IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 26, 2020, accepted May 1, 2020, date of publication May 11, 2020, date of current version June 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2993491

Bird Mating Optimizer for Combinatorial Optimization Problems

ANAS ARRAM[1], MASRI AYOB[1], GRAHAM KENDALL[2], AND ALAA SULAIMAN[1]
[1]Data Mining and Optimization Research Group (DMO), Centre for Artificial Intelligence (CAIT), Universiti Kebangsaan Malaysia, Bangi 43600, Malaysia
[2]ASAP Research Group, School of Computer Science, University of Nottingham, Nottingham NG8 1BB, U.K.

Corresponding author: Anas Arram (anas.arram@gmail.com)

This work was supported in part by the Ministry of Higher Education Malaysia under Grant FRGS/1/2018/ICT02/UKM/01/1 and in part by the Universiti Kebangsaan Malaysia under Grant DIP-2019-013.

ABSTRACT The bird mating optimizer is a new metaheuristic algorithm that was originally proposed to solve continuous optimization problems with a very promising performance. However, the algorithm has not yet been applied for solving combinatorial optimization problems. Thus, the formulation may not be able to generate a discrete feasible solution. Many continuous algorithms used random-key representation to represent the discrete solution using real numbers or a discrete variant of the algorithm is used to deal with the discrete solution of the problem. However, there is no evidence which methodology is better for solving combinatorial optimization problems. Therefore, this work proposes two variants of bird mating optimizer (random-key bird mating optimizer and the discrete bird mating optimizer), to identify which one is more efficient in solving combinatorial optimization problems. In the first one, we use a random-key encoding scheme, whilst, in the later one, we use crossover (multi-parent) and mutation operators to combine the components of the selected parents to generate new broods. The performance of these algorithms is tested on the travelling salesman problem and berth allocation problem, and are compared with the results of two well-known optimization algorithms: Genetic Algorithm and Particle Swarm Optimization. Experimental results show that the discrete bird mating optimizer is more efficient than the others on all tested benchmark instances. Indeed, it is able to attain the best-known results in some of the BAP benchmark instances. These indicate the applicability and the effectiveness of the proposed discrete bird mating optimizer in solving combinatorial optimization problems.

INDEX TERMS Heuristics, bird mating optimizer, berth allocation problem, travelling salesman problem, random-key, combinatorial optimization, metaheuristics.

I. INTRODUCTION

Combinatorial optimization problems arise in various aspects of computer science and other areas such as artificial intelligence, operational research and electronic commerce. The combinatorial optimization problem can be defined as the need to find an optimal arrangement, grouping, or ordering for a given set of discrete variables [1], [2]. Examples of such problems include scheduling public transportation, travelling salesman problems, university educational timetabling, and berth allocation problems [3]. Several optimization techniques have been proposed for solving combinatorial optimization problems efficiently [1], [4]–[7], [74], [75]. These techniques can be classified as either exact methods or approximate methods. Exact methods such as dynamic programming [8], [9], branch and cut [10] and branch and price [11] are guaranteed to find an optimal solution in a finite time and systematically search the solution space [12]. However, due to the complexity of many combinatorial problems and many are known to be non-deterministic polynomial-time (NP)-hard problems, the time needed to solve them grows exponentially as the problem size grows linearly [13]. Therefore, many researchers have focused on approximate methods to address combinatorial optimization problems [13]–[15].

The use of approximate algorithms does not guarantee that the optimal solution will be found, but empirically they have often been shown to find a nearly optimal solution

The associate editor coordinating the review of this manuscript and approving it for publication was Fuhui Zhou.

within a reasonable amount of time [13]. Approximate algorithms are classified into two families: heuristics and metaheuristics. Heuristics, such as construction algorithms, are problem dependent, that is, they are designed to be applicable to a particular problem domain [16], [17]. Heuristic algorithms can be costly to implement and may not be suitable for a large variety of optimization problems [2], [17]. In contrast, metaheuristics are more general methodologies that can work effectively across different optimization problems. A metaheuristic algorithm is a high-level heuristic methodology that provides a high-level control strategy in order to improve exploration of the search space [18]. Examples of these algorithms include simulated annealing [19], [20], great deluge [21], tabu search [22], genetic algorithm (GA) [23]–[26], harmony search [27], scatter search [28], ant colony optimization [29], [30] and particle swarm optimization [31], [32].

Recently, a nature-inspired metaheuristic population-based algorithm, called the bird mating optimizer (BMO) has been proposed [14]. The BMO is inspired by the mating strategies of bird species during mating season in which male birds tend to mate with female birds in order to breed a new brood. The strength of the BMO lies in its ability to provide a good balance between exploration and exploitation [14]. The BMO overcomes the drawback of losing population diversity and getting trapped in local optima that is inherent in other methodologies by employing five updating strategies to move through the search space [14]. Furthermore, the BMO is able to exploit the best regions by utilizing a local search at each iteration. Therefore, BMO has more capability than the GA to effectively explore and exploit the search space and find the global solution. The algorithm has demonstrated effective performance across range of optimization problems [33]–[40]. In addition, it showed a competitive result comparing to other evolutionary algorithm such as classical evolutionary programming, fast evolutionary programming, classical evolutionary strategies, fast evolutionary strategies, genetic algorithm, particle swarm optimization, and group search optimizer [14].

However, BMO was originally proposed to address continuous optimization problems and, therefore, the application of BMO to combinatorial optimization problems has not yet been investigated. The original BMO uses mathematical formulations that combine the information of the selected solutions to generate a new one. Thus, standard BMO equations may not be able to generate a discrete feasible solution because the positions are discrete values in combinatorial optimization problems. Many continuous algorithms in the literature that have been applied to solve combinatorial problems use one of the two following methods. The first method is to apply a continuous algorithm using random-key representation to represent the discrete solution using real numbers, which can deal with the continuous algorithms [41]–[44]. The second method is to construct a discrete variant of the algorithm to deal with the discrete nature of the problem [15], [43], [45], [46]. However, there is no investigation in the literature on which method is better for solving combinatorial problems. The question is how can BMO be applied to solve combinatorial optimization problems and which method is better? To answer these questions, this work proposes two variants of BMO to address combinatorial optimization problems: the random-key bird mating optimizer (RKBMO) and discrete bird mating optimizer (DBMO). We also investigate the effectiveness of each method to identify the most promising one. Two well-known combinatorial optimization problems are chosen to test the proposed methods: The Travelling Salesman problem (TSP) (Reinelt 1991 instances) [47] and the Berth Allocation Problem (BAP) (Cordeau *et al.* 2005 instances) [48] Note this research study is different from [49], in which only RKBMO was applied and only a set of BAP instances were used and they are different from those the set of BAP used in this work. In addition, no analysis study was provided. Here we utilise two problem domains (TSP and BAP) and conduct deeper analysis.

The remainder of the paper is organized as follows: the basic BMO is presented in Section II. The problem descriptions are presented in Section III. The proposed RKBMO is presented in Section IV, followed by the proposed DBMO in Section V. In Section VI, the application of the proposed DBMO on the BAP and TSP is presented. Section VII summarizes and discusses the computational results. Finally, a conclusion and details of future work are presented in Section VIII.

II. THE BIRD MATING OPTIMIZER

The BMO is a population-based stochastic search technique that was proposed by [14] to address continuous optimization problems. The behavior of this algorithm is based on simulating the mating strategies of bird species during mating season. In this algorithm, the mating process of birds involves the use of three main operators to produce a new generation: two-parent mating, multi-parent mating, and mutation. Two-parent mating is where the two parents mate with each other to breed only one new brood, whereas multi-parent mating is when a parent mates with at least two other parents to breed one new brood. Mutation, on the other hand, is a mechanism where the female parent produces a new brood without the help of a male by modifying its own genes [14].

In the BMO, the population is represented by the bird society, and each bird represents a feasible solution to the problem at hand. There are five mating types that can generate a new population:

1) Monogamy: most birds are monogamous, where a male mate with one female only.
2) Polygyny: the male tries to mate with several females.
3) Polyandry: the female tries to mate with several males.
4) Parthenogenesis: the female can produce a new brood without the help of a male.
5) Promiscuity: two birds' mate for one time only with no stable relationship (one-time visit).

Thus, a bird society is divided into males and females. The females are those birds that have the most promising genes in the society. The females are divided into two groups: parthenogenetic and polyandrous, while the males are divided into three groups: monogamous, polygynous and promiscuous [14].

Generally, in the original BMO, the assumption that each bird in the society can change its own strategy to another when a new generation is updated [14]. At each generation, the birds that have the highest fitness values are designated as parthenogenetic and polyandrous, while those that have worse fitness values are designated as promiscuous. The remaining birds in the society are considered as monogamous or polygynous birds. A more detailed explanation of each group is given below.

