Received: January 23, 2023. Revised: February 8, 2023. 440

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Multiple Interaction Optimizer: A Novel Metaheuristic and Its Application to Solve Order Allocation Problem

Purba Daru Kusuma[1]* **Astri Novianty[1]**

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

Abstract: Order allocation problem is a crucial problem in the manufacturing system. Many studies have been carried out to overcome this problem. On the other hand, future studies related to this problem are still available due to its complexity, circumstance, and methods. This work develops a new metaheuristic called a multiple interaction optimizer (MIO). MIO has distinct mechanics in finding the optimal solution. MIO consists of two phases. In the first phase, each agent interacts with some randomly selected agents in the population. The guided search is conducted in every interaction. In the second phase, each agent carries out a local search which linearly reduces the search space during the iteration. Three tests are carried out on the performance of MIO. In the first test, the MIO is challenged to solve 23 functions. The second test is performed as the investigation of the hyper parameters. In the third test, MIO is challenged to solve the order allocation problem with the objective is minimizing the total cost, total lateness, and total defect. In this test, MIO is benchmarked with five metaheuristics: particle swarm optimization (PSO), marine predator algorithm (MPA), grey wolf optimizer (GWO), slime mould algorithm (SMA), and golden search optimizer (GSO). The result indicates that MIO is superior in solving both 23 functions and order allocation problems. In the theoretical test, MIO outperforms PSO, MPA, GWO, SMA, and GSO in 22, 21, 22, 19, and 18 functions respectively. MIO is also superior in achieving minimum cost, minimum lateness, and minimum total defect in solving order allocation problems.

Keywords: Operations research, Order allocation problem, Swarm intelligence, Metaheuristic.

1. Introduction

Order allocation is a crucial part of the manufacturing system. This process consists of selecting the supplier and determining the requested quantity [1]. This process becomes the first step in affecting the production cost and quality of service. In general, three main aspects are considered when ordering raw materials, i.e., quality, cost, and time. Any companies, especially manufacturers should make orders to any suppliers who can deliver the best raw materials in the combination of the aspects. The worse quality or higher number of defects in the raw material may reduce the quantity and quality of the finished product. The purchasing cost affects the production cost directly [2]. Higher purchasing cost leads to an increase in production cost and a decrease in profit. Meanwhile, tardiness of the incoming raw material may lead to tardiness in the finished product, and it can decrease the company's quality of service. Order allocation problems and supplier selection also become an integral part of supply chain management [3]. Meanwhile, order allocation becomes more difficult due to the uncertainty because the incoming customers' request is also uncertain [4].

Nowadays, there are many studies concerning the order allocation problem. This enormous number of works comes from the enormous circumstances faced in the order allocation problem. Meanwhile, there are several common objectives in order allocation or supplier selection, such as maximizing profit [5] or minimizing total cost [2]. Besides, there are many considerations taken by any companies in selecting suppliers or determining order quantity, such as price [6], lead time [7], location [8], responsiveness [9], flexibility [10], long term relationship [11], and many more. Many studies on the supplier selection or order

allocation problem utilized metaheuristic methods, such as genetic algorithm (GA) [2], grey wolf optimizer (GWO) [1], teaching learning-based optimizer (TLBO) [12], and so on.

Many studies in operational research, especially in the manufacturing system, used metaheuristic methods for the optimization process. The popularity of metaheuristics is related to its advantage as a stochastic method. Metaheuristic does not trace all possible solutions within the space so that it can tackle the computational resource limitation [13]. In the beginning, metaheuristic starts with a randomized solution. Then, the solution is improved during the iteration. This advantage comes with the consequence that metaheuristic does not guarantee to find the global optimal solution [13]. The second advantage is that metaheuristics can be implemented in any optimization problem if the objective and constraints of the problem are well-defined.

The popularity of metaheuristics also comes from the existence of an enormous number of metaheuristics, especially the swarm-based metaheuristics. It is easy to find any studies related to the optimization problem in the manufacturing system that utilized metaheuristics, such as in flow-shop scheduling [14], parallel machine scheduling [15], and so on. Meanwhile, there is a lot of new swarm-based metaheuristics proposed in the shortcoming years. Some metaheuristics adopt the behavior of animals during mating or foraging, such as the marine predator algorithm (MPA) [16], slime mould algorithm (SMA) [17], Komodo mlipir algorithm (KMA) [18], red deer algorithm (RDA) [19], stochastic Komodo algorithm (SKA) [20], pelican optimization algorithm (POA) [21], northern goshawk optimizer (NGO) [22], butterfly optimization algorithm (BOA) [23], and so on. Some metaheuristics adopt the mechanics of human activities, such as driving training-based optimizer (DTBO) [24], modified social forces algorithm (MSFA) [25], chef-based optimization algorithm (CBOA) [26], and so on. Some metaheuristics adopt the term leader that represents the mechanism of the guided search, such as hybrid leader-based optimizer (HLBO) [27], multi leader optimizer (MLO) [28], mixed leader-based optimizer (MLBO) [29], and so on. Some metaheuristics do not use metaphors and their name represent the main concept of their mechanics, such as average subtraction-based optimizer (ASBO) [30] and golden search optimizer (GSO) [31].

The explosive development of metaheuristics can be linked to the no-free-lunch theory and the nature of the stochastic approach. The no-free-lunch theory states that there is not any method that is suitable to solve all problems. Meanwhile, the performance of any method also depends on the problem to solve. Some metaheuristics may be good to solve some problems but perform poorly when solving other problems. For example, some algorithms may be better to solve unimodal problems while others are better at solving multimodal problems. A metaheuristic may perform well in solving problems with low dimensions or narrow search space, but it performs poorly in solving problems with high dimensions or large search space.

Even though a lot of swarm-based metaheuristics already exist, in most of them, each agent carries out only a single movement in the guided search. Metaheuristic with multiple agent movements in the guided search phase is rare to find. Moreover, operational research is not commonly implemented in the first introduction of many metaheuristics. Most of the studies proposing a new metaheuristic used the engineering problem for the practical case. Based on this circumstance, proposing a new swarm-based metaheuristic using an order allocation problem for its practical problem is still available.

This work aims to propose a new metaheuristic specially to solve the order allocation problem. The proposed metaheuristic is developed based on swarm intelligence in which the system consists of a population trying to find the optimal solution autonomously. Meanwhile, the agents interact with each other to improve the optimization process. The proposed metaheuristic is called a multiple interaction optimizer (MIO). Its novel mechanics is that each agent carries out multiple interactions, and each interaction is followed by the guided search. In most metaheuristics, each agent carries out only a single guided search in every iteration. Another contribution of this work is adopting the order allocation problem as an integer-based practical problem in the introduction of a new metaheuristic.

The remainder of this paper is constructed as follows. The literature regarding the order allocation problem and metaheuristic development is reviewed in the second section. A detailed explanation of the proposed metaheuristic is presented in the third section. The fourth section presents the carried-out test to evaluate the performance of MIO and its result. The fifth section discusses the in-depth analysis of the result, the relation between the test result and the theory, and the limitation. In the end, the sixth section summarizes the conclusion and the future research potential.

