Intelligent Methods in Engineering Sciences 4(2): 29-37, 2025

IMIENS

INTELLIGENT METHODS
IN ENGINEERING SCIENCES

https://www.imiens.org

International
Open Access

August, 2025

e-ISSN 2979-9236

Research Article

https://doi.org/10.58190/imiens.2025.125

Binary Aquila Optimizer with Taper-Shaped Transfer Function: An Application for Merkle-Hellman Knapsack Cryptosystems

Gülnur YILDIZDAN [a,*]

[a] *Kulu Vocational School, Selcuk University, 42770, Konya, Turkiye*

ARTICLE INFO

Article history:
Received 7 July 2025
Accepted 11 August 2025

Keywords:
Aquila Optimizer,
Cryptosystems,
Merkle-Hellman Knapsack,
Taper-shaped transfer function

ABSTRACT

Metaheuristic algorithms are powerful methods used to solve large and complex optimization problems. Thanks to their flexibility, they provide effective results in various fields and also have an important place in security applications such as the Merkle-Hellman Knapsack Crypto System. Aquila Optimizer is an optimization algorithm inspired by the hunting behavior of aquilas. It provides fast and effective solutions to complex problems. In this study, Aquila Optimizer is discretized using taper-shaped transfer functions. Taper-shaped transfer functions help the algorithm obtain more precise and effective results by increasing its performance. Four BinAO versions obtained in this way were tested on the Merkle-Hellman Knapsack Cryptosystem. In the tests conducted for the cryptanalysis of "CAT" and "MACRO" messages, the $BinAO_{T4}$ version achieved more successful results. Additionally, tests conducted with the algorithms in the literature clearly showed that the proposed algorithm is successful and effective.

1. INTRODUCTION

Metaheuristic algorithms are powerful solution methods that are especially preferred for complex, large-scale optimization problems. These algorithms are used when classical deterministic methods fall short or when the solution space is extensive. They aim to achieve results close to the global optimum by exploring the solution space intuitively. The appeal of metaheuristic algorithms lies in their flexibility, capacity to adapt seamlessly to various types of problems, and efficiency in handling very large solution spaces [1]. Additionally, they are commonly employed in diverse fields such as logistics, production planning, route optimization, and machine learning. The Merkle-Hellman Knapsack Cryptosystem (MHKC) is an example of a problem where metaheuristic algorithms are applied. MHKC is critical for ensuring data security and enhancing the reliability of cryptographic protocols [2]. Furthermore, the large size of the solution space and the presence of numerous local minima necessitate the development of efficient optimization methods. Therefore, developing efficient and reliable solution methods for complex combinatorial problems like MHKC is crucial for real-world applications.

Studies in the literature in this field are as follows: Abdel-Basset et al. derived a binary variant of the nutcracker optimization algorithm using two different families of transfer functions: S-shaped and V-shaped [3]. Then, this algorithm, whose search performance was improved by using the local search strategy, was used in the MHKC solution. Grari et al. presented a new variant of ant colony optimization in which two different search techniques are used in the MHKC solution [4]. Kantour and Bouroubi developed a parallel genetic algorithm adapted to efficiently explore the significantly large search space for MHKC [5]. Sikdar et al. cryptanalyzed the MHKC cipher using the Cuckoo Search Algorithm [6]. The proposed algorithm was distinguished by the following methods: population generation, fitness function evaluation, mutation, perturbation, and Lévy flight. Abdel-Basset et al. employed eight well-known metaheuristic algorithms to assess the reliability of MHKC toward cryptanalysis assaults, employing knapsack sizes spanning from 8 to 32 bits [7]. Abdel-Basset et al. introduced a new version of the whale optimization algorithm for the cryptanalysis of MHKC [8]. This method converted continuous inputs to discrete values using the sigmoid function. A penalty function has been integrated into the evaluation function for suboptimal solutions. The solutions were improved via mutation. Sikdar et al. implemented the MHKC using Cuckoo Search, Grey Wolf Optimization, and Harris Hawk Optimization algorithms [9]. In the selected plaintext attack scenario, the best results were achieved with Harris Hawk Optimization.

