Hybrid Pelican Komodo Algorithm

Purba Daru Kusuma, Ashri Dinimaharawati
Computer Engineering, Telkom University, Bandung, Indonesia

***Abstract*—In this work, a new metaheuristic algorithm, namely the hybrid pelican Komodo algorithm (HPKA), has been proposed. This algorithm is developed by hybridizing two shortcoming metaheuristic algorithms: the Pelican Optimization Algorithm (POA) and Komodo Mlipir Algorithm (KMA). Through hybridization, the proposed algorithm is designed to adapt the advantages of both POA and KMA. Several improvisations regarding this proposed algorithm are as follows. First, this proposed algorithm replaces the randomized target with the preferred target in the first phase. Second, four possible movements are selected stochastically in the first phase. Third, in the second phase, the proposed algorithm replaces the agent's current location with the problem space width to control the local problem space. This proposed algorithm is then challenged to tackle theoretical and real-world optimization problems. The result shows that the proposed algorithm is better than grey wolf optimizer (GWO), marine predator algorithm (MPA), KMA, and POA in solving 14, 12, 14, and 18 functions. Meanwhile, the proposed algorithm creates 109%, 46%, 47%, and 1% better total capital gain rather than GWO, MPA, KMA, and POA, respectively in solving the portfolio optimization problem.**

Keywords*—Metaheuristic; Pelican Optimization Algorithm; Komodo Mlipir Algorithm; portfolio optimization algorithm; LQ45 index

I. Introduction

Optimization is a prevalent work that has been implemented in many areas. Optimization is essential because it aims to maximize results or minimize cost or effort. Optimization is also important because in any human process, whether the scope is individual or institution, it has a specific goal or objective. Contrary, the resources needed to execute this work are limited. The term cost can be translated in many ways, such as travel distance, consumed energy, production cost, penalty, unserved requests, and so on. On the other hand, the term result can also be translated into many ways, such as sales, profit, served customers, accuracy, and so on. In the production process, optimization is widely used, such as in the flow-shop scheduling [1], batch-shop scheduling [2], assembly line balancing [3], procurement [4], and so on. In transportation and logistics, optimization is implemented in route planning [5], storage management [6], and so on. Optimization is also implemented in finance, such as in portfolio optimization [7], option pricing [8], credit risk assessment [9], bankruptcy mitigation [10], etc.

Metaheuristic algorithm is a popular method used in many studies conducting the optimization problem. This popularity comes from its flexibility in facing the limited computation resources. Moreover, the metaheuristic algorithm is flexible enough to tackle various objective functions, from simple to complicated ones. This advantage cannot be obtained from the exact method that needs an excessive computational resource, especially in solving a complicated problem with high dimension space [11]. However, as an approximate method, a metaheuristic algorithm does not guarantee the true optimal solution but only the acceptable or near optimal one [11]. In many metaheuristic algorithms, several parameters also must be adjusted. Proper adjustment can improve its performance, while misjudgment can worsen its performance.

Many metaheuristic algorithms are inspired by nature or behavior, especially the animal behavior during mating and foraging. This circumstance occurs due to the similarity between these behaviors and the metaheuristic algorithm. An animal has a certain degree of uncertainty during mating and foraging. In foraging, even if it is searching for a food source or hunting prey, the animal still does not know the actual location of the food source or prey. Based on it, a random search with a certain degree of certainty is conducted. Although animals have a certain degree of similarity during foraging, there is a specific strategy conducted by every animal. On the other hand, the mating process can generate new descendants from the selected parents. These descendants inherit the characteristics of their parents. Some descendants are better than their parents while the others are worse. Several metaheuristic algorithms adopt this circumstance. In several algorithms, the improvement is created by mating a selected solution with the best solution. Several algorithms that adopt foraging behavior are particle swarm optimization (PSO) [12], ant colony optimization (ACO) [13], grey wolf optimization (GWO) [14], marine predator algorithm (MPA) [15], artificial bee colony algorithm (ABC) [16], Etc. Meanwhile, several algorithms that adopt the mating process are genetic algorithm (GA) [17], evolutionary algorithm (EA) [18], Etc. Several algorithms, such as the red deer algorithm (RDA), combine mating and foraging [19].

Among many shortcoming metaheuristic algorithms, there are two brand-new algorithms that is firstly introduced in 2022. The first is Komodo Mlipir Algorithm (KMA), and the second is the Pelican Optimization Algorithm (POA). The animal's behavior inspires both algorithms. The behavior of Komodo dragon during foraging and mating inspires KMA [20]. Meanwhile, POA is inspired by the behavior of pelicans during foraging [21]. In their first appearance, both algorithms beat several algorithms. POA outperformed genetic algorithm (GA), particle swarm optimization (PSO), teaching-learning based optimization (TLBO), grey wolf optimizer (GWO), whale optimization algorithm (WOA), gravitational search algorithm (GSA), tunicate swarm algorithm (TSA), and marine predator algorithm (MPA) [21]. On the other hand, KMA outperformed six algorithms: GA, success-history based parameter adaptation differential evolution (SHADE), linear population size reduction SHADE with ensemble sinusoidal differential

covariance matrix adaptation with Euclidean neighborhood (LSHADE-CnEpSIn), equilibrium Optimizer (EO), MPA, and slime mold algorithm (SMA) [20].

Despite their outstanding performance, these algorithms are still not popular as brand-new algorithms. Studies conducting these algorithms to solve optimization problems are still hard to find. Based on this, it is challenging to explore these algorithms further. Moreover, as brand-new algorithms, the opportunity to improve and modify these algorithms is widely open.

The objective and scope of this work are as follows. This work proposes a new metaheuristic algorithm that hybridizes both shortcoming algorithms: POA and KMA. Through hybridization, the proposed algorithm is hoped to combine the strength of both algorithms and, on the other hand, tackle the weakness of both algorithms too. Based on this objective, the scope of this work is to develop new algorithms that hybridize both POA and KMA and then evaluating this proposed algorithm through simulation.