Table 1. Comparison among shortcoming metaheuristics and their practical problems in their first introduction

Metaheuristic	Ref. for guided search	Phases	Practical Problem	Reference
GSO	Best solution	1	-	[31]
ASBO	Best solution, worst solution	3	-	[30]
DTBO	Randomly selected solution	3	Pressure vessel design, welded beam design	[24]
SMA	Best solution, two randomly selected solutions	1	Welded beam structure problem, pressure vessel structure problem, cantilever structure problem, I-beam structure problem	[17]
NGO	Randomly selected solution	2	Pressure vessel, welded beam, tension/compression spring, speed reducer	[22]
BOA	Best solution, two randomly selected solutions	1	Spring design, welded beam design, and gear train design	[23]
MLBO	Best solution, randomly selected solution	1	-	[29]
MLO	Several best solutions	2	-	[28]
HLBO	Best solution, randomly selected solution	2	-	[27]
POA	Random solution within the search space	2	Pressure vessel, speed reducer, welded beam, spring	[21]
MPA	Local best solution, two randomly selected solution	2	Pressure vessel, welded beam, spring, fan scheduler, building design	[16]
MSFA	Randomly solution within the search space	1	-	[25]
KMA	Best solution, several best solutions	1	-	[18]

2. Related works

Many swarm-based metaheuristics adopt the mechanics of animals during searching for food or mating. Searching for an optimal solution has similarities to searching for food. From the agent's perspective, the location of the solution is unknown so the agent should trace the location based on the limited information in the most effective way. In swarm intelligence, there is interaction, i.e., information sharing among agents so that the objective can be achieved faster. Many stochastic approaches can be explored in the exploration and exploitation strategy carried out in any metaheuristic. Many metaheuristics utilize uniform distribution while other metaheuristics use normal distribution [20], Brownian movement [16], Levy movement [16], sinusoid distribution, and so on. Many metaheuristics utilize the best solution or some best solutions as a reference for their guided search. Meanwhile, some metaheuristics choose random selected solutions as the reference for their guided search. Some swarm intelligence-based metaheuristics consists of only single phase that combines the guided search and the random search. On the other hand, some other metaheuristics separates the guided search and random search so that there are multiple phases carried out by every agent in every iteration.

Guided search is a searching mechanism carried out by an agent with a reference representing the target. Some metaheuristic implements single possible direction while some others implement two possible directions. In the single-direction approach, the agent moves toward the reference. This approach is usually implemented in metaheuristics where the reference is the best solution, local best solution, or several best solutions. It can be seen in metaheuristics, such as GWO [32], BOA [23], and GSO [31]. In the two possible direction approach, the agent may move closer to the reference or move away from the reference. This approach is usually implemented in metaheuristics where the reference is a randomly selected agent or some randomly selected agents. It can be seen in metaheuristics, such as POA [27], NGO [22], and ASBO [30]. If the reference is better than the agent, then the agent moves toward the reference. Otherwise, the agent moves away from the reference. Below are the shortcoming metaheuristics including the basic concept and the practical problems used in their first introduction.

Table 1 indicates that there are many strategies can be taken to propose a new metaheuristic. The first part is determining the reference for the guided search. The

second part is determining the number of phases taken by each agent in every iteration. Besides, two more methods can be used in proposing a new metaheuristic. The first one is determining the random number calculation used in the proposed metaheuristic, whether it follows a uniform distribution, normal distribution, Brownian movement, and so on. The second one is determining the formulae in the interaction between the corresponding agent with its reference and in the mechanism carried out in the local search. Moreover, almost all metaheuristics implement single movement in the guided search. It means that proposing a new metaheuristic where an agent carries out multiple movements during the guided search is interesting.

Table 1 also indicates that the practical problems used in many existing studies proposing new metaheuristics are mechanical engineering problems. Most of them are pressure vessels, welded beams, and springs. All these problems are floating point-based problems. Ironically, studies proposing a new metaheuristic that use operational research problem, such as the order allocation problem is rare to find. Meanwhile, many problems in the operational research are presented in the integer-based parameters. On the other hand, many metaheuristics were not tested to solve practical problems in their first introduction although the existence of practical problem tests is important. Based on this circumstance, proposing a new metaheuristic that uses the order allocation problem as the practical problem becomes a novel contribution.

3. Model

MIO consists of several agents that represent the solutions. These agents search autonomously within the search space to find the optimal solution. Each agent carries out two phases in every iteration. The first iteration is the guided search while the second phase is the local search. The guided search is designed to improve the current solution while the local search is designed to tackle the local optimal problem. MIO adopts a strict acceptance-rejection approach. This approach is implemented in both guided search and local search.

The novel mechanism of MIO is that each agent interacts with multiple other agents in the population in the first phase. These other agents can be called partners. These partners are selected randomly among the population. There is a possibility that an agent selects the same partner more than once in a single iteration because the selection is carried out randomly and there is no checking activity. Each time an agent

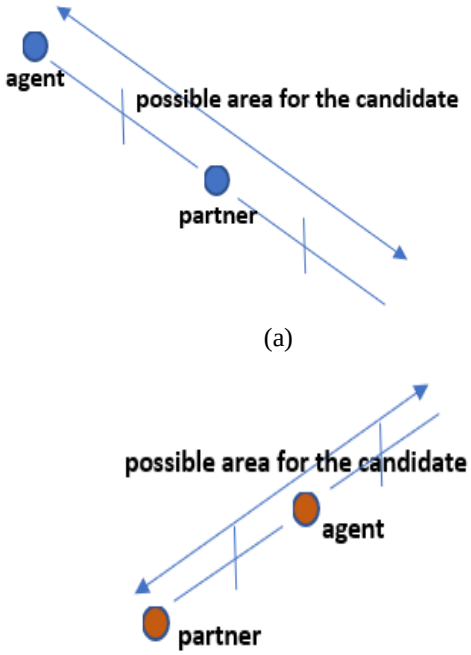

Figure. 1 Guided search: (a) movement toward the partner, and (b) movement toward the agent

interacts with a partner, the agent carries out a guided search or movement. The number of interactions in every iteration is set manually before the optimization runs. This mechanism means that the number of guided searches is linear to the number of interactions. It is different from many other metaheuristics which consider multiple agents but conduct a single search.

There are two possible directions regarding this guided search. Each search generates a candidate. If the agent is better than its partner, then the search is carried out by the movement from the agent toward or surpasses its partner. On the other hand, if the agent is not better than its partner, then the search is carried out by the partner toward or surpasses the agent. This movement is illustrated in Fig. 1.

Fig. 1 (a) represents the movement toward the partner while Fig. 1 (b) represents the movement toward the agent. This strategy reasons that the candidate is designed to improve the current solution. When the partner is better than the agent then the probability that the better solution is near the partner is higher than near the agent. Surpassing the partner is designed to find a better solution than the partner. The same reason is also applied when the agent is better than its partner.

