

Chaotic binary black widow optimization algorithm based on L-shaped transfer function to solve 0-1 knapsack problem

Hind Talaat Yaseen*, Ghalya Tawfeeq Basheer

*Department of Operations Research and Intelligent Techniques, College of Computer Sciences and Mathematics,
University of Mosul, Mosul, Iraq*

Abstract In this study, a binary chaotic Black Widow Optimization algorithm was introduced to solve the 0-1 knapsack problem. The original Black Widow Optimization algorithm, which was designed for continuous optimization, was modified to work with binary search spaces by introducing L-shaped transfer functions, chaotic initialization, and a greedy repair operator to maintain feasibility given the knapsack capacity constraint. Three transfer functions were analyzed to obtain a better binary version of the algorithm and balance exploration and exploitation. The proposed approach was tested on small-, medium-, and large-scale knapsack benchmark problems. The results of the experiments demonstrate that the improved algorithms achieve better solution quality, stability, convergence speed, and success rate than the basic BBWO-0 algorithm. The CBBWO-L2 and CBBWO-L4 algorithms consistently demonstrate high effectiveness and stability, particularly when applied to medium- and large-scale problems. The results indicate that the proposed method is an efficient metaheuristic approach for solving combinatorial-optimization problems.

Keywords 0–1 Knapsack Problem; Black Widow Optimization Algorithm (BWO); L-shaped transfer functions; Chaotic map

DOI: 10.19139/soic-2310-5070-4348

1. Introduction

The 0-1 Knapsack Problem is a well-known combinatorial optimization problem and a classical NP-hard problem. The knapsack problem has significant applications in investment decision-making, cargo loading, and resource allocation and is applied to many practical problems, including portfolio optimization, cutting stock problems, and cryptography.

The knapsack problem is NP-hard because it cannot be solved in polynomial time unless NP equals P. Currently, there are two algorithms for solving the 0-1 KP : deterministic algorithms, including branch-and-bound, dynamic programming, and backtracking algorithms, and deterministic algorithms based on these algorithms. Although deterministic algorithms are effective in solving small-scale 0-1 KP problems, they are disadvantaged by their inability to overcome the immense time consumption of solving large-scale 0-1 KP problems because they do not have polynomial time complexity. The second type is nondeterministic algorithms, primarily consisting of stochastic algorithms, approximation algorithms, such as meta-heuristic algorithms, and evolutionary algorithms. These algorithms do not guarantee that they will get an optimal solution but are extremely fast in terms of computational speed. Thus, the focus on nature-inspired algorithms was needed to obtain approximate solutions within a reasonable time [1, 2].

Meta-heuristic algorithms are uncomplicated, adaptable, and can be used to address problems with various objective functions, such as discrete, continuous, and mixed problems [3–5]. Meta-heuristic algorithms to solve

*Correspondence to: Hind Talaat Yaseen (Email: hindtalaat48@uomosul.edu.iq). Department of Operations Research and Intelligent Techniques, University of Mosul. Mosul, Iraq.

Knapsack Problem include genetic algorithm (GA) [6, 7], particle swarm optimization (PSO) [8], artificial fish swarm algorithm (AFSA) [9], flower pollination algorithm (FPA) [1], whale optimization algorithm (WOA) [10], cuckoo search algorithm (CS) [11], Monarch butterfly optimization (MBO) [3, 12], dragonfly algorithm (DF) [13], Honey Badger Algorithm (HBA) [14], snack optimization algorithm [15], Puma Optimizer (PO) [16], Harmony search (HS) [17], Light spectrum Optimizer (LSO) [18].

The Black Widow Optimization algorithm, introduced by Hayyolalam and Kazem in 2020 [19], represents a novel meta-heuristic approach designed to address continuous nonlinear optimization problems. The Black Widow Optimization (BWO) algorithm is derived from the distinctive mating behavior observed in black widow spiders. This algorithm has been employed in numerous studies to address a range of optimization problems [20]. A novel hybrid meta-heuristic algorithm has been developed, integrating black widow optimization with an iterated greedy algorithm, to address gene selection problems. [21] suggested a hybrid black widow that merges seagull optimization and deep learning to classify roads efficiently in UAV-aided intelligent transportation systems. [22] proposed an improved black widow optimization algorithm for feature selection. [23] proposed the optimization of hybrid renewable energy sources based on black widow optimization methods and uncertainty constraints [24]. Introducing a Binary Black Widow Optimization Algorithm to Tackle the Cell Formation Problem Considering Alternative Pathways and Machine Reliability.

The binary black widow optimization algorithm applies a transfer function to transform the continuous values produced by the algorithm into binary values, and thus can provide a binary BWO sufficient to strike a balance between exploration and exploitation. In this study, we applied the L-shaped transfer function and integrated it with the Black Widow Optimization algorithm to develop a new binary Black Widow Optimization algorithm. In the initialization stage, we incorporated chaotic maps with a greedy repair mechanism to enhance the algorithm's performance. The new chaotic binary black widow optimization algorithm was applied to solve the 0-1 knapsack problem.

The subsequent sections of this paper are structured as follows: Section 2 provides a detailed description of the mathematical model pertaining to the 0–1 knapsack problem. Section 3 introduces the Black widow Optimization algorithm. In Section 4, the L-shaped transfer function is presented. Section 5 introduces the proposed CBBWO algorithm. Section 6 introduces and analyzes the experimental findings of this study. The conclusions drawn from this study are outlined in Section 7.

2. 0-1 Knapsack Problem

The 0-1 knapsack problem has a single constraint that must be fulfilled in its solution. Specifically, the purpose of this problem is to determine the near-optimal set of items that can maximize the total profit and meet the knapsack capacity constraint. To solve this problem, the profit p_i and weight w_i of each item in the capacity (C) of the knapsack were provided. Subsequently, an optimization algorithm is used to identify a subset of these items that may yield the highest total profit without violating the knapsack capacity constraint. This problem can be mathematically modeled as follows [3, 25]:

$$\text{Maximize } f(x) = \sum_{i=1}^n p_i x_i \quad (1)$$

$$\begin{aligned} \text{subject to } & \sum_{i=1}^n w_i x_i \leq C \\ & x_i = \begin{cases} 1 & \text{if item } i \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (2)$$

where w_i denotes the weight of item i , p_i is the profit of the i th item, n is the number of items, C is the knapsack capacity, x is a binary decision vector used to determine the near-optimal set of items, and $f(x)$ is the objective function to be maximized.

3. Black widow Optimization Algorithm

In 2020, Hayyolalam and Kazem introduced a novel meta-heuristic algorithm, termed the Black Widow Optimization (BWO) algorithm. This algorithm is inspired by the distinctive mating behaviors and life cycles of black widow spiders. The essence of the algorithm resembles mating, sexual cannibalism, egg-laying, and sibling cannibalism. The primary objective is to address nonlinear optimization problems by effectively balancing exploration, which involves searching for new areas, and exploitation, which entails refining existing solutions.

The BWO algorithm is initiated with an initial population of spiders, where each spider represents a potential solution. In this early stage, spider pairs engage in reproduction to generate a new generation. During or after copulation, female black widows consume males. Sperm are stored in the sperm thecae and subsequently deposited into egg sacs. Spiderlings emerge from these egg sacs approximately 11 days after deposition. They remain on the maternal web for a few days to a week, during which sibling cannibalism may take place. Eventually, they are dispersed by the wind [19].

The BWO algorithm has four primary stages: initialization, procreation, cannibalism, and mutation. The procreate stage focuses on discovering the quest domain by generating numerous offspring. Next, by eliminating incorrect solutions in the cannibalism phase, the BWO algorithm can rapidly move towards the correct solution. The mutation process is such that the exploitation and discovery phases are conducted in the correct sequence [20]. In addition to the existence of a stage of cannibalism, which is one of the main aspects that allows for the sifting of weak solutions at an earlier stage, resulting in accelerated convergence and preventing local optima [19].

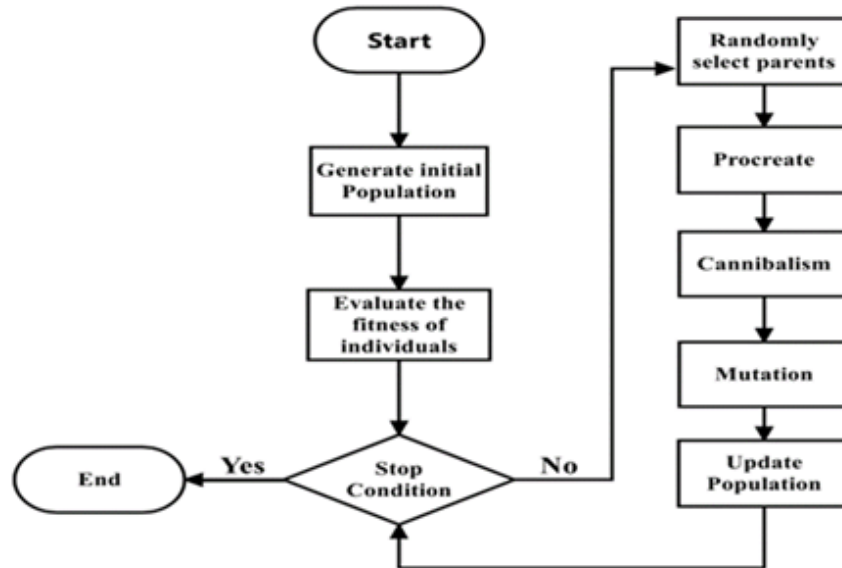


Figure 1. Flowchart of the BWO [19].

3.1. Algorithm steps (BWO stages)

Fig. (1) shows the flowchart of the BWO algorithm, which follows a sequence of four biological-inspired steps.

(1) Initialization

Similar to any population-based metaheuristic, the first step is to generate an initial population. The black widow is an array of $1 \times N_{\text{var}}$ in an N_{var} -dimensional optimization problem, which is the solution to this problem. The following is the definition of this array:

$$\text{Window} = [w_1, w_2, \dots, w_{N_{\text{var}}}] \quad (3)$$

where N_{var} is the number of decision variables in the optimization problem and W is the widow population. The widow's fitness is evaluated by applying the fitness function f to her specific position $[w_1, w_2, \dots, w_{N_{\text{var}}}]$:

$$\text{Fitness} = f(\text{window}) = f(w_1, w_2, \dots, w_{N_{\text{var}}}) \quad (4)$$

The initial population of spiders is used to generate a candidate widow matrix of size $(N_{\text{pop}}, N_{\text{var}})$ to initiate the optimization algorithm. The female black widow preys on the male black widow prior to or during mating, and pairs of parents are chosen randomly to perform the procreation part of the mating process.