The methodology conducted to this work is as follows. First, the mechanics and strategy in both KMA and POA are explored and reviewed. This exploration is needed to analyze their strength and weakness. Then, the proposed algorithm is developed by hybridizing both algorithms. After that, this proposed algorithm is challenged to solve the theoretical and real-world optimization problems so that its performance can be evaluated. The proposed algorithm is implemented to solve the 23 benchmark functions. These functions represent the theoretical optimization problem. These functions are popular in many studies that propose a new metaheuristic algorithm. Meanwhile, the portfolio optimization problem is chosen as the real-world optimization problem. In this simulation, the proposed algorithm is compared with four shortcoming metaheuristic algorithms: GWO, MPA, KMA, and POA. GWO and MPA represent algorithms that have been implemented and modified in many studies. Meanwhile, KMA and POA are chosen because this proposed algorithm is the improved version of these algorithms. Several findings regarding the simulation result are then analyzed deeper.

There are several contributions regarding this work. These contributions are as follows.

1) This work proposes a new algorithm that hybridizes two brand-new algorithms: POA and KMA.

2) This work modifies the swarm movement in the first phase of POA by replacing the randomized target with a more deterministic target.

3) This work adopts the behavior of three types of Komodo in KMA to be implemented in the swarm movement in the first phase with several modifications.

4) This work modifies the second phase by replacing the agent's current location with the problem space to control the local problem space.

The remainder of this paper is structured as follows. The mechanics of POA and KMA are reviewed in the second section to analyze their strengths and weaknesses. Based on this review, the proposed algorithm's model is presented in the third section. The simulation regarding the proposed algorithm is explained in the fourth section. The more profound analysis regarding the simulation result and the findings are discussed in the fifth section. In the end, the conclusion and future research potential regarding this work are summarized in the sixth section.

II. Related Work

Komodo Mlipir Algorithm (KMA) is a brand-new algorithm that adopts the behavior of the Komodo dragon during mating and foraging. This algorithm is a population-based algorithm consisting of several autonomous agents. Each agent represents the solution. These agents are classified into three groups based on their quality: big male, female, and small male [20]. Each type of agent has a specific role and mechanics. The big males are agents whose qualities are better. The females are agents whose quality is mediocre. In the end, the petite males are agents whose quality is worse. The proportion of these groups is fixed and set manually before the process begins. The rank to determine the group's members is updated in every iteration.

The big male adopts foraging behavior by searching for prey [20]. The big male moves based on its current location and other big males. The big male moves toward the better big males and moves away from the worse big males. The big male does not take account of the female and small male.

The female conducts the mating process. There are two possible mating strategies for every female: sexual reproduction or asexual reproduction (parthenogenesis) [20]. Sexual reproduction is achieved by mating the female with the highest quality big male. Each female produces two descendants. The first descendant is close to the female, while the second descendant is close to the highest quality big male. Then, the best descendant between them will replace the female current's location. In parthenogenesis, a female creates a descendant randomly within the problem space.

Like a big male, the small male implements foraging [20]. The small male moves toward the cumulative of big males. As a worse solution, the small male should get closer to the better solutions (big males) to improve its quality.

Meanwhile, the Pelican Optimization Algorithm is a brand-new algorithm that adopts the pelican behavior during foraging. POA is a swarm-based intelligence. This algorithm consists of a certain number of agents (pelicans). As a swarm intelligence, collective intelligence is used or shared among the pelicans [22]. In this algorithm, the randomized target represents collective intelligence. POA consists of two steps that are executed sequentially in every iteration.

There is a global target in the first phase where all pelicans will move based on this target [21]. This global target is selected randomly within the problem space at the beginning of every iteration. The pelican can choose two possible movements. If this target is better than the pelican's current location, the pelican will move toward this target. Otherwise, the pelican will move away from this target. In POA, an acceptance-rejection strategy is adopted. The pelican will move to this new location only if this new location is better than its current location.

In the second phase, the pelican flies around its current location [21]. Although this term is not relevant in some circumstances, it can be seen as a local or neighborhood search. In this phase, a new location is selected randomly within the pelican's local problem space. The width of this local problem space declines gradually due to the increase of the iteration. It means that the local problem space is wide enough in the beginning, and it can be seen as an exploration. On the other hand, this investigation moves to exploitation as the iteration goes. Besides iteration, the local problem space is also determined by the agent's current location. Near zero current location makes the width of the local problem space narrow, although in the early iteration. Like in the first phase, in this phase, the pelican moves toward the new location only if this new location is better than the pelican's current location.

Based on the detailed description of KMA and POA, the comprehensive comparison between these algorithms is as follows. KMA splits the population into three groups. Each group represents a distinct strategy. But each agent conducts only a single procedure in every iteration. On the other hand, in POA, there is not any population split. Every agent is treated equal and conducts the same strategy. Each agent acts these two actions in every iteration.

The swarm movement toward a better solution and away from the worse solution is also conducted in both algorithms. In KMA, the big males move toward better big males and move away from the resultant of worse big males. Moreover, petite males move toward the resultant of big males. On the other hand, each pelican moves toward a randomized target if this target is better than the pelican's current location and avoids this randomized target if this target is worse than the pelican's current location. This strategy can be seen as improving the current solution based on the guidance of the better solution or avoiding the possible worse solution. Both algorithms choose a different method in determining the target in the swarm movement. POA selects the target randomly within the problem space. On the other hand, in KMA, only big males can become the target.

Random search is also conducted in both algorithms but in a different way. In POA, this strategy is undertaken in the second phase so that all agents work this strategy in every iteration. In KMA, the random search is implemented only by the female when it chooses parthenogenesis. It means that with the same population size, the probability of conducting the random search in POA is higher than in KMA.

There is a difference between KMA and POA regarding the local problem space in the local search strategy. In KMA, the local problem space width is fixed based on the problem space. In POA, the local problem space is reduced gradually as the iteration increases. Reducing the local problem space during the iteration can make the system focus on the exploration in the early iteration and then transform to the exploitation. At the end of the iteration, the system focuses on exploitation. The advantage of this strategy is that the system can concentrate on exploring any space within the problem space to find the region where the optimal global solution exists. After that, the system will improve the solution within this region. Moreover, the agent will not be thrown away to any areas within the iteration in the later iteration, so it should start the searching. Contrary, fixed local problem space width is essential when the system still fails to find the region where the optimal global solution exists. The system can escape from the optimal local trap, although the iteration is not in the early phase.