In MIO, each agent carries out two phases in every iteration. The first phase is the guided search presented previously. The second phase is the local search. In the local search, an agent searches for a

algorithm 1: multiple interaction optimizer

1 determine n_i using (1)
2 **for** s **in** S
3 initiate s using (2)
4 update s_{best} using (3)
5 **end for**
6 **for** t=1 **to** t_{max}
7 **for** s **in** S
8 **for** i=1 **to** n_i
9 find s_t using (4)
10 conduct guided search using (5) and (6)
11 update s_{best} using (3)
12 **end for**
13 conduct local search using (7) and (8)
14 update s_{best} using (3)
15 **end for**
16 **end for**
17 **return** s_{best}

b_l	lower boundary
b_u	upper boundary
f	fitness function
n	set size
n_i	number of interactions
r_i	interaction ratio
r_1	uniform real random number between 0 and 1
r_2	uniform integer random number between 1 and 2
r_3	uniform real random number between -1 and 1
s	solution
S	set of solution
s_{best}	best solution
s_g	guided search-based solution
s_l	local search-based solution
s_t	selected solution
t	iteration
t_{max}	maximum iteration
U	uniform random

new solution within its local search space. In the beginning, the local search space width is twice as big as half of the search space. Then, this local search space width declines linearly due to the increase in iteration. At the end of the iteration, the local search space width is zero. The wide local search space in the early iteration is to accommodate more exploration and find the area of the global optimal solution as early as possible.

The formalization of MIO is presented in Algorithm 1. The annotations used in this work can be seen below.

The explanation of algorithm 1 is as follows. MIO can be divided into two parts: the initialization and iteration. Lines 2 to 5 represent the initialization. Meanwhile, lines 6 to 16 represent the iteration. The best solution becomes the final solution. The initialization consists of single loop. Meanwhile, the iteration consists of three loops. The outer loop represents the iteration from the first iteration to the maximum iteration. The middle loop represents the searching carried out by every agent. The inner loop represents the number of iterations carried out by every agent. Lines 8 to 12 represent the guided search while lines 13 and 14 represent the local search. The mathematical model used in MIO can be seen in Eq. (1) to Eq. (8).

$$n_i = f.n(S) \tag{1}$$

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

$$s'_{best} = \begin{cases} s, f(s) < f(s_{best}) \\ s_{best}, else \end{cases} \tag{3}$$

$$s_t = U(S) \tag{4}$$

$$s_g = \begin{cases} s + r_1(s_t - r_2.s), f(s_t) < f(s) \\ s_t + r_1(s - r_2.s_t), else \end{cases} \tag{5}$$

$$s' = \begin{cases} s_g, f(s_g) < f(s) \\ s, else \end{cases} \tag{6}$$

$$s_l = s + \left(1 - \frac{t}{t_m}\right)\left(\frac{r_3(b_u - b_l)}{2}\right) \tag{7}$$

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

The explanation of Eq. (1) to Eq. (8) is as follows. Eq. (1) is used to determine the number of iterations carried out by each agent and it is the fraction of the population. Eq. (2) states that the initial solution is generated uniformly within the solution space. Eq. (3) states that the current solution replaces the best solution only if it is better than the best solution. Eq. (4) states that the partner is selected uniformly among the population. Eq. (5) states that the direction of the guided search depends on the quality of the agent compared to its partner. Eq. (6) states that the guided search candidate replaces the current solution only if it is better than the current solution. Eq. (7) states that the local search solution is randomly selected within the local search space and the local search space width declines linearly due to the iteration. Eq. (8) states that the local search candidate replaces the current solution only if it is better than the current solution. Eq. (6) and

Eq. (8) represent the strict acceptance-rejection strategy.

4. Simulation and result

The proposed MIO is tested to solve both the theoretical problem and the practical problem. The 23 functions represent the theoretical optimization problem. These functions can be clustered into three groups: (1) seven high-dimension unimodal functions, (2) six high-dimension multimodal functions, and (3) fixed-dimension multimodal functions. Meanwhile, the order allocation problem represents the practical problem. The 23 functions are popular due to their coverage of any problems in the optimization work. They cover problems from a very narrow space to a very large. They also cover problems from the easiest ones to very difficult ones to solve. Some functions are very smooth and non-ambiguous so that the area of the global optimal is easy to find. Meanwhile, some functions are wavy or flat with a very narrow area of the global optimal. It makes many metaheuristics reach their convergence without finding the area of the global optimal. The detail description of the 23 functions can be seen in Table 2.

In this test, MIO is benchmarked with five metaheuristics: PSO, MPA, GWO, SMA, and GSO. PSO is the early version of swarm intelligence-based metaheuristic [34]. It is proven to be used in many studies solving the optimization problem in the manufacturing system. GWO represents a new well-known metaheuristic. GWO is also implemented in many works in the manufacturing system. MPA and SMA are new metaheuristics developed in recent years. GSO is the newest metaheuristic among these benchmark metaheuristics.

The parameter setup in this test is as follows. The population size is 10 and the maximum iteration is 50. This setup makes challenges for the metaheuristics to solve the problems in the low iteration and low population. This setup is implemented in all metaheuristics. The dimension for the high-dimension functions is set to 30. In PSO, the weights are 0.1. In MPA, the fishing aggregate devices are set to 0.1. In MIO, the interaction ratio is 0.5. The result is presented in Table 3, Table 4, and Table 5. These tables present the average case, best case, and worst case respectively. The best result in the average case is written in bold font. Meanwhile, the superiority comparison in every function group is presented in Table 6.

Table 3 indicates that MIO performs well in solving the 23 functions. It can find the global optimal of F9 and F11. Moreover, MIO is superior to other metaheuristics by creating the best result in 15 functions. Among these 15 functions, six functions are high-dimension unimodal functions, five functions are high-dimension multimodal functions, and four functions are fixed-dimension multimodal functions. This result indicates that MIO is almost superior in solving the high dimension functions. On the other hand, MIO is not so superior in solving the fixed-dimension multimodal functions.

Table 4 indicates that in general, these six metaheuristics can find the global optimal solution at its best case. It means that all these metaheuristics are good. The problem is in the average case and the worst case. In many runs, these metaheuristics create final solution that is too far from the acceptable solution. This condition makes the average cases of these metaheuristics, especially the benchmark metaheuristics are not so good so that these benchmark metaheuristics are outperformed by the proposed MIO.

Table 6 strengthens the superiority of MIO among the benchmark metaheuristics. MIO is almost superior to PSO, MPA, and GWO by outperforming these three metaheuristics in almost all functions. The superiority of MIO to PSO, MPA, and GWO occurs in all groups of functions. On the other hand, MIO is very superior to SMA and GSO in the first and second groups and still superior in the third groups although its superiority is not so high as in the first and second groups.

The second test is performed to investigate the hyper parameter of the proposed MIO. This test is aimed to evaluate which parameters have significant influence on the performance of MIO. In this test, the proposed MIO is also challenged to solve the 23 functions as in the first test. There are three parameters includes the maximum iteration, population size, and interaction ratio. All these parameters are positive proportional to the complexity. In the first evaluation, there are two values of population size: 20 and 40. In the second evaluation, there are two values of maximum iteration: 40 and 80. In the third evaluation, there are two values of interaction ratio: 40 and 80. The result is presented Table 7, Table 8, and Table 9.

