Received: July 4, 2022. Revised: August 19, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Guided Pelican Algorithm

Purba Daru Kusuma[1*] **Anggunmeka Luhur Prasasti**[1]

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

Abstract: This paper presented a new metaheuristic technique, namely the guided pelican algorithm (GPA). GPA has the improvements for a shortcoming algorithm, namely the pelican optimization algorithm (POA), that mimics the behaviour of pelican birds during hunting prey. It improves the original POA in three ways. First, GPA replaces the randomized target with the global best solution as a deterministic target in phase one. Second, GPA replaces the pelican's current location with the search space size in determining the local search space size in phase two. Third, GPA implements multiple candidates in both phases rather than a single candidate as it is used in the original POA. Simulation is implemented to observe GPA's performance in optimizing both theoretical and real-world problems. GPA is compared with four algorithms: marine predator algorithm (MPA), particle swarm optimization (PSO), komodo mlipir algorithm (KMA), and POA. The result shows that GPA outperforms all sparing algorithms in optimizing most benchmark functions. GPA is also implemented to optimize the portfolio problem. The result shows that GPA outperforms all sparing algorithms in optimizing the portfolio problem. It outperforms three sparing algorithms in optimizing the portfolio problem. Its performance is 9%, 11%, and 13%, better than PSO, MPA, and KMA consecutively. Meanwhile, its less than 1% worse than POA.

Keywords: Pelican optimization algorithm, Metaheuristic, Intelligent finance, Portfolio optimization problem, LQ45 index.

1. Introduction

Metaheuristic algorithms have been implemented widely in many optimization problems. In the manufacturing sector, they have been used to solve many scheduling problems, such as in flow-shop [1], job-shop [2], batch-shop [3], and so on. In the transportation sector, they also have been utilized to solve many routing problems, such as in the capacitated vehicle routing problem [4], where each vehicle has a maximum load capacity that cannot be surpassed, and in the pickup delivery problem [5] where the vehicle must tackle two activities: pickup and delivery. Metaheuristic algorithms are also implemented in many assignment problems [6], where certain jobs or tasks should be handled by several limited resources in the most efficient way, such as in the course timetabling problem [7]. This popularity comes from their characteristic as an approximating approach so that they become adaptive to the limited computational resource in solving large-scale and complicated optimization problems [8]. However, the metaheuristic algorithms only guarantee to find the sub-optimal solution at the end of the process [8].

Many shortcoming metaheuristic algorithms are developed based on swarm intelligence. Swarm intelligence-based algorithm is a derivative of the population-based algorithm where the system consists of several autonomous agents. Moreover, in swarm intelligence, communal knowledge is introduced and shared among the agents to improve its performance [9]. PSO is the early version of algorithm that adopts swarm intelligence. It mimics the behavior of a group of birds during foraging or finding foods [10]. In PSO, global best and local best represent communal knowledge [10]. The birds tend to fly to get closer to these global best and local best representations [10]. To date, many shortcoming metaheuristic algorithms, especially the algorithms that mimic the animal behavior during foraging, implement the swarm intelligence.

Getting closer to the best solution is also adopted by several shortcoming metaheuristic algorithms. In KMA, the small males move toward the big males while the big males move toward the other higher quality big males [11]. In grey wolf optimizer (GWO), the wolves move toward the three best wolves in the population [12]. In MPA, the prey moves toward its predator (elite) [13]. Besides the animal-inspired algorithms, several game-based algorithms also adopt this mechanism, such as football game-based optimizer (FBGO) [14], hide object game optimization (HOGO) [15], and darts game optimizer (DGO) [16]. This mechanism can be seen as a guided movement because there is a deterministic direction that guides the agent to move to a certain location. It is different from several other algorithms that focus on local search or neighborhood search, such as simulated annealing (SA) [17], invasive weed optimizer (IWO) [18], tabu search (TS) [19], and so on.

One of the latest metaheuristic algorithms is pelican optimization algorithm (POA). This algorithm was introduced by Trojovsky and Dehghani and was firstly published in 2022 [20]. This algorithm adopts the behavior of pelican during foraging or hunting preys. It is a population-based algorithm formed of some pelicans that fly autonomously [20]. This algorithm conducts a specific method in its exploration and exploitation strategy. There are two consecutive phases conducted for all pelicans in every iteration. In phase one, all pelicans tend to fly toward the same prey [20]. This prey is distributed randomly inside the search space at the beginning of every iteration. It is an uncommon mechanism because, in many algorithms, each agent makes a move toward the local best or global best. This first phase represents exploration. In phase two, all pelicans search for a new location inside their own local search space [20]. During the iteration, this local search space size declines gradually. Phase two represents the exploitation strategy. In its first appearance, this algorithm was compared with PSO, teaching-learning based algorithm (TLBO), GWO, WOA, MPA, TSA, gravitational search algorithm (GSA), and genetic algorithm (GA) to solve 23 benchmark functions [20].

As a new algorithm, studies conducted to implement this algorithm in solving other real-world optimization problems or modifying the POA are hard to find. Meanwhile, many possibilities can be conducted to modify this algorithm. The modification can be conducted in phase one and phase two. Implementation of this algorithm for the other real-world problem is widely open. Based on this circumstance, studies related to POA are very interesting.

The objective of this work is proposing a new optimizer, namely the guided pelican algorithm (GPA). This algorithm is developed as an improvement of the original POA. As the improvement version of the previous algorithm, the methodology that is implemented in this work is as follows. First, the strategy of POA is reviewed to analyze its pros and cons. Then, GPA is developed by maintaining the core mechanics of POA and improving the weakness of its current approach. Through GPA, the challenge to solving an optimization problem, its performance related to the POA, and other sparing algorithms can be evaluated. Finally, the analysis regarding the result will be discussed and summarized to make the new baseline for future studies.