* **Corresponding Author:** gavsar@selcuk.edu.tr

Furthermore, the results outperformed the previously used Binary Firefly and Differential Evolution algorithms. Abdel-Basset et al. developed a new algorithm, called the Binary Hybrid Equilibrium Optimizer, for binary optimization problems[10]. The proposed method aims to accelerate convergence and avoid local minima by leveraging local search operators. The method is applied to three different problems: the 0–1 knapsack problem, feature selection, and the MHKC. The results demonstrate strong and competitive performance. Abdel-Basset et al. proposed the mantis search algorithm, differential evolution, and binary versions of quadratic interpolation methods to solve the 0–1 knapsack, multidimensional knapsack, and MHKC problems [11]. Experimental results demonstrate that the hybrid quadratic interpolation algorithm outperforms MHKC and other knapsack problems.

In this study, the Aquila Optimizer, an effective algorithm originally designed for continuous problems, was adapted for binary use with four different Taper-shape transfer functions. This approach allowed for an examination of the effects of the transfer functions. Cryptanalysis of the "CAT" and "MACRO" messages was carried out, and the results obtained were evaluated. The contributions of the proposed algorithm to the literature can be listed as follows:

- The Aquila Optimizer algorithm, developed for continuous problems, was binaryized and applied to the field of cryptanalysis for the first time; (during the preparation of this study, no research using AO in MHKC solving was found).
- The effects of four different taper-shaped transfer functions on solution quality were experimentally evaluated by integrating them with the AO algorithm.
- The performance of AO on the MHKC problem was analyzed, demonstrating its potential for cryptanalysis problems.
- The method has been tested practically beyond the theoretical contribution with decryption applications for real messages ("CAT" and "MACRO").

The study is organized as follows. The second section introduces the basic concepts. The third section explains the proposed method, and the fourth section presents the experimental results. The fifth section discusses the results, and the sixth section offers a general assessment and recommendations for future research.

2. Preliminaries

2.1. Merkle–Hellman Knapsack Cryptosystem

The rapid progress in communication technology in the last twenty years and the massive increase in information flow over the internet have made it necessary to ensure the confidentiality of the transmitted information. Cryptography is defined as the study of different techniques used to encrypt information, that is, to transform it into a state that cannot be read by unwanted persons during the communication between the sender and the receiver [12]. The Merkle-Hellman Knapsack Cryptosystem (MHKC) is one of the most widely recognized cryptosystems. In MHKC, an asymmetric public-key cryptosystem encrypts a message with a public key and decrypts it with a private key so only the recipient can read it [3, 13]. The following part offers a description of the encryption and decryption processes of this asymmetric cryptosystem, shown by a specific instance of the knapsack problem introduced by Merkle and Hellman [14] in 1978.

Encryption

Equation 1 converts a superincreasing knapsack series to a trapdoor knapsack sequence.

$$A = A'_i \times r \,\%q \qquad i = 1,2,\dots\dots\dots,n \tag{1}$$

Figure 1 explains how the trapdoor knapsack sequence is produced for a given super increasing sequence with q and r values.

$\boldsymbol{A'} = \{\mathbf{2,3,7,15,29,65,125,251}\}$ $\quad \boldsymbol{q = 507}$ $\quad \boldsymbol{r = 10}$

$A_1 = 2 \times 10\,\%\,507 = 20$ $\quad A_5 = 29 \times 10\,\%\,507 = 290$
$A_2 = 3 \times 10\,\%\,507 = 30$ $\quad A_6 = 65 \times 10\,\%\,507 = 143$
$A_3 = 7 \times 10\,\%\,507 = 70$ $\quad A_7 = 125 \times 10\,\%\,507 = 236$
$A_4 = 15 \times 10\,\%\,507 = 150$ $\quad A_8 = 251 \times 10\,\%\,507 = 482$

Trapdoor knapsack sequence:

$$\boldsymbol{A} = \{\mathbf{20,30,70,150,290,143,236,482}\}$$

Figure 1. The production of the trapdoor knapsack sequence [12]

Once the trapdoor knapsack sequence, depicted in Figure 1, has been created, the public key (A) and private key (A', q, r) are ready to be used for encrypting and decrypting messages between the sender and receiver. This ensures the security of information transfer and maintains its confidentiality. For example, the encryption of the "CAT" message is done as given in Figure 2, and the encryption steps are as follows [12]:

1. An 8-bit ASCII code is used to encode each character (x_i , $i \in \{1, 2, \dots, 8\}$)
2. The encrypted text is generated by multiplying the bits associated with each character by its corresponding trapdoor element and adding the products. The encrypted text is calculated as $\sum_{i=0}^{8} a'_i \times x_i$.