(2) Procreate (Mating)

Black widows exhibit a unique mating behavior that produces a new generation. Parents are randomly selected to participate in the procreation process. Black widows in nature produce approximately 1000 eggs, but only the strongest of them survive. To replicate this behavior, an array referred to as alpha is developed whenever the widow array consists of random numbers. Subsequently, the offspring are generated using Eq. (5).

$$\begin{cases} ch1 = \alpha \times X_1 + (1 - \alpha) \times X_2, \\ ch2 = \alpha \times X_2 + (1 - \alpha) \times X_1 \end{cases} \quad (5)$$

In Eq. (5), X_1, X_2 are the parents and ch_1, ch_2 , are the children. This is repeated $N_{\text{var}}/2$ times, and randomly chosen numbers should not be duplicated. It is necessary to note that X_1 should be different from X_2 . Finally, both offspring and parents undergo cannibalization, and the survivors are organized within a matrix and ranked according to their cost values.

(3) Cannibalism

Cannibalism can be categorized into three distinct types:

- (a) **Sexual Cannibalism:** The female (stronger fitness) consumes the male during or after mating.
- (b) **Sibling Cannibalism:** Strong offspring consume their weaker siblings based on a Cannibalism Rating (CR).
- (c) **Maternal cannibalism :** is a prevalent form of cannibalism characterized by offspring consuming their mothers when they possess greater fitness.

(4) Mutation

This stage involves determining the number of populations to mutate through the mutation rate (MR), which is constant. Widows in this subpopulation experience a positional exchange between two of their decision variables, thereby creating a new widow. Fig. (2) illustrates the mutation methods.



Figure 2. Mutation method.

As in most evolutionary algorithms, there are three convergence criteria, which are looked at in BWO: (a) the number of iterations, which is set in advance; (b) no change in the fitness value of the strongest widow in many iterations; and (c) the achievement of the required degree of accuracy. The major steps of the BWO are summarized in the pseudo-code presented in Algorithm 1.

Algorithm 1 Black Widow Optimization Algorithm (BWO)

Require: Number of population N_{var} , Maximum number of iteration Maxiter, Procreating rate P_p , Cannibalism rate C_R , Mutation rate M_R

Ensure: Optimal solution found

```

1: Initialization:
2: Randomly initialize the population of black widows using Eq. (3)
3: Compute the fitness value of each solution using Eq. (4)
4: while  $t < \text{Maxiter}$  do
5:   Determine the number of reproductions,  $n_r$ , according to  $P_p$ 
6:   Choose the parent population of size  $n_r$ 
7:   for  $i = 1$  to  $n_r$  do
8:     Randomly select two solutions as parents from the  $n_r$  parents
9:     Generate two children using Eq. (5)
10:    Destroy the fathers
11:    Exclude a few children according to the cannibalism rate  $C_R$ 
12:    Rescue the remaining children and mothers into a new matrix as the new generation
13:  end for
14:  Determine the number of mutations  $n_m = n_r \times M_R$ 
15:  for  $i = 1$  to  $n_m$  do
16:    Select a solution from among the remaining children
17:    Randomly mutate one of the elements of the chosen solution to create a new solution
18:  end for
19:   $t \leftarrow t + 1$ 
20: end while
21: return the best solution found

```

4. Transfer Function

Transfer functions are important in binary metaheuristic methods, as the search operators are usually designed in the continuous domain, whereas optimization problems like the 0-1 knapsack problem, have binary decision variables. The main use of a transfer function is to convert continuous position values into probabilities that are used to determine whether a binary variable is 0 or 1.

The logarithmic transfer function is used in the proposed BBWO method to perform this mapping. The transfer function is the relationship between the binary representation of the knapsack problem and the continuous search function of the Black Widow Optimization method. The transfer function influences the probability of bit transitions and, therefore, the trade-off between exploration and exploitation. An efficient transfer function can be used to improve the algorithm's escape from local optima, avoid premature convergence, and enhance the diversity of the population. Consequently, the chosen transfer function is one of the most important parts of the proposed method, as its properties significantly influence the performance of the binary optimization process.

In 2025, (Seyma Gules et al.) introduced a novel transfer function for feature selection, the logarithmic transfer function. The results indicate that the bWSO-log method is superior or competitive in terms of the number of features selected and the classification accuracy. The results demonstrate the effectiveness of the proposed logarithmic function, as represented by the following formula [26]:

$$\text{logarithmic} = \log_2(|x| + 1) \quad (6)$$

To solve the 0-1 knapsack problem, a novel class of transfer functions called L-shaped transfer functions using the logarithmic function was introduced in 2025 [2]. Four L-shaped transfer functions were introduced, and the results revealed that LBFLA2 and LBFLA4 are more beneficial and competitive for solving the 0-1 knapsack problem. The L-shaped transfer function is listed in Table 1.

Table 1. Logarithmic transfer function [2].

L-Shaped transfer function	
L1	$L_1(x) = \log_2 \left(\frac{1}{2A}x + \frac{3}{2} \right)$
L2	$L_2(x) = \log_3 \left(\frac{1}{A}x + 2 \right)$
L3	$L_3(x) = \log_4 \left(\frac{3}{2A}x + \frac{5}{2} \right)$
L4	$L_4(x) = \log_5 \left(\frac{2}{A}x + 3 \right)$

In Fig.3, the curves of L1 to L4 within the range $[-A, A] = [-5, 5]$ are shown, clearly indicating that they share the same shape and resemble the tilted letter “L,” which is why these four new transfer functions are referred to as *L-shaped* transfer functions.

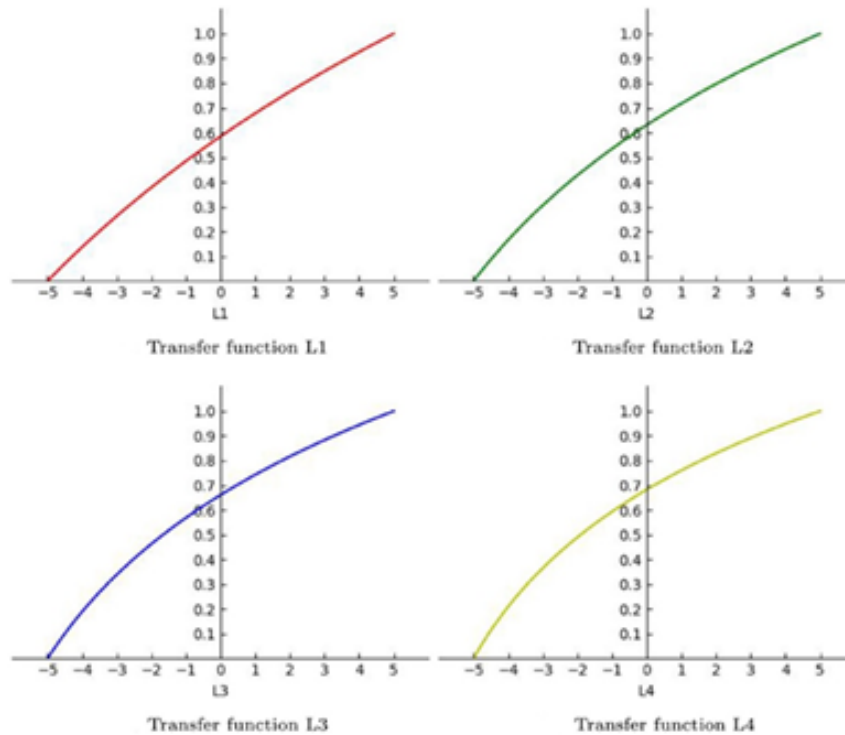


Figure 3. Curves of L1 to L4 transfer functions [2].

In this study, three different formulas for the logarithmic transfer function were used: The first and second formulas are L2 and L4, which were proposed in 2025 by [Yichao He et al.] [2]. The researcher [Yichao He et al.] tested the 4 transfer functions that he proposed in his research, and compared them with a group of transfer functions U, T, S, V. He has found that among these transfer functions, the best ones which solve 0-1 Knapsack Problem are L2 and L4. Hence, L2 and L4 transfer functions were employed in this research as they are proved to be effective when compared to other transfer functions. The third formula is shown in Eq. (6), which was proposed in 2025 by (Seyma Gules et al.) [26].

5. Proposed Algorithm

In this study, a chaotic binary black widow optimization algorithm for the 0-1 knapsack problem with logarithmic transfer functions is proposed. The first version of the Black Widow Optimization (BWO) cannot be directly

applied to binary search spaces because it was originally developed for continuous optimization problems. This restriction is overcome using binary adaptation, which involves adding a transfer function that maps the continuous search positions to binary variables. Logistic chaotic initialization in the initial stages of the search enhances population variety and increases exploration capacity. Furthermore, a repair operator is used to ensure that all solutions generated meet the knapsack capacity constraints.

The proposed method has four main stages: population initialization, procreation, cannibalism, and mutation. In the mutation phase, the suggested transfer functions are applied to produce a continuous perturbation first, and then the continuous perturbation is converted to a binary decision. Thus, the algorithm can explore the binary search space and balance the exploration and exploitation. The algorithm consists of chaotic initialization, binary representation, mutation using a transfer function, and a repair operator. All these components contribute to the search capabilities of the algorithm and the quality of the solutions generated for the combinatorial optimization problems.

5.1. Chaotic Initialization

In chaos theory, chaotic maps are used to represent the chaotic behavior of nonlinear systems. Sayed et al. (2018) define chaotic maps as being described by particles that are allowed to flow over different nonlinear, dynamic maps but don't necessarily have a well-defined and regular traveling path. The chaos technique was employed to enhance the quality of the global optimum search and prevent entrapment in local optimum solutions. Consequently, many optimization applications have utilized chaos [27–29].