Parthenogenesis is the mating type in which the female bird can produce a brood without mating with a male. In this method, each female tries to produce her brood by modifying and changing her genes with a predefined rate. Each female bird in the parthenogenetic group produces a brood as presented in equation (1) [14]:

$$\begin{aligned}
&for\ i = 1 : n \\
&\quad if\ r_1 > mcf_p \\
&\quad\quad x_{brood}(i) = x(i) + \mu \times (r_2 - r_3) \times x(i); \\
&\quad else \\
&\quad\quad x_{brood}(i) = x(i) \\
&\quad end \\
&end
\end{aligned} \tag{1}$$

where $x\ (i)$ denotes the i^{th} bird, x_{brood} is the resultant brood, n is the problem dimension (number of genes), mcf_p is the mutation control factor of parthenogenesis, r_1 denotes a random number between 0 and 1, and μ is the step size. Monogamy is a two-parent mating type in which a male tends to mate with only one female. Every male evaluates the quality of the females by using a probabilistic approach to select one of them to be its mate. Female birds with good genes have a higher probability of being selected. Equation 2 illustrates the process of producing a new brood from two selected parents [13]:

$$\begin{aligned}
&\vec{x}_i = \vec{x} + w \times \vec{r}. \times (\vec{x}^i - \vec{x}) \\
&c = a\ random\ integer\ number\ between\ 1\ and\ n \\
&if\ r_1 > mcf \\
&\quad x_{brood}(c) = l(c) - r_2 \times (l(c) - u(c)); \\
&end
\end{aligned} \tag{2}$$

where w is a time-varying weight to adjust the selected female, $\vec{r}$ is a $1 \times d$ vector in which every element is a distributed random number between 0 and 1 and this random vector influences the corresponding element of $(\vec{x}^i - \vec{x})$, n is the problem dimension, mfc denotes the mutation control factor which is distributed between 0 and 1, and u and l are the upper bounds and lower bounds of the elements, respectively.

From the first part of Equation (2), each male bird seeks to produce a good-quality brood by finding a worthy female with which to mix his genes in order to produce new genes. Then, the male bird tries to improve his produced brood by mutating one of the brood's genes with the probability of $1 - mcf$.

Polygamy is a multi-parent mating process in which a male bird tries to produce a brood by mating with two or more females. The benefit of this multi-mating process is to produce a brood with better genes. In nature, a polygynous bird mating with several females produces a number of broods, but in the BMO, only one brood results from this mating process, where the brood's genes are a combination of the male's and multiple females' genes. After selecting the females by using a selection mechanism, each male bird mates with his selected females. The resultant brood is produced by the following process [13]:

$$\begin{aligned}
&\vec{x}_i = \vec{x} + w \times \sum\nolimits_{j=1}^{n_1} \vec{r}_{j.} \times (\vec{x}_j^i - \vec{x}) \\
&C = a\ random\ number\ between\ 1\ and\ n \\
&if\ r_1 > mcf \\
&\quad x_{brood}(c) = l(c) - r_2 \times (l(c) - u(c)); \\
&end
\end{aligned} \tag{3}$$

where n_i is the number of selected female birds and x_j^i denotes the jth selected bird.

Promiscuity is a two-parent mating approach between two birds where they only mate once. This type of mating does not lead to a long relationship between the birds. This mating process indicates a chaotic social structure in which the male bird will never see the brood or the nest, and generally will not see the female for further mating activity. In promiscuity, the birds use a chaotic sequence method during the generations. However, each promiscuous bird behaves in the same way as a monogamous bird. In other words, in the BMO, promiscuous birds produce a new brood according to Equation (2).

Polyandry is a multi-parent mating system where a female bird seeks to mate with more than two males. The female performs a selection mechanism to select the males. Then, each female bird mates with her selected male birds. In the BMO, polyandrous birds produce a new brood according to Equation (3).

The procedure of the BMO is as follows [14]:

Step 1 (Parameter Initialization): initialize the following BMO parameters:

1) Society size (SS): this refers to the number of birds in the society or the number of solutions in the population of the BMO;
2) The percentage of each group in the society: that is, the percentage of monogamous, polygynous, promiscuous, polyandrous, and parthenogenetic birds;
3) The number of mates (nm) for the polygynous and polyandrous birds: this refers to the number of mates that will participate in the multi-parent mating system practiced by the polygynous and polyandrous birds;

4) Mutation control factor (*mcf*): this is the probability of mutating the generated brood after each mating process;
5) Maximum number of generations (*NG*): this refers to the termination condition of the BMO or the number of iterations.

Step 2 (Society Initialization): randomly initialize a set of feasible solutions and add them to the society. Each solution is considered as a bird and is specified by a vector, with the length of n.

Step 3 (Society Evaluation): calculate the quality of each bird using an evaluation function.

Step 4 (Ranking): rank the birds in the society based on their quality.

Step 5 (Classification): divide the society into five groups of birds: parthenogenetic, polyandrous, monogamous, polygynous and promiscuous. The parthenogenetic and polyandrous groups are considered as female and the remaining groups are considered as male. Birds with top *n1* qualities are designated as parthenogenetic and the next *n2*, *n3*, *n4* and *n5* are designated as polyandrous, monogamous, polygynous and promiscuous, respectively. The percentage of each group is defined in the parameter initialization step, as recommended by [14].

Step 6 (Breeding): each bird produces a new brood using its own pattern.

Step 7 (Replacement): if the quality of the brood is better than the quality of the birds in the society, then the brood replaces the bird, otherwise, the bird remains in the society and the brood is abandoned.

Step 8 (Termination Condition): repeat steps 4 to 7 until a predetermined number of generations (*NG*) is performed.

Step 9 (Report the Best): select the best-quality bird in the society as the best solution.

III. PROBLEM DESCRIPTION

In this study, the performance of the proposed DBMO is evaluated over two benchmark problems: The Travelling Salesman Problem (TSP) and Berth Allocation Problem (BAP). The following subsections describe these problems.

A. TRAVELLING SALESMAN PROBLEM

The TSP is a well-known combinatorial optimization problem that classified as a NP-hard problem, which means that it may take an infeasible computational time to solve it [50]–[52]. The TSP can be described as a search for the shortest path that visits each city once and only once, and finally return to the start city [53].

B. BERTH ALLOCATION PROBLEM

The BAP is a highly constrained combinatorial optimization problem which is hard to solve to optimality [48], [54]. The BAP can be classified into two types based on berth type and vessel arrival time. The berth type is considered as discrete if the quay is divided into a set of sections (berths) or continuous if not partitioned. The vessel arrival time is considered as dynamic if vessels can arrive at any time during the container operations with planning arrival time or static if all the vessels have to arrive in the harbor before the berth planning step begins. In this study, we focus on the discrete and dynamic version of the BAP [43]. This BAP deals with allocating vessels to berths in the harbor at the planned arrival time. More formally, the goal is to assign a berth for each vessel and a service time at the selected berth. The following assumptions are considered in the BAP [46]:

1) Each berth can serve only one vessel at a time;
2) Any vessel can be assigned to any berth with a given handling time taking into account that the handling time of a vessel can differ from one berth to another;
3) All vessels arrive at their berths before or after the berths' opening hours with a known arrival time;
4) When a vessel is moored in a berth, it remains there until all servicing activities have been completed.

The objective of the BAP is to minimize the overall waiting time of all the vessels that need to be serviced in the harbor, which is calculated as an objective function as follows [48]:

$$\sum_{i\in N}\sum_{k\in K}(T_i^k - a_i + P_i^k \sum_{j\in N\cup\{d\}} x_{jj}^k) \tag{4}$$

where:

- a_i: *arrival time of vessel i*
- K: *set of berths,* $K = \{1, 2, ..., K$
- N: *number of vessels that will arrive in the harbour,*
- P_i^k: *handling time of vessel i at berth k*
- T_i^k: *berthing time of vessel i at berth k*
- x_{ij}^k: *decision variable,* $x_{ij}^k = 1$ *if vessel j is serviced at berth k immediately after vessel i*

IV. THE PROPOSED RANDOM-KEY BIRD MATING OPTIMIZER

Random-key representation is a common technique that transforms a position in a continuous space and converts it onto a combinatorial space representation [41]. It uses a vector of real numbers to represent a solution in which each number is randomly generated in uniform [0, 1]. The combinatorial vector is composed of integers ordered according to the sequence of the real numbers in the first vector. This scheme is proposed by [41] and used in the literature to address different combinatorial optimization problems [44], [43], [55]. An example of this representation is shown in Fig. 1.

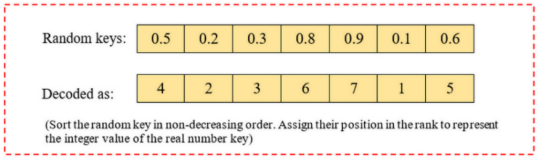

FIGURE 1. Random-key encoding scheme.

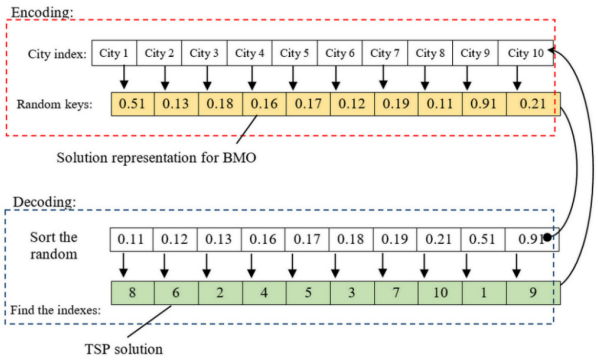

FIGURE 2. Random-key representation steps for TSP.

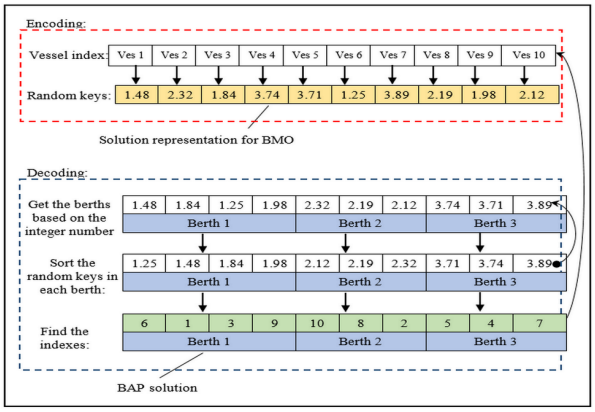

FIGURE 3. Random-key representation steps for BAP.