Trapdoor knapsack sequence	20	30	70	150	290	143	236	482	Encrypted text
C	0	1	0	0	0	0	1	1	30+236+482=**748**
A	0	1	0	0	0	0	0	1	30+482=**512**
T	0	1	0	1	0	1	0	0	30+150+143=**323**

Figure 2. Encryption of the message "CAT" [12]

Decryption

A message encrypted using the public key can only be decrypted by the recipient using the private key (A′, q, r). To achieve this, it is necessary to find which numbers correspond to 1 in the super-increasing knapsack sequence for each target total. The mentioned decryption steps are as follows [12]:

1. For each target sum ci, the value pi is calculated according to Equation 2.

$$p_i = c_i * r^{-1} \, mod \, q \quad (2)$$

where p_i is the ith character of the plaintext.

2. Continue finding the largest integer smaller than p_i in the super-increasing knapsack sequence until you reach zero, as shown in Figure 3.

3. The numbers that result in p_i reaching zero are assigned a value of 1, but the other super-increasing knapsack numbers are assigned a value of 0.

Step 1

$p_1 = 748 \times 10^{-1} \, mod \, 507 = 748 \times 355 \, mod \, 507 = 379$
$p_2 = 512 \times 10^{-1} \, mod \, 507 = 512 \times 355 \, mod \, 507 = 254$
$p_3 = 323 \times 10^{-1} \, mod \, 507 = 323 \times 355 \, mod \, 507 = 83$

Step 2

$p_1 = 379 - \mathbf{251} = 128 - \mathbf{125} = 3 - \mathbf{3} = 0$
$p_2 = 354 - \mathbf{251} = 3 - \mathbf{3} = 0$
$p_3 = 83 - \mathbf{65} = 18 - \mathbf{15} = 3 - \mathbf{3} = 0$

Step 3

$p_1 = 01000011 \quad \longrightarrow \quad C$
$p_2 = 01000001 \quad \longrightarrow \quad A$
$p_3 = 01010100 \quad \longrightarrow \quad T$

Figure 3. The decryption of (748, 512, 323) message [12]

2.2. Aquila Optimizer (AO)

In the Northern Hemisphere, aquilas are often seen as predatory birds. Aquila catches prey with its quickness, strength, strong feet, and keen claws. The Aquila uses four hunting strategies, switching between them according on the situation. The hunting techniques consist of contour flying with a brief glide assault, high soaring with a vertical stoop, low flying with a gradual descent attack, and walking and grabbing prey [15]. Abualigah et al. presented the Aquila Optimizer in 2021 by modeling these hunting strategies [16]. This method is developed for continuous optimization problems and follows these steps:

Initialization

In AO, a population-based algorithm, the population is initialized stochastically, a practice that is common to numerous other metaheuristic algorithms. The algorithm generates a population of candidate solutions that are within the upper (Ub) and lower (Lb) bounds of the given problem during the initial phase. The position of each individual ($X_{i,j}$) within the population is determined by Equation 3.

$$X_{i,j} = r_1 \times \left(Ub_j - Lb_j\right) + Lb_j \qquad i = 1, \ldots, n \qquad j = 1, \ldots, m \quad (3)$$

where Ub_j and Lb_j are the upper and lower bound values for the jth individual, r_1 is a random number in the range [0, 1], and m is the number of decision variables and n is the number of individuals. The ith value of the decision variable is denoted by X_i. Equation 4 is employed to transition AO from exploration to exploitation, where t is the current iteration and T is the total number of iterations.

$$\begin{cases} Exploration\ phase & if \quad t \leq \frac{2}{3} \times T \\ Exploitation\ phase & else \end{cases} \quad (4)$$

Expanded exploration

This initial phase (X_1) is constructed as shown in Equations 5 and 6, and it consists of the aquila flying high soar with the vertical stoop in order to identify the hunting region and choose the most attractive hunting location.

$$X_1(t+1) = X_b(t) \times \left(1 - \frac{t}{T}\right) + \left(X_{mean}(t) - X_b(t) \times r_2\right) \quad (5)$$

$$X_{mean}(t) = \frac{1}{n}\sum_{i=1}^{n} X_i(t) \qquad \forall j = 1, \ldots, m \quad (6)$$

In Equations 5 and 6, X_{mean} denotes the mean position of individuals, whereas X_b signifies the position of the best individual in the population. r_2 is a random number within the range [0, 1].