Chaos can be used to enhance the 0-1 knapsack problem, which is an optimization problem defined within the range [0, 1]. This study examines the application of a logistic chaotic map to enhance the ability of the binary Black Widow algorithm to avoid local optima and improve the convergence speed. Eq. (7) demonstrates the logistic chaotic map [30]:

$$x_{i+1} = rx_i(1 - x_i) \quad (7)$$

where $r = 4$ is the growth rate and $x_i \in [0, 1]$ is the ratio of the population in generation i and Algorithm 2 explains chaotic initialization.

Algorithm 2 Chaotic Initialization

```

1: for  $i = 1$  to  $N$  : All black widows do
2:   for  $j = 1$  to  $n$  : Dimensions of each black widow do
3:      $x(i) = r \cdot x(i) \cdot (1 - x(i))$ 
4:   end for
5:   if  $x(i) \geq 0.5$  then
6:     Population( $i, j$ ) = 1
7:   else
8:     Population( $i, j$ ) = 0
9:   end if
10: end for

```

5.2. Fitness Function

The knapsack problem can be formulated as below with the use of the penalty function:

$$\min \Phi(x) = -f(x) + \lambda \max \left(0, \sum_{i=1}^n w_i x_i - C \right) \quad (8)$$

where λ represents the penalty coefficient. In this study, the parameter was set to 10^{16} for all test functions. The penalty function is defined in Algorithm 3.

Algorithm 3 Penalty function

Input: x_i
 Calculate total weight: $\sum_{i=1}^n w_i x_i$
if $\sum_{i=1}^n w_i x_i \leq C$ **then**
 $\Phi(x) = -f(x)$
else
 $\Phi(x) = -f(x) + \lambda (\sum_{i=1}^n w_i x_i - C)$
end if

5.3. Repair Operator

Because BBWO is a randomized method and the 0-1 knapsack problem is a constrained optimization problem, employing the BBWO approach to solve 0-1KP may result in infeasible solutions that violate the constraint of. The GRO algorithm, which employs a greedy approach to fix and enhance potential solutions to workable solutions, was used to address this situation. Repair and improvement are the two components of GRO. In the repair phase, when is infeasible, the item with the lowest profit density p_j/w_j is be removed from the knapsack until is feasible. To enhance the quality of the feasible solution, each item is sequentially incorporated into the knapsack, following the order of decreasing profit-to-weight ratio p_j/w_j during the improvement phase. If an item does not exceed the capacity limit, it is placed in a knapsack. If so, alternative items are considered until all possibilities that have not been packed are exhausted. The repair operator is explained in Algorithm 4.

Algorithm 4 Greedy Repair Operator

Repair phase
Input: infeasible solution of x_i
for $i = 1$ to n **do**
 /* Dimension of each black widow */
 Calculate $R = \frac{p_i}{w_i}$ for each item
end for
 Sort R in ascending order with its corresponding item
while $\sum_{i=1}^n w_i x_i > C$ **do**
 Remove items with the lowest R
end while
Output: feasible solution x_i

Improve phase
Input: feasible solution from repair phase
for $i = 1$ to n **do**
 /* Dimension of each black widow */
 Calculate $R = \frac{p_i}{w_i}$ for each item
end for
 Sort R in descending order with its corresponding item
while $\sum_{i=1}^n w_i x_i < C$ **do**
 Add items with the highest R
end while
Output: improved feasible solution x_i

Table 2. Parameters values of BWO algorithm

Parameter	Values
N_{var}	Population size = 50
Maxiter	Maximum number of iterations = n (n is the number of items in KP)
Pp	Procreating percentage = 0.7
CR	Cannibalism rate (CR) = 0.2
MR	Mutation rate = 0.05

6. Computational results

In this section, four algorithms—Binary Black Widow Optimization with no transfer function (BBWO-0), Binary Chaotic Black Widow Optimization based on L2 transfer function (CBBWO-L2), Binary Chaotic Black Widow Optimization based on L4 transfer function (CBBWO-L4), and Binary Chaotic Black Widow Optimization based on log transfer function (CBBWO-Log)—are applied to the 0-1 knapsack problem. This study investigated knapsack problems across three distinct scales: small, medium, and large knapsack problems. In total, 30 iterations of the four algorithms examined in this study were executed for each problem using MATLAB 2023. To measure the performance of the improved algorithms in terms of some statistical measures: best, mean, worst, and standard deviation. (standard deviation), average time, average iterations, FE (average number of objective function evaluations), and success rate achieved in the runs. The optimal values are shown in the tables and are obtained by solving the knapsack problem dynamically. Table 2 displays The BBWO algorithm parameters, and for a fair comparison, the parameters were set for all algorithms and all iterations. The success rate of the algorithm was computed using the following equation:

$$\text{Success Rate (SR)} = \left(\frac{\text{Number of Successful Runs}}{\text{Total Number of Runs}} \right) \times 100\% \quad (9)$$

6.1. Small scale 0-1 knapsack problem

The performance of the improved algorithm was tested using ten small-scale 0 – 1 knapsack problems (kp1–kp10) from [31]. The dimensions of this example were 4–23. Table 3 displays the comparative outcomes of all the algorithms used to solve (kp1–kp10). Table 3 shows that all the algorithms (BBWO-0, CBBWO-L2, CBBWO-L4, and CBBWO-Log) can find the optimal solution for the small-size 0-1 knapsack problem. The proposed algorithms yielded the same results as the standard BBWO-0 in terms of the best, worst, mean, and standard deviation values. The substantial discrepancy between the performance of the proposed methods and total BBWO-0 was unforeseen, given the relatively small number of items.

6.2. Medium scale 0-1 knapsack problem

To further evaluate the efficacy of the proposed techniques, ten medium sized 0-1 knapsack problems (denoted as kp11 to kp20) from [31] were considered. The problems contain (30-75) items. Table 4 presents the comparison results of each algorithm for solving the kp11–kp20 problems.

The performance of BBWO-0 with the proposed CBBWO algorithms on problems (KP11-KP20) is compared in Table 4. The results clearly show that the improved algorithms, particularly CBBWO-L2 and CBBWO-L4, are able to obtain better solution quality, stability and reliability than the original BBWO-0 algorithm. For most of the test cases, both variants converged to the optimal objective value with a standard deviation of zero or close to zero, indicating good convergence behavior and repeatability of the results. By contrast, BBWO-0 typically yielded a poorer mean solution quality, higher variance and lower success rates, and even needed the maximum number of function evaluations to converge in some cases. In this study, CBBWO-Log also performed better in some cases, but its performance was less consistent than the other two CBBWO-L2 and CBBWO-L4. The table clearly shows

Table 3. Results small-scale 0-1 knapsack problem for all improved algorithms

Problem	D	capacity C	Optimal	Algorithm	Best	Worst	Mean	Std.	S.R%
KP1	4	20	35	BBWO-0	35	35	35	0	100%
				CBBWO-L2	35	35	35	0	100%
				CBBWO-L4	35	35	35	0	100%
				CBBWO-Log	35	35	35	0	100%
KP2	4	11	23	BBWO-0	23	23	23	0	100%
				CBBWO-L2	23	23	23	0	100%
				CBBWO-L4	23	23	23	0	100%
				CBBWO-Log	23	23	23	0	100%
KP3	5	80	130	BBWO-0	130	130	130	0	100%
				CBBWO-L2	130	130	130	0	100%
				CBBWO-L4	130	130	130	0	100%
				CBBWO-Log	130	130	130	0	100%
KP4	7	50	107	BBWO-0	107	107	107	0	100%
				CBBWO-L2	107	107	107	0	100%
				CBBWO-L4	107	107	107	0	100%
				CBBWO-Log	107	107	107	0	100%
KP5	10	269	295	BBWO-0	295	295	295	0	100%
				CBBWO-L2	295	295	295	0	100%
				CBBWO-L4	295	295	295	0	100%
				CBBWO-Log	295	295	295	0	100%
KP6	10	60	52	BBWO-0	52	52	52	0	100%
				CBBWO-L2	52	52	52	0	100%
				CBBWO-L4	52	52	52	0	100%
				CBBWO-Log	52	52	52	0	100%
KP7	15	375	481.07	BBWO-0	481.07	481.07	481.07	0	100%
				CBBWO-L2	481.07	481.07	481.07	0	100%
				CBBWO-L4	481.07	481.07	481.07	0	100%
				CBBWO-Log	481.07	481.07	481.07	0	100%
KP8	20	879	1025	BBWO-0	1025	1025	1025	0	100%
				CBBWO-L2	1025	1025	1025	0	100%
				CBBWO-L4	1025	1025	1025	0	100%
				CBBWO-Log	1025	1025	1025	0	100%
KP9	20	878	1024	BBWO-0	1024	1024	1024	0	100%
				CBBWO-L2	1024	1024	1024	0	100%
				CBBWO-L4	1024	1024	1024	0	100%
				CBBWO-Log	1024	1024	1024	0	100%
KP10	23	10000	9767	BBWO-0	9767	9767	9767	0	100%
				CBBWO-L2	9767	9767	9767	0	100%
				CBBWO-L4	9767	9767	9767	0	100%
				CBBWO-Log	9767	9767	9767	0	100%

that the proposed modifications enhance the robustness and efficiency of the original algorithm. The success rate was below 100%, for problems kp15, kp18, and kp19; however, the performance of the proposed algorithm was better than that of the other algorithms.

The results for each KP problem (see Fig. 4–13) are in the form of boxplots, which show results similar to those in Table 4. The proposed algorithms, CBBWO-L2 and CBBWO-L4, exhibited good accuracy and stability compared to BBWO-0. For most of the cases, these methods were very close to the optimal solution with only a small deviation, while BBWO-0 had a higher deviation, lower success rate, and lower convergence. In some instances, CBBWO-Log demonstrated superior performance compared to CBBWO-L2 and CBBWO-L4; however, this was not consistently observed in all cases.