As the improvement of POA, this work contributes in some places as follow.

1) GPA replaces the randomized prey with the global best in the guided movement.
2) The single candidate is improved with the multiple candidates during the guided and random movement.
3) The random movement is improved so that it becomes more adaptive to the search space size problem.

The rest of this paper is structured as follows. The strategy and approach conducted in POA are reviewed in section two. Then, section three presents the model of GPA which consists of the concept, algorithm, and mathematical models. Simulation and implementation related to GPA are presented in section four. The intense analysis regarding the simulation result is discussed in section five. Finally, the conclusions of the work and future research potentials regarding the GPA are summed up in the last section.

2. Related works

POA is a new metaheuristic optimizer built based on swarm intelligence. This algorithm consists of some pelicans that fly autonomously around the search space to hunt for the best preys. As a swarm intelligence, each agent shares a communal knowledge to improve their work. Like many other shortcoming algorithms, its behavior mimics animal behavior. In this context, POA adopts the behavior of a group of pelicans during foraging (hunting prey) [20]. This basic behavior is like PSO, wherein PSO, a group of birds flies together inside a certain distance during searching for food [10].

Like any other metaheuristic algorithm, there are two steps in POA. Step one is initialization [8]. Step two is iteration [8]. In the initialization, all pelicans are distributed randomly inside the search space [20].

Then, the location of every pelican is improved in every iteration. Specifically, there are two consecutive phases in every iteration. Phase one is called moving toward the prey [20]. Phase two is called winging the water surface [20]. Phase one represents the exploration, while phase two represents the exploitation. In both phases, the pelican moves to the new location only if this new location is better than its current solution.

In phase one, a prey is generated randomly at the beginning of every iteration [20]. It follows a uniform distribution. There is only one prey for the entire pelican population. Thus, all pelicans will move toward this prey. In this phase, there are two possible directions depending on the fitness score of the prey and the pelican. If the prey is better than the pelican, then the pelican will fly toward the prey. Else, the pelican flies away from the prey. The step size of this movement is randomized. This step can make the pelican very close to the prey (long jump) or contrary, it is just a small step so that its new location is very close to the pelican's current location (short jump). If this new location is better than its current location, then this new location will replace the current location (accept). Otherwise, the pelican remains to stay in its current location (reject).

In phase two, the pelican conducts a local search [20]. This movement relies only on the pelican's current location. The pelican searches randomly a new location inside its local search space. The local search space size depends on three aspects: iteration, static coefficient, and its current location. In POA, the space size declines gradually as the iteration goes on. In the early iteration, the local search space is wide enough to accommodate the exploration dominant strategy. Meanwhile, in the later iteration, the local search space becomes very narrow or almost zero so that the new location will be generated very close to the pelican's current location. The acceptance-rejection strategy is also implemented in this phase.

This strategy is useful in solving both unimodal and multimodal problems. Exploration dominant strategy in the early iteration means that the algorithm has enough opportunity to explore a wider area inside the search space to avoid the local optimal trap. On the other side, narrow local search space in the later iteration will make the algorithm focus on exploitation and improve the precision of the new location (solution). It is assumed that in the latter iteration, the area where the global optimal lies, has been found.

POA also implements a global best which is updated in every iteration after all pelicans update their location. The pelican with the best fitness score becomes the candidate to replace the current global best solution. However, this candidate will replace the current global best solution only if it is better than the current global best solution. After the iteration ends or the terminating criteria are met, the last value of the global best solution becomes the final solution.

There are several notes regarding POA. The movement toward a certain object or location with a certain step size is common in many algorithms inspired by the foraging mechanism of the animal. However, in general, the target is specified, and it is usually the best one. It means that the agent tends to move closer to a better location. This movement can be drawn back in PSO, where each agent (bird) moves toward the accumulation of the weighted global best solution and the weighted local best solution [10].

This movement is then adapted by many later algorithms. In KMA, the small males move toward the accumulation of the big males, and the big male moves toward the accumulation of the other better big males [11]. In GWO, the wolves move toward the three best wolves (alpha, beta, and delta) [12]. This strategy is proven to create convergence for all agents so that all agents move like a swarm. Moreover, the movement tends to be smooth.

This strategy can also be called as leader-guided strategy. This strategy is also adopted by many shortcoming metaheuristic algorithms. Although the concept is similar, each algorithm has their own distinct improvisation. In HOGO, there is a movement where a player tends to move toward the best player or player whom the coach makes the loudest voice [15]. In an average and subtraction-based optimizer (ASBO), an agent will move toward a certain location related to the best agent [21]. In golden search optimization algorithm (GSOA), the solutions are sorted in every iteration [22]. The worst solution is replaced by the randomly selected solution. Then all solutions move toward the best solution based randomized sinusoidal step size [22]. In northern goshawk optimization (NGO), the leader is randomly selected among the solutions. Then, each solution conducts neighborhood search where the local search space is reduced linearly during the iteration [23]. In hybrid leader-based optimization (HLBO), a leader of each solution is constructed from the normalized quality of three solutions: the related solution, the highest quality solution, and the randomly selected solution [24]. The normalized quality of each entity defines the portion of this entity to construct the hybrid leader. Then, each solution will follow its own hybrid leader [24]. In the mixed leader-based optimizer, there are two types of leaders [25]. In the first half of iteration, the leader is the combination between the best quality solution and a solution that is selected randomly inside the search space [25]. In the

second half of iteration, the leader is the best quality solution [25]. The multi leader optimizer (MLO) can be seen as the improvisation of GWO. In MLO, the number of solutions that constructs the leader is not only three, but it is set manually before the optimization runs [26]. In the three influential member-based optimizer (TIMBO), there are three solutions that become the leaders: the best solution, the worst solution, and the mean solution [27]. Each solution gets closer to the best solution and escape from the worst solution [27]. The acceptance-rejection is also implemented in TIMBO [27]. stochastic komodo algorithm (SKA) is the improvisation of KMA. Rather than determined based on the fitness quality as in KMA, in SKA, the big male, small male, and female are determined randomly based on certain threshold [28]. In modified honey badger algorithm (MHBA), the leader is the best solution so far [29]. But each solution follows the leader based on two options. The first option is randomized sinusoidal movement and the second option is randomized non-sinusoidal movement [29].

