 |
| Ratsrigin | 17.5424 | 11.0106 | 8.1328 |
| Ackley | 3.1073 | 2.3593 | 2.4817 |
| Griewank | 0.9317 | 0.6514 | 0.7018 |
| Penalized | 1.2564 | 0.8481 | 0.5831 |
| Penalized 2 | 2.7543 | 0.8494 | 1.0848 |
| Shekel Foxholes | 1.3601 | 0.9980 | 0.9980 |
| Kowalik | 0.0008 | 0.0004 | 0.0004 |
| Six Hump Camel | -1.0316 | -1.0316 | -1.0316 |
| Branin | 0.3980 | 0.3980 | 0.3980 |
| Goldstein-Price | 3.0000 | 3.0000 | 3.0000 |
| Hartman 3 | -0.0495 | -0.0495 | -0.0495 |
| Hartman 6 | -3.0389 | -3.3195 | -3.3216 |
| Shekel 5 | -9.8176 | -10.1143 | -10.0888 |
| Shekel 7 | -10.3370 | -10.3906 | -10.3829 |
| Shekel 10 | -10.3174 | -10.4665 | -10.4899 |

The simulation scenario is Muslim socks manufacturer in Bandung, Indonesia. This company produces 40 product items. Half of them are long socks, while half others are short socks. Six items are fast-moving products while the others are moderate ones. The most fast-moving products are the light brown socks, both short and long. The other fast-moving products are white socks and black socks. Each item should be produced within the minimum and maximum production ranges. On the other hand, there is a limitation in the storage and financial capacity so that all produced socks cannot surpass the total production quantity. The maximum total capacity is only 5,250 dozen. The characteristics of every item are shown in Table VII. The production quantity is presented in dozen while the price is presented in rupiah. The objective is to maximize total gross profit.

TABLE VII. PRODUCT DESCRIPTION

| Product ID | Min. Production (dozen) | Max. Production (dozen) | Gross Profit (rupiah/dozen) |
|---|---|---|---|
| 1 | 500 | 1000 | 24,000 |
| 2, 3 | 200 | 400 | 24,000 |
| 4-20 | 50 | 100 | 24,000 |
| 21 | 500 | 1000 | 30,000 |
| 22, 23 | 200 | 400 | 30,000 |
| 24-40 | 50 | 100 | 30,000 |

This optimization problem can be seen as a Knapsack optimization problem. The concept of the Knapsack problem is that there is a space with a limited capacity [26]. On the other hand, there are several products to pick up. The objective is to determine the items, and the quantity picked to minimize or maximize the objective parameters.

The proposed algorithm is compared with PSO, HS, KMA, and MPA in this simulation. Due to its characteristic as a multi-dimension problem with a large search space, the maximum iteration for PSO, KMA, MPA, and SMPA-MC is set at 300. Meanwhile, the maximum iteration for HS is set at 12.000. The result is shown in Table VIII.

TABLE VIII. REAL-WORLD PRODUCTION PLANNING SIMULATION RESULT

| Algorithm | Total Gross Profit (rupiah) |
|---|---|
| proposed (SMPA-MC) | 186,589,200 |
| PSO | 171,114,000 |
| HS | 188,959,800 |
| MPA | 157,104,600 |
| KMA | 143,143,100 |

Table VIII shows that the proposed SMPA-MC outperforms three algorithms in creating better total gross profit. Its total gross profit is 9% higher than PSO, 19% higher than MPA, and 30% higher than KMA. On the other hand, its performance is only 1% lower than HS. This result shows that although HS is inferior in solving a theoretical mathematic problem, its performance is superior in solving real-world high dimension problems.

## V. DISCUSSION

In general, Table III shows that the proposed SMPA-MC algorithm is better than the original MPA in solving multimodal functions. Its superiority especially occurs in solving multimodal functions with low dimension and narrow problem space, as indicated by the last ten functions. On the other hand, MPA is better at solving unimodal functions. The proposed algorithm is better at avoiding local optimal trap (exploration), while the MPA is better at finding the near-optimal solution or more precise solution. In the context of the method used in these algorithms, it is shown that the Levy movement creates more precise solutions than a uniform random or simple random walk.

There are several notes due to the competitiveness of the proposed algorithm. All metaheuristic algorithms use iteration to improve their current solution [6]. The result in Table IV shows this circumstance. Some functions can be solved faster, while others need more iteration, such as high dimension functions or functions with large problem space. Besides, an algorithm may be better in the early iteration, which means they are better in finding the convergence. On the other hand, some algorithms may be worse in the early iteration but better in the long run.

Metaheuristic is also identic with adjusted parameters. These parameters are provided to tune the algorithm's performance in the adaptation of many optimization problems. The inferior performance of PSO, HS, and KMA in Table III may come from the adjustment. By implementing different adjustments, an algorithm may perform better or worse depending on the problem it faces. It means that competing with one algorithm with the others is not the only tool to judge the algorithm's performance.

