




Article

Multi-Objective ABC-NM Algorithm for Multi-Dimensional Combinatorial Optimization Problem

Muniyan Rajeswari ¹, Rajakumar Ramalingam ², Shakila Basheer ³, Keerthi Samhitha Babu ⁴, Mamoon Rashid ^{5,*} and Ramar Saranya ⁶

¹ Department of Computer Science and Engineering, Sri Manakula Vinayagar Engineering College, Pondicherry 605107, India

² Department of Computer Science and Technology, Madanapalle Institute of Technology and Science, Madanapalle 517325, India

³ Department of Information Systems, College of Computer and Information Science, Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

⁴ Department of Computer Science and Information Technology, KL Deemed to be University, Guntur District, Vaddeswaram 522302, India

⁵ Department of Computer Engineering, Faculty of Science and Technology, Vishwakarma University, Pune 411048, India

⁶ Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli 627012, India

* Correspondence: mamoon.rashid@vupune.ac.in

Abstract: This article addresses the problem of converting a single-objective combinatorial problem into a multi-objective one using the Pareto front approach. Although existing algorithms can identify the optimal solution in a multi-objective space, they fail to satisfy constraints while achieving optimal performance. To address this issue, we propose a multi-objective artificial bee colony optimization algorithm with a classical multi-objective theme called fitness sharing. This approach helps the convergence of the Pareto solution set towards a single optimal solution that satisfies multiple objectives. This article introduces multi-objective optimization with an example of a non-dominated sequencing technique and fitness sharing approach. The experimentation is carried out in MATLAB 2018a. In addition, we applied the proposed algorithm to two different real-time datasets, namely the knapsack problem and the nurse scheduling problem (NSP). The outcome of the proposed MBABC-NM algorithm is evaluated using standard performance indicators such as average distance, number of reference solutions (NRS), overall count of attained solutions (TNS), and overall non-dominated generation volume (ONGV). The results show that it outperforms other algorithms.

Keywords: artificial bee colony; Nelder—Mead; multi-objective optimization; 0-1 knapsack problem; nurse scheduling problem

MSC: 68Q17; 68Q25; 68Q30; 68Q87



Citation: Rajeswari, M.; Ramalingam, R.; Basheer, S.; Babu, K.S.; Rashid, M.; Saranya, R. Multi-Objective ABC-NM Algorithm for Multi-Dimensional Combinatorial Optimization Problem. *Axioms* **2023**, *12*, 395. <https://doi.org/10.3390/axioms12040395>

Academic Editors: Ivan Mauricio Amaya-Contreras and José Carlos Ortiz-Bayliss

Received: 23 January 2023

Revised: 12 April 2023

Accepted: 15 April 2023

Published: 19 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Multi-objective optimization is the method of finding a single optimal result which has the potential to satisfy more than one objective for the given problem. There are three possible situations in multi-objective optimization problems [1,2]:

- Diminish all the objective functions;
- Increase all the objective functions;
- Diminish a few objectives and increase other objective functions.

$$\text{Max}f(x) = \min(-f(x)) \quad (1)$$

A multi-objective optimization problem (MOP) aims to determine a better compromising solution than a solitary individual. The vector of the decision variable $\vec{x}^* \in F$ is Pareto optimal when there is no other decision variable $\vec{x} \in F$ such that $f_i(\vec{x}) \leq f_i(\vec{x}^*)$, $i = 1, 2, \dots, n$. The vector \vec{x}^* is determined as Pareto optimal or globally non-dominated if no other solution in the set can dominate [3]. The set of solutions thus produced is said to be a Pareto-optimal set. The optimal set is specified as the Pareto-optimal front. From the multi-objective set, the user can select an ideal solution [4].

The dominance relation is used to associate two individuals. For example, a solution u is said to lead to another solution v if and only if $f_i(u) \leq f_i(v)$, for $i = 1, 2, \dots, n$ and $f_i(u) < f_i(v)$ for at least one i , it is known as a globally non-dominated set. No other solution among the set can dominate it. The Pareto dominance for the solution is to dominate other solutions, and should not get worse in any given objectives with strict efficiency than one of them [5]. The Pareto dominance among two solutions u and v can possibly occur in any one of these cases:

- The solution u dominates solution v , denoted as $u_i \prec v_i$;
- The solution u is dominated by solution v , denoted as $v_i \prec u_i$;
- Both the solutions u and v are not dominated by each other, and they are said to be non-dominated. It is denoted as $\neg(u_i \prec v_i) \wedge \neg(v_i \prec u_i)$.

Recently, several meta-heuristic algorithms have been introduced to address the multi-dimensional combinatorial optimization problem [6]. Some of the famous techniques, namely genetic algorithm [7], differential evolution [8], particle swarm optimization [9], grey wolf optimization [10], and firefly algorithm [11], have been applied in various real-time applications, including optimum design for a centrifugal pump [12,13], Optimizing Magnification Ratio for the Flexible Hinge Displacement Amplifier [14], clustering [15], economic load dispatch [16], and job scheduling [17], to determine optimal solutions. However, the algorithms must be reinforced while applying them to multi-objective problems [18]. In this work, we utilized the ABC algorithm, which is more robust in mathematical analysis and provides more adequate solutions than all other algorithms. However, the ABC algorithm consumes ample computation time due to inefficient search direction while handling multi-objective problems. To eradicate these issues, we introduced the Nelder—Mead technique with non-dominated sorting and fitness allotment methods to address the multi-objective concerns.

The main theme of this work is discussed below:

- A novel algorithm, MBABC-NM, is proposed to improve the exploitation of the artificial bee colony (ABC) technique. The algorithm incorporates a modified non-dominated sorting and fitness-sharing approach to handle multi-dimensional problems efficiently.
- The proposed MBABC-NM algorithm is tested on two different real-time datasets: the knapsack problem and the nurse scheduling problem.
- The algorithm's performance is compared with other state-of-the-art algorithms, like genetic algorithm, cyber swarm optimization, and particle swarm optimization.
- The results of the experiments demonstrate that MBABC-NM outperforms the compared algorithms significantly. This result suggests that the proposed algorithm can effectively solve real-world optimization problems.

The rest of the paper is structured such that Section 2 discusses modified non-dominated sorting and fitness-sharing techniques over the multi-dimensional problem; Section 3 illustrates the detailed working process of the proposed MBABC-NM algorithm; and Section 4 presents the experimental setup for NSP and the 0-1 knapsack problem. The empirical study and the discussion of the results are shown in Section 5, while Section 6 summarizes the work and its future directions.

2. Methodology

2.1. Modified Non-Dominated Sorting

In modified non-dominant sorting, the algorithm divides the population L , $1 \leq L \leq N$ fronts in decreasing order of their dominance $F = \{F_1, F_2, \dots, F_L\}$. Each solution in a front is non-dominated by the other. Each individual in F_l is conquered by at least one individual in its preceding front F_l . Non-dominated arrangement aids in arranging the solutions sequentially based on the dominance, as mentioned in the above relation [19]. It improves the search capability of the multi-objective approach by introducing modified non-dominated solutions into the search space. The detailed narrative of the modified non-dominated arrangement is discussed in Algorithm 1. This algorithm is specified as one function involved in Algorithm 3.

Algorithm 1: Non-Dominated Sort (Z)

```

Input:  $Z$ 
For each individual  $a \in Z$  do
  Individuals dominated by  $a$ 
   $P_a \leftarrow \emptyset$ 
   $P_b \leftarrow \emptyset$ 
  Solutions which dominate  $a$ 
   $C_a \leftarrow 0$ 
  For each solution  $b \in Z$  do
    if ( $a \prec b$ ) then
      Add the individuals  $b$  to the set of solutions dominated by  $a$ 
       $P_a \leftarrow P_a \cup \{b\}$ 
    else if ( $b \prec a$ ) then
      Increment the domination counter  $a$ 
       $C_a \leftarrow C_a + 1$ 
    End if
  end for
  if  $C_a = 0$  then
    Assign non-dominance rank as 1 for individual  $a$ 
     $a_{rank} \leftarrow 1$ 
     $L_1 \leftarrow L_1 \cup \{a\}$ 
  End if
end for
Initialize front counter
 $u \leftarrow 1$ 
While  $L_u \neq \emptyset$  do
  Members of next front  $K$ 
   $K \leftarrow \emptyset$ 
  For each solution  $a \in L_u$  do
    For each solution  $b \in P_a$  do
      Decrement the dominant counter of  $b$ 
       $C_b \leftarrow C_b - 1$ 
      if  $C_b = 0$  then
        Assign rank for the individual  $b$ 
         $b_{rank} \leftarrow u + 1$ 
         $K \leftarrow K \cup \{b\}$ 
      End if
    end for
  end for
   $u \leftarrow u + 1$ 
   $L_u \leftarrow K$ 
  The dominant solution of  $L_u$  are stored in  $\hat{L}_u$ 

```

This section discusses a modified non-domination sorting process that helps improve the multi-objective algorithm’s search capability. In addition, the fitness-sharing function that aids the population in exploring diverse groups based on individual similarity is discussed in Section 2.2.