Result in Table 7 indicates that only five functions where the quality of the solution can be improved significantly by the increasing of population size from 20 to 40. Among these five functions, four functions are the high dimension unimodal functions, and one function is fixed dimension multimodal functions. Meanwhile there is not any high dimension multimodal function in which its solution is improved significantly by increasing the population size from 20 to 40. But there are six multimodal functions where their solution is not improved because the

Table 2. Detail description of 23 benchmark functions

No	Function	Model	Dim	Space	Target
1	Sphere	$\sum_{i=1}^{d} x_i^2$	30	[-100, 100]	0
2	Schwefel 2.22	$\sum_{i=1}^{d}\|x_i\| + \prod_{i=1}^{d}\|x_i\|$	30	[-100, 100]	0
3	Schwefel 1.2	$\sum_{i=1}^{d}\left(\sum_{j=1}^{i} x_j\right)^2$	30	[-100, 100]	0
4	Schwefel 2.21	$\max\{\|x_i\|, 1 \le i \le d\}$	30	[-100, 100]	0
5	Rosenbrock	$\sum_{i=1}^{d-1}(100(x_{i+1} + x_i^2)^2 + (x_i - 1)^2)$	30	[-30, 30]	0
6	Step	$\sum_{i=1}^{d-1}(x_i + 0.5)^2$	30	[-100, 100]	0
7	Quartic	$\sum_{i=1}^{d} i\, x_i^4 + random\ [0,1]$	30	[-1.28, 1.28]	0
8	Schwefel	$\sum_{i=1}^{d} -x_i \sin\left(\sqrt{\|x_i\|}\right)$	30	[-500, 500]	-1.2569x10^4
9	Ratsrigin	$10d + \sum_{i=1}^{d}\left(x_i^2 - 10cos(2\pi x_i)\right)$	30	[-5.12, 5.12]	0
10	Ackley	$-20 \cdot exp\left(-0.2 \cdot \sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i{}^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos 2\pi x_i\right) + 20 + exp(1)$	30	[-32, 32]	0
11	Griewank	$\frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} cos\left(\frac{x_i}{\sqrt{i}}\right)$+1	30	[-600, 600]	0
12	Penalized	$\frac{\pi}{d}\left\{10\sin(\pi y_1) + \sum_{i=1}^{d-1}\left((y_i - 1)^2(1 + 10\ sin^2(\pi y_{i+1}))\right) + (y_d - 1)^2\right\} + \sum_{i=1}^{d} u(x_i, 10,100,4)$	30	[-50, 50]	0
13	Penalized 2	$0.1\left\{sin^2(3\pi x_1) + \sum_{i=1}^{d-1}\left((x_i - 1)^2(1 + sin^2(3\pi x_{i+1}))\right) + (x_d - 1)^2(1 + sin^2(2\pi x_d))\right\} + \sum_{i=1}^{d} u(x_i, 5,100,4)$	30	[-50, 50]	0
14	Shekel Foxholes	$\left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	[-65, 65]	1
15	Kowalik	$\sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$	4	[-5, 5]	0.0003
16	Six Hump Camel	$4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5, 5]	-1.0316
17	Branin	$\left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	2	[-5, 5]	0.398
18	Goldstein-Price	$\left(1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right).\left(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right)$	2	[-2, 2]	3
19	Hartman 3	$-\sum_{i=1}^{4}\left(c_i exp\left(-\sum_{j=1}^{d}\left(a_{ij}(x_j - p_{ij})^2\right)\right)\right)$	3	[1, 3]	-3.86
20	Hartman 6	$-\sum_{i=1}^{4}\left(c_i exp\left(-\sum_{j=1}^{d}\left(a_{ij}(x_j - p_{ij})^2\right)\right)\right)$	6	[0, 1]	-3.32
21	Shekel 5	$-\sum_{i=1}^{5}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$	4	[0, 10]	-10.1532
22	Shekel 7	$-\sum_{i=1}^{7}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$	4	[0, 10]	-10.4028
23	Shekel 10	$-\sum_{i=1}^{10}\left(\sum_{j=1}^{d}(x_j - c_{ji})^2 + \beta_i\right)^{-1}$	4	[0, 10]	-10.5363

final solution is the global optimal solution or very close to the global optimal solution. This circumstance means that in general, high population size scenario does not improve the quality of the solution for multimodal functions.

Result in Table 8 indicates that only five functions where the quality of the solution can be improved significantly by the increasing of maximum iteration from 40 to 80. All these five functions are high dimension unimodal functions. Meanwhile, there is not any multimodal functions where their solution is improved due to the increasing of maximum iteration from 40 to 80. Like in the previous test, there are six multimodal functions where the final solution is the

Table 3. Average case comparison in solving 23 functions

Function	Average Fitness Score					
	PSO [34]	MPA [16]	GWO [32]	SMA [17]	GSO [31]	MIO
1	1.065×10^{4}	2.890×10^{1}	6.235×10^{2}	1.449×10^{3}	8.733×10^{3}	**3.188×10^{-44}**
2	**0.0000**	2.457×10^{-36}	**0.000**	**0.000**	1.823×10^{37}	6.110×10^{-259}
3	4.416×10^{4}	5.242×10^{2}	3.098×10^{3}	1.161×10^{4}	2.064×10^{4}	**2.087×10^{-14}**
4	3.961×10^{1}	3.948	1.744×10^{1}	1.323×10^{1}	3.260×10^{1}	**2.805×10^{-19}**
5	7.680×10^{6}	4.391×10^{2}	5.123×10^{3}	1.534×10^{6}	6.445×10^{6}	**2.891×10^{1}**
6	9.795×10^{3}	2.866×10^{1}	4.904×10^{2}	1.391×10^{3}	8.909×10^{3}	**5.466**
7	3.451	4.418×10^{-2}	2.029	1.441×10^{2}	3.223	**3.932×10^{-3}**
8	-2.246×10^{3}	-2.896×10^{3}	-3.717×10^{1}	**-6.722×10^{3}**	-3.735×10^{3}	-2.536×10^{3}
9	2.628×10^{2}	1.766×10^{1}	1.648	2.136×10^{1}	1.766×10^{2}	**0.000**
10	1.492×10^{1}	1.313	1.114×10^{1}	6.437	1.785×10^{1}	**3.997×10^{-15}**
11	9.077×10^{1}	1.121	9.772×10^{-1}	9.519	7.964×10^{1}	**0.000**
12	2.153×10^{6}	1.913	1.219×10^{5}	2.845×10^{4}	1.307×10^{6}	**6.882×10^{-1}**
13	1.427×10^{7}	5.113	6.094×10^{4}	1.461×10^{6}	1.307×10^{7}	**2.952**
14	9.555	9.131	2.513×10^{1}	**4.306**	9.222	5.341
15	8.840×10^{-2}	1.194×10^{-2}	2.149×10^{-1}	1.176×10^{-1}	1.373×10^{-2}	**6.500×10^{-4}**
16	-4.784×10^{-2}	-9.824×10^{-1}	-1.779×10^{-4}	-3.935×10^{-2}	**-1.032**	-1.030
17	5.560×10^{1}	2.080	5.575×10^{1}	6.376×10^{-1}	**3.981×10^{-1}**	3.998×10^{-1}
18	5.995×10^{2}	1.990×10^{1}	5.776×10^{2}	1.550×10^{1}	**3.001**	3.009
19	-1.258×10^{-3}	**-3.490**	-2.350×10^{-3}	-4.954×10^{-2}	-4.474×10^{-2}	-4.954×10^{-2}
20	-5.089×10^{-3}	-1.583	-5.089×10^{-3}	-1.163	-2.977	**-3.168**
21	-2.731×10^{-1}	-1.100	-2.731×10^{-1}	-4.612	**-5.483**	-5.309
22	-2.936×10^{-1}	-1.072	-2.936×10^{-1}	-5.304	-5.518	**-5.587**
23	-3.217×10^{-1}	-1.142	-3.217×10^{-1}	-4.289	-4.900	**-5.128**