Table 4. Results medium-scale 0-1 knapsack problem for all improved algorithms

Problem	D	Optimal	Algorithm	Best	Worst	Mean	Std.	Average Time	Average Iterations	FE	S.R%
KP11	30	1437	BBWO-0	1437	1410	1426.9	9.0909	0.04772	82.33	4116	33.33%
			CBBWO-L2	1437	1437	1437	0	0.0045	1	50	100%
			CBBWO-L4	1437	1437	1437	0	0.00373	1	50	100%
			CBBWO-Log	1437	1437	1437	0	0.00579	1	50	100%
KP12	35	1689	BBWO-0	1689	1658	1678.4	7.5509	0.065103	95.23	4761	13.33%
			CBBWO-L2	1689	1689	1689	0	0.03296	21.87	1093	100%
			CBBWO-L4	1689	1689	1689	0	0.04038	26.20	1310	100%
			CBBWO-Log	1689	1684	1684.1	0.9129	0.10504	97.17	4858	3.33%
KP13	40	1821	BBWO-0	1821	1771	1798.7	14.5737	0.05582	97.77	4888	6.67%
			CBBWO-L2	1821	1821	1821	0	0.03039	16.83	841	100%
			CBBWO-L4	1821	1821	1821	0	0.041087	23	1150	100%
			CBBWO-Log	1817	1817	1817	0	0.11696	100	5000	0%
KP14	45	2033	BBWO-0	2020	1954	1989.2	18.1999	0.05915	100	5000	0%
			CBBWO-L2	2033	2033	2033	0	0.00449	1	50	100%
			CBBWO-L4	2033	2033	2033	0	0.00388	1	50	100%
			CBBWO-Log	2033	2033	2033	0	0.00363	1	50	100%
KP15	50	2440	BBWO-0	2434	2335	2381.7	25.2036	0.05596	100	5000	0%
			CBBWO-L2	2440	2438	2439.9	0.3651	0.05821	36.93	1846	96.67%
			CBBWO-L4	2440	2438	2439.8	0.6103	0.05453	36.23	1811	90%
			CBBWO-Log	2440	2438	2438.06	0.3651	0.11291	99.33	4966	3.33%
KP16	55	2651	BBWO-0	2617	2540	2577.6	17.3473	0.05711	100	5000	0%
			CBBWO-L2	2651	2651	2651	0	0.00396	1	50	100%
			CBBWO-L4	2651	2651	2651	0	0.00433	1	50	100%
			CBBWO-Log	2651	2651	2651	0	0.00423	1	50	100%
KP17	60	2917	BBWO-0	2877	2755	2806.2	34.6996	0.06064	100	5000	0%
			CBBWO-L2	2917	2917	2917	0	0.00422	1	50	100%
			CBBWO-L4	2917	2917	2917	0	0.00406	1	50	100%
			CBBWO-Log	2917	2917	2917	0	0.00407	1	50	100%
KP18	65	2818	BBWO-0	2762	2656	2708.1	28.044	0.06168	100	5000	0%
			CBBWO-L2	2818	2817	2817.8	0.4068	0.08151	50.53	2526	80%
			CBBWO-L4	2818	2817	2817.8	0.3457	0.08734	53.27	2663	86.67%
			CBBWO-Log	2818	2817	2817.06	0.2537	0.12467	95.73	4786	6.67%
KP19	70	3223	BBWO-0	3181	3000	3096.2	36.338	0.06118	100	5000	0%
			CBBWO-L2	3223	3221	3222.6	0.7581	0.07063	39.73	1986	83.33%
			CBBWO-L4	3223	3221	3222.6	0.8137	0.08423	49.27	2463	80%
			CBBWO-Log	3220	3220	3220	0	0.13727	100	5000	0%
KP20	75	3614	BBWO-0	3526	3348	3436.6	48.8428	0.07015	100	5000	0%
			CBBWO-L2	3614	3614	3614	0	0.00455	1	50	100%
			CBBWO-L4	3614	3614	3614	0	0.00468	1	50	100%
			CBBWO-Log	3614	3614	3614	0	0.00512	1	50	100%

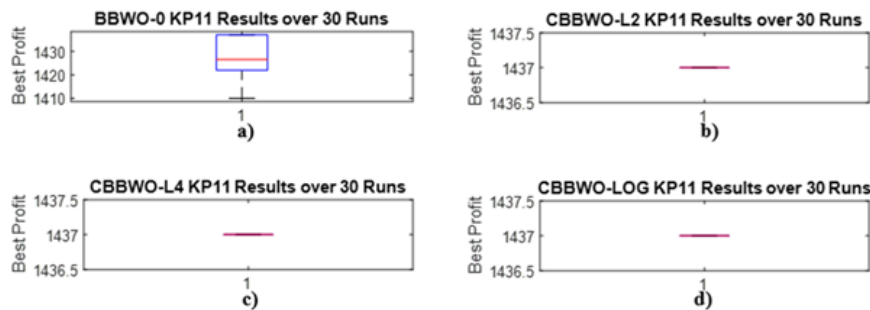


Figure 4. The boxplot of KP11 for all algorithms.

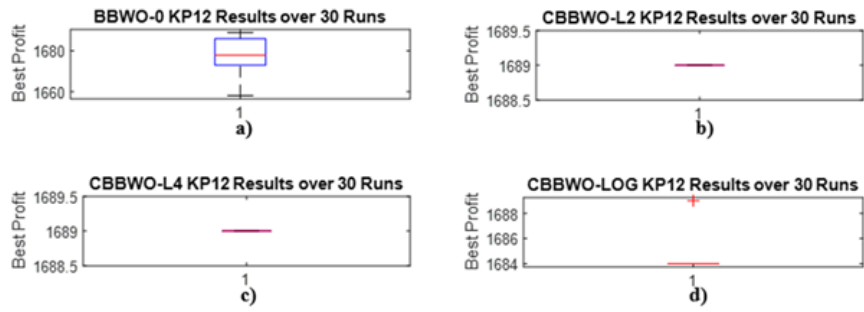


Figure 5. The boxplot of KP12 for all algorithms.

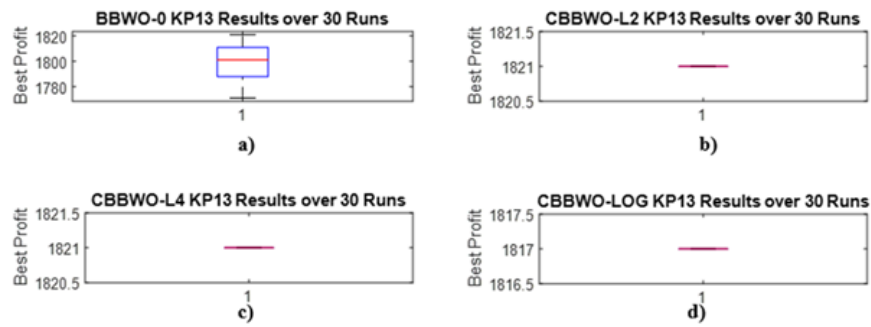


Figure 6. The boxplot of KP13 for all algorithms.

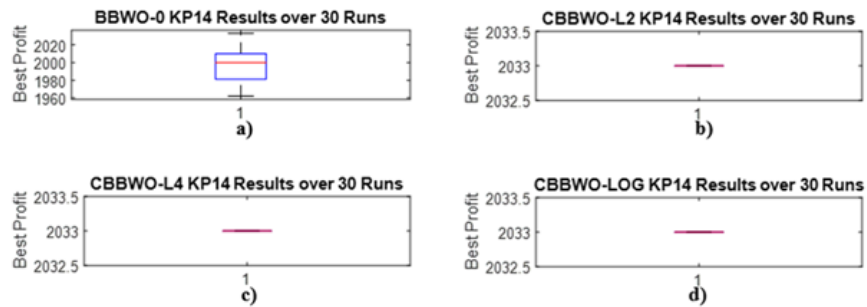


Figure 7. The boxplot of KP14 for all algorithms.

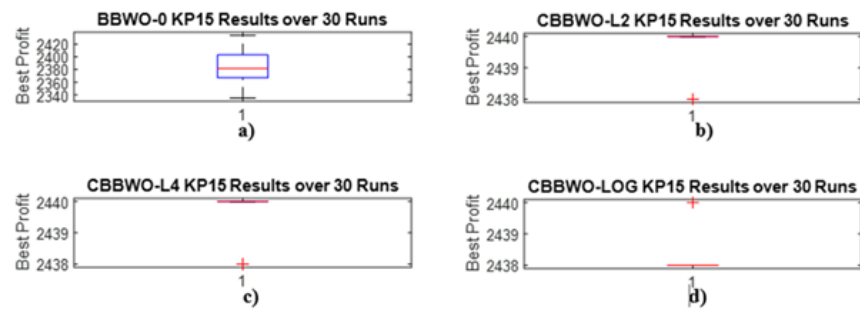


Figure 8. The boxplot of KP15 for all algorithms.

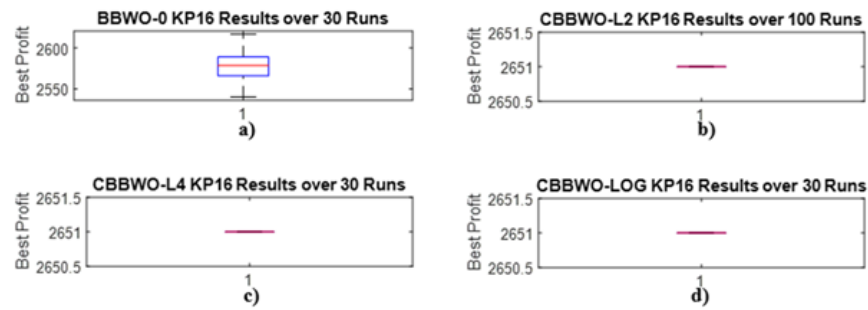


Figure 9. The boxplot of KP16 for all algorithms.

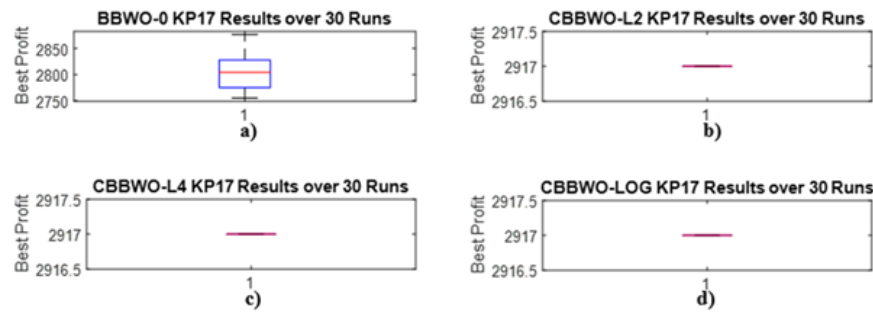


Figure 10. The boxplot of KP17 for all algorithms.