2.2. Fitness Sharing

Fitness sharing in evolutionary computing is used for isolating the population into diverse groups based on individual similarity [1]. It transforms an individual’s fitness into the shared fitness value; usually, it is a lower value than the original. Only a limited amount of fitness value is available in each niche, and individuals in the same niche will share fitness value. The shared fitness $f_{shared}(i)$ of food particle i with fitness fit_i can be measured by

$$f_{shared}(i) = \frac{fit_i}{n_i} \tag{2}$$

where n_i is the niche total, which counts the number of food particles with fitness fit_i shared. The niche count can be calculated by summing the distribution function over the swarm.

$$n_i = \sum_{j=1}^{FP} \varphi(d_{ij}) \tag{3}$$

where FP denotes number of food particles and (d_{ij}) is the distance between the food particles i and j . The sharing function φ computes the relationship between two food particles. The sharing function returns one if the food particles are identical and return 0 if the distance (d_{ij}) is greater than a threshold of dissimilarity value. The distribution function can be represented as

$$\varphi(d_{ij}) = f(x) = \begin{cases} 1 - \left(\frac{d_{ij}}{\theta_r}\right)^\varphi, & d < \theta_r \\ 0, & otherwise \end{cases} \tag{4}$$

where θ_r represents the sharing radius, which defines the size of the niche and threshold of dissimilarity. The food particles within this sharing radius are like each other and share their fitness. φ is the constant which normalizes the shape of the distribution function. d_{ij} is the distance between two food particles measured based on genotypic or phenotypic resemblance. The genotypic similarity is based on bit-string and is usually measured using Hamming distance. The phenotypic resemblance measures accurate parameters available in the search space using Euclidean distance.

$$d(a, b)^\psi = \left(\left[(p_a - p_b)^2 + (q_a - q_b)^2 \right]^{\frac{1}{2}} \right)^\psi \tag{5}$$

The Euclidean distance $d(a, b)$ is the distance between the nodes a and b , (p_a, q_a) are the coordinates of the node a , and (p_b, q_b) are the coordinates of the node b . The minimum transmission energy TE_{sol_i} contains the near-optimal solution. Fitness distribution based on phenotypic resemblance provides an improved outcome compared to distribution based on genotypic similarity [20–22].

In our algorithm, every individual finds a new solution. If a new solution dominates the original individual, it is entered into the external archive. If both do not dominate, then solutions are chosen randomly. When many non-dominated solutions exceed the archive size, our proposed algorithm uses a niching technique to truncate the crowded member and maintain uniform distribution among the archive members. Maintaining diversity among archive members is a complex task. Thus, in our proposed algorithm, we incorporated a fitness-sharing technique based on niche count, to ensure the diversity of the population.

The niching method maintains diversity and permits the algorithm to examine multiple summits in parallel. It also prevents the algorithm from being stuck in the local optima of search space, and can be viewed as the subspace of the population. For each niche in

our proposed algorithm, the fitness is finite and shared among the population. It is the process of optimizing the entire domain set. Fitness sharing transforms the raw fitness of a solution into a shared one. It helps to sustain diversity among the population, and thus our algorithm explores a better search space. The proposed fitness sharing technique is shown in Algorithm 2. Algorithm 3 invokes Algorithm 2 as a function while processing the execution.

Algorithm 2: Fitness Sharing (L_u)

Number of solutions in Front counter L
 $g \leftarrow |L_u|$
For $k \leftarrow 1$ to g **do**
 $L_u(\text{Share}_k) \leftarrow 0$
 For each objective m **do**
 Sort population with respect to all objectives
 $L_u \leftarrow \text{sort}(L_u, m)$
 $L_u[1] \leftarrow \infty$
 $L_u[g] \leftarrow \infty$
 For $k \leftarrow 2$ to $g - 1$ **do**
 Calculate Shared fitness of the k^{th} solution with fit_k
 $L_u(\text{Share}_k) \leftarrow \frac{fit_k}{n_k}$
 Niche count can be measured by
 $n_k \leftarrow \sum_{j=1}^{|L|} \varphi(d_{kj})$
 The sharing function between two population elements can be measured using

$$\varphi(d_{kj}) \leftarrow \begin{cases} 1 - \left(\frac{d_{kj}}{\theta_r}\right)^\rho, & d < \theta_r \\ 0, & \text{otherwise} \end{cases}$$

 End for
End for
End for

3. Multi-Objective BABC-NM for a Multi-Dimensional Combinatorial Problem

Multi-Objective BABC-NM consists of an algorithm explained in this thesis and Algorithms 3–6 in this chapter. Algorithm 3 is modified based on a multi-objective perspective, and the pseudocode of the proposed MBABC-NM was described in detail in Algorithm 3. The working process of the formulated MBABC-NM is portrayed in Figure 1. The mapping process of this algorithm involves the following steps:

Initialization: The algorithm randomly generates a population of candidate solutions to the MDCOP. Each candidate solution is represented as a vector of decision variables.

Fitness Evaluation: The fitness of each candidate solution is evaluated by computing its objective function values. In multi-objective optimization, multiple objective functions usually need to be optimized simultaneously. Thus, the fitness of a candidate solution is represented as a vector of accurate function values.

Employed Bees: In this step, some bees are selected to perform the exploration process. The selected bees modify the solutions in the population by adding or subtracting a random value from the decision variables. It generates a new solution for each bee.

Onlooker Bees: Some other bees are selected to perform the exploitation process in this step. The selected bees choose solutions from the population based on their fitness and then modify them similarly to the employed bees. It generates a new solution for each onlooker bee.

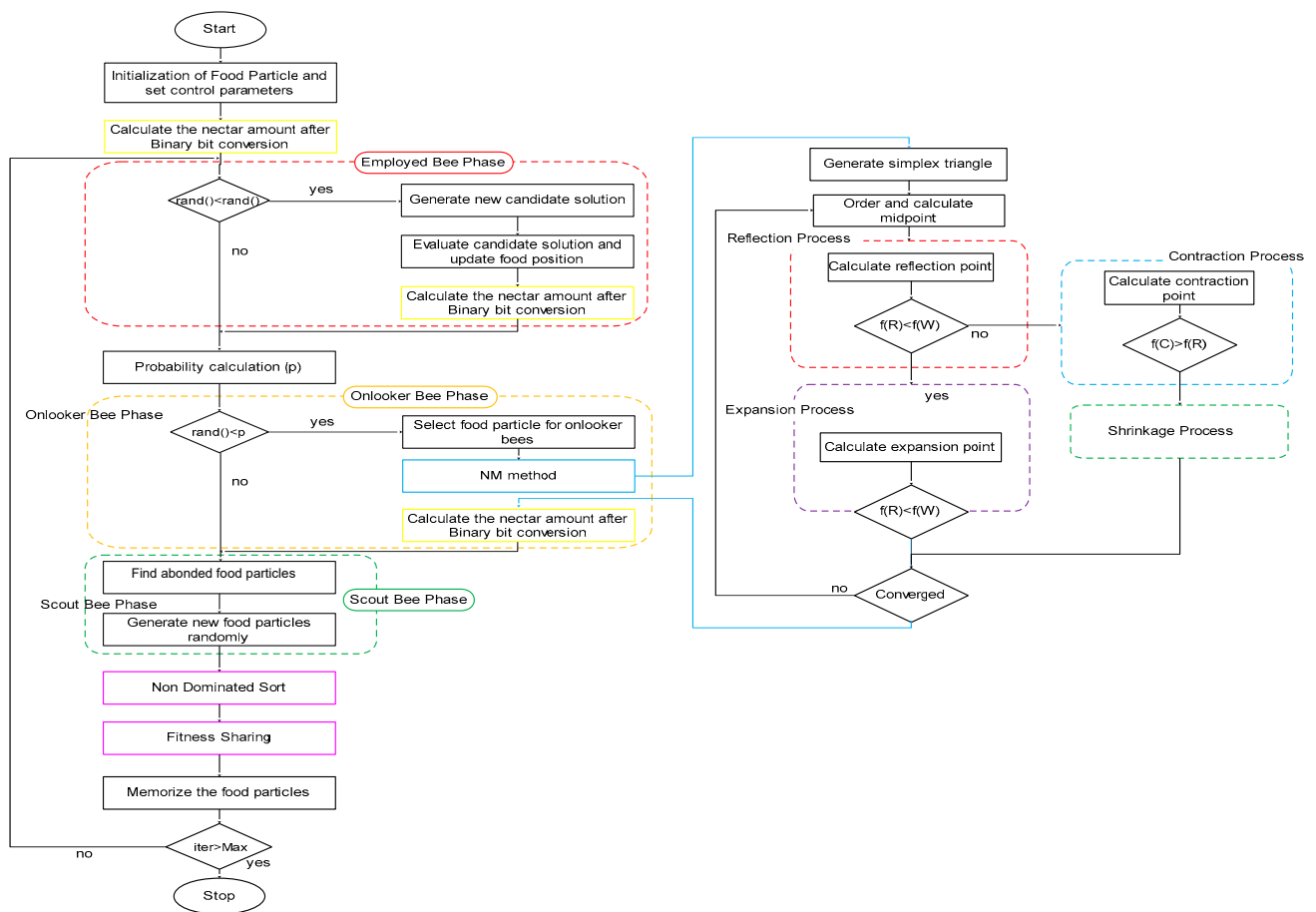


Figure 1. Workflow of MBABC-NM.