Table 4. Best case comparison in solving 23 functions

Function	Best Fitness Score					
	PSO [34]	MPA [16]	GWO [32]	SMA [17]	GSO [31]	MIO
1	7.061×10^{3}	2.699×10^{-1}	2.395×10^{-4}	8.665	3.000×10^{3}	4.402×10^{-48}
2	0.0000	0.000	0.000	0.000	0.000	0.000
3	1.922×10^{4}	1.618×10^{1}	1.040×10^{-2}	2.242×10^{3}	6.899×10^{3}	1.017×10^{-20}
4	2.987×10^{1}	3.045×10^{-1}	2.823×10^{-2}	0.000	2.449×10^{1}	2.018×10^{-20}
5	3.016×10^{6}	3.012×10^{1}	2.892×10^{1}	2.282×10^{3}	1.128×10^{6}	2.881×10^{1}
6	5.162×10^{3}	7.640	6.509	1.325×10^{1}	4.172×10^{3}	4.979
7	1.177	2.074×10^{-3}	1.070×10^{-2}	3.128×10^{1}	7.366×10^{-1}	1.937×10^{-5}
8	-3.277×10^{3}	-3.734×10^{3}	-3.803×10^{2}	-8.008×10^{3}	-5.171×10^{3}	-3.277×10^{3}
9	2.085×10^{2}	2.683×10^{-1}	9.104	2.000	8.667×10^{1}	0.000
10	1.324×10^{1}	3.690×10^{-1}	2.544×10^{-3}	3.082	1.304×10^{1}	3.997×10^{-15}
11	4.772×10^{1}	2.789×10^{-2}	1.193×10^{-6}	0.000	4.447×10^{1}	0.000
12	2.739×10^{5}	1.116	1.108	2.479×10^{-1}	1.375×10^{2}	4.332×10^{-1}
13	1.394×10^{6}	3.234	2.974	1.172×10^{1}	1.698×10^{6}	2.546
14	2.012	3.806	1.267×10^{1}	9.980×10^{-1}	1.992	0.998
15	1.678×10^{-3}	1.722×10^{-3}	8.912×10^{-2}	4.375×10^{-2}	9.944×10^{-4}	3.128×10^{-4}
16	-9.853×10^{-1}	-1.030	-6.420×10^{-3}	-5.740×10^{-1}	-1.032	-1.032
17	5.560×10^{1}	5.531×10^{-1}	5.542×10^{1}	5.070×10^{-1}	3.981×10^{-1}	3.981×10^{-1}
18	5.918×10^{2}	3.448	4.338×10^{1}	3.000	3.000	3.000
19	-1.617×10^{-2}	-3.925	-1.997×10^{-2}	-4.954×10^{-2}	-4.954×10^{-2}	-4.954×10^{-2}
20	-5.089×10^{-3}	-2.584	-5.089×10^{-3}	-2.714	-3.314	-3.310
21	-2.731×10^{-1}	-4.180	-2.731×10^{-1}	-1.015×10^{1}	-1.014×10^{1}	-9.866
22	-2.936×10^{-1}	-1.807	-2.936×10^{-1}	-1.040×10^{1}	-1.040×10^{1}	-9.220
23	-3.217×10^{-1}	-2.508	-3.217×10^{-1}	-1.054×10^{1}	-1.054×10^{1}	-9.521

Table 5. Worst case comparison in solving 23 functions

Function	Worst Fitness Score					
	PSO [34]	MPA [16]	GWO [32]	SMA [17]	GSO [31]	MIO
1	$1.447x10^{4}$	$1.911x10^{2}$	$2.651x10^{3}$	$3.823x10^{3}$	$1.631x10^{4}$	$5.121x10^{-44}$
2	0.0000	$5.406x10^{-35}$	0.000	0.000	$2.323x10^{38}$	$1.939x10^{-172}$
3	$1.172x10^{5}$	$1.955x10^{3}$	$5.162x10^{3}$	$3.919x10^{4}$	$3.659x10^{4}$	$1.391x10^{-13}$
4	$5.611x10^{1}$	$1.034x10^{1}$	$1.000x10^{2}$	$3.700x10^{1}$	$4.230x10^{1}$	$1.122x10^{-18}$
5	$1.873x10^{7}$	$2.503x10^{3}$	$5.380x10^{4}$	$7.750x10^{6}$	$3.316x10^{7}$	$2.894x10^{1}$
6	$1.455x10^{4}$	$1.061x10^{2}$	$3.294x10^{3}$	$4.237x10^{3}$	$1.640x10^{4}$	5.905
7	6.826	$1.161x10^{-1}$	$3.502x10^{1}$	$2.107x10^{2}$	7.233	$1.768x10^{-2}$
8	$-1.276x10^{3}$	$-2.237x10^{3}$	$2.658x10^{1}$	$5.700x10^{1}$	$-2.411x10^{3}$	$-1.820x10^{3}$
9	$2.923x10^{2}$	$4.662x10^{1}$	$4.506x10^{1}$	9.561	$2.283x10^{2}$	0.000
10	$1.672x10^{1}$	3.902	8.930	$3.606x10^{1}$	$2.004x10^{1}$	$3.997x10^{-15}$
11	$1.340x10^{2}$	4.804	6.310	$3.107x10^{5}$	$2.023x10^{2}$	0.000
12	$1.319x10^{7}$	3.216	$2.538x10^{6}$	$2.845x10^{4}$	$1.434x10^{7}$	$9.303x10^{-1}$
13	$5.444x10^{7}$	8.271	$1.309x10^{6}$	$9.058x10^{6}$	$4.410x10^{7}$	3.116
14	$3.591x10^{1}$	$1.267x10^{1}$	$2.867x10^{2}$	$1.267x10^{1}$	$1.645x10^{1}$	9.234
15	$6.369x10^{-1}$	$3.407x10^{-2}$	1.699	$1.484x10^{-1}$	$4.961x10^{-2}$	$2.437x10^{-3}$
16	$7.380x10^{-5}$	$-7.362x10^{-1}$	$2.485x10^{-3}$	0.000	-1.032	-1.022
17	$5.568x10^{1}$	6.078	$5.773x10^{1}$	$6.438x10^{-1}$	$3.981x10^{-1}$	$4.045x10^{-1}$
18	$6.015x10^{2}$	$5.894x10^{1}$	$6.525x10^{2}$	$2.780x10^{2}$	3.015	3.139
19	$-2.821x10^{-15}$	-2.659	$-9.346x10^{-21}$	$-4.954x10^{-2}$	$-6.072x10^{-3}$	$-4.954x10^{-2}$
20	$-5.089x10^{-3}$	$-4.839x10^{-1}$	$-5.089x10^{-3}$	$-3.066x10^{--1}$	-2.289	-3.003
21	$-2.731x10^{-1}$	$-4.533x10^{-1}$	$-2.731x10^{-1}$	$-8.809x10^{-1}$	-2.453	-1.860
22	$-2.936x10^{-1}$	$-5.383x10^{-1}$	$-2.936x10^{-1}$	$-9.097x10^{-1}$	-2.723	-3.537
23	$-3.217x10^{-1}$	$-6.899x10^{-1}$	$-3.217x10^{-1}$	$-9.487x10^{-1}$	-2.334	-2.444