A. RKBMO FOR TSP

Fig. 2 illustrates the steps of encoding and decoding process of generating a TSP solution using random keys. To encode the solution for BMO, we generate m random numbers from [0, 1]m for each bird in the society, where m is the size of the TSP instance. Each gene in the bird has an integer index (see Fig. 4). In the decoding process, we follow the steps as presented in [44], [41]:

1) Sort the random keys in ascending order.
2) Set the indexes of the sorted numbers as the city index array, i.e. the integer indexes correspond to the city index. The sequence of the generated real numbers represents the order of the genes in a bird. Thus, a bird contains a set of genes (i.e. city index).

During the search process, some numbers might be outside the feasible solution space. We handle this limitation by doing the following: where the solution is infeasible, if the generated number is less than 0 or bigger than 1, a new random number between 0 and 1 is generated and replaces with the infeasible one [44].

B. RKBMO FOR BAP

For BAP solution representation, the number of vessels is considered as the number of genes in the bird (solution), where every vessel represents a gene [43], [41]. In order to assign vessel to a berth, we generate, for each vessel, a random integer number in $\{1, 2, \ldots, m\}$ and add a uniform number from [0, 1]. In decoding, the integer part of any random key represents the berth assignment for that vessel, and the fraction parts represent a vessel sequence for each berth.

Fig. 3 shows an example of how the BAP is represented using a random-key scheme. The example consists of a BAP instance with 10 vessels and three berths. In the encoding step, a random integer number is generated from [1], [3] for each vessel, and a uniform (0, 1) deviate is added to the number. Again, the integer part of each number represents the berth assignment for that vessel, which means that the same integer part represents the set of vessels at the same berth, and the fractional part represents the vessel sequence for the berth. To decode the solution, we follow the steps as presented in [55] and [56]:

1) The berth number of each vessel is obtained based on the integer part of the random key (see Fig. 3).
2) The random keys are sorted for each berth in ascending order to find the vessel sequences.
3) The index of each vessel is obtained to represent the final solution of BAP.

Therefore, berth 1 will serve vessels {6, 1, 3, 9}, berth 2 will serve vessels {10, 8, 2} and berth 3 will serve vessels {5, 4, 7}. Sorting the fractional values in ascending order for berth 1 results in the vessel sequence {7, 9, 8}, and for berth 2 the vessel sequence {2, 6, 5}, and for berth 3 the vessel sequence {3, 1, 4}.

During the search process, some numbers might be outside the feasible solution space. We handle this limitation by doing the following: if the value of the gene is less than zero, we randomly generate a random number, *rnd* between (0, 1) and add it to the first berth number (i.e. $1+rnd$). If the value of the gene is bigger than the number of berths, we randomly generate a random number in (0, 1) and subtract this number from the number of berths (i.e. 3 - *rnd*) [43].

V. THE PROPOSED DISCRETE BIRD MATING OPTIMIZER

In order to propose a discrete version of the BMO, it is necessary to understand the main components of BMO and how the algorithm works. BMO has three main operators which it can use to produce a new solution. These are [14]: self-recombination, two-parent recombination and multi-parent recombination. These operators are implemented using the five mating strategies: monogamous, promiscuous, polyandrous, polygynous and parthenogenetic, as discussed in Section 2. In addition, the BMO applies a mutation operator with probability after the following strategies [14]: monogamous, promiscuous, polyandrous and polygynous. Therefore, the mutation operator is one of the main operators in the BMO that needs to be mapped onto a discrete space.

Before presenting the proposed discrete version of the BMO, the following subsections describe these three discrete operators in more detail.

A. SELF-RECOMBINATION IN DBMO

In the original BMO, the self-recombination operator generates a new solution by modifying the genes of a selected solution [14]. One of the most popular operators that can substitute the basic self-recombination is the basic version of the hill-climbing (HC) algorithm. The HC algorithm starts with an initial solution and improves it by iteratively generating a neighbor solution. The current solution is replaced by the neighbor solution if the quality of the neighbor solution is better than the current one. The HC algorithm stops when the termination criterion is met [13], [57]. Hence, the basic HC algorithm (see Algorithm 1) is similar to parthenogenetic improvement behavior, where the HC algorithm generates a new solution by making some changes to the genes of the current solution. However, parthenogenetic improvement requires a probability rate to decide either to perform the parthenogenetic improvement or to skip the procedure.

Algorithm 1 Pseudocode of the Modified HC Algorithm

Input: x as the starting solution, mcf_p.
Improve = true.
Generate random number r between 0 and 1
If r $< mcf_p$ **then**
  Repeat
    Generate ($N(x)$). *//generate a candidate neighbor//*
    if x' better than x **then** *//x' is the generated solution//*
      $x = x'$.
      improve = true.
    else
      improve = false.
    end
  Until improve = true.
Else $x = x$.
Output: final solution found x.

B. TWO-PARENT RECOMBINATION IN DBMO

In the original BMO, the two-parent recombination operator generates a new solution by mixing and recombining the genes of two parents [14]. This reproduction strategy is similar to the standard crossover operator where two parents produce a new child by combining their genes. In other population-based approaches, several crossover methods have been developed for solving combinatorial problems. For example, partially matched crossover [58], cycle crossover [58] and order crossover (OX) [58], [59]. Experimental results conducted by [60] revealed that OX performs better than others. Therefore, OX is used as a two-parent recombination operator in the DBMO. More details about the steps of OX are reported in [60].

C. MULTI-PARENT RECOMBINATION IN DBMO

The multi-parent recombination operator in the original BMO generates a new solution by combining the genes of more than two parents [14]. In order to convert the multi-parent recombination operator from a continuous space onto a discrete space, we need a multi-parent recombination operator that can deal with combinatorial problems. Two methods have been proposed in the literature to tackle this issue: multi-parent partially mapped crossover (MPPMX) [61] and adjacency-based crossover [62]. The experimental results presented in [62] demonstrated that adjacency-based crossover is not efficient in producing a good-quality solution for more than two-parent crossovers [62]. Therefore, the MPPMX was chosen in this work to substitute the multi-parent recombination operator in the original BMO. The MPPMX was proposed in order to extend the basic two-parent partially mapped crossover (PMX) into a multi-parent crossover for better performance [61]. There are four main steps in MPPMX to construct a new offspring: substring selection, substring exchange, mapping list determination and offspring legalization [61]. The details of these steps are given in [61].

D. MUTATION OPERATOR IN DBMO

In the original BMO, the mutation operator is applied with probability after the following strategies: monogamous, promiscuous, polyandrous and polygynous [14] have been performed. The mutation operator is applied using an equation that can deal with continuous problems only (see Section 2). Therefore, an insertion operator is introduced in the DBMO to substitute the continuous mutation operator in the original BMO. In the insertion operator, an element is randomly selected to be removed and is added to another randomly selected position. Note that the same mutation probability (mcf) of the original BMO is applied in the DBMO.

E. THE DBMO FREMWORK

The following steps explain the DBMO procedure, which was adapted from the basic BMO procedures [14]:

Step 1 (Parameter Initialization): DBMO uses the same parameters as the basic BMO (see Section II).

Step 2 (Society Initialization): randomly initialize a set of feasible solutions and add them to the society. The initialization strategy is related to the problem domain.

Step 3 (Society Evaluation): calculate the quality of each bird using the objective function of the problem.

Step 4 (Ranking): rank the birds in the society based on their quality.

Step 5 (Classification): classify the birds in the society into five groups based on their quality into: parthenogenetic, polyandrous, monogamous, polygynous, or promiscuous birds. The classification process is the same as in the original BMO (see step 5 in Section II).

Step 6 (Breeding): each bird produces a new brood using its own pattern, see Fig. 4.

	Species	Improvement	Updating pattern
Birds society	parthenogenetic	Self-recombination	Modified HC
	polyandrous	One female + multi-male	MPPMX
	monogamous	One male + one female	OX
	polygynous	One male + multi-female	MPPMX
	promiscuous	One male + one female	OX

FIGURE 4. Overview of step 6 in DBM.

Step 7 (Replacement): If the quality of the brood is better than the quality of the bird, the brood replaces the bird. Otherwise, the bird remains in the society, and the brood is removed.

Step 8 (Termination Condition): repeat steps 4 to 7 repeated until a predetermined number of generations (NG) is performed.

Step 9 (Report the Best): select the bird with the best quality in the society as the best solution.

The pseudocode of the DBMO is illustrated in Algorithm 2.

VI. DBMO FOR TSP AND BAP

In this section, we present the solution representation for both BAP and TSP using the DBMO.

A. DBMO FOR THE TSP

In this study, the solution is represented using the simple popular representation for the TSP, i.e. a one-dimensional array with the length of N, where N is the number of cities as in [43,68]. Each gene in the solution represents a city number. Fig. 5 shows an example of the solution representation for the instance of 10 cities. A solution with length $M = 10$, where $\{1, 2, 3, \ldots 10\}$ represents the indexes of the cities, and $\{4, 1, 8, \ldots 7\}$ represents the visiting order for the cities in the tour path.