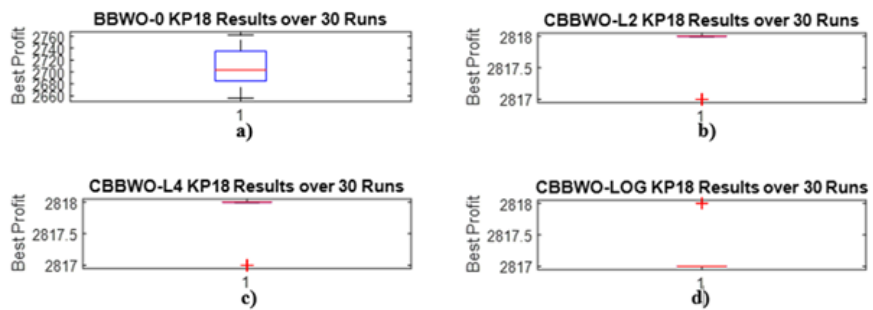


Figure 11. The boxplot of KP18 for all algorithms.

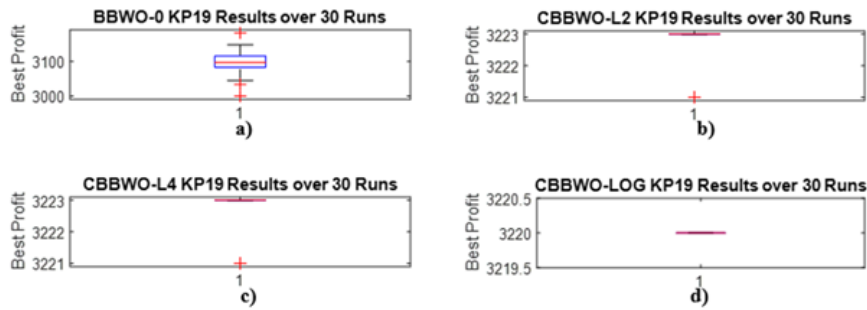


Figure 12. The boxplot of KP19 for all algorithms.

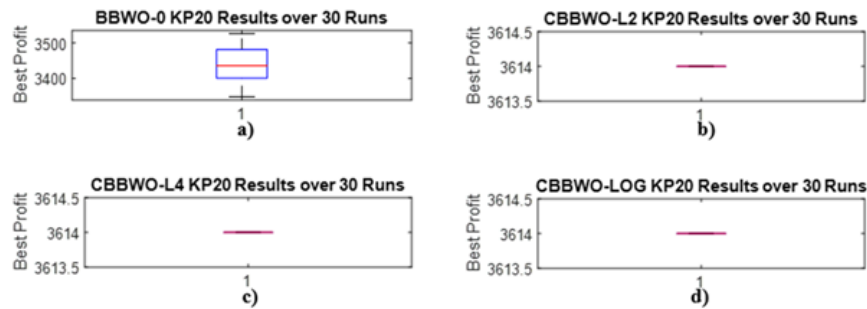


Figure 13. The boxplot of KP10 for all algorithms.

6.3. Large scale 0-1 knapsack problem

In this section, 15 randomly generated instances of the knapsack problem are considered for analysis. A substantial random test set was generated to assess the performance of the proposed algorithms. Based on three possible correlations: weakly correlated, strongly correlated, and uncorrelated between the weights and earnings, we created random test problems. [32] introduced a description of these correlations. Table 5 compares BBWO-0, CBBWO-L2, CBBWO-L4, and CBBWO-Log for the five strongly correlated random-test problems. There are 200-2000 items.

Table 5 shows that the proposed CBBWO algorithms significantly enhanced the performance of the basic BBWO-0 algorithm for problems (KP21–KP25). In all cases, BBWO-0 did not attain optimum or near-optimum solutions, whereas the CBBWO algorithms attained optimum or near-optimum solutions with significantly lower variance, higher success rates, and fewer evaluations.

The boxplots in Figs. (14-18) are used to illustrate the variation in the results over the runs. For the CBBWO algorithms, a narrow box indicates that the algorithm is consistent; this is the case for the algorithms with a zero standard deviation in the table. However, BBWO-0 has a wider spread and lower central values, as it has a worse mean and a 0% success rate.

Table 5. Results weakly correlated 0-1 knapsack problem for all improved algorithms

Problem	D	Optimal	Algorithm	Best	Worst	Mean	Std.	Average Time	Average Iterations	FE	S.R%
KP21	200	2043	BBWO-0	2042	2037	2040.1	1.4162	0.27811	200	10000	0%
			CBBWO-L2	2043	2043	2043	0	0.04554	10.57	522	100%
			CBBWO-L4	2043	2043	2043	0	0.03213	5.43	271	100%
			CBBWO-Log	2043	2042	2042.7	0.4498	0.13100	44.57	2228	73.33%
KP22	500	4978	BBWO-0	4776	4713	4745.5	13.3849	0.8274	500	25000	0%
			CBBWO-L2	4978	4978	4978	0	0.02177	1	50	100%
			CBBWO-L4	4978	4978	4978	0	0.02168	1	50	100%
			CBBWO-Log	4978	4978	4978	0	0.02086	1	50	100%
KP23	1000	10490	BBWO-0	10487	10480	10483	2.0457	7.91258	1000	50000	0%
			CBBWO-L2	10490	10490	10490	0	0.18101	8.43	421	100%
			CBBWO-L4	10490	10490	10490	0	0.1979	9.33	466	100%
			CBBWO-Log	10490	10490	10490	0	0.42975	24.47	1223	100%
KP24	1500	15700	BBWO-0	15685	15670	15677	4.0385	4.331	1500	75000	0%
			CBBWO-L2	15700	15700	15700	0	0.10298	1	50	100%
			CBBWO-L4	15700	15700	15700	0	0.10332	1	50	100%
			CBBWO-Log	15700	15700	15700	0	0.10029	1	50	100%
KP25	2000	21209	BBWO-0	21160	21124	21144.1	7.9816	1.82818	2000	100000	0%
			CBBWO-L2	21209	21209	21209	0	1.15569	183	9191	100%
			CBBWO-L4	21209	21209	21209	0	3.20958	53.43	2671	100%
			CBBWO-Log	21209	21209	21209	0	0.25718	2.3	115	100%

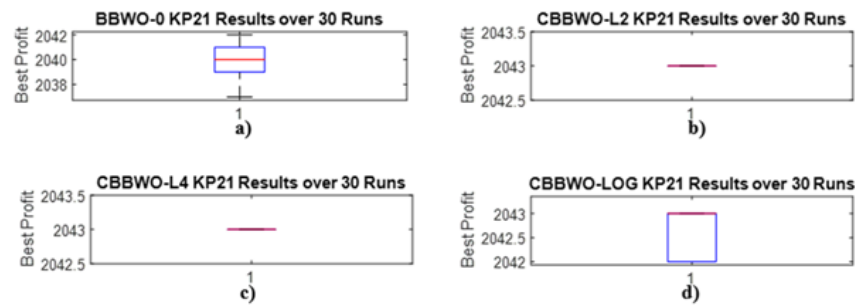


Figure 14. The boxplot of KP21 for all algorithms.

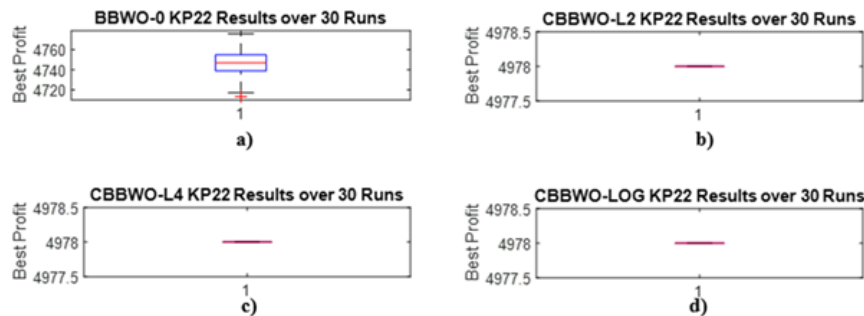


Figure 15. The boxplot of KP22 for all algorithms.

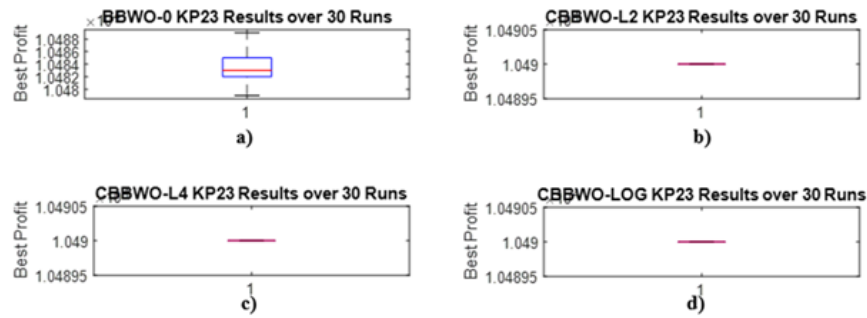


Figure 16. The boxplot of KP23 for all algorithms.



Figure 17. The boxplot of KP24 for all algorithms.

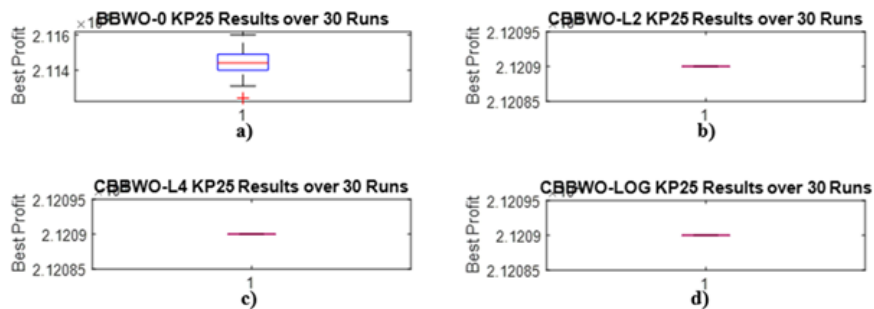


Figure 18. The boxplot of KP25 for all algorithms.