Table 6. Superiority of MIO based on the cluster of functions

Cluster	Number of functions where MIO is superior				
	PSO [34]	MPA [16]	GWO [32]	SMA [17]	GSO [31]
1	6	7	6	6	7
2	6	5	6	5	5
3	10	9	10	8	6
Total	22	21	22	19	18

Table 7. Relation between population size and average fitness score

Function	Average Fitness Score		Significantly Improved?
	$n(S) = 20$	$n(S) = 40$	
1	$8.014x10^{-77}$	$3.349x10^{-132}$	yes
2	0.000	0.000	no
3	$4.784x10^{-29}$	$1.010x10^{-53}$	yes
4	$1.062x10^{-32}$	$2.989x10^{-57}$	yes
5	$2.888x10^{1}$	$2.885x10^{1}$	no
6	4.823	4.365	no
7	$2.202x10^{-3}$	$1.010x10^{-3}$	yes
8	$-2.682x10^{3}$	$-2.526x10^{3}$	no
9	0.000	0.000	no
10	$3.997x10^{-15}$	$3.997x10^{-15}$	no
11	0.000	0.000	no
12	$5.193x10^{-1}$	$4.334x10^{-1}$	no
13	2.838	2.695	no
14	2.257	1.092	yes
15	$4.034x10^{-4}$	$3.216x10^{-4}$	no
16	-1.031	-1.031	no
17	$3.985x10^{-1}$	$3.981x10^{-1}$	no
18	3.003	3.000	no
19	$-4.954x10^{-2}$	$-4.954x10^{-2}$	no
20	-3.238	-3.275	no
21	-5.075	-5.774	no
22	-6.375	-6.505	no
23	-5.695	-5.677	no

global optimal solution or very close to the global optimal solution.

Result in Table 9 indicates that only eight functions where the quality of the solution can be improved significantly by the increasing of interaction ratio from 0.2 (low interaction ratio) to 0.8 (high interaction ratio). Among these eight functions, five functions are high dimension unimodal functions, two functions are high dimension multimodal functions, and one function is fixed dimension multimodal function. Like in the previous test, there are six multimodal functions where the final solution is the global optimal solution or very close to the global optimal solution.

In the third test, MIO is challenged to solve the order allocation problem. The scenario adopted from [2] represents a single customer, multiple vendors, and a single product. A manufacturer should make an

Table 8. Relation between maximum iteration and average fitness score

Function	Average Fitness Score		Significantly Improved?
	t_{max} = 40	t_{max} = 80	
1	$1.527x10^{-34}$	$3.171x10^{-74}$	yes
2	$6.146x10^{-168}$	0.000	yes
3	$1.754x10^{-9}$	$6.952x10^{-24}$	yes
4	$5.733x10^{-15}$	$1.243x10^{-31}$	yes
5	$2.892x10^{1}$	$2.890x10^{1}$	no
6	5.289	5.310	no
7	$8.101x10^{-3}$	$2.633x10^{-3}$	yes
8	$-2.376x10^{3}$	$-2.742x10^{3}$	no
9	0.000	0.000	no
10	$3.997x10^{-15}$	$3.997x10^{-15}$	no
11	0.000	0.000	no
12	$7.173x10^{-1}$	$6.324x10^{-1}$	no
13	2.976	2.892	no
14	5.187	3.019	no
15	$5.437x10^{-4}$	$4.541x10^{-4}$	no
16	-1.030	-1.031	no
17	$3.996x10^{-1}$	$3.991x10^{-1}$	no
18	3.014	3.004	no
19	$-4.954x10^{-2}$	$-4.954x10^{-2}$	no
20	-3.105	-3.218	no
21	-4.274	-5.760	no
22	-5.065	-6.659	no
23	-4.827	-5.596	no

Table 9. Relation between interaction ratio and average fitness score

Function	Average Fitness Score		Significantly Improved?
	r_i = 0.2	r_i = 0.8	
1	$2.028x10^{-18}$	$9.689x10^{-67}$	yes
2	$4.531x10^{-69}$	0.000	yes
3	$7.355x10^{-3}$	$2.824x10^{-24}$	yes
4	$2.924x10^{-8}$	$2.973x10^{-28}$	yes
5	$2.893x10^{1}$	$2.891x10^{1}$	no
6	5.533	5.090	no
7	$1.210x10^{-2}$	$3.348x10^{-3}$	yes
8	$-2.551x10^{3}$	$-2.819x10^{3}$	no
9	0.000	0.000	no
10	$1.803x10^{-12}$	$3.997x10^{-15}$	yes
11	$1.734x10^{-7}$	0.000	yes
12	$7.418x10^{-1}$	$6.583x10^{-1}$	no
13	3.059	2.856	no
14	6.292	3.864	no
15	$2.143x10^{-3}$	$5.444x10^{-4}$	yes
16	-1.029	-1.030	no
17	$3.992x10^{-1}$	$3.986x10^{-1}$	no
18	3.023	3.006	no
19	$-4.954x10^{-2}$	$-4.954x10^{-2}$	no
20	-3.160	-3.189	no
21	-4.028	-4.850	no
22	-4.435	-5.215	no
23	-4.079	-4.877	no

Table 10. Performance of MIO in solving order allocation problem

Metaheuristic	Total Cost	Total Late	Total Defect
PSO [34]	195.9	101.6	71.9
MPA [16]	194.8	97.9	69.5
GWO [32]	201.5	125.3	80.5
SMA [17]	191.9	90.4	64.8
GSO [31]	193.6	91.8	65.1
MIO	191.7	90.2	64.1

order of 2,000 components from six vendors. Each vendor has a minimum and maximum order quantity. It means that orders to a certain vendor should not below the minimum quantity but also should not surpass the minimum quantity. If the order is below the minimum quantity, then the corresponding vendor will reject it. On the other hand, if the order is above the maximum quantity, then the vendor will fail to fulfil it. Each vendor has its own cost which is different from others. Meanwhile, each vendor has its own defect ratio and late ratio.