Solution length N = 10

City index:	1	2	3	4	5	6	7	8	9	10
City order:	4	1	8	2	6	9	3	10	5	7

FIGURE 5. Example of solution representation for TSP.

In this work, the TSP solution is generated in a random manner. First, we generate an array that contains the total number of cities. Next, we generate an empty array, randomly pick a number from the first array, and add it to the empty array. The selected number is then removed from the first array. This process is repeated until all the cities are added into the empty array.

B. DBMO FOR THE BAP

In the BAP, the solution is presented using a string of integer numbers as in [64]. The length of the solution is the number of vessels n plus the number of berths (m) minus one. That is, the integer numbers contain m segments separated by 'zeros'. Each segment represents a service sequence for a number

Algorithm 2 Pseudocode of the DBMO Algorithm

Determine the society size (*SS*), maximum number of generation (), number of mates (*nm*) and mutation control factor (*mcf*).
Generate *SS* feasible birds.
for t = 1 to gen_{max} do
  Rank the birds in ascending order based on their quality.
  Classify the society into five groups: parthenogenetic polyandrous, monogamous, polygynous and promiscuous. //the classification is based on the quality of each bird.
  for i = 1 to *SS* do
    case parthenogenetic:
      Produce new brood using modified HC (Algorithm 4.3)
    case polyandrous:
      Select nm birds from the male group.
      Produce new brood using MPPMX.
      If r < *mcf* then // r is a random number between 0 and 1//
        Mutate the new brood using Insertion operator.
      end
    case monogamous:
      Select one mate bird from female group.
      Produce new brood using OX.
      If r < *mcf* then
        Mutate the new brood using Insertion Operator
      end
    case polygynous:
      Select nm birds from female group.
      Produce new brood using MPPMX.
      If r < *mcf* then
        Mutate the new brood using Insertion operator.
      end
    case promiscuous:
      Select one mate bird from female group.
      Produce new brood using OX.
      If r < *mcf* then
        Mutate the new brood using Insertion operator.
      end
    end
  end
  Perform replacement stage: replace the new generated broods with their parents if they have better quality. //update the society for the next generation
end
Return the best bird

of particular vessels at the assigned berth. Fig. 6 shows an example of the solution representation of an instance of the BAP that contains 13 vessels and three berths. The solution has a length of 15 and contains two zeros to separate the berths. For example, vessels 4, 3, 1 and 8 will be serviced at berth 1 in that order.

Solution length N = 10

City index:	1	2	3	4	5	6	7	8	9	10
City order:	4	1	8	2	6	9	3	10	5	7

FIGURE 6. Example of solution representation for TSP.

The solution is generated in three steps (as in [32]). First, we create an array with the number of vessels n and zeros m-1. Next, an empty solution array is created and then the elements of the first array are randomly selected and moved to the solution array. Finally, the vessel sequence for each berth is sorted in ascending order based on vessel arrival time. The pseudocode of the DBMO initialization steps for the BAP solution is given in Algorithm 3.

Algorithm 3 Initialize the BAP Solution

Input: set of N vessels and M zeros; // M zeros refers to the number of berths
Let V = set of N vessels + (M-1) zeros;
S = [ ] // S is a BAP solution
While V is not empty do
    Select a vessel from V at random;
    Add the selected element into S;
    Remove the vessel from V;
End while
For each berth in S do
    Sort the sequence of the vessels in ascending order based on their arrival time
End
Output: TSP solution S.

VII. COMPUTATIONAL RESULT

In this section, we discuss the results of an evaluation of the effectiveness of the proposed BMO variants RKBMO and DBMO and identify which of the two variants performs best. We also compare the performance of the proposed algorithms with that of the basic GA and PSO [65], [66]. The GA is chosen for this comparison because the proposed RKBMO and DBMO are evolutionary based and the GA is the standard algorithm for evolutionary algorithms [67], [68], while PSO has been widely improved and used to address many combinatorial optimization problems [24], [43], [46], [69], [70]. The GA has been implemented using roulette wheel selection with order crossover [71]. All algorithms (RKBMO, DBMO, GA and PSO) are tested on two combinatorial optimization problems: TSP and BAP. A statistical analysis is also conducted to verify the obtained results. First, however, the parameter settings and the experimental design for both the RKBMO and DBMO are presented.

A. PARAMETER SETTINGS AND EXPERIMENTAL DESIGN

The proposed algorithms were implemented using Java NetBeans IDE version 8.1 on a personal computer (Intel Pentium (R) Core i5 CPU at 3.40 GHz with 4 GB RAM), running a Windows 10 operating system (64-bit). The parameter settings of both the original BMO and DBMO for TSP and BAP are presented in Table 1. To ensure a fair comparison of the tested algorithms, we used the same parameter settings for both the basic BMO and DBMO. The values of SS and NG were obtained based on preliminary experiments as follows: 200 and 4000 for the TSP and 100 and 2000 for the BAP, see Tables A1 and A2 in Appendix A. The remaining parameters were fixed in line with [14] (see Table 1). Likewise, for the GA and PSO, for a fair comparison, the maximum number of generations and the population size were set as the same as those for the RKBMO and DBMO.

TABLE 1. Parameter settings for basic BMO and DBMO.

parameter	Value for TSP	Value for BAP
Society size (SS)	200	100
Number of generations (NG)	4000	2000
Number of mates for polygynous and polyandrous birds (nm)	3	3
Mutation control factor (mcf)	0.1	0.1
The percentage of each group of birds in the society:		
Parthenogenetic	5%	5%
Polyandrous	5%	5%
Monogamous	50%	50%
Polygynous	30%	30%
promiscuous	10%	10%

All algorithms (GA, PSO, RKBMO and DBMO) were executed over 30 independent runs with different random seeds for all instances of the problem domains. The reason for executing the proposed algorithms on 30 runs was to obtain a good indication of algorithm consistency, and to enable a robust statistical analysis of algorithm performance [13]. In the subsequent tables, the results for each instance are presented as average (Avg.), average time (Avg Time), standard deviation (Std), and percentage deviation (Gap%) with respect to the quality of solution produced by the compared algorithms. The best results are highlighted in bold and the Gap is calculated as follows:

$$\frac{BCA - BKS}{BKS} \tag{5}$$

where BCA show the best obtained from the compared algorithms and BKS is stand for best known solution in the literature.

TABLE 2. Comparison results for GA, PSO, RKBMO and DBMO on TSP (with respect to solution quality (Gap and Avg), and average time when the best-quality solution is obtained).

Instances	GA				PSO				RKBMO				DBMO			
	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.
eil51	11.74	558.2	5.39	33.5	2.82	473.23	4.66	6	20.66	631	14.86	90.08	**1.17**	**438.8**	**1.98**	**4.5**
St70	34.52	1107.9	8.82	82.1	6.52	874.8	5.98	24.22	51.11	1864	12.027	486.1	**0.89**	**692.5**	**3.59**	**7.1**
eil76	73.42	1021.6	13.65	46.6	43.49	698.07	8.38	16.3	64.5	1319	25.49	291.82	**22.86**	**675.7**	**7.11**	**6**
Pr76	37.41	172918.9	10.8	12922.1	5.97	149111.1	8.25	4328.8	78.32	311000	14.142	73202.4	**0.21**	**110211.1**	**4.88**	**773.1**
kroA100	70.98	43664.5	13.96	3585.1	27.42	37737.33	11.94	1744.32	174.79	88866	34.41	22071.25	**1**	**21959.8**	**7.51**	**307.7**
kroB100	62.76	43084.6	13.78	3309.1	22.2	37672.37	13.44	1897.5	138.75	84003	35.2	21558.68	**1.58**	**22963**	**8.17**	**263**
kroC100	75.24	42826.3	14.13	3425	24.54	37790.73	10.98	1515.32	161.27	87059	35.55	24459.1	**0.16**	**21408.9**	**7.12**	**277.5**
kroD100	75.3	42213.6	14.09	3454.7	23.09	38235.7	11.29	1645.72	163.06	85501	34.37	19025.78	**1.8**	**22161.3**	**8.11**	**230.7**
kroE100	73.92	43417.1	14.13	2830	29.97	37408.43	11.3	1128.31	157.43	83983	35.01	24318.39	**1.38**	**22946.4**	**11.51**	**282.7**
rd100	70.78	14907.5	14.19	1000.1	24.38	14166.2	10.84	452.73	168.57	37712	35.16	5680.5	**1.24**	**8234.8**	**11.8**	**127.5**
eil101	13.2	781.7	10.71	40.7	19.78	950.23	10.46	33.96	148.65	2390	35.36	314.68	**5.09**	**675.7**	**7.11**	**6**
lin105	77.31	30991.1	14.59	2834.7	20.88	27967.33	12.54	1453.44	205.88	59400	37.32	16851.3	**0.23**	**14774.2**	**7.37**	**206.6**
ch130	83.26	12599.3	17.14	762.9	40.5	13382.77	16.34	623.99	390.34	36562	42.35	1798.89	**3**	**6449.6**	**12.26**	**79.6**
ch150	91.18	14492.9	**18.96**	878.2	51.85	15874	28.7	813.58	419.7	43924	41.61	1987.46	**4.14**	**7009.1**	19.08	**86.8**
kroA150	119.09	63260.9	**19.01**	3738.1	65.98	69097.37	32.78	3016.59	333.83	191367	46.89	27454.15	**3.49**	**28134.1**	19.37	**372.9**
kroB150	118.36	63742.4	**18.97**	3687.3	61.44	64758.47	34.32	3465.36	410.46	185189	47.55	24341.07	**2.42**	**27425.8**	21.08	**335.2**
D198	161.85	47280.1	**24.39**	4349	86.52	51826.73	82.38	2987.4	456.56	146965	50.529	14363.6	**2.33**	**16334.4**	33.03	**98.5**
kroA200	165.22	85228.5	**24.46**	4021	114.75	97383.2	99.24	4565.48	676.24	280521	48.37	10406.81	**4.39**	**31487.2**	39.66	**383.4**
kroB200	143.6	83389.5	**24.56**	5080.8	101.21	98317.43	101.39	5329.42	689.49	272432	55.99	12368.69	**4.96**	**31627**	40.03	**336.6**
lin318	243.86	159719.8	**41.37**	7866.4	271.26	222557.2	348.05	10015.11	1106.41	520575	79.48	4199.3	**6.51**	**45764.3**	98.09	**546.9**
Average	90.15			3197.37	52.22			2253.17	300.8			15263.5	3.44			236.61