Narrowed exploration

In the second phase (X_2), known as contour flight with a short glide attack, the aquila circles above the target prey, prepares the area, and strikes. The algorithm illustrates this strategy using the equations provided in Equation 7-14.

$$X_2(t+1) = X_b(t) \times Levy(D) + X_R(t) + (y - x) \times r_3 \quad (7)$$

In Equation 7, X_R denotes the position of a randomly selected individual from the population, while r_3 refers to a random real number within the range [0, 1]. D is the variable number, and Levy (D) represents the Levy flight distribution function outlined in Equations 8 and 9.

$$Levy(D) = s \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \quad (8)$$

$$\sigma = \frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\frac{\beta-1}{2}}} \quad (9)$$

In these equations, s is a constant valued at 0.01. u and σ are random variables inside the interval [0, 1]. β is a constant, with a value of 1.5. The variables x and y are used to generate the spiral configuration and are computed using Equations 10 and 11. U and ω are equal to 0.00565 and 0.005, respectively. ϵ is a number ranging from 1 to 20. D_1 is an integer that varies from 1 to the length of the search space (m).

$$x = \psi \times \sin(\theta) \quad (10)$$

$$y = \psi \times \cos(\theta) \quad (11)$$

$$\psi = \epsilon + U \times D_1 \quad (12)$$

$$\theta = -\omega \times D_1 + \theta_1 \quad (13)$$

$$\theta_1 = \frac{3\times\pi}{2} \quad (14)$$

Expanded exploitation

Upon readiness to assault and having accurately identified the hunting location, the aquila descends vertically in preparation for a frontal attack during the third stage. The mathematical description of it, which is termed low flying with sluggish descending attack, is expressed in Equation 15. The equation employs random values ranging from 0 to 1 for r_4 and r_5. α and δ are constants, each set at a value of 0.1.

$$X_3(t+1) = (X_b(t) - X_{mean}(t)) \times \alpha - r_4 + ((U_b - L_b) \times r_5 + L_b) \times \delta \quad (15)$$

Narrowed exploitation

After reaching the prey in the fourth phase, the aquila initiates an assault from above, tracking the prey's random movements over the land. Equation 16 offers a mathematical depiction of the walking and grasping behaviors associated with prey.

$$X_4(t+1) = X_b(t) \times q_F - (\mathfrak{g}_1 \times X(t) \times r_6) - \mathfrak{g}_2 \times Levy(D) + r_7 \times \mathfrak{g}_1 \quad (16)$$

In Equation 16, q_F denotes the quality function and is computed in accordance with Equation 17. Equation 18 represents $\mathfrak{g}_1$, which is employed to track prey during escape. $\mathfrak{g}_2$ is used to track prey throughout an escape, from beginning to end. The value representing the aquila's flight slope is calculated using Equation 19 and exhibits a decline from 2 to 0. The variables r_6, r_7, and r_8 are random values ranging from 0 to 1.

$$Q_F = t^{\frac{2\times r_7 - 1}{(1-T)^2}} \quad (17)$$

$$\mathfrak{g}_1 = 2 \times r_8 - 1 \quad (18)$$

$$\mathfrak{g}_2 = 2 \times (1 - \frac{t}{T}) \quad (19)$$

3. Proposed Method

Aquila Optimizer (AO) is an effective algorithm for continuous optimization problems, but for binary optimization problems, the solution space needs to be discretized. Transfer functions perform this discretization by transforming continuous variables into specific binary values [17]. The transfer function transforms continuous values into values that can be represented in binary form, thus making AO adaptable to binary optimization problems.