The adjustment also affects to the performance as it is shown in Table V and Table VI. Although exploration is important to avoid the local optimal trap, targeted exploration is proven more effective rather than the fully randomized exploration, especially in the later iteration. Higher fishing aggregate devices makes the probability of the fully randomized exploration higher. It means, the searching process will restart at location somewhere in the problem space and it is not productive in the later iteration. In some circumstance, the number of candidates gives positive results. But, after the algorithm reaches its peak performance, the increase of the number of candidates does not improve the algorithm's performance significantly.

Table III also strengthens the no-free-lunch theory [27]. Although, in general, PSO and HS are inferior compared to HOGO, MPA, and the proposed SMPA-MC, they are still superior in solving several functions. PSO is superior in solving four functions, while HS is superior and can find the true optimal solution in solving three functions.

As shown in Table VIII, the real-world simulation result demonstrates that superiority in solving a high-precision mathematical problem may not work in solving real-world problems. In theoretical mathematic problems, the parameters are usually represented in floating-point numbers. Very little difference between two floating-point numbers may give a significant gap in the result. An algorithm can achieve better performance by generating a more precise floating-point number. This process can usually be conducted by making small and high-precision step sizes during the guided movement. Small step size is usually achieved by generating a more precise random number, for example by using Levy movement or normal distribution. On the other hand, uniform random is usually less precise.

On the other hand, many real-world problems do not need very precise floating-point numbers. Many of them usually use integer numbers, especially in operations research. Many studies in operations research use integer numbers, for example, to find the number of products that should be produced or ordered. It is impossible to produce goods, for example, shoes, cars, and so on, in a fractional quantity. This circumstance makes the high precision optimization algorithm, such as KMA, MPA, or HOGO, lose their advantage.

Moreover, the objective function in real-world problems is simpler than the theoretical mathematic problems. In real-world problems, especially in operations research problems, most of their objectives can be presented in multi-variate linear functions, such as minimizing total tardiness [28], production cost [29], travel distance [30], and so on. This objective can be achieved by accumulating these parameters in all dimensions, for example by accumulating all due date penalties of all executed orders or accumulating the total quantity of all unexecuted orders due to limited production or storage capacity. This objective is even simpler than Sphere and Schwefel 2.22 functions, which are the simplest among 23 benchmark functions. But in real-world problems, some optimization problems use a multi-objective model.

This circumstance is also related to the popularity of the algorithm. Many studies in optimization, especially operations research, still use old-fashioned algorithms, such as genetic algorithm, tabu search, simulated annealing, or variable neighborhood search. This phenomenon indicates that these algorithms are still well-proven and competitive enough to solve real-world problems. However, they are often beaten by the shortcoming algorithms in solving mathematical functions. Besides, the mechanism of these old-fashioned algorithms is simple so that they are easy to modify or hybridize.

## VI. Conclusion

This work has demonstrated that the proposed algorithm, the stochastic marine predator algorithm with multiple candidates, has proven as a good metaheuristic algorithm. It has achieved two main objectives of metaheuristic algorithm: finding a near-optimal solution and tackling the local optimal. The simulation result shows that its performance is competitive in solving optimization problems theoretically and practically. Among 23 benchmark functions, it achieves true optimal solution in solving 5 functions. Compared with other algorithms, its performance is also superior in solving 10 functions. This algorithm also outperforms the original form of the marine predator algorithm in solving 13 functions, which means 57 percent of total functions. Practically, it is also competitive in solving real-world problems. It outperforms particle swarm optimization, marine predator algorithm, and Komodo mlipir algorithm in optimizing production planning problems. Its performance is 9%, 19%, and 30% better than these three algorithms consecutively.

This work has shown that improving the existing algorithm is also important compared to proposing a new algorithm. This improvement can be conducted by modifying the current form of the algorithm or hybridizing this algorithm with another algorithm to combine the advantage of every algorithm. In the future, modifying the marine predator algorithm is still possible and challenging. Besides, implementing this proposed algorithm to solve more real-world optimization problems is still potential, especially in solving combinatorial problems, such as scheduling, timetabling, etc.

## Acknowledgment

This work was financially supported by Telkom University, Indonesia.

## References

[1] V. Fernandez-Viagas, “A speed-up procedure for the hybrid flow shop scheduling problem”, Expert Systems with Applications, vol. 187, ID: 115903, pp. 1-11, 2022.

[2] T. Chargui, A. Bekrar, M. Reghioui, and D. Trentesaux, “Scheduling trucks and storage operations in a multiple-door cross-docking terminal considering multiple storage zones”, International Journal of Production Research, vol. 60, no. 4, pp. 1153-1177, 2022.