Neighbourhood Mutation: In this step, the solutions generated by the employed and onlooker bees are subjected to a neighbourhood mutation process. It involves selecting a neighbourhood around each solution and generating a new solution within that neighbourhood.

Scout Bees: In this step, if a solution has not been improved after a certain number of iterations, it is considered a non-promising solution and is replaced by a new random solution generated by a scout bee.

Pareto Optimization: After generating the new solutions, the algorithm performs a Pareto optimization process to determine the best solutions. The Pareto optimization process identifies solutions not dominated by any other solution in the population.

Termination: The algorithm continues to iterate through steps 3 to 7 until a termination criterion is met. It could be a maximum number of iterations or a satisfactory level of solution quality.

Algorithm 3: MBABC-NM**Input**

FS: Number of Food Sources
 MI: Maximum iteration
 Limit: number of predefined trials

Iter = 0

Prepare the population

For $i = 1$ to FS **do**

For $j = 1$ to S **do**

 Produce $x_{i,j}$ solution

$$x_{i,j} \leftarrow x_{min,j} \pm rand(0,1) * (x_{max,j} - x_{min,j})$$

 Where $x_{min,j}$ and $x_{max,j}$ are the min and max bound of the dimension j

$\hat{x}_{i,j} \leftarrow \text{BinaryConv}(x_{i,j})$ using Algorithm 5

For $h = 1$ to M **do**

 Evaluate the fitness of the population for a M number of Objectives

$$f_h \leftarrow f_h(\hat{x}_{i,j})$$

End for

$$trial(i) \leftarrow 0$$

End for

End for

iter $\leftarrow 1$

Repeat

{

 // *Employed Bee process* //

For each food source i **do**

 Create new individual v_i using

$$v_{i,j} \leftarrow x_{i,j} + \varnothing_{i,j} (x_{i,j} - x_{k,j})$$

$\hat{v}_{i,j} \leftarrow \text{BinaryConv}(v_{i,j})$ using Algorithm 5

 Evaluate $f(\hat{v}_i)$

 Select between $f(\hat{v}_i)$ and $f(\hat{x}_i)$ using greedy method

If $f(\hat{v}_i) < f(\hat{x}_i)$

$$x_i \leftarrow v_i$$

$$f(\hat{x}_i) \leftarrow f(\hat{v}_i)$$

$$trial(i) \leftarrow 0$$

Else

$$trial(i) \leftarrow trial(i) + 1$$

End if

End For

 // *Onlooker Bee Phase* //

If iter = 1

 Set $r = 0, i = 1;$

While ($r \leq \text{FS}$)

 Calculate Probabilities for onlooker bees using Algorithm 4

If $rand(0,1) < Pro_i$

$$r \leftarrow r + 1$$

For each food source, i **do**

 Generate new individual v_i using Algorithm 6

 NM method (v_i)

$\hat{v}_{i,j} \leftarrow \text{BinaryConv}(v_{i,j})$ using Algorithm 5

 Evaluate $f(\hat{v}_i)$

 Select between $f(\hat{v}_i)$ and $f(\hat{x}_i)$ using greedy method

If $f(\hat{v}_i) < f(\hat{x}_i)$

$$x_i \leftarrow v_i$$

$$f(\hat{x}_i) \leftarrow f(\hat{v}_i)$$

$$trail(i) \leftarrow 0$$

Algorithm 3: Cont.

```

                                Else
                                trial(i) ← trial(i) + 1
                                End if
                                End For
                                End if
                                i ← (i + 1) mod FS
                                End while
Else
    For each food source, i do
        Generate new individual vi using Algorithm 6
        NM method (Lu)
                                u ∈ Lu
        Divide {Lu} into |Lu| equal chunks
                                Su ←  $\frac{\{L_u\}}{|L_u|}$ 
                                ∀Lui, i ∈ 1, 2, ..., |Lu|
        Txi ← Rank (Lui, Sui)
        Txi ← Delete least rank individual (Txi)
        vi ← celltomat {Txi}
    End For
End if
/*Scout Bee Phase*/
q = {i : trial(i) = max (trial)}
If trial(q) > limit
    Abandon the food source xi
    xq,j ← xmin,j ± rand(0, 1) * (xmax,j - xmin,j)
    x̂q,j ← BinaryConv(xq,j) using Algorithm 5
    For h = 1 to M do
        Evaluate the fitness of the population for a M number
of Objectives
                                fh ← fh(x̂q)
    End for
    trial(q) ← 0
End if
Add the new solution obtained to Zi
Non-Dominated Sort (Zi) using Algorithm 1
                                L ← Zi
Fitness Sharing (L) using Algorithm 2 // density estimation where L denotes dense population
around the individual i
Memorize the best solution obtained so far
iter ← iter + 1
}
Until iter = MI
Output: Optimal value of the objective function

```

Algorithm 4: Probability Computation

```

For i = 1 to FS, do
    Compute the probability Pij for the individual vi,j
    Proi ←  $\frac{fit_i}{\sum_{j=1}^{FS} fit_j}$ 
    fiti ←  $\begin{cases} \frac{1}{1+f_i}, & f_i \geq 0 \\ 1 + abs(f_i), & f_i < 0 \end{cases}$ 
End for

```

Algorithm 5: BinaryConv($x_{i,j}$)

```

For  $i = 1$  to FS do
  For  $j = 1$  to S do
     $bit(x_{i,j}) = \sin(2\pi x_{i,j} \cos(2\pi x_{i,j}))$ 
     $\hat{x}_{i,j} = \begin{cases} 1, & \text{if } bit(x_{i,j}) > 0 \\ 0, & \text{otherwise} \end{cases}$ 
  End for
End for

```

Algorithm 6: NM method (v_i)

```

Generate new food source  $v_i$  using modified NM technique
Let  $v_i$  denotes list of vertices
 $\zeta, \mu, \lambda$  and  $\zeta$  are the coefficients of reflection, expansion, contraction, and shrinkage
 $f$  is the objective function to be minimized
For  $i = 1, 2, \dots, n + 1$  vertices, do
  Arrange the values from lowest fit value  $f(v_1)$  to highest fit value  $f(v_{n+1})$ 
   $f(v_1) \leq f(v_2) \leq \dots \leq f(v_{n+1})$ 
  Compute mean for best two summits
   $v_m \leftarrow \sum \frac{v_i}{n}$ , where  $i = 1, 2, \dots, n$ 
  /*Likeness point  $v_r$ */
     $v_r \leftarrow v_m + \zeta(v_m - v_{n+1})$ 
  If  $f(v_1) \leq f(v_r) \leq f(v_n)$  then
     $v_n \leftarrow v_r$  and go to end condition
  End if
  /*Enlargement point  $v_e$ */
  If  $f(v_r) \leq f(v_1)$  then
     $v_e \leftarrow v_r + \mu(v_r - v_m)$ 
  End if
  If  $f(v_e) < f(v_r)$  then
     $v_n \leftarrow v_e$  and go to end condition

  Else
     $v_n \leftarrow v_r$  and go to end condition

  End if
  /*Reduction point  $v_c$ */
  If  $f(v_n) \leq f(v_r) \leq f(v_{n+1})$  then
    Compute outside reduction
     $v_c \leftarrow \lambda v_r + (1 - \lambda)v_m$ 
  End if
  If  $f(v_r) \geq f(v_{n+1})$  then
    Compute inside reduction
     $v_c \leftarrow \lambda v_{n+1} + (1 - \lambda)v_m$ 
  End if
  If  $f(v_r) \geq f(v_n)$  then
    Contraction is done among  $v_m$  and the best vertex among  $v_r$  and  $v_{n+1}$ .
  End if
  If  $f(v_c) < f(v_r)$  then
     $v_n \leftarrow v_c$  and go to end condition
  Else go to Shrinkage part
  End if
  If  $f(v_c) \geq f(v_{n+1})$  then
     $v_{n+1} \leftarrow v_c$  and go to end condition
  Else go to Shrinkage part
  End if

```

Algorithm 6: *Cont.*

```

/* Shrinkage part */
Shrink towards the best solution with new vertices
 $v_i \leftarrow \zeta v_i + v_1(1 - \zeta)$ , where  $i = 2, \dots, n + 1$ 
End condition
Arrange and rename the newly constructed simplex's summits according to their fit
values, then carry on with the reflection phase.
    
```

4. Experimental and Environment Setup

This section specifies the experimental structure of the proposed approach and other techniques. In addition, the projected outcomes with other methods are compared, to confirm the model's efficacy.