The POA chooses a different mechanism. Rather than choosing the global best solution, the target (prey) is determined randomly inside the search space. It means that in every iteration, the movement tends to be chaotic because the target can change extremely. It is shown as a massive local or global search, depending on the generated step size. Fortunately, this chaotic search is still controlled by the acceptance-rejection strategy. In this algorithm, the role of the global best is just as a storage entity. It stores the global best solution so far which is not utilized as the guidance or compass for agents to determine their direction.

Phase two of POA also can be seen as a local or global search depending on the iteration. In the beginning, phase two implements a global search. Then, it turns to local search gradually. The difference from phase one is the number of locations (solutions) considered. In phase one, the movement depends on the current location and the target. On the other hand, in phase two, the movement depends only on its current location.

Phase two is like the eddy formation in MPA as an iteration-controlled search [13] but with a certain difference. The similarity between POA and MPA is that the local search space declines gradually as the iteration goes on. In MPA, there are two options in the eddy formation. The prey can move randomly inside the local search space or move toward the two randomly chosen prey [13]. Meanwhile, in POA, the pelican moves only inside its local search space. Another difference is that in MPA, the local search space size is also affected by the search space size while in POA, the local search space size is also affected by the pelican's current location.

Based on this review, there is a wide opportunity to improve the original POA. The improvement can be conducted in the method implemented in phase one or phase two. In phase one, there is an opportunity to transform the chaotic movement into a guided movement. In phase two, there is a possibility of modifying the local search space size. This review also shows that there are a lot of opportunities to develop a new metaheuristic algorithm based on the leader-based approach.

3. Model

This section presents the model of GPA. It consists of three parts. The first part is the concept. The second part is pseudocode. The third part is mathematical model. The conceptual model explains the concept and movement adopted in the model. The algorithm is presented by using pseudocode. More detailed mechanics of GPA is explained by using a mathematical model.

Like POA, GPA also consists of some agents that search for optimal solutions autonomously based on their objective and the environment. In this context, the search space represents the environment. GPA is also a swarm intelligence-based algorithm. It means that there is communal knowledge that is shared among agents during the optimization process. In GPA, the global best becomes the communal knowledge. Different from the POA, in GPA, the global best is not only used to store the best solution so far and then used as the final solution, moreover, the global best also becomes the main target (prey) for all agents during their guided movement. The use of the global best as the main target will increase the probability of all agents improving their current solution or moving to a better place.

Like POA, GPA consists of two phases in every iteration. Phase one is guided movement, and phase two is randomized movement. The global best is updated in every iteration.

In GPA, in phase one, the target is the global best. It is different from the POA where the target is randomized inside the search space. Then, a certain number of candidates are generated along the path toward the target. The distance between the adjacent candidates is the same. Then, a candidate whose fitness score is the highest becomes the best candidate. This best candidate then replaces the agent's current location (solution) only if it is better than the agent's current location. It is also different from the original POA, where there is only one candidate generated in

algorithm 1: GPA algorithm

output: x_{best}
begin
  //initialization
  for i=1 **to** $n(X)$ **do**
    set x using Eq. (1)
    update g_{best} using Eq. (6)
  end for
  //iteration
  for t=1 **to** t_{max} **do**
    for i=1 **to** $n(X)$ **do**
      //first phase
      for j=1 **to** $n(C)$ **do**
        generate c_j using Eq. (2) and Eq. (3)
      end for
      find c_{best} using Eq. (4)
      update x using Eq. (5)
      update x_{best} using Eq. (6)
      //second phase
      for j=1 **to** $n(C)$ **do**
        generate c_j using Eq. (7) and Eq. (8)
      end for
      find c_{best} using Eq. (4)
      update x using Eq. (5)
      update x_{best} using Eq. (6)
    end for
  end for
end

this phase.

Like POA, GPA also conducts random movement near the current location (neighborhood search) in phase two. GPA also adopts narrowing local search space. It means that the search space size declines gradually as the iteration goes on. However, in GPA, the local search space size is different from POA. In POA, the local search space size depends on the agent. In contrast, in the significance of GPA, the local search space size depends on the search space size. In this phase, rather than a single candidate as in POA, in GPA, several candidates are generated randomly inside the local search space. The best candidate is selected among them based on its fitness score. If this best candidate is better than the agent's current location, then this best candidate becomes the replacement.

This conceptual model is then used to create the algorithm and mathematical model. The algorithm is presented in algorithm 1. Several annotations are used in them. These annotations are described as follows.

b_l	lower bound
b_u	upper bound
b_w	search space size
c	candidate
C	set of candidates
c_{best}	best candidate
f	fitness
x	agent's current location
X	set of agents
x_{best}	global best
x_{tar}	target
U	uniform random

Based on algorithm 1, the complexity of this algorithm can be presented as $O(2t_{max}.n(X).n(C))$. It shows that its complexity is proportional to the maximum iteration, the number of candidates, and population size. Then, the accumulation is doubling because there are two phases in every iteration.