There are different transfer functions for discretization, one of which is the taper transfer function. The advantageous aspects of transfer functions compared to other transfer functions can be listed as follows [18]:

- Due to the symmetry of the Y-axis, it is more appropriate for the discretization of evolutionary algorithms compared to transfer functions without this characteristic.
- On the intervals $[-A, 0]$ and $[0, A]$, the transfer functions exhibiting smooth curvature transitions are superior to those with non-smooth curvature transitions.
- Discrete evolutionary algorithms provide more benefits when the transfer function's domain matches the value range of an individual's component.

The taper transfer function was implemented in this study as a result of the advantages mentioned above. In this way, the Aquila Optimizer was made more effective in binary optimization problems. Figure 4 presents taper-shaped transfer functions and their graphics.

Taper-Shaped (A=[-1, 1])	
T1	$F(a) = \frac{\sqrt{\lvert a \rvert}}{\sqrt{\lvert A \rvert}}$
T2	$F(a) = \frac{\lvert a \rvert}{\lvert A \rvert}$
T3	$F(a) = \frac{\sqrt[3]{\lvert a \rvert}}{\sqrt[3]{\lvert A \rvert}}$
T4	$F(a) = \frac{\sqrt[4]{\lvert a \rvert}}{\sqrt[4]{\lvert A \rvert}}$

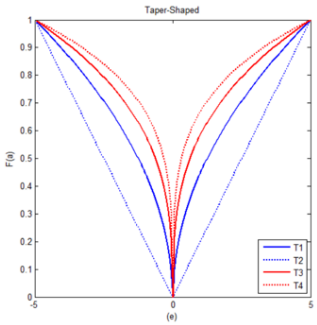

Figure 4. Taper-shaped functions and their graphics [18]

In the binary search space, each dimension represents a segment of the bit sequence. For the MHKC problem, a value of "1" indicates that the relevant element is included in the encryption process, while a value of "0" indicates that it is not. The process of transforming the continuous search space into a binary structure suitable for the knapsack problem (binarization)—that is, determining the selection status of each element—is shown in Figure 5. This transformation is achieved through the thresholding function defined by Equation 20 [19]. In this way, the values in the continuous space are converted to a binary representation, making them suitable for the MHKC.

$$X_{binary} = \begin{cases} 1, & if \quad F(a) > rand \\ 0, & else \end{cases} \quad (20)$$