In Table 6, the performances of BBWO-0, CBBWO-L2, CBBWO-L4, and CBBWO-Log are compared for the five strongly correlated random test problems. The number of items was 200-2000 items.

Table 6. Results strongly correlated 0-1 knapsack problem for all improved algorithms

Instance	D	Optimal	Algorithm	Best	Worst	Mean	Std.	Average Time	Average Iterations	FE	S.R%
KP26	200	3526	BBWO-0	2994	2824	2903.2	47.624	0.18459	200	10000	0%
			CBBWO-L2	3526	3526	3526	0	0.01037	1	50	100%
			CBBWO-L4	3526	3526	3526	0	0.01315	1	50	100%
			CBBWO-Log	3526	3526	3526	0	0.01009	1	50	100%
KP27	500	8942	BBWO-0	8191	7913	8032.6	70.630	0.44332	500	25000	0%
			CBBWO-L2	8942	8942	8942	0	0.02089	1	50	100%
			CBBWO-L4	8942	8942	8942	0	0.02066	1	50	100%
			CBBWO-Log	8942	8942	8942	0	0.02048	1	50	100%
KP28	1000	13450	BBWO-0	13046	12983	13013.4	17.539	3.08523	1000	50000	0%
			CBBWO-L2	13450	13450	13450	0	0.05327	1	50	100%
			CBBWO-L4	13450	13450	13450	0	0.05081	1	50	100%
			CBBWO-Log	13450	13450	13450	0	0.05449	1	50	100%
KP29	1500	20549	BBWO-0	19873	19788	19830.6	21.483	7.5056	1500	75000	0%
			CBBWO-L2	20549	20549	20549	0	0.12002	2	100	100%
			CBBWO-L4	20549	20549	20549	0	0.7775	29.67	7576	100%
			CBBWO-Log	20549	20549	20549	0	0.14645	2.97	148	100%
KP30	2000	27118	BBWO-0	26151	26041	26098.7	30.06	12.3709	2000	100000	0%
			CBBWO-L2	27118	27118	27118	0	2.3889	56.27	2813	100%
			CBBWO-L4	27118	27118	27118	0	0.21123	2	100	100%
			CBBWO-Log	27118	27118	27118	0	0.2653	3.27	163	100%

The results in Table 6 show that the proposed CBBWO algorithms, namely CBBWO-L2, CBBWO-L4, and CBBWO-Log, perform better than the original BBWO-0 on all strongly correlated 0-1 knapsack problems ranging from KP26 to KP30. In all cases, the improved algorithms attained the optimal value exactly, with a zero standard deviation and a success rate of 100%, indicating that they were accurate and stable. BBWO-0 showed no optimality in all cases, had lower mean values, and no successful runs. The boxplots in Figs. 19–23 also visually reflect these results. The outcomes obtained from BBWO-0 were not consistent, as they were more scattered and had lower values than the optimum. The CBBWO algorithms, on the other hand, yielded results that were very close to the optimum value, and there was almost no difference between the runs. The proposed improvements make the algorithm significantly more robust and efficient for benchmark problems.

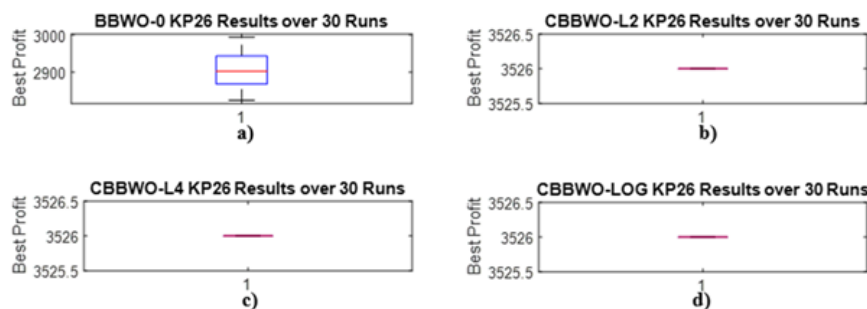


Figure 19. The boxplot of KP26 for all algorithms.

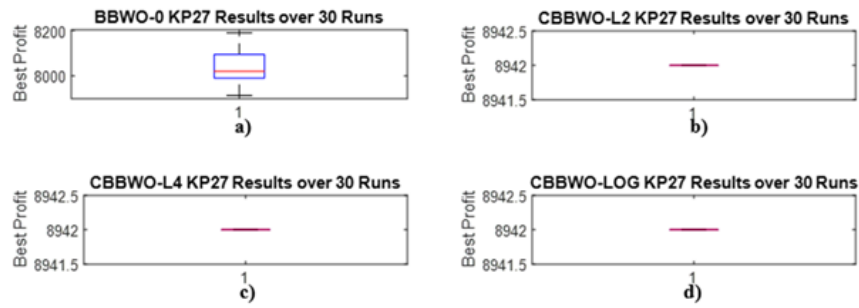


Figure 20. The boxplot of KP27 for all algorithms.



Figure 21. The boxplot of KP28 for all algorithms.

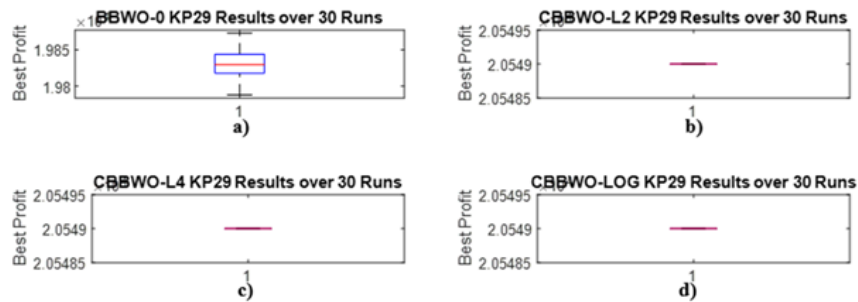


Figure 22. The boxplot of KP29 for all algorithms.

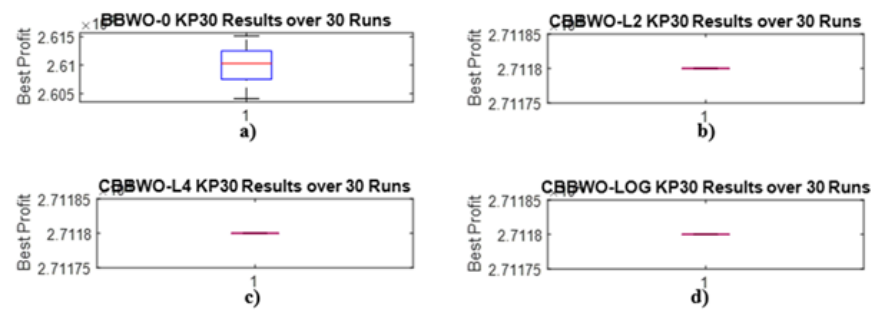


Figure 23. The boxplot of KP30 for all algorithms.

Table 7 presents a comparative analysis of the performances of BBWO-0, CBBWO-L2, CBBWO-L4, and CBBWO-Log across five uncorrelated random test problems. The problems contain (200-2000) items.

Table 7. Results uncorrelated 0-1 knapsack problem for all improved algorithms

Instance	D	Optimal	Algorithm	Best	Worst	Mean	Std.	Average Time	Average Iterations	FE	S.R%
KP31	200	4435	BBWO-0	4403	4279	4348.9	27.2466	0.20626	200	10000	0%
			CBBWO-L2	4435	4435	4435	0	0.01163	1	50	100%
			CBBWO-L4	4435	4435	4435	0	0.00948	1	50	100%
			CBBWO-Log	4435	4435	4435	0	0.0101	1	50	100%
KP32	500	10286	BBWO-0	9675	9494	9595.7	42.386	0.91339	500	25000	0%
			CBBWO-L2	10286	10286	10286	0	0.02075	1	50	100%
			CBBWO-L4	10286	10286	10286	0	0.02055	1	50	100%
			CBBWO-Log	10286	10286	10286	0	0.02208	1	50	100%
KP33	1000	20862	BBWO-0	18808	18467	18644.2	90.356	3.33644	1000	50000	0%
			CBBWO-L2	20862	20862	20862	0	0.55369	39.47	1973	100%
			CBBWO-L4	20862	20862	20862	0	0.49196	34.57	1728	100%
			CBBWO-Log	20862	20862	20862	0	0.99553	75.57	3778	100%
KP34	1500	30853	BBWO-0	27292	26536	26839.9	154	8.04066	1500	75000	0%
			CBBWO-L2	30853	30853	30853	0	0.02948	1	50	100%
			CBBWO-L4	30853	30853	30853	0	0.03105	1	50	100%
			CBBWO-Log	30853	30853	30853	0	0.03076	1	50	100%
KP35	2000	41144	BBWO-0	35239	34293	34796.6	230.1	13.7433	2000	100000	0%
			CBBWO-L2	41144	41144	41144	0	0.7992	397.57	19878	100%
			CBBWO-L4	41144	41144	41144	0	0.63002	189.93	9496	100%
			CBBWO-Log	41144	41144	41144	0	0.34568	7.70	385	100%

Table 7 shows that the proposed CBBWO algorithms outperform the original BBWO-0 on uncorrelated knapsack problems KP31–KP35. The mean solutions for BBWO-0 were lower, the standard deviations were larger, and there was no success rate for any of the cases, indicating poor reliability and convergence. Conversely, CBBWO-L2, CBBWO-L4, and CBBWO-Log consistently achieved optimal outcomes in most instances, exhibiting minimal or no variation between runs and attaining a 100% success rate in most cases. The box plots in Figs. 24–28 confirm these findings, indicating that the better algorithms are grouped closely around the optimum value, whereas BBWO-0 is spread out with lower central values. In general, Table 7 and Figs. 24–28 show that the chaotic binary versions are more accurate, stable, and efficient in solve uncorrelated 0-1 knapsack problems.