In the initialization, the pelican's location is uniformly distributed inside the search space. This process is formalized by using Eq. (1).

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

In phase one, the agent moves toward the global best. Several candidates are generated in this phase with the same distance between the adjacent candidates. This process is formalized by using Eq. (2) to Eq. (6).

$$x_{tar} = \begin{cases} x + 2(x_{best} - x), f(x_{best}) < f(x) \\ x + (x - x_{best}), else \end{cases} \tag{2}$$

$$c_i = c + \frac{i}{n(C)}(x_{tar} - x) \tag{3}$$

$$c_{best} = c \in C \wedge \min\big(f(c)\big) \tag{4}$$

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

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

The explanation of Eq. (2) to Eq. (6) is as follows. Eq. (2) determines the target of the guided movement. There are two possibilities. The target will be toward the global best if the global best is better than the current location. Otherwise, the target will be away from the global best. Eq. (3) states that the candidates are spread along the path from the current location and the target. Meanwhile, the distance of the adjacent candidates is reverse proportional to the number of candidates. Eq. (4) states that the candidate whose fitness score is the best will be chosen as the best candidate. Eq. (5) states that the best candidate replaces the current location if it is better than the current location. Eq. (6) states that the new location

Table 1. Benchmark functions

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

replaces the global best if this new location is better than the global best.

In phase two, the randomized movement is conducted. In this phase, several candidates are generated by using Eq. (7) and Eq. (8). Then, candidate selection, agent's location update, and global best update are conducted by using Eq. (4) and Eq. (6). Eq. (5) is not used because, in phase two, the best candidate replaces the current location without considering the fitness score between the best candidate and the current location.

$$b_w = \frac{b_u - b_l}{2} \tag{7}$$

$$c = x + \left(1 - \frac{t}{t_{max}}\right)(1 - 2U)b_w \tag{8}$$

The explanation of Eq. (7) and Eq. (8) is as follows. Eq. (7) states that the iteration-free local search space size is half of the search space. Eq. (8) states that the candidate will be generated randomly inside the local search space where its size is reduced gradually due to the increase of the iteration.

4. Simulation and result

GPA is then implemented into simulation. As a metaheuristic algorithm, GPA must be able to meet two considerations. The first one is finding the sub-optimal or acceptable solution. The second one is escaping from the local optimal entrapment. There are several simulations performed in this work. Simulation one is to evaluate GPA's performance in solving the 23 benchmark functions. This simulation can be seen as a simulation related to the theoretical optimization problem. The second and third simulations are performed to analyze GPA's sensitivity. Like simulation one, the second and third simulations are also performed by implementing GPA to solve the 23 benchmark functions. Simulation four is conducted to evaluate GPA in optimizing the real-case problem. The portfolio optimization problem represents the real-world optimization problem, especially in the financial sector.

In simulation one, GPA is challenged to solve 23 benchmark functions. Seven functions are unimodal while sixteen functions are multimodal. The unimodal functions represent problems with a single global optimal [30]. Although the unimodal functions are simple in general, the different shape of the curve represents a different challenge. The key consideration in solving this problem is how fast the algorithm moves toward the global optimal and how precise the algorithm is so that it can find a solution that is as near as possible to the exact global optimal. Contrary, the multimodal functions represent problems with several to many optimal solutions [30]. There is only one global optimal and the rest are the local optimal. The key concern in solving these problems is avoiding the local optimal. The algorithm

Table 2. Simulation result

Funct.	PSO	MPA	KMA	POA	GPA	Better Than
1	$7.879x10^{2}$	$7.380x10^{1}$	$6.279x10^{2}$	$3.372x10^{3}$	**$5.915x10^{-2}$**	PSO, MPA, KMA, POA
2	$1.457x10^{2}$	**0**	$1.148x10^{2}$	**0**	**0**	PSO, KMA
3	$1.687x10^{3}$	$1.351x10^{2}$	$2.046x10^{3}$	$5.412x10^{3}$	**$1.155x10^{-1}$**	PSO, MPA, KMA, POA
4	$1.449x10^{1}$	$5.196x10^{-1}$	$1.202x10^{1}$	$3.321x10^{1}$	**$1.747x10^{-1}$**	PSO, MPA, KMA, POA
5	$1.246x10^{5}$	**$2.261x10^{1}$**	$3.2629x10^{4}$	$5.569x10^{6}$	$3.458x10^{1}$	PSO, MPA, KMA, POA
6	$4.352x10^{2}$	$5.867x10^{1}$	$3.933x10^{2}$	$2.347x10^{3}$	**$5.102x10^{-2}$**	PSO, MPA, KMA, POA
7	$2.775x10^{-2}$	$2.499x10^{-2}$	$1.942x10^{-1}$	$7.446x10^{-1}$	**$2.554x10^{-3}$**	PSO, MPA, KMA, POA
8	$-1.609x10^{3}$	$-1.896x10^{3}$	$-3.079x10^{3}$	$-1.927x10^{3}$	**$-3.256x10^{3}$**	PSO, MPA, KMA, POA
9	$5.410x10^{1}$	$3.020x10^{1}$	$4.373x10^{1}$	$6.965x10^{1}$	**6.625**	PSO, MPA, KMA, POA
10	8.628	4.604	8.755	$1.595x10^{1}$	**$5.054x10^{-1}$**	PSO, MPA, KMA, POA
11	9.156	1.877	5.363	$3.487x10^{1}$	**$4.421x10^{-1}$**	PSO, MPA, KMA, POA
12	$4.579x10^{1}$	3.965	7.101	$1.179x10^{6}$	**$3.071x10^{-2}$**	PSO, MPA, KMA, POA
13	$1.431x10^{5}$	$1.477x10^{1}$	$4.575x10^{3}$	$7.359x10^{6}$	**$3.030x10^{-2}$**	PSO, MPA, KMA, POA
14	5.160	4.799	9.774	1.584	**$9.980x10^{-1}$**	PSO, MPA, KMA, POA
15	$1.641x10^{-2}$	$4.976x10^{-3}$	$1.772x10^{-2}$	**$2.999x10^{-3}$**	$1.572x10^{-3}$	PSO, MPA, KMA
16	-1.031	-1.024	-1.025	-1.029	**-1.032**	PSO, MPA, KMA, POA
17	$7.189x10^{-1}$	$6.809x10^{-1}$	$5.298x10^{-1}$	$4.028x10^{-1}$	**$3.981x10^{-1}$**	PSO, MPA, KMA, POA
18	3.251	4.094	5.721	3.078	**3.000**	PSO, MPA, KMA, POA
19	$-5.392x10^{-3}$	**-3.806**	$-7.194x10^{-1}$	$-4.954x10^{-2}$	$-4.954x10^{-2}$	PSO
20	-2.405	-2.014	-2.867	-2.943	**-3.277**	PSO, MPA, KMA, POA
21	-4.101	-2.092	**-7.637**	-3.354	-7.622	PSO, MPA, POA
22	-3.034	-1.879	-7.581	-3.773	**-9.645**	PSO, MPA, KMA, POA
23	-5.010	-2.018	-5.672	-3.998	**-9.512**	PSO, MPA, KMA, POA