B. RESULTS FOR THE TSP DATASET

The experimental results of the GA, PSO, RKBMO and DBMO on the TSP dataset are presented in Table 2. From this table, it can be seen that the DBMO outperformed the GA, PSO and RKBMO across all instances (20 out of 20 instances) in terms of the gap and the average of the obtained results. The average of all the percentage deviations of the GA, PSO, RKBMO and DBMO are 90.1, 52.22, 300.8 and 3.44, respectively, which means that the solutions obtained from the DBMO over 30 runs are relatively close to the best-known solutions. (The best-known solutions along with the TSP dataset can be found at https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/). Furthermore, the average of all the standard deviations of GA, PSO, RKBMO and DBMO are 3197.37, 2253.17, 15263.5 and 236.6, respectively, which indicates that the DBMO is more stable and consistent than the GA, PSO and RKBMO.

In addition, the computational time of the DBMO was much better than that of the GA, PSO and RKBMO in most instances (13, 20 and 20 out of 20 instances, respectively), but it was sometimes more expensive than the GA especially for large-size instances. The reason for the high computational time of the DBMO in some instances is that it uses multi-parent partially mapped crossover and local search, which improves the solution quality at the expense of increased computation time. However, the computational time of the DBMO increased in only seven out of 20 instances and the overall performance of DBMO was superior to that of the GA, PSO and RKBMO. Similar to the results for the reported for the BAP, these results indicate the positive impact of the improvement operators of the DBMO on performance.

In order to verify whether the DBMO is significantly different from GA, PSO and RKBMO, a multiple comparison test was performed using Friedman and Quade statistical tests at a significance level of 0.05 [72]. (Note that a non-parametric test was performed based on normality tests and indicated that the data was not normally distributed.) The Friedman test is performed first, and if significant differences are found (the P-value of Friedman or Iman-Davenport statistic is less than the critical level 0.05), the Friedman and Quade tests are conducted to calculate the average rank of the compared algorithms to estimate the best performing one (the lower the better).

Then, we perform post-hoc procedures to obtain the adjusted p-values for each comparison between the control algorithm (the first ranking one) and the rest. (more details are reported in [72]). The p-value computed through Friedman and Quade tests (0.0000 and 0.0000) and the Iman-Davenport (0.000) are less than the critical level 0.05, therefore, we performed post-hoc procedures to detect the significant difference between all tested methods using Holm and Hochberg statistical tests.

TABLE 3. The average rank of the DBMO with GA, PSO and RKBMO for TSP.

No.	Algorithm	Friedman	QUADE
1	DBMO	1	1
2	PSO	2.45	2.56
3	GA	2.55	2.43
4	RKBMO	4	4

Table 3 shows the average ranking (the lower the better) of Friedman and Quade statistical tests for the compared

methods on TSP datasets. DBMO obtained the first rank followed by PSO, GA and RKBMO, in the given order. Therefore, DBMO will be the controlling method in Holm and Hochberg statistical tests. The Holm and Hochberg statistical test of Friedman and Quade tests [72] is conducted to detect the significant differences for each comparison between DBMO, GA, PSO and RKBMO. Table 4 shows the adjusted p-values of Holm and Hochberg statistical tests where DBMO is the controlling method. The table demonstrate that DBMO outperforms all other methods with critical level of 0.05 (adjusted p-values < 0.05) for all instances.

TABLE 4. The average rank of the DBMO with GA, PSO and RKBMO for TSP.

No.	Algorithm	Unadjusted P	P Holm
1	GA	0.0001	0.0002
2	PSO	0.0003	0.0003
4	RKBMO	0.0000	0.0000
No.	Algorithm	Unadjusted P	P Holm

The distribution of the results obtained from the GA, PSO, RKBMO and DBMO for four TSP instances (eil51, ch130, ch150 and kroA200) after running the program 30 times is presented in Figure A1 (see Appendix A). The graphs present the maximum and minimum values. The boxes represent the centre 50 percent of the data. The lower quartile of the boxes shows the lower median of the data while upper quartile presents the median of the upper part of the data. The line across the boxes relates to the median of the data. This is to show the distribution of the solutions obtained from the GA, PSO, RKBMO and DBMO. The figure shows that the distribution of the solutions is symmetric and distributed around the median in most instances. Based on these graphs, it can be seen that, for most instances, the DBMO is more consistent than the GA, PSO and RKBMO.

C. RESULTS FOR THE BAP DATASET

The results reported in Table 5 indicate that the DBMO, across all instances, outperforms the GA, PSO and RKBMO not only in terms of the best solution quality, but also with respect to the average quality and the average of all standard deviations of GA, PSO, RKBMO and DBMO is equal to 54.77, 36.20, 222.17 and 7.42. In addition, the DBMO finds the best-known solution (Gap% $= 0$) in seven instances (i03, i04, i05, i14, i15, i17 and i18) while the RKBMO and GA didn't find any best-known solution. (best-known solution here is the optimal value in the literature and can be found at [73]). This result demonstrates the effectiveness of the improvement process of the DBMO, which uses different discrete operators (two-parent, multi-parent and local search) to modify and improve the solution of the problem at hand. The RKBMO, on the other hand, uses mathematical equations to search the space and improve the solutions. The basic GA only uses two-parent crossover and mutation operators. The GA took less computational time than the DBMO and, basically because the use of multi-parent crossover in DBMO algorithm consumes extra computational time.

We performed a multiple comparison test between GA, PSO, RKBMO, and DBMO using Friedman and Quade statistical tests. Table 6 shows the average ranking (the lower the better) computed by Friedman and Quade tests for all compared methods. The table highlighted that DBMO obtained the first rank out of four compared methods followed by PSO, GA and RKBMO, respectively. The p-value computed through Friedman and Quade tests are less than 0.05 (0.0000 and 0.0000) and the Iman-Davenport (<0.05) prove that there is a significant difference among the methods. Therefore, post-hoc procedures (Holm and Hochberg) are performed to detect the significant difference between all tested methods. The adjusted p-values of Holm and Hochberg statistical tests presented in Table 7 (where DBMO is the controlling method) demonstrate that DBMO outperforms all other methods with critical level of 0.05 (adjusted p-values < 0.05).

The box-whisker plot in Fig. A2 (see Appendix A) shows the distribution of the results for 30 runs of the DBMO, RKBMO and GA for four instances of the BAP (i01, i02, i03 and i04). It can be seen from these graphs that the distribution of the results for most of the instances in the case of the DBMO is skewed to the lower end (i.e. i01, i03 i04). However, there are some cases where the solution distribution is symmetric; the solutions are distributed evenly around the median (i.e., i02). While the distribution of the solutions obtained by the GA, PSO and RKBMO in most instances is symmetric, the distribution is split at the median. In addition, in the case of the RKBMO one solution distribution is skewed to the upper end (worst solution) (i01). These distributions demonstrate that the DBMO is more consistent than the GA, PSO and RKBMO.

To conclude, it is clear that in both problem domains (TSP and BAP), the DBMO outperformed the GA, PSO and RKBMO and matched the best-known results in some instances. In addition, for both domains the standard deviation was relatively small. Also, the percentage deviation (Gap) showed that the DBMO results were very close to the best-known solutions in both domains. This favorable outcome reveals the benefit of using the discrete version of the BMO rather than the original one with random-key representation. The DBMO was able to achieve a significant difference in performance compared to the RKBMO for all instances of both the TSP and BAP. This is due to the improvement procedures in the DBMO: two parent recombination using OX and multi-parent recombination using MPPMX which gives a possibility to combine the components of many solutions and generate new one. Another feature of DBMO is the use of Hill Climbing heuristic for self-recombination which improves the ability

TABLE 5. Comparison results for 30 runs of GA, PSO, RKBMO and DBMO on BAP (with respect to solution quality (Gap and Avg), and average time when the best-quality solution is obtained).