Acceptance-rejection strategy is conducted only in POA. Meanwhile, KMA does not adopt this strategy. Acceptance-rejection has strengths and weaknesses, so not all algorithms adopt this strategy. By implementing this strategy, the agent moves to a new solution only if the new solution is better than its current solution. There is no probability of a worsening situation. But the system may be stuck in a case, such as the local optimal, when it fails to improve the current solution. On the other hand, without accepting this strategy, the system may go to a worse situation. Some algorithms, such as MPA, partially adopt this strategy. In MPA, the prey may move toward the worse solution. Contrary, the predator moves to a new location, only this new location is better than the predator's current location.

This review shows that both KMA and POA have several strengths and weaknesses. Based on this circumstance, there is the possibility of improvement by hybridizing these algorithms. Several parts that can be modified are as follows. First, modification can be conducted in the swarm movement. Second, change also can be shown in the random search.

III. Proposed Model

This section will present the detailed model of the proposed algorithm. This model consists of the conceptual model, pseudocode, and mathematical model. The conceptual model explains the framework and general mechanics of the algorithm. The pseudocode formalizes the structure of the proposed algorithm. In the end, the mathematical model describes the detailed formulation of processes and methods within the algorithm.

The conceptual model of the proposed algorithm is as follows. This proposed algorithm uses POA as its main framework. The proposed algorithm consists of two phases. The first phase is the swarm movement toward the target. The second phase is the randomized movement within the local problem space. Like in POA, these phases are conducted sequentially in every iteration.

In the first phase, four possible movements can be chosen by every agent. The first movement is the movement toward the global best solution. The second movement moves to the middle between the current location and the international best solution. The third movement is the movement related to the randomly selected agent. The fourth movement is jumping across the global best solution. KMA inspires the first, second, and third movements. The first movement is the modification of the minor male movement. The second movement is the modification of the mating process of the female with the best quality big male. The third movement is the modification of the significant male movement. MPA inspires the fourth movement. In the fourth movement, the agent's new location is obtained based on the current global best solution movement away from the related agent. The main objective of the fourth movement is to improve the global best solution. In this first

phase, the agent will move to the new location, whether this new location is better or worse. It is different from POA, where the pelican will move to the new location only if this new location is better than the pelican's current location.

In the second phase, every agent searches for a new location within its local problem space. This method also occurs in POA. In this phase, the similarity between the proposed algorithm and POA is that the local problem space is reduced gradually due to the increase of the iteration. The exploration to exploitation strategy is also adopted in the proposed algorithm. Meanwhile, there is a difference between the proposed algorithm and POA. In this proposed algorithm, the local problem space width also depends on the problem space width. It is different from POA, where the agent's current location affects its local problem space width. Like in POA, in this phase, the agent moves to the new location only if it is better than the current location.

Like many metaheuristic algorithms, this proposed algorithm consists of two steps. The first step is initialization. The second step is iteration. The agent's initial location is randomized within the problem space during the initialization. It follows uniform distribution so that the opportunity of every place is equal. The improvement is conducted during the iteration. Each time an agent moves to a new location, the global best solution will be updated in every process. The global best solution is an entity that stores the best answer so far. This best solution is applied among all agents. This international best solution is updated to its new value only if this new solution is better than the current global best solution. The global best solution becomes the final solution at the end of an iteration.

This framework is then transformed into the pseudocode and the mathematical model. The pseudocode of the proposed algorithm is shown in Algorithm 1. There are several annotations used in the pseudocode and mathematical model. These annotations are as follows.

b_l	lower bound
b_u	upper bound
d	space divider
f	objective function
r	generated random number
s	step size
x	agent
X	set of agents
x_c	candidate
x_{tar}	target
x_{sel}	selected agent
x_{best}	global best solution
t	iteration
t_{max}	maximum iteration
T_1	first threshold
T_2	second threshold
T_3	third threshold
U	uniform distribution

Algorithm 1: HPKA Algorithm

```
output: x_best
begin
 //initialization
 for all X do
 initialize x using (1)
 end for
 //iteration
 for t=1 to t_max do
 for all X do
 //first phase
 generate r using (2)
 if r < T_1 then
  first movement using (3)
 else if T_1 ≤ r < T_2 then
  second movement using (4)
 else if T_2 ≤ r < T_3 then
  third movement using (5) and (6)
 else
  fourth movement using (7)
 end if
 update x_best using (8)
 //second phase
 search within local problem space using (9) and (10)
 update x_best using (8)
 end for
end
```

All agents' initial location is determined randomly within the problem space in the initialization. This process is formalized using (1). Equation (1) shows that the lower and upper bound to become the boundaries of the problem space. These boundaries represent the single dimension problem space. Each dimension has its limits in the multiple dimension problem space, and (1) is applied in all dimensions.

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

In the first phase, the movement is selected randomly based on the value of a generated random number. The distribution of this random number follows a uniform distribution. This random number is formalized by using (2). Then, the movement is selected based on the location of this random number related to the thresholds.

$$r = U(0,1) \tag{2}$$

In the first movement, the agent moves toward the global best solution. This first movement is chosen if the generated random number is less than the first threshold. This process is formalized by using (3). Equation (3) shows that the movement length is uniformly randomized. It also depends on the step size. A bigger step size makes the agent moves closer to the global best solution. On the other hand, a smaller step size makes the agent moves closer to its current location.

$$x' = x + s.U(0,1).(x_{best} - x) \tag{3}$$

In the second movement, the agent moves to the middle between its current location and the global best solution. This movement is chosen if the generated random number is between the first and second threshold. This movement represents the deterministic version of the first movement. This movement is formalized using (4).

$$x' = \frac{x_{best}+x}{2} \quad (4)$$

In the third movement, the agent moves related to the selected agent. This agent is chosen randomly among the set of agents. This movement is selected if the generated random number is between the second and third threshold. If this agent chosen is better than the agent's current location, then this agent will move toward the selected agent. Else, this agent will move away from the designated agent. This process is formalized by using (5) and (6). Equation (5) formalizes the agent selection. Equation (6) formalized the movement related to the selected agent.