must find the area where the global optimal lies. These functions also have their shape and difficulties. Some functions are like areas with multiple ripples. In this circumstance, the algorithm may avoid the current local optimal but then is trapped inside the other local optimal. Other functions can be seen as a wide flat area with a single narrow slope. In this flat area, it will be difficult to find a solution that is better than the current solution. These 23 benchmark functions are commonly used to evaluate many shortcomings in other metaheuristic algorithms, such as KMA [11], ASBO [21], POA [20], HOGO [15], MPA [13], FBGO [14], DGO [16], and so on. These functions represent various types of problems with their own characteristics and difficulties, from simple to complicated.

These functions are categorized into three clusters: (1) high dimension unimodal functions, (2) high dimension multimodal functions, and (3) fixed dimension multimodal functions. In the high dimension functions, the dimension is flexible so that it can range from two to thousands. Meanwhile, the fixed functions have static dimensions so that they cannot be expanded or reduced. In this work, the dimension of the high dimension functions is set at 10. The detailed description of these functions is shown in Table 1.

The 23 functions represent optimization problems with various ranges of search space, from the narrow ones to the wide ones. These various ranges of search space represent different challenges or difficulties. In general, a wider search space means that the algorithm must spend more effort or time to find the solution. Wider search space must be compensated with a higher maximum iteration or longer step size. On the other side, a wider longer step size may reduce the precision of the result.

In this work, GPA is compared with other metaheuristic algorithms: PSO, KMA, MPA, and POA. PSO represents the early metaheuristic algorithm that implements guided movement through a simple calculation. KMA represents a metaheuristic algorithm that combines mating and foraging mechanics [11]. The mating process in KMA can be seen as a movement toward the global best solution (highest quality big male) that generates two possibilities or candidates [11]. The first candidate is close to the current solution and the second candidate is close to the global best. Subsequently, the better candidate replaces the current solution. The guided movement of the small males represents the movement toward all better solutions. Meanwhile, the guided movement of the big males represents the movement toward the better solutions and away from the worst solutions [11].

MPA represents the metaheuristic algorithm that implements guided movement toward the local best. Every predator engages with its own prey [13]. In the guided movement, the better or worse solution is not

Table 3. Relation between maximum iteration and GPA's performance

Func.	Average Fitness Score		
	t_{max} = 60	t_{max} = 120	t_{max} = 180
1	$1.639x10^{-1}$	$4.451x10^{-2}$	$1.689x10^{-2}$
2	0	0	0
3	$3.236x10^{-1}$	$1.114x10^{-1}$	$5.153x10^{-2}$
4	$3.098x10^{-1}$	$1.553x10^{-1}$	$1.076x10^{-1}$
5	$3.325x10^{1}$	$5.807x10^{1}$	8.959
6	$1.202x10^{-1}$	$2.529x10^{-2}$	$1.405x10^{-2}$
7	$5.235x10^{-3}$	$2.745x10^{-3}$	$1.264x10^{-3}$
8	$-3.200x10^{3}$	$-2.954x10^{3}$	$-3.380x10^{3}$
9	4.240	5.637	5.987
10	$7.409x10^{-1}$	$2.634x10^{-1}$	$1.644x10^{-1}$
11	$6.028x10^{-1}$	$3.769x10^{-1}$	$3.067x10^{-1}$
12	$1.757x10^{-1}$	$2.913x10^{-2}$	$3.204x10^{-4}$
13	$9.865x10^{-2}$	$2.970x10^{-2}$	$7.634x10^{-3}$
14	$9.980x10^{-1}$	$9.980x10^{-1}$	$9.980x10^{-1}$
15	$2.236x10^{-3}$	$5.415x10^{-4}$	$5.087x10^{-4}$
16	-1.032	-1.032	-1.032
17	$3.981x10^{-1}$	$3.981x10^{-1}$	$3.981x10^{-1}$
18	3.000	3.000	3.000
19	$-4.954x10^{-2}$	$-4.954x10^{-2}$	$-4.954x10^{-2}$
20	-3.312	-3.257	-3.257
21	-8.170	-8.206	-7.198
22	-9.694	-9.587	-9.342
23	-8.882	-8.741	-8.745

considered. This better or worse solution is considered only when the predators are updated. Moreover, exploration is also conducted besides the guided movement by implementing the eddy formation. The global best is calculated after the iteration ends. It is different from PSO, KMA, and POA where the global best solution is updated in every iteration.