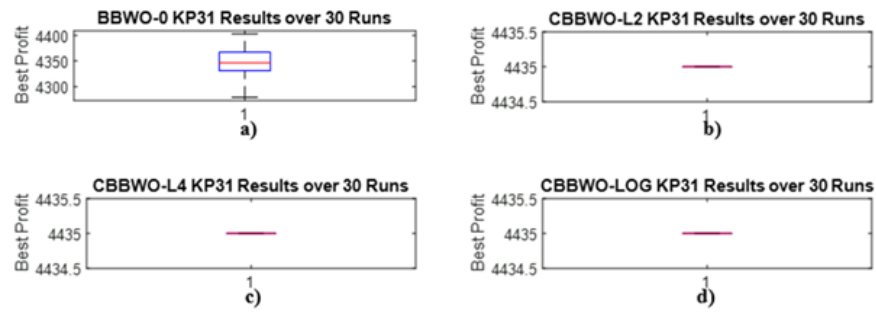


Figure 24. The boxplot of KP31 for all algorithms.

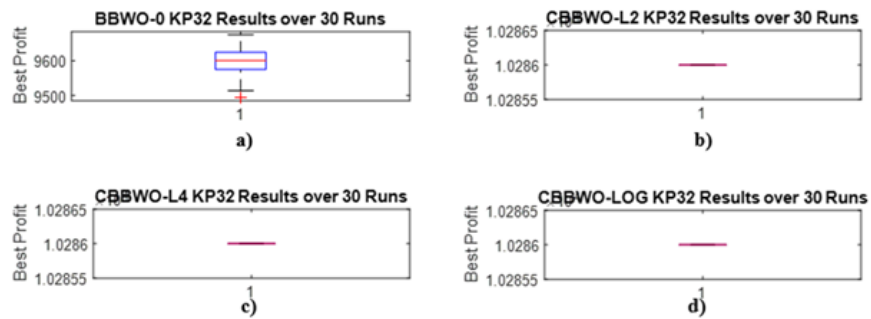


Figure 25. The boxplot of KP32 for all algorithms.

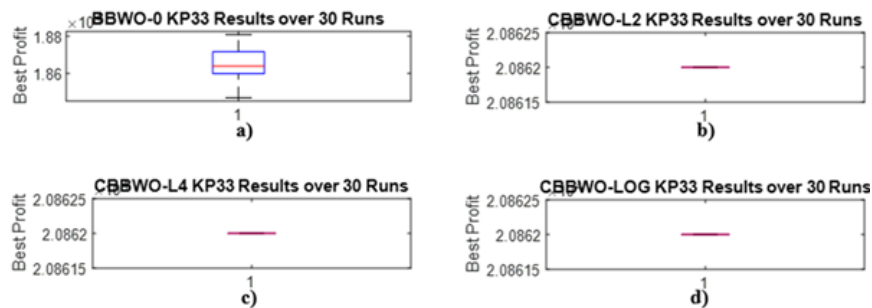


Figure 26. The boxplot of KP33 for all algorithms.

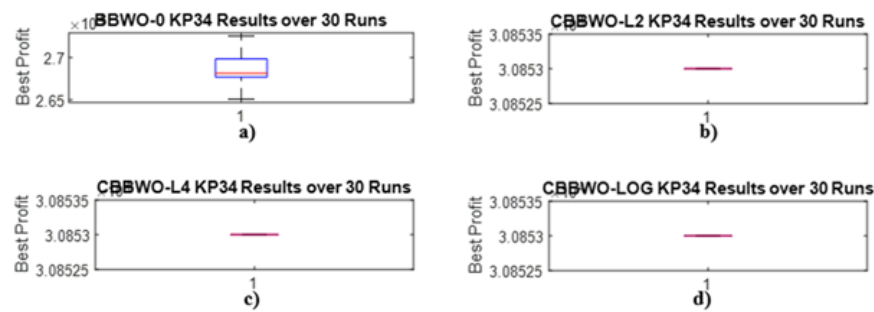


Figure 27. The boxplot of KP34 for all algorithms.

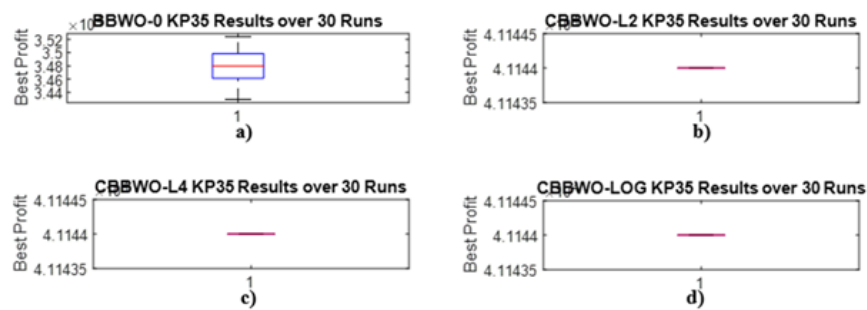


Figure 28. The boxplot of KP35 for all algorithms.

Two additional algorithms have been implemented in order to compare their performance with that of CBBWO-L2 of the large-scale KP instances. The binary of particle swarm optimization (BPSO) is the first algorithm. The binary gray wolf optimization (BGWO) algorithm is the second algorithm. The algorithms are tested with 30 independent runs, and the best, mean, worst and SD values are recorded, as explain in Table 8.

Table 8. Comparison between BPSO, BGWO and CBBWO-L2 of the large-scale KP instances

Instance	BPSO				BGWO				CBBWO-L2			
	best	worst	mean	Std.	best	worst	mean	Std.	best	worst	mean	Std.
KP21	2035	2029	2031.2	1.74	2005	1982	1992.3	5.42	2043	2043	2043	0
KP22	4923	4903	4913.3	4.909	4861	4832	4846.2	7.42	4978	4978	4978	0
KP23	10360	10336	10348.6	5.46	10223	10170	10199.3	13.39	10490	10490	10490	0
KP24	15464	15434	15449.3	7.61	15272	15207	15233.2	15.52	15700	15700	15700	0
KP25	20833	20795	20812.7	10.12	20597	20498	20536.2	21.09	21209	21209	21209	0
KP26	3526	3525	3525.9	0.15	3526	3524	3525.85	0.175	3526	3526	3526	0
KP27	8942	8941	8941.88	0.17	8942	8940	8942.9	0.025	8942	8942	8942	0
KP28	13434	13425	13428.4	2.41	13404	13379	13389.8	5.72	13450	13450	13450	0
KP29	20519	20509	20514.8	2.12	20480	20468	20473.4	3.62	20549	20549	20549	0
KP30	27068	27063	27066.2	2.18	27022	27000	27009.9	5.52	27118	27118	27118	0
KP31	4435	4434	4434.7	0.16	4435	4434	4434.9	0.045	4435	4435	4435	0
KP32	10231	10199	10211.6	8.1	10003	9900	9957.03	27.95	10286	10286	10286	0
KP33	20585	20519	20549.9	16.5	20153	19928	20039.7	50.6	20862	20862	20862	0
KP34	30369	30260	30308.7	25.29	29829	29547	29637.5	50.74	30853	30853	30853	0
KP35	40382	40260	40309.5	36.4	39673	39374	39453.2	68.5	41144	41144	41144	0

The results Table 8 indicate that CBBWO-L2 had the best, worst and mean scores on all instances. For example, on KP35, CBBWO-L2 had a mean value of 41144, whereas BPSO had a mean value of 40309.5, and BGWO had a mean value of 39453.2. The proposed algorithm also showed high robustness with standard deviation of zero for all the instances, while BPSO and BGWO had nonzero standard deviation on most of the problems. This means that the solution produced by CBBWO-L2 was of high quality and remained the same for the independent runs. The results of the comparison are also supported by the results presented in Figure 29, which indicates that CBBWO-L2 is able to converge to higher solution values and stabilizes better than BPSO and BGWO on KP21–KP25. As the problem size grows larger, the performance gap becomes larger, which shows that the proposed algorithm has better exploration and exploitation performance due to chaotic initialization and L2 transfer function.

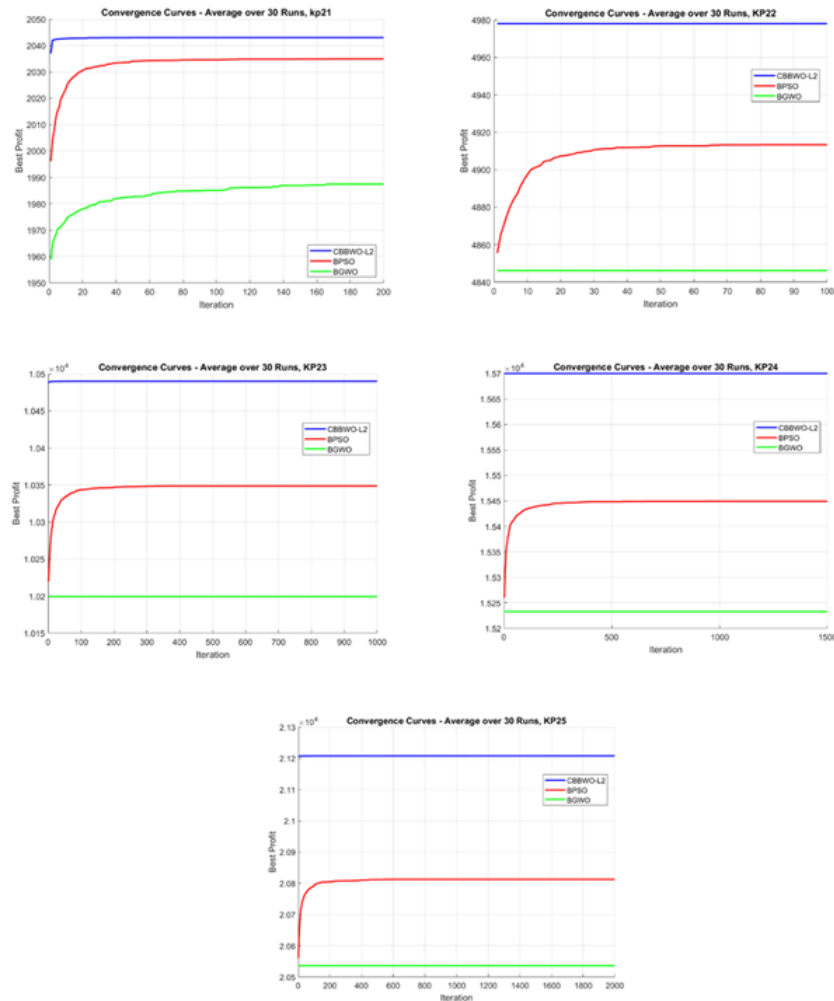


Figure 29. Convergence curves of large-scale weakly correlated KP instances.