$$x_{sel} = U(X) \quad (5)$$

$$x' = \begin{cases} x + s.U(0,1).(x_{sel} - x), f(x_{sel}) < f(x) \\ x + s.U(0,1).(x - x_{sel}), else \end{cases} \quad (6)$$

In the fourth movement, the agent's new location is obtained from the direction of the global best away from the agent's current location. This movement is chosen if the generated random number is higher than the third threshold. This process is formalized by using (7). This movement is conducted to exploit the location near the global best.

$$x' = x_{best} + s.U(0,1).(x_{best} - x) \quad (7)$$

This agent's new location is then used to update the global best. As mentioned in the conceptual model, the new solution will replace the global best current solution only if this new solution is better than the global best solution. This process is formalized by using (8).

$$x'_{best} = \begin{cases} x, f(x) < f(x_{best}) \\ x_{best}, else \end{cases} \quad (8)$$

The agent searches for a new location within its local problem space in the second phase. This process is formalized by using (9) and (10). Equation (9) formalizes the candidate for the new location. Equation (10) states that this candidate will only replace the agent's current location if it is better than its current location.

$$x_c = x + \left(1 - \frac{t}{t_{max}}\right)(2U - 1)\left(\frac{b_u - b_l}{d}\right) \quad (9)$$

$$x' = \begin{cases} x_c, f(x_c) < f(x) \\ x, else \end{cases} \quad (10)$$

Based on this explanation, the complexity of the proposed algorithm can be presented in the Big O notation as $O(2t_{max}.n(X))$. Based on this notation, it is shown that the complexity of the proposed algorithm is linearly proportional to the maximum iteration or the population size. The number 2 represents the two phases that are conducted in every iteration.

IV. Simulation and Result

Four simulations are conducted to evaluate the proposed algorithm's performance in this work. The first simulation is conducted to evaluate the proposed algorithm's performance in solving the theoretical mathematic optimization problem. The second simulation is conducted to assess the sensitivity of the algorithm, related to its performance. The third simulation is conducted to evaluate the proposed algorithm's performance in solving the real-world optimization problem. The fourth simulation is conducted to evaluate the convergence of the algorithm in solving the real-world optimization problem.

In the first simulation, the proposed algorithm is challenged to solve the 23 benchmark functions representing the theoretical optimization problem. These functions are commonly used in many studies that suggest new metaheuristic algorithms, such as darts game optimizer (DGO) [23], hide objects game optimizer (HOGO) [24], KMA [20], RDA [19], POA [21], and so on. The list of these functions can be seen in Table I. These functions can be clustered into three groups based on their similar characteristics. The first group represents the high dimension unimodal functions. This group consists of function one to function seven. The second group represents the high dimension multimodal functions. This group consists of function eight to function thirteen. The third group represents the fixed dimension multimodal functions. This group consists of function 14 to function 23.

TABLE I. Benchmark Functions

No	Function	Dim	Problem Space	Target
1	Sphere	10	[-100, 100]	0
2	Schwefel 2.22	10	[-100, 100]	0
3	Schwefel 1.2	10	[-100, 100]	0
4	Schwefel 2.21	10	[-100, 100]	0
5	Rosenbrock	10	[-30, 30]	0
6	Step	10	[-100, 100]	0
7	Quartic	10	[-1.28, 1.28]	0
8	Schwefel	10	[-500, 500]	-4189.8
9	Ratsrigin	10	[-5.12, 5.12]	0
10	Ackley	10	[-32, 32]	0
11	Griewank	10	[-600, 600]	0
12	Penalized	10	[-50, 50]	0
13	Penalized 2	10	[-50, 50]	0
14	Shekel Foxholes	2	[-65, 65]	1
15	Kowalik	4	[-5, 5]	0.0003
16	Six Hump Camel	2	[-5, 5]	-1.0316
17	Branin	2	[-5, 5]	0.398
18	Goldstein-Price	2	[-2, 2]	3
19	Hartman 3	3	[1, 3]	-3.86
20	Hartman 6	6	[0, 1]	-3.32
21	Shekel 5	4	[0, 10]	-10.1532
22	Shekel 7	4	[0, 10]	-10.4028
23	Shekel 10	4	[0, 10]	-10.5363

The more detailed explanation related to the characteristic of these functions is as follows. The unimodal function is a function that has only one optimal solution [25], which is the optimal global solution. There is not any optimal local solution in this function. Contrary, the multimodal function is a function that has multiple optimal solutions [25]. One optimal is the optimal global solution that becomes the target of the optimization. The other optimal solutions are the local optimal.

In this function, the algorithm can be trapped in the local optimal so that the global optimal cannot be found until the iteration ends [25]. The high dimension function represents the function that has a flexible number of adjusted parameters that construct the solution. The dimension can be one and up to unlimited (hundreds or thousands). A higher dimension makes the problem more challenging to optimize. It means that more iteration or population size is needed. The fixed dimension function represents the function that its measurement is static or final. Although the dimension is static and usually low, it does not mean that this function is easy to solve.

These 23 benchmark functions also represent optimization problems with various problem space. The problem space ranges from very narrow, such as in Quartic and Hartman 6, to the very large, such as in Schwefel and Griewank. Most of these functions are centralized at 0. Meanwhile, in several functions, such as Shekel 5 and Hartman 3, the problem space central is not at 0.

In this simulation, the proposed algorithm is compared with four other algorithms: GWO, MPA, KMA, and POA. In general, these four algorithms are new. All these algorithms adopt the foraging mechanism of the animal. Meanwhile, these four algorithms have their distinct mechanics. GWO represents algorithms that every agent moves toward certain (three) best solutions or three global best solutions. MPA represents the movement of several couples of predators and preys where the predator represents the local best solution for its prey. KMA represents algorithm that combines the foraging and mating. POA represents the algorithm that all agents move toward the randomized global target. GWO and MPA also represent the shortcoming algorithms that have been widely studied, improved, and implemented. Meanwhile, KMA and POA represent brand new algorithms that are not popular yet.