POA is chosen because GPA is the improved version of POA. As the improvement, it is important to evaluate the improved version with the original version. Moreover, it is also important to evaluate the advantage and disadvantage of GPA.

In this simulation, the setting of several general parameters is as follows. The maximum iteration is set at 100 which represents the low maximum iteration. The reason is to evaluate the algorithms in the limited computational process. The population size is set at 20. Meanwhile, the specific parameters setting is as follows. In GPA, the number of candidates is set to 10. In PSO, all weights are set at 0.1. This circumstance represents low-speed guided movement to overcome the high precision result. In KMA, the big male portion is 40%. There is only one female. The rest are small males. The Mlipir rate is 0.5. In MPA, the fishing aggregate devices are set to 0.5. This circumstance represents the balance portion in the exploration process between finding the new solution inside the local search space and in the middle among randomly selected agents. The result is shown in Table 2.

Table 2 shows that GPA is proven as a preferred algorithm for solving theoretical mathematic problems. It can tackle two challenges: finding the acceptable solution and escaping from the local optimal. This circumstance happens in all 23 benchmark problems. It also happens in all three clusters of functions. Moreover, it can find the global optimal in four functions: schwefel 2.22, six hump chamel, branin, and goldstein-price. One is a high dimension unimodal function whereas the other three functions are fixed dimension multimodal functions. The result also shows that the search space does not affect GPA's performance. GPA is still good in solving functions with narrow, medium, and large search space. GPA is also proven good in solving functions with the optimal location at 0 or somewhere else.

Table 2 also shows that GPA is competitive among the sparing algorithms. GPA creates the best result in solving eighteen functions. It is better than PSO in solving all functions. Moreover, it is also better than MPA, KMA, and POA in almost all functions. However, it is only better than PSO in solving Hartman 3 function.

The next simulation is performed to observe the sensitivity analysis of parameters constructing the algorithm. In this work, two parameters will be analysed: maximum iteration and population size. Theoretically, these parameters have a positive relationship with GPA. It is a common approach that many metaheuristic studies are to improve its performance by expanding the size of the population and the maximum iteration. The consideration of these parameters' significance is related to GPA's performance.

Simulation two is performed to evaluate the relation between the maximum iteration and GPA's performance. In this simulation, there are three values of the maximum iteration: 60, 120, and 180. Like simulation one, GPA is challenged to optimize the 23 benchmark functions. Table 3 shows the result.

Table 3 shows that in general, the maximum iteration has a positive relationship with the algorithm's performance. The fitness score tends to decline due to the increase in the maximum iteration. Moreover, the result also shows that the trend converges in the low maximum iteration for 13 benchmark functions.

Simulation three is performed to evaluate the relation between the population size and the algorithm performance. In this simulation, there are three values of population size: 10, 30, and 50. Like

Table 4. Relation between population size and GPA's performance

Func.	Average Fitness Score		
	$n(X) = 10$	$n(X) = 30$	$n(X) = 50$
1	1.130×10^{-1}	5.324×10^{-2}	4.111×10^{-2}
2	0	0	0
3	3.397×10^{-1}	6.256×10^{-2}	5.568×10^{-2}
4	2.675×10^{-1}	1.646×10^{-1}	1.394×10^{-1}
5	2.988×10^{1}	2.178×10^{1}	9.303
6	9.003×10^{-2}	3.638×10^{-2}	2.957×10^{-2}
7	4.054×10^{-3}	1.767×10^{-3}	9.823×10^{-4}
8	-3.166×10^{3}	-3.358×10^{3}	-3.186×10^{3}
9	6.764	3.467	2.739
10	4.888×10^{-1}	1.054×10^{-1}	3.446×10^{-1}
11	5.797×10^{-1}	3.848×10^{-1}	3.249×10^{-1}
12	6.267×10^{-2}	5.777×10^{-2}	3.974×10^{-4}
13	5.246×10^{-2}	1.752×10^{-2}	1.357×10^{-2}
14	9.980×10^{-1}	9.980×10^{-1}	9.980×10^{-1}
15	6.473×10^{-4}	4.838×10^{-4}	4.769×10^{-4}
16	-1.032	-1.032	-1.032
17	3.981×10^{-1}	3.981×10^{-1}	3.981×10^{-1}
18	3.000	3.000	3.000
19	-4.954×10^{-2}	-4.954×10^{-2}	-4.954×10^{-2}
20	-2.999	-3.279	-3.301
21	-6.021	-6.705	-8.771
22	-8.747	-8.001	-8.960
23	-1.053×10^{1}	-9.642	-9.5613

Table 5. Data of six companies in LQ45 index

Stock	Start Price	End Price	Capital Gain
BBCA	6,420	8,125	1,705
BBNI	5,575	9,225	3,650
BBRI	3,951	4,870	919
BBTN	1,640	1,845	205
BMRI	5,975	8,950	2,975
BFIN	665	1,285	620

Table 6. Simulation result of the portfolio optimization problem

Algorithm	Total Capital Gain (rupiah)
PSO	310,031,004
MPA	307,448,014
KMA	300,567,285
POA	341,894,419
GPA	340,711,745

simulation two, GPA is challenged to solve 23 benchmark functions in this simulation. The result is shown in Table 4.