4.1. Experimental Setup

The proposed MBABC-NM algorithm to solve NSP and 0-1 knapsack problems, is demonstrated concisely in this division. The simulation is conducted on various optimization algorithms with similar environmental constraints, and the outcomes are analyzed. The technique proposed to handle NSP and 0-1 knapsack problems is implemented using a MATLAB 2018a tool under a Windows Intel I7 processor with 8GB of RAM. The experimental analysis will set the bounds of the formulated work. The parameters are considered based on the trial and error method. We used the standard dataset for both the NSP and the 0-1 knapsack problem. The compared algorithms are selected to ensure the performance of the formulated technique for the NSP in Table 1. The heuristic parameters and the consistent values are symbolized in Table 2.

Table 1. List of competitors' methods of comparing an NSP dataset for MBABC-NM.

Type	Method	Reference
M1	Multi-objective genetic algorithm: NSGA-II	Zhang et al., 2021 [23]
M2	Multi-objective cyber swarm optimization algorithm	Yin et al., 2013 [24]
M3	Multi-objective particle swarm optimization	Han et al., 2021 [25]
M4	Multi-objective ABC	Li et al., 2015 [26]

Table 2. Configuration parameters of MBABC-NM for experimental evaluation.

Type	Method
number of bees	100
maximum iterations	1000
initialization technique	binary
stop criteria	maximum iterations
run	20
heuristic	Nelder—Mead method
likeness factor	$\alpha > 0$
enlargement factor	$\gamma > 1$
reduction factor	$0 > \beta > 1$
shrinkage factor	$0 < \delta < 1$

4.2. Standard 0-1 Knapsack Problem Dataset

This work performs experiments on standard instances of the 0-1 knapsack problem from OR-Library to evaluate the performance of the proposed algorithm MBABC-NM. We used nine different instance classes to illustrate the outcomes of the proposed approach. Each problem suite is classified based on the number of knapsack constraints and object items used. A detailed description of the problem suite is discussed in Table 3. Table 3, column 2 describes the number of knapsacks constraints available in the corresponding problem suite. Column three represents the number of available object items within it, and

column four shows the known optimum solution provided by the standard OR-Library. The compared algorithms are selected to ensure the efficacy of the formulated model on the 0-1 knapsack problem, as shown in Table 4.

This section discussed the experimental setup and dataset description used for implementing the proposed algorithm and other compared techniques. Moreover, the performance of the proposed algorithm with other compared techniques is discussed in Section 5.

Table 3. The features of MKP datasets for MBABC-NM.

Instance	No. of Objectives	No. of Items
kn250_2	2	250
kn250_3	3	250
kn250_4	4	250
kn500_2	2	500
kn500_3	3	500
kn500_4	4	500
kn750_2	2	750
kn750_3	3	750
kn750_4	4	750

Table 4. List of competitors' techniques to associate MKP dataset for MBABC-NM.

Type	Method	Reference
M1	Pareto evolutionary algorithm	Luo et al., 2019 [27]
M2	GRASP	Yuan et al., 2021 [28]
M3	Genetic Tabu search for MKP	Alharbi et al., 2018 [29]
M4	ACO for MKP	Fidanova et al., 2020 [30]

5. Experimental Result Analysis and Discussion

5.1. Standard NSP Dataset

The experimental outcomes achieved by the MBABC-NM algorithm on solving the standard NSP dataset are presented in Tables 5 and 6. The performance of the proposed algorithm is compared with existing multi-objective algorithms listed in Table 1 for M1, M2, M3, and M4. The value present in the table specifies the ONGV value attained via the consistent system. The multi-objectives of the NSP are the minimization of cost, maximizing nurse preferences, and minimizing the deviation between the number of nurses required and the least numeral of nurses for the shift; day shift followed by night shift is not permissible. To legalize the proposed algorithm, we utilized 15 test cases of various sizes with multiple issues. It is proven that projected MBABC-NM accomplished maximum ONGV values for a maximum number of instances. The experimentation has been carried out on four different algorithms with the same simulation parameters.

Table 5 reviews the comparison and assessment of ONGV performance indicators attained by our proposed technique MBABC-NM associated with other methods, as shown in Table 1.

On comparing the mean values of ONGV for the NSP dataset, our proposed MBABC-NM outperforms existing algorithms for smaller datasets, with 14.99% against genetic NSGA, 59.67% against the cyber swarm, 28.70% against PSO, and 63.24% against the MABC algorithm. Our proposed MBABC-NM also outperforms existing algorithms for medium-sized datasets, with 84.75% against genetic NSGA, 23.12% against the cyber swarm, 60.43% against PSO, and 54.21% against the MABC algorithm. For larger-sized datasets, it achieved 43.15% against genetic NSGA, 44.27% against the cyber swarm, 25.14% against PSO, and 16.50% against the MABC algorithm.

Table 6 reviews the comparison and valuation of the SP performance indicator and shows the proposed MBABC-NM with another competitor's technique, as shown in

Table 1. Our proposed algorithm achieved minimized Euclidean distance among the Pareto solutions.

Table 5. Experimental result of NSP dataset in terms of ONGV.

Case	Type	Instance	MBABC-NM	M1	M2	M3	M4
C-1	N25	1	135	121	92	104	53
C-1	N25	7	132	125	86	106	63
C-1	N25	12	131	119	91	115	51
C-1	N25	19	128	119	88	101	50
C-1	N25	25	128	122	83	104	56
C-2	N25	2	143	118	86	99	67
C-2	N25	5	142	121	80	113	69
C-2	N25	9	136	124	85	116	69
C-2	N25	15	149	124	78	115	60
C-2	N25	27	146	124	79	99	63
C-3	N25	1	145	123	77	97	61
C-3	N25	3	150	125	82	97	65
C-3	N25	16	151	125	77	99	71
C-3	N25	27	146	121	91	113	73
C-3	N25	35	151	117	93	107	70
C-4	N25	5	139	122	92	98	63
C-4	N25	10	136	117	88	112	71
C-4	N25	25	150	120	78	111	65
C-4	N25	38	151	121	97	110	59
C-4	N25	41	135	122	79	99	72
C-5	N25	7	150	122	78	97	59
C-5	N25	11	127	118	92	107	70
C-5	N25	30	135	120	80	114	61
C-5	N25	42	135	121	91	104	71
C-5	N25	47	148	118	83	100	64
C-6	N50	1	192	40	90	109	73
C-6	N50	4	229	47	91	107	60
C-6	N50	12	222	35	87	125	73
C-6	N50	26	244	47	96	114	66
C-6	N50	29	223	41	87	126	76
C-7	N50	3	242	36	96	65	57
C-7	N50	6	248	42	90	60	60
C-7	N50	12	246	34	87	67	66
C-7	N50	26	233	36	88	62	65
C-7	N50	36	214	39	89	72	61
C-8	N50	4	251	43	95	55	71
C-8	N50	9	255	48	98	74	55
C-8	N50	15	249	34	97	65	58
C-8	N50	40	196	37	87	57	57
C-8	N50	47	228	47	88	57	73
C-9	N60	5	225	36	94	63	61
C-9	N60	10	210	49	89	60	58
C-9	N60	23	207	33	99	73	63
C-9	N60	29	203	41	91	65	72
C-9	N60	40	183	37	100	73	67
C-10	N60	6	196	49	94	76	58
C-10	N60	14	180	47	90	65	66
C-10	N60	20	208	49	95	54	64
C-10	N60	32	184	42	91	64	60
C-10	N60	41	218	39	92	69	63
C-11	N60	2	349	82	137	123	129
C-11	N60	8	374	98	151	126	121
C-11	N60	14	316	83	144	113	111
C-11	N60	20	364	96	145	118	118
C-11	N60	32	292	96	139	112	134

Table 5. Cont.