This third test consists of three subtests that represent different objectives. The objective of the first sub-test is to minimize the total cost. The objective of the second sub-test is to minimize the total number of products that come late. The objective of the third sub-test is to minimize the total number of defective products. This test can be seen as an integer-based optimization problem because the product quantity and cost are discrete and presented in integers. It is different from the first test where the solution space is presented in a floating-point number. The result is presented in Table 10.

Table 10 indicates that MIO is superior to all benchmark metaheuristics in solving the order allocation problem. On the other hand, GWO becomes metaheuristic with poorest performance. Comparing between MIO and GWO, MIO creates 4.8%, 2.8%, and 20.4% lower than GWO in total cost, total lateness, and total defect consecutively. Meanwhile, the gap between MIO and SMA is very near although MIO is still better. MIO creates 0.1%, 0.2%, and 1.1% lower than SMA in total cost, total lateness, and total defect.

5. Discussion

In general, the test result indicates that MIO performs well and is competitive as a swarm intelligence-based metaheuristic. This statement is supported by several test results. First, MIO can find an acceptable solution in both 23 functions and the order allocation problem. Moreover, MIO can find the global optimal solution of F9 and F11. MIO is

superior to all benchmark metaheuristics by outperforming PSO, MPA, GWO, SMA, and GSO in 22, 21, 22, 19, and 18 functions consecutively. Meanwhile, MIO becomes the best metaheuristic for solving the order allocation problem. MIO is superior to PSO, MPA, and GWO in all groups of functions. Meanwhile, MIO is superior to SMA and GWO in the first and second groups. This proposed MIO also performs well in the low population size and low maximum iteration as it is presented in Table 7 and Table 8.

The in-depth analysis can be drawn back to the main concept of the two-phase strategy adopted by MIO. The multiple interaction strategy carried out in the first phase performs well in achieving excellent results in solving functions in the first and second groups. Multiple interactions followed by multiple guided searches in MIO make the improvement faster than single or limited interaction carried out in the guided search as in PSO and MPA. The result also shows that multiple interaction strategy is more powerful than the use of a leader or the best solution in the guided search. All these benchmark metaheuristics utilize the best solution for the guidance, whether it is the global best solution as in SMA [17], the local best solution as in MPA, the combination of global best and local best as in PSO [34], or several best solutions as in GWO [32]. The approach of getting closer to the better solution and avoiding the worse solution is more relevant as it is adopted in many shortcoming metaheuristics, such as in KMA [18], MLO [28], DTBO [24], HLBO [27], and many more. The random search in the second phase is proven important to tackle the local optimal problem. The guided search is proven enough to solve the basic multimodal problems as in the second group. Meanwhile, the random search as in the second phase plays important role in finding the global optimal solution where the global optimal solution is in a very narrow area, or this area is ambiguous as in the third group. This circumstance makes many shortcoming metaheuristics deploy multiple strategies whether it is implemented into multiple phase strategy as in NGO [22] or ASBO [30], or multiple role strategy as in KMA [18].

The result in Table 3 and Table 8 strengthens the argument that any metaheuristic should be tested not only by using theoretical or mathematical functions but also by the practical problem. It is presented that making improvements in solving practical problems is more difficult rather than solving theoretical functions. The improvement created by MIO is much more significant than PSO where the gap is very wide. Meanwhile, Table 8 shows that making more than 10 percent improvement is very difficult. The main problem comes from the type of number used in the practical problem. Many operations research problem, such as the order allocation problem, is developed based on the integer number. It is because the quantity of any goods, whether they are products, components, or vehicles, cannot be split into floating point numbers. On the other hand, high advancement in solving theoretical problems comes from the high precision floating point number. Meanwhile, high-precision floating point number is not common in many engineering optimization problems because it is not practical.

The complexity of MIO can be presented as $O(t_{max}.n_i.n(S))$. It means the complexity of MIO is linear to the maximum iteration, several interactions, and population size. In general, the complexity of all metaheuristics is linear to the maximum iteration because of the nature of metaheuristics that improves the solution by iterating the searching process. The linearity of population size and complexity is also common in all population-based metaheuristics, especially swarm intelligence-based metaheuristics. This argument comes from the nature of the swarm intelligence-based metaheuristic where all agent search for improvement in every iteration. Meanwhile, the complexity of MIO is higher than the basic swarm intelligence-based metaheuristic, such as PSO or MPA, where the interaction of every agent is limited. Meanwhile, the complexity of MIO is comparable to other swarm intelligence-based metaheuristics where each agent generates several candidates in every iteration. When the number of interactions is equal to the number of candidates, then the complexity of MIO is equal to this metaheuristic. On the other hand, the complexity of MIO is lower than metaheuristics that implement sorting mechanism in the beginning of every iteration, for example in GWO [32], dart game optimizer (DGO) [33], or MLO [28].

There are several limitations regarding this work and the proposed metaheuristic. The scenario in the order allocation problem used in this work is simple. Its simplicity comes from the single-product scenario. Meanwhile, the need is known in advance. The more complicated scenario can be implemented in future studies by implementing the multi-product scenario where there is not any vendor that can provide all products. The complicated scenario can also be conducted based on the objective. In this work, the objective in every optimization process is only one, whether it is minimizing the total cost, total lateness, or total defect. This single objective can be transformed into a multi-objective optimization problem. Moreover, the considered parameters can be increased, for example, the limited storage

capacity, manufacturing capacity, or perishable products. In this work, the use case is the order allocation problem. Meanwhile, there is broader use case in the optimization related to manufacturing systems, for example, the scheduling process during production, whether it can be flow-shop, job-shop, assembly line, and so on. The use case can be moved forward to the logistic and transportation problem, for example by implementing MIO to solve warehouse, fleet, and routing problems. This explanation means that future studies regarding this current study are still available and challenging.

6. Conclusion

This study has presented the excellent performance of the proposed metaheuristic, the multiple interaction optimizer (MIO). Its excellence comes from the two-phase strategy adopted by MIO. The multiple interaction strategy conducted in the first phase contributes to achieving the highest improvement. Meanwhile, the local search carried out in the second phase contributes to overcoming the local optimal problem. Moreover, the strict acceptance-rejection strategy contributes to avoiding the optimization process from the worsening circumstance. The test result indicates that MIO can find an acceptable solution in solving 23 functions and order allocation problems. MIO outperforms PSO, MPA, GWO, SMA, and GSO in 22, 21, 22, 19, and 18 functions. Moreover, MIO outperforms these five metaheuristics by creating minimum total cost, minimum lateness, and minimum total defect in solving the order allocation problem. The result also indicates that achieving significant improvement in solving the integer-based optimization problem as in the order allocation problem is more difficult than in the floating point-based optimization problem as in the 23 functions.

This study can become the baseline for future studies in two directions. The first direction is the improvement of the proposed metaheuristic. Future studies regarding the modification or hybridization of MIO are necessary to improve its performance. The modification of MIO can also be carried out by transforming the original form of MIO into a new form suitable to solve a combinatorial optimization problem. Besides, implementing MIO in many optimization problems in the manufacturing system is also important. The second direction is related to the order allocation problem, for example in the multiple product or multiple objective order allocation problems.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

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

Acknowledgments

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

References

[1] A. A. Reyes, E. Cuevas, A. Rodriguez, A. Mendoza, and E. O. Benitez, “An Improved Grey Wolf Optimizer for a Supplier Selection and Order Quantity Allocation Problem”, *Mathematics*, Vol. 8, p. 1457, 2020.