Table 4 shows that in general, GPA is acceptable for low population size. GPA finds its convergence in the low population size in solving 16 functions. Most of the functions that are converged in the low population size are the multimodal functions. Meanwhile, the result is still improved in high population size in solving three unimodal functions: Sphere, Schwefel 1.2, and Schwefel 2.21.

Simulation four is performed to evaluate GPA in solving the real-world optimization problem. In this work, the portfolio optimization problem represents this real-world optimization problem, especially in the financial sector. A portfolio can be defined as a collection of investments owned by a person or institution [31]. These investments can be stocks, bonds, and any other assets. The common objectives of the portfolio optimization problem are maximizing return and minimizing risk.

In this simulation, the portfolio optimization problem focuses on the composition of six stocks that are listed in the LQ45 index. LQ45 index is a list of the 45 most preferred companies to be invested in Indonesia [32]. The criteria of this index are determined by the indonesia stock exchange. In general, the main criteria are market capitalization and liquidity. In this work, six companies listed in the LQ45 represent the dimension of the optimization problem. These companies are in the financial sector (banking and finance).

The goal is to maximize the total capital gain. The capital gain is the difference between the start price and end price in a one-year timespan. The price data is collected from Google. The price on 10 May 2021 represents the start price and the price on 28 April 2022 represents the end price. This data is shown in Table 6. The start price, end price, and capital gain are expressed in rupiah per share.

The investment scenario is as follows. The optimization handles one billion rupiah that can be used to buy stocks. The purchasing unit is in a lot that represents 100 shares. The purchasing price refers to the end price. The portfolio must consist of all stocks in Table 5 to distribute the investment risk. The number of lots in every stock must range from 100 lots to 400 lots. The purchasing volume cannot surpass the one billion rupiahs of investment.

The simulation scenario is as follows. GPA is compared with the four algorithms: PSO, MPA, KMA, and POA. The population size is set 20. The maximum iteration is 100. The result is shown in Table 6.

Table 6 shows that GPA is very competitive in solving the portfolio optimization problem. It outperforms all sparing algorithms. Its performance is 9%, 11%, and 13%, better than PSO, MPA, and KMA consecutively. Meanwhile, its less than 1% worse than POA. It means that the performance gap between GPA and PSO, MPA, and KMA is significant. On the other hand, the performance gap between GPA and POA is less significant.

5. Discussion

There are some findings related to the simulation result. First, in general, GPA becomes a good metaheuristic algorithm regarding its success to find the sub-optimal solution and tackle the local optimal entrapment. Second, GPA is very competitive because it outperforms the sparing algorithms in solving most benchmark functions. This circumstance happens in solving unimodal functions and multimodal functions. It means that GPA has enough time to find the area where the global optimal lies and to escape from the area where the local optimal lies. Moreover, its precision is high enough so that its performance is the best in solving most unimodal functions. Third, GPA is shown to be able to solve the theoretical optimization problem with low maximum iteration and a low number of candidates. Fourth, GPA is also competitive in solving the portfolio optimization problem. The in-depth analysis regarding these findings will be discussed below.

Generating multiple candidates along the path toward the global best candidate gives the advantage in scanning the possible better global best candidate faster. Due to the fixed size step between the adjacent candidates, the probability of missing this better candidate will be reduced. It is different from the stochastic-based guided movement with a single candidate for the next location, which is adopted in many metaheuristic algorithms. Neglecting the mechanism of determining the next location (fixed step, uniform, or normal distribution), the single candidate-based guided movement can scan only one possibility.

The guided movement implemented in GPA is proven much better than the randomized movement implemented in the original form of POA. Through this guided movement, GPA focuses on exploring the search space toward the global best solution without considering the other area. In general, it is proven that the probability of a better solution toward the best solution (local or global) is higher than any area inside the search space. In contrast, the randomized (unguided) movement as adopted in POA makes the agent's movement unconcentrated in a certain area. Although the unguided movement still can find the acceptable solution, it needs more iteration or a higher population size.

The approach adopted in GPA during phase two is better than the original POA. GPA's local search space tends to be wider than the original POA, especially in the earlier iteration. This circumstance gives better exploration although a gradual decline of the local search space is implemented in both algorithms. However, a wider local search space makes the algorithm performs better in avoiding the local optimal trap. In GPA, the disadvantage of the wider local search space is tackled by generating multiple candidates and by choosing the best candidate for the replacement. Although there is a possibility of the worse candidate being chosen as the replacement, overall, the guided movement toward the global best solution tackles this problem.

6. Conclusion

This work has demonstrated that GPA is proven a good and competitive algorithm. GPA can tackle two important in metaheuristic algorithms: finding the sub-optimal solution and escaping from the local optimal entrapment. In the theoretical optimization problem, GPA outperforms all benchmark algorithms in solving eighteen functions which are most of the 23 benchmark functions. GPA also outperforms all benchmark algorithms in solving the portfolio optimization problem. It outperforms all sparing algorithms. Its performance is 9%, 11%, and 13%, better than PSO, MPA, and KMA consecutively. Meanwhile, its less than 1% worse than POA.

There are future research opportunities regarding this work. Both POA and GPA are still widely possible to modify. These algorithms can also be combined or hybridized with other methods, such as metaheuristic, heuristic, or exact methods. More implementation of these algorithms to tackle many more real-world optimization problems is also challenging.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization: Kusuma; methodology: Kusuma, software: Kusuma, validation: Kusuma, formal analysis: Kusuma and Prasasti; writing-original paper draft: Kusuma; writing-review and editing: Prasasti, project administration: Kusuma, funding acquisition: Kusuma.

Acknowledgments

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

References

[1] J. Y., Lee, “A Genetic Algorithm for a Two-machine Flowshop with a Limited Waiting Time Constraint and Sequence-dependent Setup

Times", *Mathematical Problems in Engineering*, ID: 8833645, pp. 1-13, 2020.