Case	Type	Instance	MBABC-NM	M1	M2	M3	M4
C-12	N60	3	327	98	140	121	115
C-12	N60	12	335	94	151	121	125
C-12	N60	19	351	98	145	120	124
C-12	N60	23	384	78	144	111	118
C-12	N60	34	289	98	140	121	107
C-13	N60	1	450	97	138	126	108
C-13	N60	4	438	87	141	118	109
C-13	N60	19	446	99	149	122	133
C-13	N60	29	347	81	152	126	109
C-13	N60	40	464	88	152	120	121
C-14	N60	5	335	100	141	108	121
C-14	N60	9	420	96	139	124	107
C-14	N60	15	400	90	146	116	115
C-14	N60	30	398	99	144	108	108
C-14	N60	43	483	94	140	117	110
C-15	N60	6	380	87	150	119	136
C-15	N60	15	433	87	141	123	125
C-15	N60	26	481	88	151	125	108
C-15	N60	35	477	90	151	124	136
C-15	N60	44	469	99	149	123	115

Table 6. Experimental results of NSP dataset in terms of SP.

Case	Nurse	Instance	MBABC-NM	M1	M2	M3	M4
C-1	N25	1	1.1×10^{-4}	3.9×10^{-4}	2.0×10^{-4}	2.1×10^{-4}	4.2×10^{-5}
C-1	N25	7	1.1×10^{-5}	8.8×10^{-4}	2.4×10^{-4}	9.2×10^{-5}	3.1×10^{-4}
C-1	N25	12	1.1×10^{-4}	9.8×10^{-4}	2.2×10^{-4}	9.4×10^{-5}	8.4×10^{-5}
C-1	N25	19	1.9×10^{-4}	7.0×10^{-4}	1.1×10^{-4}	8.1×10^{-6}	1.3×10^{-5}
C-1	N25	25	1.1×10^{-4}	2.6×10^{-4}	1.0×10^{-4}	5.6×10^{-4}	3.3×10^{-4}
C-2	N25	2	1.8×10^{-4}	3.1×10^{-4}	4.7×10^{-4}	5.5×10^{-4}	2.3×10^{-4}
C-2	N25	5	9.7×10^{-5}	6.1×10^{-4}	3.4×10^{-4}	1.1×10^{-4}	2.9×10^{-5}
C-2	N25	9	7.4×10^{-5}	8.2×10^{-4}	4.0×10^{-4}	3.4×10^{-4}	9.5×10^{-6}
C-2	N25	15	6.6×10^{-5}	9.0×10^{-5}	3.5×10^{-4}	2.5×10^{-4}	3.3×10^{-4}
C-2	N25	27	3.7×10^{-5}	4.3×10^{-5}	2.7×10^{-4}	1.6×10^{-4}	1.0×10^{-4}
C-3	N25	1	7.2×10^{-5}	7.6×10^{-4}	4.4×10^{-4}	7.4×10^{-5}	1.0×10^{-4}
C-3	N25	3	1.4×10^{-4}	9.0×10^{-4}	1.0×10^{-4}	4.7×10^{-4}	3.5×10^{-4}
C-3	N25	16	1.5×10^{-4}	3.2×10^{-4}	2.0×10^{-4}	1.7×10^{-4}	1.4×10^{-4}
C-3	N25	27	6.6×10^{-5}	2.3×10^{-4}	3.5×10^{-4}	2.3×10^{-4}	2.7×10^{-4}
C-3	N25	35	4.5×10^{-5}	5.6×10^{-4}	3.8×10^{-4}	1.9×10^{-4}	2.7×10^{-4}
C-4	N25	5	1.9×10^{-4}	5.1×10^{-4}	7.0×10^{-5}	5.6×10^{-4}	1.9×10^{-5}
C-4	N25	10	9.4×10^{-5}	9.1×10^{-4}	8.4×10^{-5}	5.0×10^{-4}	1.0×10^{-4}
C-4	N25	25	1.6×10^{-4}	6.9×10^{-4}	6.3×10^{-5}	5.4×10^{-4}	2.0×10^{-4}
C-4	N25	38	3.2×10^{-5}	4.6×10^{-4}	1.0×10^{-4}	7.8×10^{-5}	8.1×10^{-5}
C-4	N25	41	1.4×10^{-4}	5.3×10^{-4}	1.9×10^{-4}	7.2×10^{-5}	1.3×10^{-4}
C-5	N25	7	1.8×10^{-4}	3.8×10^{-5}	2.4×10^{-4}	9.9×10^{-5}	1.2×10^{-4}
C-5	N25	11	3.5×10^{-5}	4.3×10^{-4}	4.2×10^{-4}	2.4×10^{-4}	1.2×10^{-4}
C-5	N25	30	1.2×10^{-4}	2.3×10^{-4}	4.1×10^{-4}	4.9×10^{-4}	2.0×10^{-4}
C-5	N25	42	2.1×10^{-5}	2.5×10^{-4}	4.6×10^{-4}	1.0×10^{-4}	2.2×10^{-5}
C-5	N25	47	1.8×10^{-4}	2.9×10^{-4}	6.2×10^{-5}	1.4×10^{-4}	1.0×10^{-4}
C-6	N50	1	1.2×10^{-4}	4.4×10^{-4}	1.7×10^{-4}	5.3×10^{-4}	2.3×10^{-4}
C-6	N50	4	1.9×10^{-4}	6.8×10^{-4}	6.3×10^{-5}	2.5×10^{-5}	3.2×10^{-4}
C-6	N50	12	1.4×10^{-4}	3.2×10^{-5}	2.5×10^{-5}	2.5×10^{-4}	1.5×10^{-4}
C-6	N50	26	9.4×10^{-5}	8.4×10^{-4}	1.1×10^{-4}	2.6×10^{-4}	3.3×10^{-4}
C-6	N50	29	3.4×10^{-5}	2.2×10^{-4}	1.8×10^{-4}	2.9×10^{-4}	3.4×10^{-4}

Table 6. Cont.