The setup of all these five algorithms is as follows. The maximum iteration is set 200 that represent low iteration. The population size is set 20. In MPA, the fishing aggregate devices are set 0.5. The reason is to make balance strategy between finding the alternative randomly within the local problem space and the two randomly selected predators. In KMA, the proportion of the big males is 40%. The reason of this proportion is to make almost balance population between the big males and the small males. Meanwhile, the only one female configuration is chosen based on the recommendation in the first appearance of KMA. There is only one female. The rest population are the small males. The mlipir rate is set 0.5. This rate is chosen to speed up the movement of the small males. Meanwhile, there is not any parameter setting in GWO and POA because these algorithms do not have any adjusted parameter. In the proposed algorithm (HPKA), the proportion is equal, and the step size is set 2. This step size is chosen to so that the local problem space width is wide enough but not too wide. This parameter setting is also can be seen in Table II. Meanwhile, the first, second, and third thresholds are set to make balance proportion between among the movements. The simulation result is shown in Table III. The best result is written in bold font.

The result shows that the proposed algorithm is a good metaheuristic algorithm. It can find the acceptable optimal solution in all 23 benchmark functions. It means that the proposed algorithm is good in solving both unimodal functions and multimodal functions. Moreover, the proposed algorithm also can find the true optimal solution in solving the Six Hump Camel.

Table III also shows that the proposed algorithm is competitive enough compared with other sparing algorithms. It performs the best in solving five functions: Step, Penalized 2, Six Hump Camel, Branin, and Hartman 6. One function is the high dimension unimodal function while the other four functions are the fixed dimension multimodal functions. Compared with other four algorithms, the proposed algorithm is better than GWO, MPA, KMA, and POA in solving 14, 12, 14, and 18 functions respectively. It is also shown the GWO is very powerful in solving the high dimension unimodal functions but weak in solving the fixed dimension multimodal functions. Contrary, KMA is very powerful in solving the Shekel 5, Shekel 7, and Shekel 10.

The second simulation is conducted to evaluate the algorithm sensitivity. In this work, the sensitivity analysis is focused on the formation of the agents due to four possibilities of action chosen by every agent. Like in the first simulation, in this simulation, the proposed algorithm is implemented to solve the 23 benchmark functions. Meanwhile, the maximum iteration and the population size are not chosen to be explored deeper. It is because based on the general model of metaheuristic algorithm, where the quality of the algorithm can be improved by increasing the maximum iteration or the population size theoretically but with the expense of the computational resource and time. On the other hand, the formation does not affect to the complexity or computational consumption. In Table IV, the proportion is presented in a set that contains the proportion of the first, second, third and fourth options consecutively. The first scenario represents the first movement dominant strategy. The second scenario represents the second movement dominant strategy. The third scenario represents the third movement dominant strategy. The fourth scenario represents the fourth movement dominant strategy. The result can be seen in Table IV. The best result is written in bold font.

TABLE II. PARAMETER SETTING

Parameter	**Value**
$n(X)$	20
t_{max}	200
s	2
T_1	0.25
T_2	0.5
T_3	0.75

TABLE III. SIMULATION RESULT (MEANS)

Function	GWO	MPA	KMA	POA	HPKA	Better Than
1	**1.326x10^{-10}**	4.467x10^{1}	4.047x10^{2}	3.030x10^{3}	1.098x10^{-9}	MPA, KMA, POA
2	**0**	**0**	2.505	**0**	1.044x10^{-19}	KMA
3	**7.583x10^{-16}**	1.003x10^{2}	1.704x10^{3}	4.085x10^{3}	4.757x10^{-1}	MPA, KMA, POA
4	**2.804x10^{-9}**	2.764x10^{-1}	1.226x10^{1}	3.057x10^{1}	4.254x10^{-1}	KMA, POA
5	**9.000**	1.004x10^{1}	1.320x10^{4}	8.090x10^{5}	9.073x10^{1}	KMA, POA
6	2.25	3.698x10^{1}	3.291x10^{2}	1.998x10^{3}	**6.243x10^{-11}**	GWO, MPA, KMA, POA
7	3.971x10^{-2}	**1.546x10^{-2}**	4.244x10^{-1}	5.733x10^{-1}	8.457x10^{-2}	KMA, POA
8	1.244x10^{-13}	-1.922x10^{3}	**-3.240x10^{3}**	-2.166x10^{3}	-2.786x10^{3}	GWO, MPA, POA
9	**0**	2.174x10^{1}	3.364x10^{1}	6.631x10^{1}	4.004x10^{1}	POA
10	**6.534x10^{-15}**	4.019	8.343	1.506x10^{1}	6.035	POA
11	**0**	1.425	4.176	2.446x10^{1}	4.471x10^{-1}	MPA, KMA, POA
12	2.639	**2.182**	2.184x10^{2}	3.030x10^{3}	2.540	GWO, KMA, POA
13	3.139	9.062	8.641x10^{3}	3.119x10^{6}	**1.167**	GWO, MPA, KMA, POA
14	1.267x10^{1}	3.002	4.152	**1.364**	6.768	GWO
15	1.484x10^{-1}	**2.947x10^{-3}**	1.954x10^{-2}	2.959x10^{-3}	4.002x10^{-3}	GWO, KMA
16	-1.326x10^{-18}	-1.029	-1.031	-1.030	**-1.032**	GWO, MPA, KMA, POA
17	5.560x10^{1}	5.676x10^{-1}	4.455x10^{-1}	3.992x10^{-1}	**3.981x10^{-1}**	GWO, MPA, KMA, POA
18	6.000x10^{2}	3.399	4.338	**3.019**	8.143	GWO
19	-1.936x10^{-3}	**-3.875**	-5.637x10^{-1}	-4.954x10^{-2}	-4.954x10^{-2}	GWO
20	-5.089x10^{-3}	-2.151	-3.015	-3.030	**-3.150**	GWO, MPA, KMA, POA
21	-0.273	-2.452	**-7.943**	-4.657	-4.894	GWO, MPA, POA
22	-0.294	-2.474	**-8.979**	-4.279	-5.817	GWO, MPA, POA
23	-0.322	-2.219	**-6.590**	-4.214	-5.843	GWO, MPA, POA