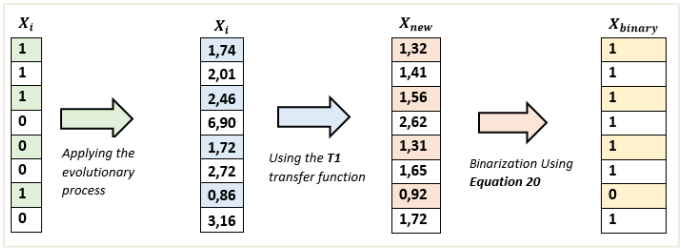

Figure 5. Example of a scheme for binarization

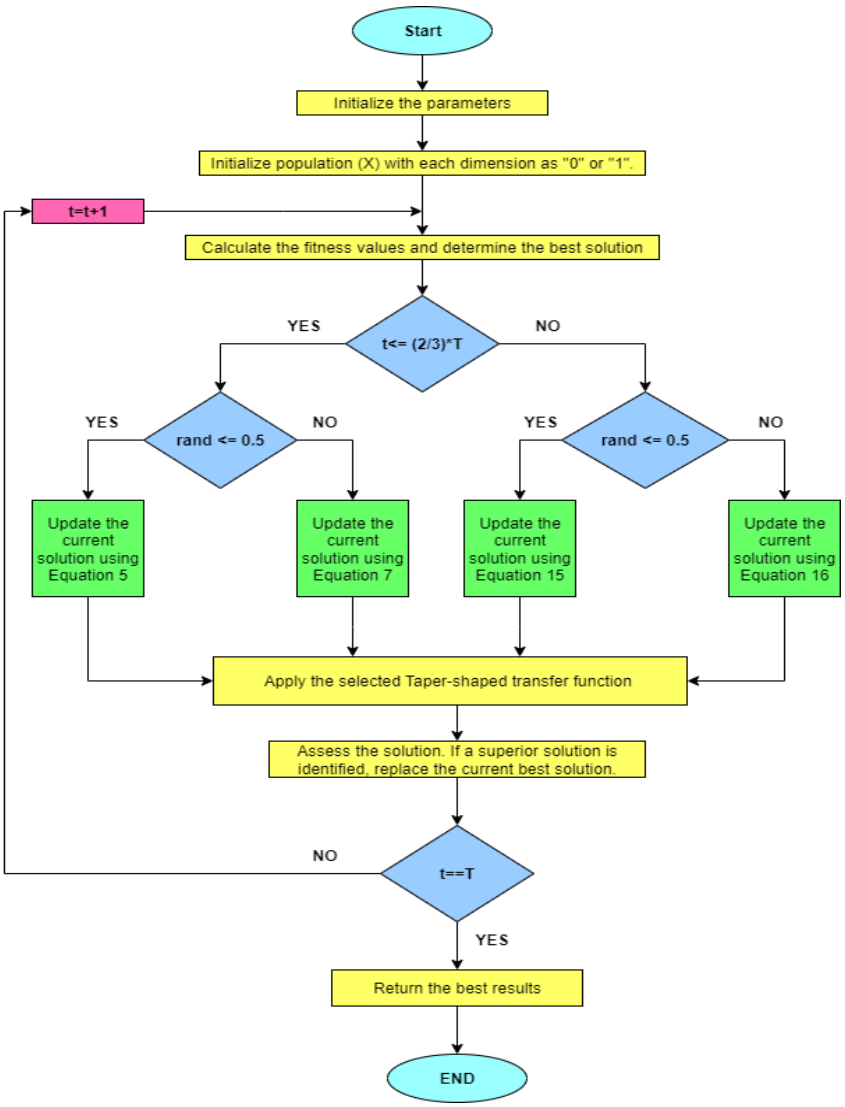

Figure 6. Flowchart of proposed method

The proposed method's flowchart is illustrated in Figure 6 to facilitate a more systematic and clear explanation of the algorithm's operation.

4. Experimental Results

4.1. Performance of BinAO on Merkle–Hellman Knapsack Cryptosystem

This section tests the performance of the proposed versions of BinAO with integrated taper-shaped transfer functions on MHKC problems. The parameters of *BinAO* are set with the following values: maximum number of iterations = 100 and population size = 20. The algorithm's parameter values were determined by referencing values frequently used in similar optimization problems in the literature. This approach ensures comparability of the obtained results with previous studies and a fair assessment of the effectiveness of the proposed method. Figure 7 shows the cipher text for the message "CAT" and each character's ASCII code [12]. An 8-bit sequence of 0s and 1s is used to encode each character. After the message is encrypted, the recipient receives the cipher text. MHKC encrypts and decrypts the "CAT" message as follows (Equation 21) [3]:

A' = [8 15, 29, 65, 125, 251]

q = 507, r =10, r^{-1} = 355,

A = {20, 30, 70, 150, 290, 143, 236, 482} (21)

Character	ASCII code	Cipher text
C	01000011	748
A	01000001	512
T	01010100	323

Figure 7. MHKC encryption of the message "CAT"[12]

Table 1 presents comparative results for BinAO for the "CAT" message cryptosystem when different taper-shaped transfer functions are used. Algorithms were evaluated based on the results of five runs (Run#). In the table, *IN* is

the number of iterations needed to determine the accurate ASCII code for each character, while M_IN represents the mean number of iterations per character. M_time gives the mean of the time taken for five runs. The results showed that B$inAO_{T4}$ performs better overall. B$inAO_{T4}$ obtained the smallest M_IN value for all characters. It also obtained a smaller value for M_time for two out of three characters. In other words, it was more successful in cryptanalyzing the character in fewer iterations and in less time.