Case	Nurse	Instance	MBABC-NM	M1	M2	M3	M4
C-7	N50	3	9.2×10^{-5}	3.5×10^{-4}	1.9×10^{-4}	2.4×10^{-4}	3.5×10^{-4}
C-7	N50	6	1.3×10^{-4}	4.5×10^{-4}	8.8×10^{-5}	2.5×10^{-4}	3.5×10^{-4}
C-7	N50	12	3.1×10^{-5}	4.0×10^{-4}	2.2×10^{-4}	8.0×10^{-5}	9.3×10^{-5}
C-7	N50	26	3.4×10^{-5}	1.8×10^{-4}	1.3×10^{-4}	4.3×10^{-4}	4.1×10^{-5}
C-7	N50	36	1.2×10^{-4}	2.1×10^{-4}	2.7×10^{-4}	3.7×10^{-4}	9.1×10^{-5}
C-8	N50	4	9.2×10^{-5}	1.3×10^{-4}	4.9×10^{-4}	3.5×10^{-4}	2.7×10^{-4}
C-8	N50	9	1.9×10^{-4}	1.2×10^{-5}	4.0×10^{-5}	8.6×10^{-5}	3.7×10^{-5}
C-8	N50	15	1.8×10^{-4}	1.2×10^{-4}	9.3×10^{-5}	4.3×10^{-4}	1.6×10^{-4}
C-8	N50	40	9.1×10^{-5}	5.7×10^{-4}	3.1×10^{-4}	2.1×10^{-4}	1.2×10^{-4}
C-8	N50	47	2.0×10^{-4}	8.6×10^{-4}	2.1×10^{-4}	2.3×10^{-5}	3.1×10^{-4}
C-9	N60	5	1.0×10^{-4}	9.6×10^{-4}	2.0×10^{-4}	3.5×10^{-5}	2.8×10^{-4}
C-9	N60	10	4.4×10^{-5}	6.2×10^{-4}	1.3×10^{-4}	4.5×10^{-4}	1.3×10^{-4}
C-9	N60	23	1.6×10^{-4}	5.5×10^{-4}	3.2×10^{-4}	3.6×10^{-4}	1.3×10^{-5}
C-9	N60	29	9.8×10^{-5}	6.6×10^{-4}	6.9×10^{-5}	2.3×10^{-4}	6.7×10^{-5}
C-9	N60	40	1.2×10^{-4}	2.0×10^{-4}	7.8×10^{-5}	2.6×10^{-4}	3.0×10^{-4}
C-10	N60	6	3.9×10^{-5}	2.0×10^{-5}	4.1×10^{-4}	5.4×10^{-5}	3.3×10^{-4}
C-10	N60	14	7.2×10^{-5}	2.3×10^{-4}	1.9×10^{-4}	1.4×10^{-4}	1.4×10^{-4}
C-10	N60	20	1.5×10^{-4}	8.8×10^{-4}	4.2×10^{-5}	2.1×10^{-4}	2.4×10^{-4}
C-10	N60	32	1.5×10^{-4}	8.4×10^{-4}	3.0×10^{-4}	7.0×10^{-5}	1.4×10^{-4}
C-10	N60	41	3.6×10^{-5}	7.0×10^{-4}	9.8×10^{-5}	2.4×10^{-4}	2.8×10^{-4}
C-11	N60	2	4.0×10^{-5}	9.4×10^{-4}	4.9×10^{-4}	4.1×10^{-4}	7.3×10^{-5}
C-11	N60	8	1.4×10^{-5}	4.5×10^{-4}	1.8×10^{-4}	3.8×10^{-4}	1.7×10^{-4}
C-11	N60	14	8.7×10^{-5}	2.1×10^{-4}	3.5×10^{-4}	2.7×10^{-4}	2.0×10^{-4}
C-11	N60	20	1.9×10^{-4}	4.3×10^{-4}	4.8×10^{-4}	2.3×10^{-4}	4.7×10^{-5}
C-11	N60	32	5.2×10^{-5}	2.3×10^{-4}	8.0×10^{-5}	5.4×10^{-4}	3.1×10^{-4}
C-12	N60	3	5.8×10^{-5}	1.1×10^{-4}	3.7×10^{-4}	4.5×10^{-5}	2.9×10^{-4}
C-12	N60	12	1.9×10^{-4}	9.1×10^{-4}	2.8×10^{-4}	2.0×10^{-4}	1.5×10^{-4}
C-12	N60	19	3.4×10^{-5}	7.5×10^{-4}	1.0×10^{-4}	3.3×10^{-4}	2.6×10^{-4}
C-12	N60	23	1.8×10^{-4}	3.4×10^{-4}	2.9×10^{-4}	9.3×10^{-5}	7.2×10^{-5}
C-12	N60	34	1.2×10^{-4}	5.2×10^{-4}	4.0×10^{-4}	5.3×10^{-5}	1.7×10^{-4}
C-13	N60	1	5.4×10^{-5}	9.8×10^{-4}	3.3×10^{-4}	3.6×10^{-4}	3.5×10^{-4}
C-13	N60	4	1.3×10^{-4}	6.7×10^{-4}	2.3×10^{-4}	3.5×10^{-4}	2.0×10^{-4}
C-13	N60	19	6.6×10^{-6}	7.5×10^{-4}	1.9×10^{-4}	1.4×10^{-5}	2.0×10^{-4}
C-13	N60	29	2.0×10^{-4}	4.9×10^{-4}	1.3×10^{-4}	4.3×10^{-4}	1.5×10^{-4}
C-13	N60	40	1.1×10^{-4}	5.8×10^{-4}	1.9×10^{-4}	1.3×10^{-4}	1.3×10^{-5}
C-14	N60	5	1.1×10^{-4}	7.5×10^{-4}	2.6×10^{-4}	5.0×10^{-4}	3.3×10^{-4}
C-14	N60	9	2.0×10^{-4}	3.9×10^{-4}	1.7×10^{-4}	3.0×10^{-4}	3.0×10^{-4}
C-14	N60	15	1.1×10^{-4}	1.9×10^{-4}	8.2×10^{-5}	1.3×10^{-4}	3.2×10^{-4}
C-14	N60	30	2.3×10^{-4}	6.5×10^{-4}	2.1×10^{-4}	1.1×10^{-4}	3.3×10^{-4}
C-14	N60	43	1.5×10^{-4}	8.1×10^{-4}	4.4×10^{-4}	5.4×10^{-4}	1.9×10^{-4}
C-15	N60	6	2.1×10^{-6}	5.7×10^{-5}	2.7×10^{-4}	5.5×10^{-4}	3.2×10^{-4}
C-15	N60	15	1.5×10^{-4}	3.5×10^{-4}	3.4×10^{-4}	1.4×10^{-5}	1.2×10^{-4}
C-15	N60	26	1.2×10^{-4}	7.7×10^{-4}	3.8×10^{-4}	1.6×10^{-5}	2.6×10^{-4}
C-15	N60	35	2.2×10^{-5}	1.3×10^{-4}	2.2×10^{-4}	8.1×10^{-5}	1.0×10^{-4}
C-15	N60	44	9.6×10^{-5}	8.4×10^{-5}	1.5×10^{-4}	1.6×10^{-4}	9.7×10^{-7}

Comparing the mean values of SP for the NSP dataset, our proposed MBABC-NM outperforms existing algorithms for smaller datasets, with 80.90% against genetic NSGA, 68.48% against the cyber swarm, 60.46% against PSO, and 43.86% against MABC algorithm. Our proposed MBABC-NM also outperforms existing algorithms for medium-sized datasets, with 79.55% against Genetic NSGA, 59.14% against the cyber swarm, 68.37% against PSO, and 57.84% against the MABC algorithm. For larger-sized datasets, it achieved 74.30% against Genetic NSGA, 53.72% against the cyber swarm, 63.22% against PSO, and 32.98% against the MABC algorithm.

5.2. Standard 0-1 Knapsack Problem

The experimental outcome achieved by the MBABC-NM algorithm on solving the standard 0-1 knapsack problem was presented in Tables 7 and 8. The performance of the proposed algorithm was compared with existing multi-objective meta-heuristic methods listed in Table 3. The values in the table specify the mean value attained for number of the reference solution and the total number of solutions using the corresponding algorithm. Our proposed MBABC-NM obtained better results for a maximum number of instances shown in [31].

Table 7. Experimental results of MKP dataset in terms of NRS and TNS.

Instance	NRS					TNS				
	MBABC-NM	M1	M2	M3	M4	MBABC-NM	M1	M2	M3	M4
kn250_2	288.45	125.77	120.09	3.77	61.9	304.96	156.16	201.62	194.21	198
kn250_3	549.19	191.68	183.63	0.69	92.2	663.75	207.34	376.82	1472.45	925
kn250_4	742.71	215.63	207.90	1.28	104.6	791.22	254.71	617.90	3958.62	2288
kn500_2	5019.36	1505.53	1500.14	1222.24	1361.2	5198.89	1543.83	1761.34	257.31	1009
kn500_3	6751.66	2986.63	2980.58	1827.03	2403.8	6935.97	3032.85	3601.36	2368.15	2985
kn500_4	17,156.62	4282.56	4277.53	2258.97	3268.2	17,255.48	5455.92	4930.68	5705.94	5318
kn750_2	18,236.52	4247.75	4240.96	3765.41	4003.2	20,515.01	6087.35	4525.30	6362.16	5444
kn750_3	33,682.95	8035.34	8029.33	5336.95	6683.1	34,520.30	9102.70	8297.50	7915.18	8106
kn750_4	58,129.46	11,307.37	11,299.73	6515.64	8907.7	60,293.70	13,065.30	11,648.42	6976.39	9312

Table 8. Experimental results of MKP dataset in terms of the reference solution and Davg.

Instance	R	Davg				
		MBABC-NM	M1	M2	M3	M4
kn250_2	320	2.10×10^{-4}	9.70×10^{-3}	3.20×10^{-3}	1.48×10^{-2}	7.50×10^{-3}
kn250_3	564	3.50×10^{-4}	1.50×10^{-3}	4.60×10^{-3}	2.02×10^{-2}	1.06×10^{-2}
kn250_4	778	1.00×10^{-4}	3.14×10^{-3}	3.10×10^{-3}	3.24×10^{-2}	6.32×10^{-3}
kn500_2	8844	7.20×10^{-4}	1.78×10^{-2}	4.50×10^{-3}	1.61×10^{-2}	2.36×10^{-2}
kn500_3	11978	6.00×10^{-4}	1.41×10^{-2}	2.20×10^{-3}	3.24×10^{-2}	1.28×10^{-2}
kn500_4	33374	2.50×10^{-3}	1.01×10^{-2}	3.60×10^{-3}	5.60×10^{-2}	3.00×10^{-3}
kn750_2	34890	6.40×10^{-3}	2.46×10^{-2}	6.80×10^{-3}	3.17×10^{-2}	2.15×10^{-2}
kn750_3	74504	9.50×10^{-3}	3.12×10^{-2}	7.80×10^{-3}	2.80×10^{-2}	9.80×10^{-2}
kn750_4	105161	7.20×10^{-3}	2.93×10^{-2}	1.32×10^{-2}	3.18×10^{-2}	1.01×10^{-2}

The experimentation has been carried out on four different algorithms with the same simulation parameters. The outcome attained by the proposed technique MBABC-NM and another competitor algorithm is presented in Table 7. The values in the table represent the number of reference solutions obtained for corresponding algorithms. TNS symbolizes an overall count of attained solutions, and NRS defines the number of reference solutions for the instances. Table 8 describes the experimental work gained by our projected technique MBABC-NM and another competitor algorithm. |R| represents several Pareto optimal or reference sets obtained for our proposed algorithm. Davg denotes the average distance between the non-dominated individual and the reference set.

Compared with an existing algorithm, our proposed algorithm MBABC-NM generated a maximum number of reference solutions from the total solutions, as illustrated in Figures 2 and 3. The mean value of NRS for a smaller dataset with 250 objects, our proposed algorithm had achieved 38% more than other competitor algorithms. The mean value of TNS was 40% against the competitor algorithm. For a medium dataset with 500 objects, the NRS was 29%, and the TNS was 21% against the competitor algorithm. For a larger dataset with 750 objects, our proposed algorithm MBABC-NM achieved 43% of NRS and 38% of TNS against the competitor algorithm.