Instances	GA				PSO				RKBMO				DBMO			
	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.	Gap%	Average	Avg. Time(s)	Std.
i01	16.24	1637.8	**1.774**	63	5.82	1552.77	8.274	31.35	7.95	1893	19.47	354.2	**0.35**	**1428.1**	3.625	**9.39**
i02	12.72	1421.4	**1.786**	57.2	5	1365.33	8.105	33.82	5.79	1491	18.88	197.91	**0.08**	**1278.8**	2.568	**8.88**
i03	9.43	1235.5	**1.735**	42.3	3.37	1211.63	8.923	35.46	3.81	1305	18.98	131.95	**0**	**1139**	2.432	**4.87**
i04	9.76	1429.1	**1.637**	36.9	2.92	1395.2	4.993	38.66	4.15	1572	18.99	273.71	**0**	**1312.2**	2.367	**7.35**
i05	9.82	1325.5	**1.675**	49.6	2.32	1273.47	6.687	24.32	2.49	1424	19.17	189.29	**0**	**1220.1**	2.463	**6.56**
i06	12.24	1415.4	**1.62**	43.1	3.89	1353.47	9.499	26.2	5.79	1492	18.57	189.23	**0.87**	**1281.6**	2.505	**4.75**
i07	9.98	1406.7	**1.616**	45.7	1.8	1356.67	9.304	28.68	7.04	1455	18.23	71.98	**0.08**	**1287.7**	1.937	**7.61**
i08	13.85	1478.9	**1.587**	55	3.7	1415.73	8.451	41.29	6.39	1533	18.49	134.01	**0.23**	**1313.6**	2.398	**5.89**
i09	12.21	1620.3	**1.638**	58.6	4.5	1554.73	8.444	32.34	4.71	1735	18.45	266.08	**0.14**	**1460.5**	2.263	**7.01**
i10	14.08	1383.8	**1.586**	66.6	2.39	1314.53	8.408	33.27	5.03	1531	18.84	283.86	**0.08**	**1221.6**	2.984	**5.16**
i11	13.59	1553.9	**1.574**	55.2	3.95	1481.37	20.44	31.69	9.21	1714	18.68	279.79	**0.37**	**1382.1**	2.031	**6.64**
i12	12.69	1493.1	**1.569**	54.6	4	1449.23	3.986	36.38	8.23	1777	18.89	328.52	**0.68**	**1348.5**	2.988	**8.33**
i13	10.98	1509.3	**1.616**	51.1	2.5	1466.93	5.292	36.94	6.25	1612	18.32	167.49	**0.15**	**1378.2**	2.764	**8.53**
i14	13.53	1399.8	**1.624**	50.7	2.6	1307.53	5.343	31.8	7.06	1604	19.12	288.49	**0**	**1241.3**	2.452	**6.63**
i15	11.92	1449.3	**1.593**	57.9	3.94	1395.67	8.722	37.86	5.71	1664	18.78	274.43	**0**	**1304.1**	1.953	**5.79**
i16	17.26	1599.4	**1.662**	73.9	6.3	1508.13	8.772	43.69	9.24	1974	19.13	355.87	**0.59**	**1393.5**	3.458	**12.99**
i17	7.46	1378.7	**1.733**	44.6	1.71	1343.67	8.7	23.09	4.75	1428	18.23	68.85	**0**	**1289.2**	1.924	**5.11**
i18	11.58	1500.7	**1.609**	65.5	4.91	1458.73	6.436	73.36	5.2	1528	18.13	84.37	**0**	**1362.4**	2.243	**7.52**
i19	15.33	1576.6	**1.645**	49.5	4.24	1507.43	4.715	49.35	7.68	1640	18.11	139.78	**0.44**	**1387.5**	3.542	**9.66**
i20	15.07	1528.1	**1.703**	67.5	3.69	1436.07	6.819	35.62	6.48	1631	18.88	180.85	**0.23**	**1338.2**	3.545	**6.89**
i21	12.92	1514.2	**1.661**	63.6	4.18	1448.27	5.863	30.93	4.85	1631	19.21	252.75	**0.45**	**1354.9**	1.938	**3.73**
i22	15.35	1529.5	**1.611**	50.7	4.68	1452.37	7.662	37.43	4.9	1755	19.08	344.73	**0.23**	**1353.7**	**3.959**	12.26
i23	12.34	1422.2	**1.608**	47.3	5.21	1378.73	7.514	37.76	8.93	1527	18.25	195.61	**0.16**	**1288**	**2.536**	7.31
i24	11.44	1404.2	**1.658**	41.7	3.41	1357.43	4.669	35.85	6.35	1501	18.74	202.77	**0.08**	**1272.53**	**3.013**	6.12
i25	12.46	1547.4	**1.746**	43.6	3.34	1492.97	6.602	39.64	7.27	1862	19.56	345.47	**0.07**	**1393.27**	**3.249**	10.5
i26	12.65	1484.7	**1.632**	61	5.16	1432.27	4.835	32.52	4.63	1650	18.85	233	**0.3**	**1334.57**	**4.388**	6.44
i27	11.51	1406.1	**1.61**	54.7	3.97	1352.23	7.328	36.99	6.98	1440	18.31	95.11	**0.56**	**1271.53**	**2.02**	3.62
i28	13.52	1542.8	**1.601**	62.8	3.31	1478.57	7.013	47.6	7.58	1717	18.96	240.34	**0.22**	**1378.97**	**2.528**	13.18
i29	13.23	1449.3	**1.601**	54.6	3.13	1377.97	7.846	29.09	4.69	1575	18.79	256.3	**0.16**	**1291.93**	**3.656**	6.7
i30	14.46	1538.3	**1.595**	74.6	5.36	1467.87	6.524	33.18	7.96	1675	18.94	238.63	**0.3**	**1365.77**	**3.584**	7.43
Average	12.65			54.77	3.84			36.20	6.23			222.17	0.227			7.42

TABLE 6. The average rank of the DBMO with GA, PSO and RKBMO for BAP.

No.	Algorithm	Friedman	QUADE
1	DBMO	1	1
2	PSO	2	2
3	GA	3	3
4	RKBMO	4	4

TABLE 7. The adjusted p-values computed by Friedman test for DBMO against GA, PSO and RKBMO for BAP.

No.	Algorithm	Unadjusted P	P Holm	P Hochberg
1	GA	0.0000	0.0000	0.0000
2	PSO	0.0027	0.0027	0.0027
4	RKBMO	0.0000	0.0000	0.0000
No.	Algorithm	Unadjusted P	P Holm	P Hochberg

of the DBMO for exploiting the most promising regions of the search space. In contrast to the RKBMO, which uses mathematical equations through random-key representation of the solution to move in the search space and improve the solution, the DBMO uses discrete operators that can directly deal with the solution and improve it. The superior results produced by the DBMO clearly answer the question of which variant of the BMO that should be used for combinatorial optimization. problems as well as the best operators and configuration for applying it to such problems.

VIII. CONCLUSION

In this work, we proposed two variants of the BMO for solving combinatorial optimization problems: the random-key bird mating optimizer (RKBMO) and discrete bird mating optimizer (DBMO). The RKBMO used random-key representation to construct a direct mapping relationship between the algorithm and the problem at hand. In contrast, the DBMO used three main operators to generate new solution: the hill-climbing algorithm, two-parent order crossover, and multi-parent order crossover. To evaluate the proposed methods, we tested them on the travelling salesman problem and berth allocation problem using the same parameter settings. In addition, a comparative study was conducted with two widely studied optimization algorithms in the literature: GA and PSO. The computational results demonstrated that the DBMO was able to outperform the RKBMO across all instances in both problem domains as well as GA and PSO. Furthermore, the DBMO matched the best-known results in some instances. In conclusion, we proved that the DBMO can be used to solve combinatorial optimization problems and achieve satisfactory results.

In future work, it might be beneficial to investigate the effectiveness of combining the three main DBMO operators (two-parent crossover, multi-parent crossover, and local search) in a pool in order to select one of them randomly and employ it to improve the whole population. In addition, changing the selected operator to another one could be done according to the search status. This may improve the DBMO to allow it to effectively tackle any problem with minimal modification.

ACKNOWLEDGEMENT

The authors wish to thank the Ministry of Higher Education Malaysia for supporting and funding this work (grant ID: FRGS/1/2018/ICT02/UKM/01/1) and Universiti Kebangsaan Malaysia (grant ID: DIP-2019-013).