Table 1. The "CAT" message cryptanalysis's results

		Run_1	Run_2	Run_3	Run_4	Run_5		
Character		***IN***	***IN***	***IN***	***IN***	***IN***	***M_IN***	***M_time***
C	$\mathbf{BinAO_{T1}}$	21	5	1	91	4	24,4	0,0513
	$\mathbf{BinAO_{T2}}$	7	16	3	1	44	14,2	0,0209
	$\mathbf{BinAO_{T3}}$	1	50	1	1	2	11	0.0164
	$\mathbf{BinAO_{T4}}$	3	1	1	1	8	**2,8**	**0,0052**
A	$\mathbf{BinAO_{T1}}$	24	1	2	41	3	14,2	0,0255
	$\mathbf{BinAO_{T2}}$	1	5	1	3	5	**3**	**0,0053**
	$\mathbf{BinAO_{T3}}$	6	3	1	5	2	3,4	0,0066
	$\mathbf{BinAO_{T4}}$	6	2	2	4	1	**3**	0,0059
T	$\mathbf{BinAO_{T1}}$	3	3	5	2	21	6,8	0,0149
	$\mathbf{BinAO_{T2}}$	1	8	14	24	1	9,6	0,0207
	$\mathbf{BinAO_{T3}}$	1	2	9	11	4	5,4	0,0132
	$\mathbf{BinAO_{T4}}$	7	1	7	1	2	**3,6**	**0,0111**

Secondly, in this section, the performance of BinAO versions in encrypting the "MACRO" message (8 bits) with MHKC, which is widely used in the literature [20], was examined. The cipher text for the message "MACRO" and the ASCII code of each character are illustrated in Figure 8. MHKC encrypts and decrypts the " MACRO " message as follows(Equation 22) [12, 21]:

A' = {4,5,13,23, 48, 96, 193, 384}

q = 776, r = 13, r^{-1} = 597

A = {52, 65, 169, 299, 624, 472, 181, 336} (22)

Character	ASCII code	Cipher text
M	01001101	1497
A	01000001	401
C	01000110	582
R	01010010	545
O	01001111	1678

Figure 8. MHKC encryption of the message "MACRO"[12]

Table 2 presents comparative results for BinAO for the "MACRO" message cryptosystem when different taper-shaped transfer functions are used. Upon examination of the results in the table, B$inAO_{T4}$ in three characters and B$inAO_{T3}$ in the remaining two characters have obtained a smaller M_IN value. The optimum time was achieved by B$inAO_{T1}$ in two characters, B$inAO_{T3}$ in two characters, and B$inAO_{T4}$ in the remaining one character when evaluated according to the M_time criterion. In the event that a comprehensive analysis is carried out, it is determined that the B$inAO_{T4}$ version is more successful than the other versions. This success is due to the fact that the T4 transfer function has a smoother transition curve and provides a more balanced exploration and exploitation process in the search space compared to other functions that cannot suppress extreme values.

Ultimately, this section presents a comparison of BinAO (i.e., the most successful B$inAO_{T4}$ version results) with the algorithms that are currently available in the literature. For this, the results of the algorithms in the study of Abdel-Basset et al. [12] were taken. The comparison is made with the modified version of WOA (MWOA) [12], binary firefly algorithm (FA) [21], genetic algorithm (GA)[21], differential evolution algorithm (DE)[22]. Table 3 illustrates the results of the comparison. Results in the table were calculated based on results from 5 runs of each algorithm. According to the comparison in Table 3, the smallest M_IN value for each character was found by $BinAO$. In this way, BinAO ranked first among the algorithms and proved its success.