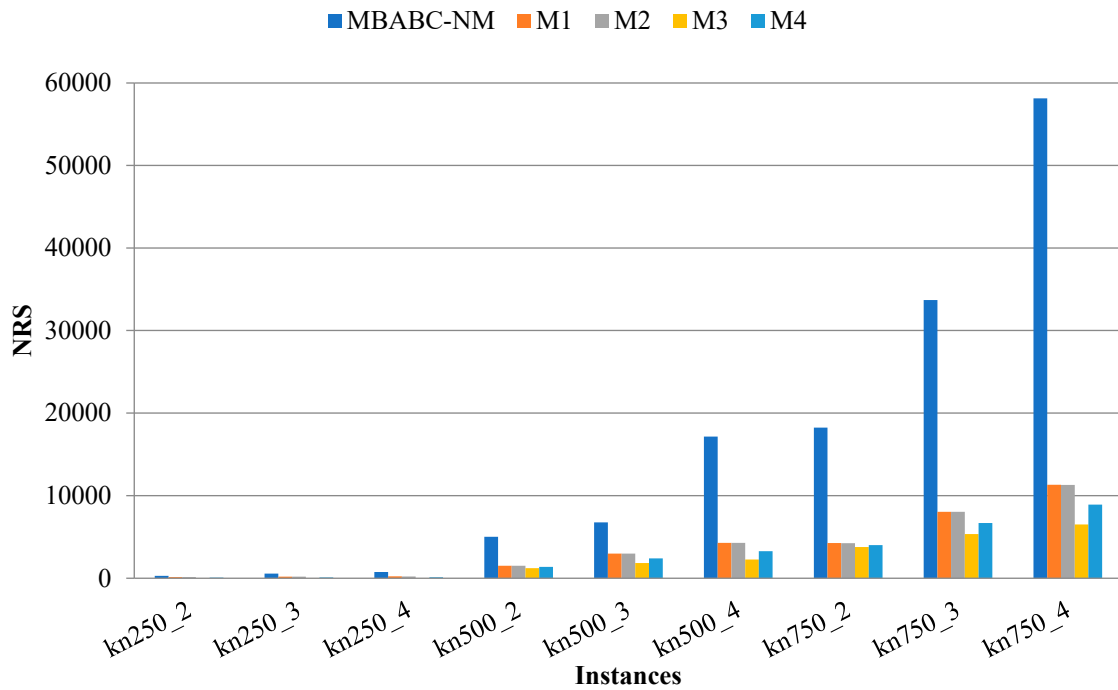


Figure 2. Performance of MBABC-NM w.r.t NRS.

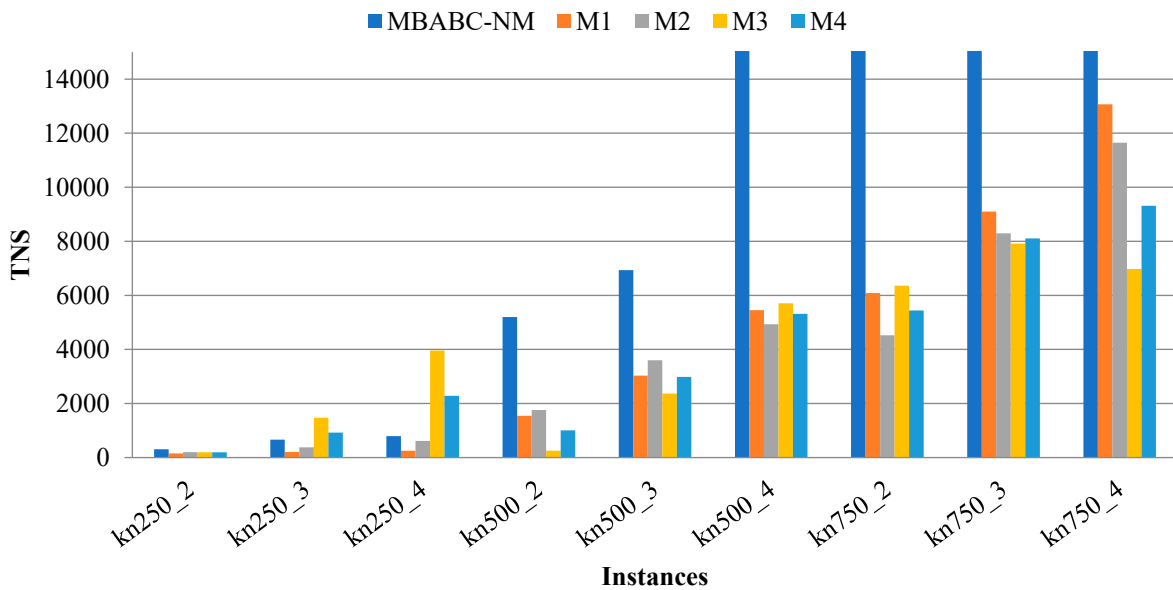


Figure 3. Performance of MBABC-NM with respect to TNS.

On comparing the mean values of D_{avg} for the 0-1 knapsack problem dataset, as shown in Figure 4, our proposed MBABC-NM outperforms existing algorithms for smaller datasets with 250 objects, as it achieved 95.07% against Local search, 93.94% against GRASP, 98.87% against Genetic Tabu search, and 97.30% against the ACO algorithm. For a medium-sized dataset with 500 objects, our proposed algorithm achieved 90.90% against Local search, 62.91% against GRASP, 96.34% against Genetic Tabu search, and 90.30% against the ACO algorithm. For a larger dataset with 750 objects, we achieved 72.85% against Local search, 16.90% against GRASP, 74.75% against Genetic Tabu search, and 82.17% against the ACO algorithm.

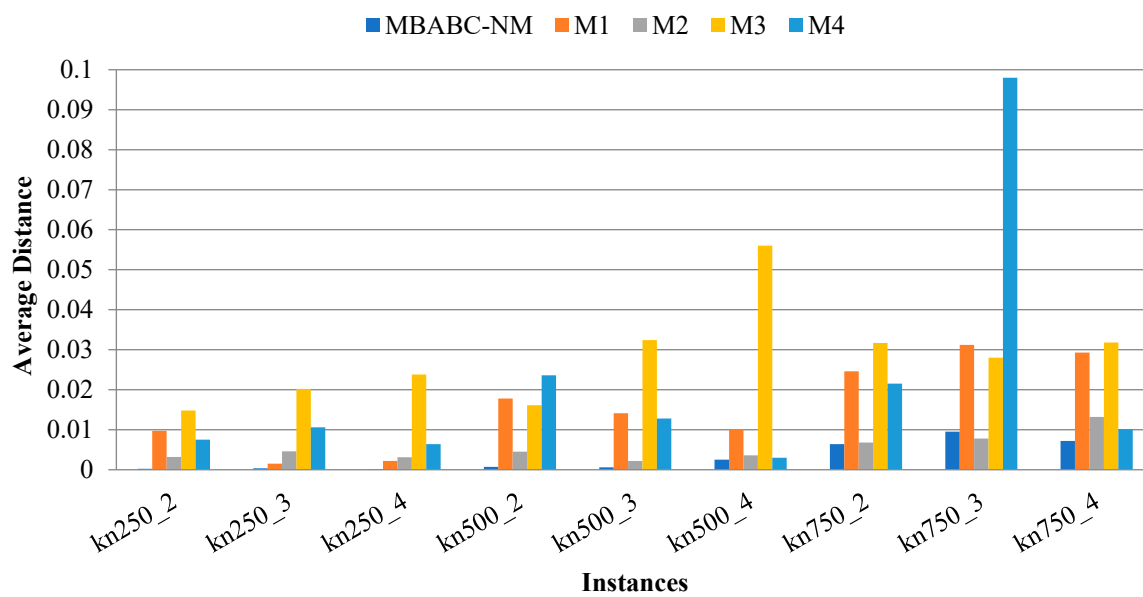


Figure 4. Performance of MBABC-NM with respect to average distance.

6. Conclusions

This paper proposed multi-objective BABC-NM on the multi-dimensional combinatorial problem. The proposed multi-objective BABC-NM with fitness sharing and modified non-dominated sorting algorithm have been incorporated. The experimentation is carried out on a MATLAB 2018a. In addition, we consider the experimental setup to assess the outcome of the projected approach MBABC-NM. The practical results and discussions on the obtained effects prove the significance of the projected work. In all three experimental methodology stages, the projected algorithm MBABC-NM's enhanced outcome was outclassed by attaining precise and satisfactory outcome factors. The projected multi-objective binary ABC with Nelder—Mead (MBABC-NM) is outstripped for all the test cases when associating with other standard classical algorithms. These studies indeed confirmed the competence of the projected algorithm in all perceptions. The proposed algorithm could be extended to handle more complex real-time optimization problems, including scheduling and resource allocation. In addition, the algorithm could be further optimized to reduce its computational complexity and improve its scalability.