[2] M. M. Ahmadian, A. Salehipour, and T. C. E. Cheng, "A Meta-heuristic to Solve the Just-in-time Job-shop Scheduling Problem", *European Journal of Operational Research*, Vol. 288, No. 1, pp. 14-29, 2021.

[3] 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.

[4] Z. J. Peya, M. A. H. Akhnand, T. Sultana, and M. M. H. Rahman, "Distance Based Sweep Nearest Algorithm to Solve Capacitated Vehicle Routing Problem", *International Journal of Advanced Computer Science and Application*, Vol. 10, No. 10, pp. 259-264, 2019.

[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. T. Ngo, J. Jaafar, I. A. Aziz, and B. N. Anh, "A Compromise Programming for Multi-Objective Task Assignment Problem", *Computers*, Vol. 10, ID: 15, pp. 1-16, 2021.

[7] P. D. Kusuma and A. S. Albana, "University Course Timetabling Model in Joint Courses Program to Minimize the Number of Unserved Request", *International Journal of Advanced Computer Science and Applications*, Vol. 12, No. 10, pp. 121-127, 2021.

[8] 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.

[9] 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.

[10] D. Freitas, L. G. Lopes, and F. M. Dias, "Particle Swarm Optimisation: A Historical Review up to the Current Developments", *Entropy*, Vol. 22, pp. 1-36, 2020.

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

[12] F. Rezaei, H. R. Safavi, M. A. Elaziz, S. H. A. E. Sappagh, M. A. A. Betar, and T. Abuhmed, "An Enhanced Grey Wolf Optimizer with a Velocity-Aided Global Search Mechanism", *Mathematics*, Vol. 10, ID: 351, pp. 1-32, 2022.

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

[14] M. Dehghani, M. Mardaneh, J. S. Guerrero, O. P. Malik, and V. Kumar, "Football Game Based Optimization: An Application to Solve Energy Commitment Problem", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 5, pp. 514-523, 2020, doi: 10.22266/ijies2020.1031.45.

[15] M. Dehghani, Z. Montazeri, S. Saremi, A. Dehghani, O. P. Malik, K. A. Haddad, and J. M. Guerrero, "HOGO: Hide Objects Game Optimization", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 4, pp. 216-225, 2020, doi: 10.22266/ijies2020.0831.19.

[16] M. Dehghani, Z. Montazeri, H. Givi, J. M. Guerrero, and G. Dhiman, "Darts Game Optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 5, pp. 286-294, 2020, doi: 10.22266/ijies2020.1031.26.

[17] W. Chang and W. Cheng, "A Mathematical Model and a Simulated Annealing Algorithm for Balancing Multi-manned Assembly Line Problem with Sequence-Dependent Setup Time", *Mathematical Problems in Engineering*, Vol. 2020, ID: 8510253, pp. 1-16, 2020.

[18] A. Ibrahim, F. Anayi, M. Packianather, and O. A. Alomari, "New Hybrid Invasive Weed Optimization and Machile Learning Approach for Fault Detection", *Energies*, Vol. 15, ID: 1488, 2022.

[19] F. H. Awad, A. A. kubaisi, and M. Mahmood, "Large-scale Timetabling Problems with Adaptive Tabu Search", *Journal of Intelligent Systems*, Vol. 31, No. 1, pp. 168-176, 2022.

[20] 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.

[21] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "A New Optimization Algorithm based on Average and Subtraction of the Best and Worst Members of the Population for Solving Various Optimization Problems", *PeerJ Computer Science*, Vol. 8, ID: e910, pp. 1-29, 2022.

[22] M. Noroozi, H. Mohammadi, E. Efatinasab, A. Lashgari, M. Eslami, and B. Khan, "Golden

Search Optimization Algorithm", *IEEE Access*, Vol. 10, pp. 37515-37532, 2022.
[23] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "Northern Goshawk Optimization: A New Swarm-Based Algorithm for Solving Optimization Problems", *IEEE Access*, pp. 162059-162080, 2021.
[24] M. Dehghani and P. Trojovsky, "Hybrid Leader Based Optimization: A New Stochastic Optimization Algorithm for Solving Optimization Applications", *Scientific Reports*, Vol. 12, ID: 5549, pp. 1-16, 2022.
[25] F. A. Zeidabadi, S. A. Doumari, M. Dehghani, and O. P. Malik, "MLBO: Mixed Leader Based Optimizer for Solving Optimization Problems", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 472-479, 2021, doi: 10.22266/ijies2021.0831.41.
[26] M. Dehghani, Z. Montazeri, A. Dehghani, R. A. R. Mendoza, H. Samet, J. M. Guerrero, and G. Dhiman, "MLO: Multi Leader Optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 6, pp. 364-373, 2020, doi: 10.22266/ijies2020.1231.32.
[27] F. A. Zeidabadi, M. Dehghani, and O. P. Malik, "TIMBO: Three Influential Members Based Optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 5, pp. 121-128, 2021, doi: 10.22266/ijies2021.1031.12.
[28] P. D. Kusuma and M. Kallista, "Stochastic Komodo Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 4, pp. 156-166, 2022, doi: 10.22266/ijies2022.0831.15.
[29] S. A. Yasear and H. M. A. Ghanimi, "A Modified Honey Badger Algorithm for Solving Optimal Power Flow Optimization Problem", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 4, pp. 142-155, 2022, doi: 10.22266/ijies2022.0831.14.
[30] 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, Nos. 4-2, pp. 218-223, 2017.
[31] 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.
[32] A. S. 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.