TABLE IV. RELATION BETWEEN FORMATION AND THE FITNESS SCORE

Function	Fitness Score			
	0.4:0.2:0.2:0.2	**0.2:0.4:0.2:0.2**	**0.2:0.2:0.4:0.2**	**0.2:0.2:0.2:0.4**
1	**3.534x10^{-11}**	1.977x10^{-10}	1.359x10^{-2}	6.729x10^{-9}
2	**0**	4.532x10^{-30}	**0**	4.802x10^{-13}
3	4.448x10^{2}	**4.857**	4.302x10^{2}	1.528x10^{3}
4	**3.038x10^{-1}**	1.267	8.684x10^{-1}	6.504x10^{-1}
5	2.597x10^{2}	4.506x10^{1}	1.344x10^{2}	**3.151x10^{1}**
6	**3.928x10^{-15}**	3.278x10^{-9}	9.676x10^{-3}	1.806x10^{-9}
7	1.572x10^{-1}	1.337x10^{-1}	**2.893x10^{-2}**	1.075x10^{-1}
8	-2.643x10^{3}	-2.888x10^{3}	**-2.974x10^{3}**	-2.971x10^{3}
9	4.399x10^{1}	**2.907x10^{1}**	2.946x10^{1}	3.787x10^{1}
10	8.203	7.871	**4.262**	7.120
11	6.029x10^{-1}	3.370x10^{-1}	2.377x10^{-1}	**2.053x10^{-1}**
12	1.726	2.846	**5.808x10^{-1}**	7.402x10^{-1}
13	2.145	4.561	**9.026x10^{-2}**	1.211
14	1.086x10^{1}	9.169	**1.510**	6.208
15	**2.225x10^{-3}**	3.593x10^{-3}	5.864x10^{-3}	4.831x10^{-3}
16	**-1.032**	**-1.032**	**-1.032**	**-1.032**
17	**3.981x10^{-1}**	**3.981x10^{-1}**	**3.981x10^{-1}**	**3.981x10^{-1}**
18	1.040x10^{1}	1.200x10^{1}	**3.000**	6.857
19	**-4.954x10^{-2}**	**-4.954x10^{-2}**	**-4.954x10^{-2}**	**-4.585x10^{-2}**
20	**-3.269**	-3.268	-3.131	-3.227
21	-4.232	-3.965	**-5.858**	-4.206
22	-4.189	-5.505	-5.324	**-5.601**
23	**-4.657**	-4.207	-4.309	-4.104

Table IV shows that the relation between the proportion of the options and the algorithm's performance is various; depends on the problem (function) to be solved. There is not any proportion that is the best among other proportions. In some functions, a proportion may be better. But in other function, other proportion is better. Meanwhile, the different proportion affects significantly, especially in solving the unimodal functions. A proportion produces much better result rather than other proportions. Meanwhile, the different proportion affects less significantly in solving multimodal functions. Moreover, the proportion does not affect the result in solving the Six Hump Camel, Branin, and Hartman 3 functions.

The third simulation is conducted to evaluate the proposed algorithm in solving the real-world problem. In this work, the proposed algorithm is challenged to tackle the portfolio optimization problem. A portfolio is a set of valuable and productive assets that is owned by individual or institutions [26]. This asset can be property, stock, bond, gold, and so on. Portfolio represents the wealth of the entity. As a portfolio, an individual or institution should distribute its asset into several options [26]. The objective of this arrangement is to protect its value in the context of maximizing the profit and avoiding the lost. The profit may come from the revenue that is generated from the utilization of the asset or the increasing value of the asset in certain timespan. On the other side, lost may come from the value depreciation or reduction of the asset. Based on it, the portfolio optimization problem can be defined the arrangement of assets in the most optimal way in facing its objective.

In this work, the portfolio optimization problem focuses on the stock. The stock represents the ownership of a proportion of a company. The profit of stock comes in two ways: capital gain and dividend. Capital gain is the increasing value of a share at the end of a certain timespan. The common timespan can be daily, monthly, year-to-date (YTD), year-on-year (YOY), and five years. The dividend is a portion of net profit distributed to the company's owner or stockholder. The stock price represents the market value of a share of a company.

The selected stocks are the ten best companies listed in the LQ45 index. LQ45 index is a list that consists of 45 companies whose share is traded on the Indonesian Stock Exchange (IDX) [27]. These companies are selected because their market capitalization is the biggest, and they are very liquid [27]. These ten companies come from several industrial sectors, such as oil and gas, mining, and banking. The list of these companies is shown in Table V. Table V contains three information: the company's code, current price, and year-to-date capital gain. The current price and capital gain are presented in rupiah per share. The data is obtained from Google, which refers to the Indonesian Stock Exchange.

The stock optimization problem scenario in this work is as follows. The objective is maximizing the total capital gain. The total capital gain is obtained by accumulating the capital gain earned from all held shares. The capital gain refers to the year-to-date capital gain in Table IV. On the other side, there are several constraints used in this optimization. The allocated investment is one billion rupiahs. It means that the bought stocks cannot surpass the total investment. All stocks in Table IV must be represented in the investment portfolio. The purchasing price refers to the current price in Table IV. The purchasing unit for every stock is presented in the lot. A lot refers to 100 shares. The investment ranges from 50 to 200 lots in every stock. Based on this scenario, this portfolio optimization problem can be seen as a high dimensional problem. The number of dimensions is 10. The problem space for every dimension is between 50 and 200.

The simulation scenario related to this portfolio optimization problem is as follows. The population size is set at 20. The maximum iteration is set at 200. The proportion among possible actions is equal. Like in the first simulation, this proposed algorithm is benchmarked with four algorithms: GWO, MPA, KMA, and POA. The result is shown in Table VI.

Table VI shows that the proposed HPKA algorithm is very competitive among algorithms in solving the portfolio optimization problem. Its total capital gain is the highest among GWO, MPA, KMA, and POA. The total capital gain created by the proposed algorithm is 109%, 46%, 47%, and 1% better than the GWO, MPA, KMA, and POA respectively.