Author Contributions: Conceptualization, M.R. (Muniyan Rajeswari); methodology, M.R. (Muniyan Rajeswari) and R.R.; validation, M.R. (Mamoon Rashid) and S.B.; formal analysis, K.S.B. and R.S.; writing—original draft preparation, M.R. (Muniyan Rajeswari); writing—review and editing, M.R. (Mamoon Rashid) and S.B., supervision, R.R. Funding Acquisition, S.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R195), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data in this research paper will be shared upon request with the corresponding author.

Acknowledgments: This research is supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R195), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goldberg, D.E.; Korb, B.; Deb, K. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Syst.* **1989**, *3*, 493–530.
2. Tharwat, A.; Houssein, E.H.; Ahmed, M.M.; Hassanien, A.E.; Gabel, T. MOGOA algorithm for constrained and unconstrained multi-objective optimization problems. *Appl. Intell.* **2018**, *48*, 2268–2283. [[CrossRef](#)]
3. Zitzler, E.; Deb, K.; Thiele, L. Comparison of multi-objective evolutionary algorithms: Empirical results. *Evol. Comput.* **2000**, *8*, 173–195. [[CrossRef](#)] [[PubMed](#)]
4. Von Lüken, C.; Barán, B.; Brizuela, C. A survey on multi-objective evolutionary algorithms for many-objective problems. *Comput. Optim. Appl.* **2014**, *58*, 707–756. [[CrossRef](#)]
5. Manzoor, A.; Javaid, N.; Ullah, I.; Abdul, W.; Almogren, A.; Alamri, A. An intelligent hybrid heuristic scheme for smart metering-based demand side management in smart homes. *Energies* **2017**, *10*, 1258. [[CrossRef](#)]
6. Mirjalili, S.Z.; Mirjalili, S.; Saremi, S.; Faris, H.; Aljarah, I. Grasshopper optimization algorithm for multi-objective optimization problems. *Appl. Intell.* **2018**, *48*, 805–820. [[CrossRef](#)]
7. Tamaki, H.; Kita, H.; Kobayashi, S. Multi-objective optimization by genetic algorithms: A review. In Proceedings of the IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 20–22 May 1996; IEEE: Piscataway, NJ, USA; pp. 517–522.
8. Zhang, Y.; Gong, D.-W.; Gao, X.-Z.; Tian, T.; Sun, X.-Y. Binary differential evolution with self-learning for multi-objective feature selection. *Inf. Sci.* **2020**, *507*, 67–85. [[CrossRef](#)]
9. Wang, Y.; Yang, Y. Particle swarm optimization with preference order ranking for multi-objective optimization. *Inf. Sci.* **2009**, *179*, 1944–1959. [[CrossRef](#)]
10. Mirjalili, S.; Saremi, S.; Mirjalili, S.M.; Coelho, L.D.S. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Syst. Appl.* **2016**, *47*, 106–119. [[CrossRef](#)]
11. Lv, L.; Zhao, J.; Wang, J.; Fan, T. Multi-objective firefly algorithm based on compensation factor and elite learning. *Future Gener. Comput. Syst.* **2019**, *91*, 37–47. [[CrossRef](#)]
12. Wang, C.-N.; Yang, F.-C.; Nguyen, V.T.T.; Vo Nhut, T.M. CFD analysis and optimum design for a centrifugal pump using an effectively artificial intelligent algorithm. *Micromachines* **2022**, *13*, 1208. [[CrossRef](#)] [[PubMed](#)]
13. Huynh, N.T.; Nguyen, T.V.T.; Nguyen, Q.M. Optimum Design for the Magnification Mechanisms Employing Fuzzy Logic-ANFIS. *CMC-Comput. Mater. Contin.* **2022**, *73*, 5961–5983.
14. Huynh, N.-T.; Nguyen, T.V.T.; Tam, N.T.; Nguyen, Q.-M. Optimizing Magnification Ratio for the Flexible Hinge Displacement Amplifier Mechanism Design. In *Proceedings of the 2nd Annual International Conference on Material, Machines and Methods for Sustainable Development (MMMS2020)*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 769–778.
15. Ramalingam, R.; Saleena, B.; Basheer, S.; Balasubramanian, P.; Rashid, M.; Jayaraman, G. EECHS-ARO: Energy-efficient cluster head selection mechanism for livestock industry using artificial rabbits optimization and wireless sensor networks. *Electron. Res. Arch.* **2023**, *31*, 3123–3144. [[CrossRef](#)]
16. Ramalingam, R.; Karunanidhy, D.; Alshamrani, S.S.; Rashid, M.; Mathumohan, S.; Dumka, A. Oppositional Pigeon-Inspired Optimizer for Solving the Non-Convex Economic Load Dispatch Problem in Power Systems. *Mathematics* **2022**, *10*, 3315. [[CrossRef](#)]
17. Kuppusamy, P.; Kumari, N.M.J.; Alghamdi, W.Y.; Alyami, H.; Ramalingam, R.; Javed, A.R.; Rashid, M. Job scheduling problem in fog-cloud-based environment using reinforced social spider optimization. *J. Cloud Comput.* **2022**, *11*, 99. [[CrossRef](#)]
18. Thirugnanasambandam, K.; Ramalingam, R.; Mohan, D.; Rashid, M.; Juneja, K.; Alshamrani, S.S. Patron-Prophet Artificial Bee Colony Approach for Solving Numerical Continuous Optimization Problems. *Axioms* **2022**, *11*, 523. [[CrossRef](#)]
19. Bao, C.; Xu, L.; Goodman, E.D.; Cao, L. A novel non-dominated sorting algorithm for evolutionary multi-objective optimization. *J. Comput. Sci.* **2017**, *23*, 31–43. [[CrossRef](#)]
20. Ye, T.; Si, L.; Zhang, X.; Cheng, R.; He, C.; Tan, K.C.; Jin, Y. Evolutionary large-scale multi-objective optimization: A survey. *ACM Comput. Surv.* **2021**, *54*, 1–34.
21. Luo, J.; Liu, Q.; Yang, Y.; Li, X.; Chen, M.-R.; Cao, W. An artificial bee colony algorithm for multi-objective optimization. *Appl. Soft Comput.* **2017**, *50*, 235–251. [[CrossRef](#)]
22. Salazar-Lechuga, M.; Rowe, J.E. Particle swarm optimization and fitness sharing to solve multi-objective optimization problems. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Edinburgh, UK, 2–5 September 2005; IEEE: Piscataway, NJ, USA; Volume 2, pp. 1204–1211.
23. Zhang, P.; Qian, Y.; Qian, Q. Multi-objective optimization for materials design with improved NSGA-II. *Mater. Today Commun.* **2021**, *28*, 102709. [[CrossRef](#)]
24. Yin, P.-Y.; Chiang, Y.-T. Cyber swarm algorithms for multi-objective nurse rostering problem. *Int. J. Innov. Comput. Inf. Control* **2013**, *9*, 2043–2063.
25. Han, F.; Chen, W.-T.; Ling, Q.-H.; Han, H. Multi-objective particle swarm optimization with adaptive strategies for feature selection. *Swarm Evol. Comput.* **2021**, *62*, 100847. [[CrossRef](#)]
26. Li, Y.; Huang, W.; Wu, R.; Guo, K. An improved artificial bee colony algorithm for solving multi-objective low-carbon flexible job shop scheduling problem. *Appl. Soft Comput.* **2020**, *95*, 106544. [[CrossRef](#)]
27. Luo, R.-J.; Ji, S.-F.; Zhu, B.-L. A Pareto evolutionary algorithm based on incremental learning for a kind of multi-objective multi-dimensional knapsack problem. *Comput. Ind. Eng.* **2019**, *135*, 537–559. [[CrossRef](#)]

28. Yuan, J.; Li, Y. Solving binary multi-objective knapsack problems with novel greedy strategy. *Memetic Comput.* **2021**, *13*, 447–458. [[CrossRef](#)]
29. Alharbi, S.T. A hybrid genetic algorithm with tabu search for optimization of the traveling thief problem. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 276–287. [[CrossRef](#)]
30. Fidanova, S. Hybrid Ant Colony Optimization Algorithm for Multiple Knapsack Problem. In Proceedings of the 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, India, 1–3 December 2020; IEEE: Piscataway, NJ, USA; pp. 1–5.
31. Beasley, J.E. OR-Library Collection of Test Data Sets for a Variety of OR Problems. World Wide Web. 2005. Available online: <http://people.brunel.ac.uk/mastjbjb/jeb/orlib/scpinfo.html> (accessed on 20 December 2022).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.