[2] I. Ahmad, Y. Liu, D. Javeed, and S. Ahmad, “A Decision-Making Technique for Solving Order Allocation Problem Using a Genetic Algorithm”, *IOP Conference Series: Materials Science and Engineering*, Vol. 853, p. 012054, 2020.

[3] E. Demiralay and T. Paksoy, “Strategy Development for Supplier Selection Process with Smart and Sustainable Criteria in Fuzzy Environment”, *Cleaner Logistics and Supply Chain*, Vol. 5, p. 100076, 2022,

[4] Z. Zhang, C. Guo, W. Ruan, W. Wang, and P. Zhou, “An Intelligent Stochastic Optimization Approach for Stochastic Order Allocation Problems with High-Dimensional Order Uncertainties”, *Computers & Industrial Engineering*, Vol. 167, p. 108008, 2022.

[5] Y. Sun, S. C. Guo, and X. Li, “An Order-Splitting Model for Supplier Selection and Order Allocation in a Multi-Echelon Supply Chain”, *Computers & Operations Research*, Vol. 137, p. 105515, 2022.

[6] A. Mohammed, I. Harris, A. Soroka, M. Naim, T. Ramjaun, and M. Yazdani, “Gresilient Supplier Assessment and Order Allocation Planning”, *Annals of Operations Research*, Vol. 296, No. 1, pp. 335-362, 2020.

[7] V. G. Venkatesh, A. Zhang, E. Deakins, S. Luthra, and S. Mangla, “A Fuzzy AHP-TOPSIS Approach to Supply Partner Selection in Continuous Aid Humanitarian Supply Chains”,

Annals of Operations Research, Vol. 283, pp. 1517-1550, 2019.

[8] T. T. Tham, N. T. T. Duc, T. T. M. Dung, and T. H. P. Nguyen, "An Integrated Approach of ISM and Fuzzy TOPSIS for Supplier Selection", *International Journal of Procurement Management*, Vol. 13, No. 5, pp. 701-735, 2020.

[9] G. Pishchulov, A. Trautrims, T. Chesney, S. Gold, and L. Schwab, "The Voting Analytic Hierarchy Process Revisited: A Revised Method with Application to Sustainable Supplier Selection", *International Journal of Production Economics*, Vol. 211, pp. 166-179, 2019.

[10] N. Janatyan, M. Zandieh, A. A. Tabriz, and M. Rabieh, "A Rapid Method for Sustainable Supplier Selection in Pharmaceutical Distribution Companies Under Uncertainty Circumstance", *International Journal of Procurement Management*, Vol. 12, No. 5, pp. 572-591, 2019.

[11] M. T. Ahmad and S. Mondal, "Dynamic Supplier Selection Approach for Mining Equipment Company", *Journal of Modelling in Management*, Vol. 14, No. 1, pp. 77-105, 2019.

[12] T. Niranjan, B. Singaravel, and S. S. Raju, "Integrated Fuzzy Criteria Evaluation with Metaheuristic Optimization for Green Supplier and Order Allocation", *IOP Conference Series: Materials Science and Engineering*, Vol. 1057, p. 012074, 2021.

[13] J. Swan, S. Adriaensen, A. E. I. Brownlee, K. Hammond, C. G. Johnson, A. Kheiri, F. Krawiec, J. J. Merelo, L. L. Minku, E. Ozcan, G. L. Pappa, P. G. Sanchez, K. Sorensen, S. Vob, M. Wagner, and D. R. White, "Metaheuristics in the Large", *European Journal of Operational Research*, Vol. 297, pp. 393-406, 2022, doi: 10.1016/J.Ejor.2021.05.042.

[14] H. Aydilek, A. Aydilek, M. Allahverdi, and A. Allahverdi, "More Effective Heuristics for a Two-Machine No-Wait Flowshop to Minimize Maximum Lateness", *International Journal of Industrial Engineering and Computations*, Vol. 13, No. 4, pp. 543-556, 2022.

[15] Y. Chen, Z. Guan, C. Wang, F. D. Chou, and L. Yue, "Bi-Objective Optimization of Identical Parallel Machine Scheduling with Flexible Maintenance and Job Release Times", *International Journal of Industrial Engineering and Computations*, Vol. 13, No. 4, pp. 457-472, 2022.

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

[17] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime Mould Algorithm: A New Method for Stochastic Optimization", *Future Generation Computer Systems*, Vol. 111, pp. 300-323, 2020.

[18] S. Suyanto, A. A. Ariyanto, And A. F. Ariyanto, "Komodo Mlipir Algorithm", *Applied Soft Computing*, Vol. 114, p. 108043, 2022.

[19] A. M. F. Fard, M. H. Keshteli, and R. T. Moghaddam, "Red Deer Algorithm (RDA): A New Nature-Inspired Meta-Heuristic", *Soft Computing*, Vol. 24, No. 19, pp. 14637–14665, 2020.

[20] 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.

[21] P. Trojovský and M. Dehghani, "Pelican Optimization Algorithm: A Novel Nature-Inspired Algorithm for Engineering Applications", *Sensors*, Vol. 22, p. 855, 2022.

[22] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "Northern Goshawk Optimization: A New Swarm-Based Algorithm for Solving Optimization Problems", *IEEE Access*, Vol. 9, pp. 162059–162080, 2021.

[23] S. Arora and S. Singh, "Butterfly Optimization Algorithm: A Novel Approach for Global Optimization", *Soft Computing*, Vol. 23, pp. 715-734, 2019.

[24] M. Dehghani, E. Trojovská, and P. Trojovský, "A New Human-Based Metaheuristic Algorithm for Solving Optimization Problems on the Base of Simulation of Driving Training Process", *Scientific Reports*, Vol. 12, p. 9924, 2022.

[25] P. D. Kusuma and D. Adiputra, "Modified Social Forces Algorithm: From Pedestrian Dynamic to Metaheuristic Optimization", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 3, pp. 294–303, 2022, doi: 10.22266/ijies2022.0630.25.

[26] E. Trojovska and M. Dehghani, "A New Human-based Metaheuristic Optimization Method based on Mimicking Cooking Training", *Scientific Reports*, Vol. 12, p. 14861, 2022.

[27] M. Dehghani and P. Trojovský, "Hybrid Leader Based Optimization: A New Stochastic Optimization Algorithm for Solving Optimization Applications", *Scientific Reports*, Vol. 12, p. 5549, 2022.

[28] 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.

[29] F. Zeidabadi, S. Doumari, M. Dehghani, and O. P. M., 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.

[30] 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, p. e910, 2022.

[31] 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.

[32] H. Faris, I. Aljarah, M. A. A. Betar, and S. Mirjalili, "Grey Wolf Optimizer: A Review of Recent Variants and Applications", *Neural Computing and Applications*, Vol. 30, pp. 413-435, 2018.

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

[34] A. G. Gad, "Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review", *Archives of Computational Methods in Engineering*, Vol. 29, pp. 2531-2561, 2022.