Based on the statistic comparison, it is shown that the proposed algorithm is more stable than POA due to its lower standard deviation. Meanwhile, MPA performs as the most stable algorithm due to its lowest standard deviation. Besides, the stability of KMA is also low and it is close to MPA. Ironically, GWO becomes the most unstable algorithm.

The fourth simulation is conducted to observe the convergence of the proposed algorithm in solving the portfolio optimization problem. In this simulation, there are three values of the maximum iteration: 50, 100, and 150. In this simulation, the proposed algorithm is still compared with these fourth algorithms. The result is shown in Table VII.

TABLE V. TEN BEST COMPANIES IN LQ45 INDEX

No	Code	Current Price	YTD Capital Gain
1	MEDC	545	83
2	ITMG	29,975	10,350
3	ADRO	3,180	810
4	INCO	6,850	2,090
5	PTBA	3,710	1,040
6	UNTR	29,775	7,950
7	MDKA	4,610	570
8	ANTM	2,340	0
9	BBNI	8,450	1,725
10	HRUM	10,125	-375

TABLE VI. PORTFOLIO OPTIMIZATION PROBLEM SIMULATION RESULT

No	Algorithm	Total Capital Gain	
		Average	Standard Deviation
1	GWO	191,827,306	48,601,823
2	MPA	274,425,133	5,323,910
3	KMA	273,280,387	7,949,071
4	POA	398,494,240	20,245,548
5	HPKA	401,824,087	16,252,005

TABLE VII. SIMULATION FOR CONVERGENCE ANALYSIS

No	Algorithm	Total Capital Gain		
		t_{max} = 50	t_{max} = 100	t_{max} = 150
1	GWO	208,336,875	189,973,268	192,619,037
2	MPA	272,722,150	273,976,306	276,707,756
3	KMA	259,277,675	268,501,294	268,033,687
4	POA	402,265,771	416,313,431	408,763,771
5	HPKA	416,847,431	404,039,878	406,785,637

Result in Table VII shows that all five algorithms achieve their convergence in the low maximum iteration. It means that these five algorithms do not need high maximum iteration to find the near optimal solution or acceptable solution. Comparing between POA and HPKA, the gap between these two algorithms is narrow.

V. DISCUSSION

In general, the result proves that the proposed algorithm is a good and competitive metaheuristic algorithm. It is very competitive in solving theoretical optimization problem and real-world optimization problem. Its performance is better than KMA and POA in solving most of benchmark functions and the portfolio optimization problem. It means that this hybrid version is better that its origins, whether it is POA or KMA. More profound analysis regarding the findings will be discussed in the following paragraphs.

Table II shows that the proposed algorithm is better than the basic POA. This circumstance happens in most functions in all three groups: high dimension unimodal, high dimension multimodal, and fixed dimension multimodal functions. This result proves that selecting the best solution for the target is better than the randomized target for the swarm movement. Through guided movement toward the global best solution, the probability of the improvement will be higher than the randomized movement, whether it is the randomized jump as conducted in the first option or the half jump as conducted in the second option.

Table II also shows that the proposed algorithm is better than the KMA. This circumstance also occurs in most benchmark functions, exceptionally high dimension unimodal and high dimension multimodal functions. The proposed algorithm is less competitive than the KMA in solving fixed dimension multimodal functions. This circumstance shows that the mechanics of the proposed algorithm consists of four optional movements and the iteration-controlled exploration-exploitation strategy is better than the three fixed movements in KMA.

The result also strengthens the no free lunch theory. As stated in this theory, developing a general-purpose algorithm better for solving all problems is almost impossible [28]. The proposed algorithm may be less competitive than GWO in solving the high dimension unimodal functions where GWO is superior in these functions. On the other hand, GWO loses its superiority in most multimodal operations, whether they are high dimension or fixed dimension. The proposed algorithm is also significantly superior to GWO in solving the portfolio optimization problem. On the other hand, the proposed algorithm is slightly better than the POA in solving a portfolio optimization problem. However, the proposed algorithm is significantly superior to POA in solving theoretical optimization problems.

The simulation result shows that the effectiveness of specific algorithms should not be measured by challenging them to solve only the theoretical optimization problem. In the end, any optimization algorithm must be challenged to solve the real-world optimization problem. On the other hand, the circumstance in real-world problems is various. Many problems, especially in the operational research or finance, are simpler to be presented using integer or mixed-integer programming. The problem space is often integer, such as the number of production units, vehicles, assigned employees, shares, and so on. Moreover, the objective function is also simple, such as maximizing the total sales or profit. This objective can be presented by accumulating the weighted parameters. As an integer problem, precision is not needed. It is difficult to achieve a much better result in the integer-based optimization problem. This circumstance also becomes the reason why many well-known old-fashioned algorithms, such as genetic algorithms, are still used widely in many studies in operational research and finance. It is different from the engineering optimization problem, where many parameters are presented in floating-point numbers. In this case, the high precision algorithm becomes more relevant.

The simulation result also shows that the effectiveness of the metaheuristic algorithm also depends on the tuning mechanism of its adjusted parameters. Many metaheuristic algorithms are equipped with several adjusted parameters. The algorithm will perform well when these parameters are adjusted properly. On the other hand, the algorithm will perform poorly when these parameters are not adjusted properly. This circumstance becomes the nature of metaheuristic algorithms so that they can tackle many optimization problems in flexible ways. Based on this circumstance, it is not wise to judge some algorithms are better than others. However, the phenomenon of beating the elder algorithms is common in many shortcoming studies that propose new metaheuristic algorithms. Although the old-fashioned algorithms, such as genetic algorithm, simulated annealing, tabu search, and PSO, have been beaten many times, their popularity is still high because they are simple and flexible to modify. Commonly, the effectiveness of an algorithm can be improved simply by increasing the iteration or enlarging the population size.

There are several challenges and questions regarding this circumstance. Many metaheuristic algorithms are designed based on fixed adjusted parameters. It means that these adjusted parameters can be changed manually. It will be challenging in the future to propose an adaptive algorithm where the parameters can be tuned automatically during the iteration. It means there is logic in this future algorithm that can learn the behavior of the optimization environment (objective and problem space), and then it reacts based on its knowledge.