[3] K. Huang, L. Xu, Y. Chen, Q. Cheng, and K. An, “Customized bus route optimization with the real-time data”, Journal of Advanced Transportation, vol. 2020, ID: 8838994, pp. 1-9, 2020.

[4] 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, pp. 121-127, 2021.

[5] A. Mahmoudi, L. Hashemi, M. Jasemi, and J. Pope, “A comparison on particle swarm optimization and genetic algorithm performances in deriving the efficient frontier of stocks portofolios based on a mean-lower partial moment model”, International Journal of Finance & Economics, vol. 26, no. 4, pp. 5659-5665, 2020.

[6] 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.

[7] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer”, Advances in Engineering Software, vol. 69, pp. 46-61, 2014.

[8] S. Mirjalili, “Dragonfly algorithm: a meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems”, Neural Computing and Applications, vol. 27, no. 4, pp. 1053-1073, 2016.

[9] S. Mirjalili and A. Lewis, “The whale optimization algorithm”, Adcances in Engineering Software, vol. 95, pp. 51-67, 2016.

[10] Suyanto, A. A. Ariyanto, and A. F. Ariyanto, “Komodo mlipir algorithm”, Applied Soft Computing, vol. 114, pp. 1-17, 2022.

[11] 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.

[12] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, “Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems”, Engineering with Computers, vol. 29, pp. 17-35, 2013.

[13] S. Arora and S. Singh, “Butterfly optimization algorithm: a novel approach for global optimization”, Soft Computing, vol. 23, pp. 715-734, 2019.

[14] M. A. Farag, M. A. El-Shorbagy, I. Eldesoky, and A. A. El-Sawy, “Binary-real coded genetic algorithm based k-means clustering for unit commitment problem”, Applied Mathematics, vol. 6, no. 11, pp. 1873-1890, 2015.

[15] K. Deb, A. Pratap, S. Agarwai, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA II”, IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, 2002.

[16] Z. Sylia, C. Gueguen, B. Lamine, T. Larbi, and K. Abdelkrim, “New strategy for resource allocation using PSO-PFS hybrid”, International Journal of Wireless and Mobile Computing, vol. 18, no. 2, pp. 175-182, 2020.

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

[18] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, “Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications”, IEEE Transactions on Industrial Informatics, vol. 17, no. 7, pp. 5068-5076, 2021.

[19] M. A. Sobhy, A. Y. Abdelaziz, H. M. Hasanien, and M. Ezzat, “Marine predators algorithm for load frequency control of modern interconnected power systems including renewable energy sources and energy storage units”, Ain Shams Engineering Journal, vol. 12, no. 4, pp. 3843-3857, 2021.

[20] G. Chen, Y. Xiao, F. Long, X. Hu, and H. Long, “An improved marine predators algorithm for short-term hydrothermal scheduling”, IAENG International Journal of Applied Mathematics, vol. 51, no. 4, pp. 1-14, 2021.

[21] N. E. Humphries, N. Queiroz, J. R. M. Dyer, N. G. Pade, M. K. Musyl, K. M. Schaefer, D. W. Fuller, J. M. Brunnschweiler, T. K. Doyle, J. D. R. Houghton, G. C. Hays, C. S. Jones, L. R. Noble, V. J. Wearmouth, E. J. Southall, and D. W. Sims, “Environmental context explains Levy and Brownian movement patterns of marine predators”, Nature, vol. 465, pp. 1066-1069, 2010.

[22] T. Guilmeau, E. Chouzenoux, and V. Elvira, “Simulated annealing: a review and a new scheme”, IEEE Statistical Signal Processing Workshop (SSP), Rio de Janeiro, Brazil, 2021.

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

[24] 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.

[25] 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 & Systems, vol. 13, no. 4, pp. 216-225, 2020.

[26] L. E. Schafer, T. Dietz, M. Barbati, J. R. Figueira, S. Greco, and S. Ruzika, “The binary knapsack problem with qualitative levels”, European Journal of Operations Research, vol. 289, no. 2, pp. 508-514, 2021.

[27] 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.

[28] B. Mor, G. Mosheiov, and D. Shabtay, “Minimizing the total tardiness and job rejection cost in a proportionate flow shop with generalized due dates”, Journal of Scheduling, vol. 24, no. 6, pp. 553-567, 2021.

[29] H. M. Al-Hattami, J. D. Kabra, and M. Lokhande, “Reducing cost in manufacturing firm by using target costing technique”, International Journal of Business Excellence, vol. 22, no. 1, pp. 69-82, 2020.

[30] P. D. Kusuma and M. Kallista, “Dispatching model in online taxi system by using stable marriage and first price sealed bid auction models”, International Journal of Intelligent Engineering & Systems, vol. 14, no. 5, pp. 166-176, 2021.