REFERENCES

[1] H. Masum, "Review of how to solve it: Modern heuristics," *ACM SIGACT News*, vol. 32, no. 1, p. 8, Mar. 2001, doi: 10.1145/568438.568443.
[2] F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stützle, "ParamILS: An automatic algorithm configuration framework," *J. Artif. Intell. Res.*, vol. 36, pp. 267–306, 2009.
[3] T. Urli. (2014). *Hybrid Meta-Heuristics for Combinatorial Optimization*. Accessed: Oct. 6, 2016. [Online]. Available: https://dspace-uniud.cineca.it/handle/10990/445
[4] T. Stützle, "Applying iterated local search to the permutation flow shop problem," 1998.
[5] S. S. Shreem, S. Abdullah, and M. Z. A. Nazri, "Hybridising harmony search with a Markov blanket for gene selection problems," *Inf. Sci.*, vol. 258, pp. 108–121, Feb. 2014.
[6] M. Masood, M. M. Fouad, R. Kamal, I. Glesk, and I. U. Khan, "An improved particle swarm algorithm for multi-objectives based optimization in MPLS/GMPLS networks," *IEEE Access*, vol. 7, pp. 137147–137162, 2019.
[7] B. V. Babu and M. M. L. Jehan, "Differential evolution for multi-objective optimization," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 4, Dec. 2003, pp. 2696–2703, doi: 10.1109/CEC.2003.1299429.
[8] R. Bellman, "The theory of dynamic programming," *Bull. Amer. Math. Soc.*, vol. 60, no. 6, pp. 503–516, Nov. 1954, doi: 10.1090/S0002-9904-1954-09848-8.
[9] C. Mu, Z. Ni, C. Sun, and H. He, "Air-breathing hypersonic vehicle tracking control based on adaptive dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 584–598, Mar. 2017. Accessed: Dec. 21, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7398119/
[10] J. E. Mitchell, "Branch-and-cut algorithms for combinatorial optimization problems," in *Handbook of Applied Optimization*, vol. 1. 2002, pp. 65–77.
[11] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Oper. Res.*, vol. 46, no. 3, pp. 316–329, 1998, doi: 10.1287/opre.46.3.316.
[12] B. Beheshti, O. A. Prokopyev, and E. L. Pasiliao, "Exact solution approaches for bilevel assignment problems," *Comput. Optim. Appl.*, vol. 64, no. 1, pp. 215–242, 2016, doi: 10.1007/s10589-015-9799-4.
[13] S. Abdullah and H. Turabieh, "Generating university course timetable using genetic algorithms and local search," in *Proc. 3rd Int. Conf. Converg. Hybrid Inf. Technol.*, Busan, South Korea, Nov. 2008, pp. 254–260.
[14] A. Askarzadeh, "Bird mating optimizer: An optimization algorithm inspired by bird mating strategies," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 4, pp. 1213–1228, Apr. 2014, doi: 10.1016/j.cnsns.2013.08.027.
[15] M. B. Dowlatshahi, H. Nezamabadi-pour, and M. Mashinchi, "A discrete gravitational search algorithm for solving combinatorial optimization problems," *Inf. Sci.*, vol. 258, pp. 94–107, Feb. 2014, doi: 10.1016/j.ins.2013.09.034.
[16] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *J. Oper. Res. Soc.*, vol. 64, no. 12, pp. 1695–1724, Dec. 2013, doi: 10.1057/jors.2013.71.
[17] K. Chakhlevitch and P. Cowling, "Hyperheuristics: Recent developments," in *Adaptive and Multilevel Metaheuristics*. Berlin, Germany: Springer, 2008, pp. 3–29.
[18] F. Glover and M. Laguna, "Tabu search*," in *Handbook of Combinatorial Optimization*, P. Pardalos, D. Z. Du, and R. Graham, Eds. New York, NY, USA: Springer, 2008.
[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983, doi: 10.1126/science.220.4598.671.
[20] V. Cerný, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *J. Optim. Theory Appl.*, vol. 45, no. 1, pp. 41–51, Jan. 1985, doi: 10.1007/BF00940812.
[21] G. Dueck, "New optimization heuristics," *J. Comput. Phys.*, vol. 104, no. 1, pp. 86–92, Jan. 1993, doi: 10.1006/jcph.1993.1010.
[22] F. Glover, "Tabu search—Part II," *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, Feb. 1990, doi: 10.1287/ijoc.2.1.4.
[23] C. E. Taylor, "Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial Intelligence. Complex adaptive systems. John H. Holland," *Quart. Rev. Biol.*, vol. 69, no. 1, pp. 88–89, Mar. 1994, doi: 10.1086/418447.
[24] M. K. Dhadwal, S. N. Jung, and C. J. Kim, "Advanced particle swarm assisted genetic algorithm for constrained optimization problems," *Comput. Optim. Appl.*, vol. 58, no. 3, pp. 781–806, Jul. 2014, doi: 10.1007/s10589-014-9637-0.
[25] J. Meena, M. Kumar, and M. Vardhan, "Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint," *IEEE Access*, vol. 4, pp. 5065–5082, 2016.
[26] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, vol. 1. Cambridge, MA, USA: MIT Press, 1992.
[27] E. Taha Yassen, M. Ayob, M. Z. Ahmad Nazri, and N. R. Sabar, "Meta-harmony search algorithm for the vehicle routing problem with time windows," *Inf. Sci.*, vol. 325, pp. 140–158, Dec. 2015, doi: 10.1016/j.ins.2015.07.009.
[28] F. Glover, "Heuristics for integer programming using surrogate constraints," *Decis. Sci.*, vol. 8, no. 1, pp. 156–166, Jan. 1977, doi: 10.1111/j.1540-5915.1977.tb01074.x.
[29] D. C. Andrus, "Toward a complex adaptive intelligence community-central intelligence agency," *Studies*, 2005.
[30] C. Blum, M. Dorigo, and T. Stützle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2005, p. 300. Accessed: May 16, 2019. [Online]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=M.+Dorigo%2C+T.+Sttzle%2C+Ant+Colony+Optimization%2C+MIT+Press%2C+Cambridge%2C+MA+%282004%29%2C+300+pp&btnG=
[31] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, Oct. 1995, pp. 39–43, doi: 10.1109/MHS.1995.494215.
[32] K. Deb and N. Padhye, "Enhancing performance of particle swarm optimization through an algorithmic link with genetic algorithms," *Comput. Optim. Appl.*, vol. 57, no. 3, pp. 761–794, Apr. 2014, doi: 10.1007/s10589-013-9605-0.
[33] A. Askarzadeh and A. Rezazadeh, "Extraction of maximum power point in solar cells using bird mating optimizer-based parameters identification approach," *Sol. Energy*, vol. 90, pp. 123–133, Apr. 2013, doi: 10.1016/j.solener.2013.01.010.
[34] Q. Zhang, G. Yu, and H. Song, "A hybrid bird mating optimizer algorithm with teaching-learning-based optimization for global numerical optimization," *Statist., Optim. Inf. Comput.*, vol. 3, no. 1, pp. 54–65, 2015.
[35] N. K. S. Behera, A. R. Routray, J. Nayak, and H. S. Behera, "Bird mating optimization based multilayer perceptron for diseases classification," in *Smart Innovation, Systems and Technologies*, vol. 33. 2015, pp. 305–315, doi: 10.1007/978-81-322-2202-6_27.
[36] A. Askarzadeh and L. dos Santos Coelho, "Determination of photovoltaic modules parameters at different operating conditions using a novel bird mating optimizer approach," *Energy Convers. Manage.*, vol. 89, pp. 608–614, Jan. 2015, doi: 10.1016/j.enconman.2014.10.025.
[37] H. Li, J. K. Liu, and Z. R. Lu, "Bird mating optimizer in structural damage identification," in *Advances in Swarm and Computational Intelligence* (Lecture Notes in Computer Science), vol. 9140. 2015, pp. 49–56, doi: 10.1007/978-3-319-20466-6_5.

[38] S. Corde, V. R. Chifu, I. Salomie, E. S. Chifu, and A. Iepure, "Bird mating optimization method for one-to-n skill matching," in *Proc. IEEE 12th Int. Conf. Intell. Comput. Commun. Process. (ICCP)*, Sep. 2016, pp. 155–162, doi: 10.1109/ICCP.2016.7737139.

[39] T.-C. Ou, W.-F. Su, X.-Z. Liu, S.-J. Huang, and T.-Y. Tai, "A modified bird-mating optimization with hill-climbing for connection decisions of transformers," *Energies*, vol. 9, no. 9, p. 671, Aug. 2016, doi: 10.3390/en9090671.

[40] D. Goswami and S. Chakraborty, "Multi-objective optimization of electrochemical discharge machining processes: A posteriori approach based on bird mating optimizer," *OPSEARCH*, vol. 54, no. 2, pp. 306–335, Jun. 2017, doi: 10.1007/s12597-016-0285-2.

[41] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," *ORSA J. Comput.*, vol. 6, no. 2, pp. 154–160, May 1994, doi: 10.1287/ijoc.6.2.154.

[42] M. F. Tasgetiren, P. N. Suganthan, Q.-K. Pan, and Y.-C. Liang, "A genetic algorithm for the generalized traveling salesman problem," in *Proc. IEEE Congr. Evol. Comput.*, vol. 174, Sep. 2007, pp. 2382–2389, doi: 10.1109/CEC.2007.4424769.

[43] C.-J. Ting, K.-C. Wu, and H. Chou, "Particle swarm optimization algorithm for the berth allocation problem," *Expert Syst. Appl.*, vol. 41, no. 4, pp. 1543–1550, Mar. 2014, doi: 10.1016/j.eswa.2013.08.051.

[44] A. Ouaarab, B. Ahiod, and X.-S. Yang, "Random-key cuckoo search for the travelling salesman problem," *Soft Comput.*, vol. 19, no. 4, pp. 1099–1106, Apr. 2015, doi: 10.1007/s00500-014-1322-9.

[45] Q.-K. Pan, M. F. Tasgetiren, and Y.-C. Liang, "A discrete differential evolution algorithm for the permutation flowshop scheduling problem," in *Proc. Genetic Evol. Comput. Conf. (Gecco)*, vols. 1–2, Jul. 2007, pp. 126–133. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-34548126517&partnerID=40&md5=800f6dea51695f25d7646b05f5f71541

[46] W.-h. Zhong, J. Zhang, and W.-n. Chen, "A novel discrete particle swarm optimization to solve traveling salesman problem," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2007, pp. 3283–3287, doi: 10.1109/CEC.2007.4424894.

[47] G. Reinelt, "TSPLIB—A traveling salesman problem library," *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, Nov. 1991, doi: 10.1287/ijoc.3.4.376.

[48] J.-F. Cordeau, G. Laporte, P. Legato, and L. Moccia, "Models and tabu search heuristics for the berth-allocation problem," *Transp. Sci.*, vol. 39, no. 4, pp. 526–538, Nov. 2005, doi: 10.1287/trsc.1050.0120.