VI. Conclusion

The proposed algorithm, namely the hybrid pelican Komodo algorithm, has been proposed in this work. This algorithm is developed by hybridizing the pelican optimization and Komodo Mlipir Algorithms. The work has demonstrated the outstanding performance of the proposed algorithm as a metaheuristic algorithm. It can tackle the two main objectives: finding near-optimal (acceptable) solutions and avoiding local optimal. Through simulation, the proposed algorithm is successful in solving the theoretical optimization problem and real-world optimization problem. It best solves five functions: Step, Penalized 2, Six Hump Camel, Branin, and Hartman 6. The proposed algorithm is better than GWO, MPA, KMA, and POA in solving 14, 12, 14, and 18 functions, respectively. In solving the portfolio optimization problem, the proposed algorithm creates 109%, 46%, 47%, and 1% better total capital gain than the GWO, MPA, KMA, and POA, respectively. Based on its positive result, this work shows that improving the current algorithms through modification or hybridization is as important as proposing a new algorithm with a new name.

There are several future research potentials regarding this work. This work is just one modification of the existing algorithms (KMA and POA). There are many other ways to modify and improve these two shortcoming algorithms. These algorithms can be hybridized with other battle proven algorithms. Besides, it will be challenging to implement these two algorithms to solve many other optimization problems so that the effectiveness of these two algorithms can be observed better to make the ground base for further development.

Acknowledgment

This work was financially supported by Telkom University, Indonesia.

References

[1] C. -L. Hsu, W. C. Lin, L. Duan, J. R. Liao, C. C. Wu, and J. H. Chen, “A robust two-machine flow-shop scheduling model with scenario-dependent processing times”, Discrete Dynamics in Nature and Society, ID: 3530701, pp. 1-16, 2020.

[2] J. W. Fowler and L. Monch, “A survey of scheduling with parallel batch (p-batch) processing”, European Journal of Operational Research, vol. 298, no. 1, pp. 1-24, 2022.

[3] F. Pilati, E. Ferrari, M. Gamberi, and S. Margelli, “Multi-manned assembly line balancing: workforce synchronization for big data sets through simulated annealing”, Applied Sciences, vol. 11, ID: 2523, 2021.

[4] D. M. Utama, I. Santoso, Y. Hendrawan, and W. A. P. Dania, “Integrated procurement-production inventory model in supply chain: a systematic review”, Operations Research Perspectives, vol. 9, ID: 200221, pp. 1-21, 2022.

[5] P. D. Kusuma and M. Kallista, “Pickup and delivery problem in the collaborative city courier service by using genetic algorithm and nearest distance”, Bulletin of Electrical Engineering and Informatics, vol. 11, no. 2, pp. 1026-1036, 2022.

[6] S. Zhang, L. Fu, R. Wang, and R. Chen, “The optimization of the location of the cargo in three-dimension shelf: employing the FP-tree and the artificial fish swarm algorithms”, Journal of Control Science Engineering, vol. 2020, ID: 8832691, pp. 1-15, 2020.

[7] W. Bakry, A. Rashid, S. Al-Mohamad, N. El-Kanj, “Bitcoin and portfolio diversification: a portfolio optimization approach”, Journal of Risk and Financial Management, vol. 14, no. 7, ID: 282, pp. 1-24, 2021.

[8] C. Bayer, R. Tempone, and S. Wolfers, “Pricing American options by exercise rate optimization”, Quantitative Finance, vol. 20, no. 11, pp. 1749-1760, 2020.

[9] X. Wang, “Analysis of bank credit risk evaluation model based on BP neural network”, Computational Intelligence and Neuroscience, vol. 2022, ID: 2724842, pp. 1-11, 2022.

[10] A. Ansari, I. S. Ahmad, A. A. Bakar, and M. R. Yaakub, “A hybrid metaheuristic method in training artificial neural network for banckruptcy prediction”, IEEE Access, vol. 8, pp. 176640-176650, 2020.

[11] H. R. Moshtaghi, A. T. Eshlagy, 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. 63-69, 2021.

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

[13] 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, ID: e0257317, pp. 1-16, 2021.

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

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

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

[17] X. Ding, M. Zheng, and X. Zheng, “The application of genetic algorithm in land use optimization research: a review”, Land, vol. 10, ID: 526, pp. 1-21, 2021.

[18] A. Slowik and H. Kwasnicka, “Evolutionary algorithms and their applications to engineering problems”, Neural Computing and Applications, vol. 32, pp. 12363-12379, 2020.

[19] A. M. Fathollahi-Fard, M. Hajiaghaei-Keshteli, and R. Tavakkoli-Moghaddam, “Red deer algorithm (RDA): a new nature-inspired meta-heuristic”, Soft Computing, vol. 19, pp. 14638-14665, 2020.

[20] Suyanto, A. A. Ariyanto, and A. F. Ariyanto, “Komodo mlipir algorithm”, Applied Soft Computing, vol. 114, ID: 108043, 2022.

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

[22] Y. Qawqzeh, M. T. Alharbi, A. Jaradat, and K. N. A. Sattar, “A review of swarm intelligence algorithms deployment for scheduling and optimization in cloud computing environments”, PeerJ Computer Science, vol. 7, pp. 1-17, 2021.

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

[24] 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.

[25] K. Hussain, M. N. M. Salleh, S. Cheng, and R. Naseem, “Common benchmark functions for metaheuristic evaluation: a review”, International Journal on Informatic Visualization, vol. 1, no. 4-2, pp. 218-223, 2017.

[26] L. Chin, E. Chendra, and A. Sukmana, “Analysis of portfolio optimization with lot of stocks amount constraint: case study Index LQ45”, IOP Conference Series: Materials Science and Engineering, vol. 300, pp. 1-6, 2018.

[27] A. Santi Samasta, S. P. Lestari, and T. D. Arsanda, “The analyze of LQ45 companies stock price in 2018-2020”, Journal of Management and Digital Business, vol. 1, no. 2, pp. 64-72, 2021.

[28] W. G. Macready, “No free lunch theorems for optimization”, IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 67-82, 1997.