To evaluate the performance of the CBBWO-L2 algorithm, a statistical analysis is conducted by conducting a non-parametric statistical test called Wilcoxon signed-rank test. It is used to see if there is a statistically significant difference between CBBWO-L2 and the other algorithms (BPSO and BGWO). The results are shown at the significance level of $\alpha = 0.05\%$ in Table 9. Table 9 shows the results, where all the p-values for the pairwise comparisons between CBBWO-L2 and the other approaches are found to be statistically significant with p-values below 0.05.

Table 9. Results of Wilcoxon signed-rank test of CBBWO-L2 algorithm with other algorithms

Instance	CBBWO-L2 vs BPSO		CBBWO-L2 vs BGWO	
	p-value		p-value	
KP21	1.3727	$\times 10^{-6}$	1.4244	$\times 10^{-6}$
KP22	1.4139	$\times 10^{-6}$	1.4931	$\times 10^{-6}$
KP23	1.5228	$\times 10^{-6}$	1.5876	$\times 10^{-6}$
KP24	1.6987	$\times 10^{-6}$	1.7257	$\times 10^{-6}$
KP25	1.7138	$\times 10^{-6}$	1.73	$\times 10^{-6}$
KP26	0.0278		0.0144	
KP27	0.0495		0.0255	
KP28	1.2073	$\times 10^{-6}$	1.6458	$\times 10^{-6}$
KP29	1.0227	$\times 10^{-7}$	1.6167	$\times 10^{-6}$
KP30	1.0492	$\times 10^{-6}$	1.6583	$\times 10^{-6}$
KP31	0.03458		0.0336	
KP32	1.7159	$\times 10^{-6}$	1.73	$\times 10^{-6}$
KP33	1.7235	$\times 10^{-6}$	1.7311	$\times 10^{-6}$
KP34	1.7311	$\times 10^{-6}$	1.7289	$\times 10^{-6}$
KP35	1.7322	$\times 10^{-6}$	1.73	$\times 10^{-6}$

7. Conclusion

In this study, a binary chaotic Black Widow Optimization (BBWO) algorithm for solving the 0-1 knapsack problem is proposed. This method uses L-shaped transfer functions, chaotic initialization, and a repair operator. The goal was to improve the binary search part of the original Black Widow Optimization algorithm and adapt it to a constrained combinatorial optimization. The computational results showed that the proposed algorithms are more accurate, robust, and converge faster than the standard BBWO-0. The best-performing and most successful solutions were CBBWO-L2 and CBBWO-L4, whereas BBWO-0 was less successful and effective in most of the test cases. The results indicate that the improvements help the algorithm to explore the search space efficiently and effectively, leading to the production of high-quality, feasible solutions. The proposed binary chaotic framework is a promising approach for solve the 0-1 knapsack problem and other similar optimization problems. The computational results demonstrated that the proposed algorithms outperformed the standard BBWO-0, particularly in terms of accuracy, robustness, and convergence behavior. CBBWO-L2 and CBBWO-L4 were the best-performing and most successful solutions in most of the test cases, whereas BBWO-0 was less successful and effective. The findings suggest that the proposed enhancements facilitate the algorithm's effective exploration of the search space, resulting in the generation of high-quality, feasible solutions. The proposed binary chaotic framework is a promising approach for solve the 0-1 knapsack problem and other similar optimization problems.

REFERENCES

1. Abdel-Basset, M., D. El-Shahat, and I. El-Henawy, Solving 0–1 knapsack problem by binary flower pollination algorithm. *Neural Computing and Applications*, 2018. 31(9): p. 5477-5495.
2. He, Y., et al., Binary Fick's Law Algorithm Based on L-Shaped Transfer Function for Solving the 0-1 Knapsack Problem. *SN Computer Science*, 2025. 7(1): p. 11.
3. Basheer, G., L. Mohammed, and Z. Algamal, Solving 0–1 knapsack problem by an improved binary monarch butterfly algorithm. *Statistics, Optimization & Information Computing*, 2025.
4. Mohammed, L.J., H.T. Yaseen, and E.M.N. Mahmood, An integrated heuristic-seeded adaptive large neighborhood search for transportation problems: Theoretical analysis and empirical validation. *Statistics, Optimization & Information Computing*, 2026.
5. Yang, X.-S., Cuckoo search and firefly algorithm, in *Studies in computational intelligence*. 2014, Springer.

6. Changdar, C., G. Mahapatra, and R.K. Pal, An improved genetic algorithm based approach to solve constrained knapsack problem in fuzzy environment. *Expert Systems with Applications*, 2015. 42(4): p. 2276-2286.
7. Zhao, J., et al. Genetic algorithm based on greedy strategy in the 0-1 knapsack problem. in *2009 Third International Conference on Genetic and Evolutionary Computing*. 2009. IEEE.
8. Bansal, J.C. and K. Deep, A modified binary particle swarm optimization for knapsack problems. *Applied Mathematics and Computation*, 2012. 218(22): p. 11042-11061.
9. Azad, M.A.K., A.M.A. Rocha, and E.M. Fernandes, Solving large 0–1 multidimensional knapsack problems by a new simplified binary artificial fish swarm algorithm. *Journal of Mathematical Modelling and Algorithms in Operations Research*, 2015. 14(3): p. 313-330.
10. Abdel-Basset, M., D. El-Shahat, and A.K. Sangaiah, A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem. *International Journal of Machine Learning and Cybernetics*, 2017. 10(3): p. 495-514.
11. Gherboudj, A., A. Layeb, and S. Chikhi, Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm. *International Journal of Bio-Inspired Computation*, 2012. 4(4): p. 229-236.
12. Feng, Y., et al., Solving 0–1 knapsack problem by a novel binary monarch butterfly optimization. *Neural computing and applications*, 2017. 28(7): p. 1619-1634.
13. Wang, L., R. Shi, and J. Dong, A hybridization of dragonfly algorithm optimization and angle modulation mechanism for 0-1 knapsack problems. *Entropy*, 2021. 23(5): p. 598.
14. Büyüköz, G.O. and H. Haki, Binary honey badger algorithm for 0-1 knapsack problem. *Journal of Intelligent Systems: Theory and Applications*, 2023. 6(2): p. 108-118.
15. Hu, Q. A New Method for Solving the 0-1 Knapsack Problem Based on Discrete Snake Optimization Algorithm. in *Proceedings of the 9th International Conference on Electronic Information Technology and Computer Engineering*. 2025.
16. Ihsan, A. and T. Sag, Binary Puma Optimizer: A Novel Approach for Solving 0-1 Knapsack Problems and the Uncapacitated Facility Location Problem. *Applied Sciences*, 2025. 15(18): p. 9955.
17. Tuo, S., L. Yong, and F.a. Deng, A Novel Harmony Search Algorithm Based on Teaching-Learning Strategies for 0-1 Knapsack Problems. *The Scientific World Journal*, 2014. 2014(1): p. 637412.
18. Abdel-Basset, M., et al., Binary light spectrum optimizer for knapsack problems: an improved model. *Alexandria Engineering Journal*, 2023. 67: p. 609-632.
19. Hayyolalam, V. and A.A.P. Kazem, Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. *Engineering Applications of Artificial Intelligence*, 2020. 87: p. 103249.
20. Alweshah, M., et al., Hybrid black widow optimization with iterated greedy algorithm for gene selection problems. *Heliyon*, 2023. 9(9).
21. Rajaganapathi, R., et al., Hybrid Black Widow Combined Seagull Optimization with Deep Learning for Efficient Road Classification in UAV-Aided Intelligent Transportation Systems. *Tehnički vjesnik*, 2026. 33(1): p. 383-390.
22. Hu, G., et al., An enhanced black widow optimization algorithm for feature selection. *Knowledge-Based Systems*, 2022. 235: p. 107638.
23. Divya, S., et al., Hybrid renewable energy source optimization using black widow optimization techniques with uncertainty constraints. *Measurement: Sensors*, 2024. 31: p. 100968.
24. Figueroa-Torrez, P., et al., A binary black widow optimization algorithm for addressing the cell formation problem involving alternative routes and machine reliability. *Mathematics*, 2023. 11(16): p. 3475.
25. Abdel-Basset, M., et al., Binary metaheuristic algorithms for 0–1 knapsack problems: Performance analysis, hybrid variants, and real-world application. *Journal of King Saud University-Computer and Information Sciences*, 2024. 36(6): p. 102093.
26. Gules, S., et al., A Logarithmic Transfer Function for Binary Swarm Intelligence Algorithms: Enhanced Feature Selection with White Shark Optimizer. *Applied Sciences*, 2025. 15(15): p. 8710.
27. Alkhateeb, A.N. and Z.Y. Algamal, Variable selection in gamma regression model using chaotic firefly algorithm with application in chemometrics. *Electronic Journal of Applied Statistical Analysis*, 2021. 14(1): p. 266-276.
28. Abed, K.A., A better approximation to Sawada-Kotera via differential transform method with hybrid grey Wolf and Cuckoo algorithm. *JOURNAL OF INTERDISCIPLINARY MATHEMATICS*, 2024. 27(6): p. 1231-1241.
29. Alkhateeb, A.N. and Q.N.N. Al-Qazaz, Variable selection in Weibull accelerated survival model based on chaotic sand cat swarm algorithm. *Statistics, Optimization & Information Computing*, 2025. 13(5): p. 2105-2118.
30. Gao, S. and B. Kan, Chaos of the new multiplicative logistic map. *Scientific Reports*, 2025. 15(1): p. 43743.
31. Basheer, G.T. and Z.Y. Algamal, Improving flower pollination algorithm for solving 0–1 knapsack problem. in *Journal of Physics: Conference Series*. 2021. IOP Publishing.
32. Martello, S., D. Pisinger, and P. Toth, New trends in exact algorithms for the 0–1 knapsack problem. *European Journal of Operational Research*, 2000. 123(2): p. 325-332.