[49] A. Arram, M. Z. A. Nazri, M. Ayob, and A. Abunadi, "Bird mating optimizer for discrete berth allocation problem," in *Proc. Int. Conf. Electr. Eng. Informat. (ICEEI)*, Aug. 2015, pp. 450–455, doi: 10.1109/ICEEI.2015.7352543.

[50] Y. Wang, "The hybrid genetic algorithm with two local optimization strategies for traveling salesman problem," *Comput. Ind. Eng.*, vol. 70, pp. 124–133, Apr. 2014, doi: 10.1016/j.cie.2014.01.015.

[51] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, pp. 231–247, Jun. 1992, doi: 10.1016/0377-2217(92)90138-Y.

[52] A. Arram, M. Ayob, and M. Zakree, "Comparative study of meta-heuristic approaches for solving traveling salesman problems," *Asian J. Appl. Sci.*, vol. 7, no. 7, pp. 662–670, Jul. 2014.

[53] G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*. New York, NY, USA: Springer-Verlag, 1994. Accessed: May 2, 2019. [Online]. Available: https://scholar.google.com/

[54] C. Bierwirth and F. Meisel, "A follow-up survey of berth allocation and quay crane scheduling problems in container terminals," *Eur. J. Oper. Res.*, vol. 244, no. 3, pp. 675–689, Aug. 2015, doi: 10.1016/j.ejor.2014.12.030.

[55] Q. Luo, Y. Zhou, J. Xie, M. Ma, and L. Li, "Discrete bat algorithm for optimal problem of permutation flow shop scheduling," *Scientific World J.*, vol. 2014, pp. 1–15, 2014, doi: 10.1155/2014/630280.

[56] S.-W. Lin and C.-J. Ting, "Solving the dynamic berth allocation problem by simulated annealing," *Eng. Optim.*, vol. 46, no. 3, pp. 308–327, Mar. 2014, doi: 10.1080/0305215X.2013.768241.

[57] M. Lones, "Sean luke: Essentials of metaheuristics," *Genetic Program. Evolvable Mach.*, vol. 12, no. 3, pp. 333–334, Sep. 2011, doi: 10.1007/s10710-011-9139-0.

[58] D. E. Goldberg, "A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing," *Complex Syst.*, vol. 4, no. 4, pp. 445–460, 1990.

[59] I. Ono, M. Yamamura, and S. Kobayashi, "A genetic algorithm for job-shop scheduling problems using job-based order crossover," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1996, pp. 547–552, doi: 10.1109/ICEC.1996.542658.

[60] K. Bryant and A. Benjamin, "Advisor genetic algorithms and the traveling salesman problem," Dept. Math., Harvey Mudd College, Claremont, CA, USA, Tech. Rep., 2000, pp. 11–18.

[61] C.-K. Ting, C.-H. Su, and C.-N. Lee, "Multi-parent extension of partially mapped crossover for combinatorial optimization problems," *Expert Syst. Appl.*, vol. 37, no. 3, pp. 1879–1886, Mar. 2010, doi: 10.1016/j.eswa.2009.07.082.

[62] A. Eiben, P. Raue, and Z. Ruttkay, "Genetic algorithms with multi-parent recombination," in *Parallel Problem Solving From Nature—PPSN III*. 1994. Accessed: May 29, 2016. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-58484-6_252

[63] L.-P. Wong, M. Y. H. Low, and C. Soon Chong, "Bee colony optimization with local search for traveling salesman problem," in *Proc. 6th IEEE Int. Conf. Ind. Informat.*, vol. 19, Jul. 2008, pp. 1019–1025, doi: 10.1109/INDIN.2008.4618252.

[64] S.-W. Lin, K.-C. Ying, and S.-Y. Wan, "Minimizing the total service time of discrete dynamic berth allocation problem by an iterated greedy heuristic," *Sci. World J.*, vol. 2014, pp. 1–12, Sep. 2014, doi: 10.1155/2014/218925.

[65] K.-P. Wang, L. Huang, C.-G. Zhou, and W. Pang, "Particle swarm optimization for traveling salesman problem," in *Proc. Int. Conf. Mach. Learn. Cybern.*, Nov. 2003, pp. 1583–1585, doi: 10.1109/ICMLC.2003.1259748.

[66] Q.-K. Pan, M. F. Tasgetiren, and Y.-C. Liang, "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2807–2839, 2008.

[67] B. Q. Pinto, C. C. Ribeiro, I. Rosseti, and A. Plastino, "A biased random-key genetic algorithm for the maximum quasi-clique problem," *Eur. J. Oper. Res.*, vol. 271, no. 3, pp. 849–865, Dec. 2018, doi: 10.1016/j.ejor.2018.05.071.

[68] Q. Lihong and L. Shengping, "An improved genetic algorithm for integrated process planning and scheduling," *Int. J. Adv. Manuf. Technol.*, vol. 58, pp. 727–740, Jan. 2012, doi: 10.1007/S00170-011-3409-0.

[69] Y.-J. Gong, J. Zhang, O. Liu, R.-Z. Huang, H. S.-H. Chung, and Y.-H. Shi, "Optimizing the vehicle routing problem with time windows: A discrete particle swarm optimization approach," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 254–267, Mar. 2012, doi: 10.1109/TSMCC.2011.2148712.

[70] T.-C. Ou and C.-M. Hong, "Dynamic operation and control of microgrid hybrid power systems," *Energy*, vol. 66, pp. 314–323, Mar. 2014, doi: 10.1016/j.energy.2014.01.042.

[71] T. Grueninger and D. Wallace, "Multimodal optimization using genetic algorithms," in *Handbook of Genetic Algorithms*. 1996, pp. 332–349. Accessed: Dec. 5, 2016. [Online]. Available: http://ci.nii.ac.jp/naid/10000000876/

[72] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011, doi: 10.1016/j.swevo.2011.02.002.

[73] K. Buhrkal, S. Zuglian, S. Ropke, J. Larsen, and R. Lusby, "Models for the discrete berth allocation problem: A computational comparison," *Transp. Res. E, Logistics Transp. Rev.*, vol. 47, no. 4, pp. 461–473, Jul. 2011, doi: 10.1016/j.tre.2010.11.016.

[74] S. Gupta, K. Deep, A. A. Heidari, H. Moayedi, and H. Chen, "Harmonized salp chain-built optimization," *Eng. with Comput.*, vol. 2019, pp. 1–31, Oct. 2019.

[75] S. Gupta and K. Deep, "A novel random walk grey wolf optimizer," *Swarm Evol. Comput.*, vol. 44, pp. 101–112, Feb. 2019.

ANAS ARRAM received the Ph.D. degree in computer science with The National University of Malaysia, in 2017. He is currently working as a Computer Vision Engineer with FitLens Company. He has published more than six papers at international journals peer-reviewed international conferences. His main research areas include meta-heuristics, hyper-heuristics, scheduling and timetabling, machine learning, and deep learning. He is a member of the Data Mining and Optimization Research Group (DMO), Centre for Artificial Intelligent (CAIT), National University of Malaysia. He received an Outstanding Researcher Award upon completing his Ph.D.

MASRI AYOB received the Ph.D. degree in computer science from the University of Nottingham, in 2005. She was a member of the ASAP Research Group, University of Nottingham. She has been a Lecturer with the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), since 1997. She is currently a Principle Researcher with the Data Mining and Optimisation Research Group (DMO), Centre for Artificial Intelligent (CAIT), UKM. She has published more than 150 articles at international journals and at peer-reviewed international conferences. Her main research areas include meta-heuristics, hyper-heuristics, scheduling and timetabling, especially educational timetabling, healthcare scheduling, and routing problems. She has been served as a program committee for more than 50 international conferences and a Reviewer for high impact journals.

GRAHAM KENDALL received the B.Sc. degree (Hons.) in computation from the University of Manchester Institute of Science and Technology (UMIST), U.K., in 1997, and the Ph.D. degree from the School of Computer Science, University of Nottingham, in 2000. Before entering academia, he spent almost 20 years in the IT industry, working for various U.K. companies (including the Co-operative Wholesale Society and Provincial Insurance), undertaking a variety of roles including a Computer Operator, a Technical Support Manager, and a Service Manager. He is currently a Professor of computer science. He is also with the University of Nottingham's Malaysia campus, where he is also serving as the Vive-Provost of research and knowledge transfer. He is also a member of the Automated Scheduling, Optimisation and Planning Research Group, School of Computer Science. He has edited and authored 9 books and has published almost 50 refereed journal articles (the vast majority in ISI ranked journals) and more than 90 articles in peer reviewed conferences. His expertise includes operational research, meta- and hyper-heuristics, evolutionary computation, and artificial intelligence, with a specific interest in scheduling, including timetabling, sports scheduling, cutting and packing, and rostering. He is a Fellow of the Operational Research Society. He has been awarded externally funded grants worth over £5.5M from a variety of sources including the Engineering and Physical Sciences Research Council and commercial organizations. He chairs the Steering Committee of the Multidisciplinary International Conference on Scheduling: Theory and Applications, in addition to chairing several other international conferences in recent years. He is an Associate Editor of eight international journals (including two the IEEE TRANSACTIONS).

AIAA SULAIMAN received the B.A. degree in electrical and computer engineering from AN-Najah National University, Nablus, Palestine, in 2003, and the M.Sc. degree in scientific computing minor in computer science and engineering from Bierzit University, Ramallah, Palestine, in 2010. He is currently pursuing the Ph.D. degree with the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM). His area of interests include machine learning, image processing, and pattern recognition.