Table 2. The "MACRO" message cryptanalysis's results

		Run_1	Run_2	Run_3	Run_4	Run_5		
Character		***IN***	***IN***	***IN***	***IN***	***IN***	***M_IN***	***M_time***
M	$\boldsymbol{BinAO_{T1}}$	2	30	9	31	9	15,2	**0,0176**
	$\boldsymbol{BinAO_{T2}}$	7	2	199	5	12	45	0,0783
	$\boldsymbol{BinAO_{T3}}$	13	37	50	2	2	20,8	0,0222
	$\boldsymbol{BinAO_{T4}}$	4	23	10	19	1	**11,4**	0,0190
A	$\boldsymbol{BinAO_{T1}}$	6	5	1	3	2	3,4	**0,0054**
	$\boldsymbol{BinAO_{T2}}$	4	2	3	6	3	3,6	0,0091
	$\boldsymbol{BinAO_{T3}}$	1	11	2	1	1	3,2	0,0090
	$\boldsymbol{BinAO_{T4}}$	1	3	1	3	1	**1,8**	0,0055
C	$\boldsymbol{BinAO_{T1}}$	21	1	31	1	1	11	0,0098
	$\boldsymbol{BinAO_{T2}}$	1	3	1	6	4	3	0,0079
	$\boldsymbol{BinAO_{T3}}$	4	3	3	2	2	**2,8**	**0,0078**
	$\boldsymbol{BinAO_{T4}}$	3	7	2	2	2	3,2	0,0088
R	$\boldsymbol{BinAO_{T1}}$	2	2	18	1	7	6	0,0239
	$\boldsymbol{BinAO_{T2}}$	2	41	1	1	48	18,6	0,0259
	$\boldsymbol{BinAO_{T3}}$	2	1	1	1	7	**2,4**	**0,0053**
	$\boldsymbol{BinAO_{T4}}$	3	4	1	9	3	4	0,0089
O	$\boldsymbol{BinAO_{T1}}$	1	10	4	12	2	5,8	0,0103
	$\boldsymbol{BinAO_{T2}}$	35	1	3	1	34	14,8	0,0186
	$\boldsymbol{BinAO_{T3}}$	1	15	3	10	25	10,8	0,0151
	$\boldsymbol{BinAO_{T4}}$	11	1	1	3	4	**4**	**0,0051**

Table 3. Comparative results of algorithms in the literature for "MACRO" message cryptanalysis

	Algorithms				
Characters	BinAO	MWOA	FA	GA	DE
	M_IN	***M_IN***	***M_IN***	***M_IN***	***M_IN***
M	11,4	16	15	258,4	31,4
A	1,8	5,8	12	220	29,2
C	3,2	11,4	12,4	260,8	21,6
R	4	5,6	8,6	192,2	28,2
O	4	7,8	5,2	100,6	11,2
Total	**24,4**	46,6	53,2	1032	121,6
Rank	**1**	2	3	5	4

5. Discussion

In this study, the proposed BinAO algorithm was tested in the Merkle-Hellman encryption system with four different Taper transfer functions, and the results were evaluated in detail. Experiments on both "CAT" and "MACRO" messages showed that the B$inAO_{T4}$ version was the most successful, with the lowest mean iteration count and mean time. In comparison with the widely used MWOA, FA, GA, and DE algorithms in the literature, BinAO ranked first with lower iteration values across all characters. These results demonstrate the proposed approach's high solution capability and processing efficiency in cryptanalysis problems.

Furthermore, significant differences were observed between the Taper functions. In particular, B$inAO_{T4}$ produced more stable and faster solutions compared to versions like B$inAO_{T1}$, which produced more unstable results. This demonstrates that the structure of the transfer function directly affects success in discrete solution space problems. The results support BinAO as a powerful and competitive method in terms of both temporal efficiency and accurate character resolution.

6. Conclusion

Aquila Optimizer is an effective algorithm proposed for continuous optimization problems. In this study, this algorithm, which operates in binary solution spaces, was adapted with four different Taper-shaped transfer

functions and tested on the decryption problem of the Merkle-Hellman Knapsack Cryptosystems. In experiments conducted on "CAT" and "MACRO" messages, the mean number of iterations required to reach the correct ASCII value of each character and the mean running time were evaluated as key performance metrics. The results, in particular, showed that the BinAO_T4 configuration reached the target solution in fewer iterations and a shorter time compared to other variants. This demonstrates the critical role of transfer functions in algorithm performance, both in guiding and accelerating the search process. Furthermore, the proposed method was found to achieve similar or better performance with lower computational cost compared to other heuristic algorithms commonly used in the literature. These results demonstrate that the method can achieve successful and reliable performance in both academic research and real-world applications.

In the future, the proposed algorithm is planned to be applied to various binary decision problems such as feature selection and facility location. Additionally, studies will be conducted on the automatic optimization and adaptability of transfer function parameters. Because the parameters are kept constant in this study, the effects of different parameter settings on algorithm performance need to be investigated in detail. Furthermore, since only a limited number of transfer functions were tested, the aim is to compare the algorithm with different binarization techniques.

Declaration of Ethical Standards

The article complies with ethical rules.

Credit Authorship Contribution Statement

Gülnur YILDIZDAN: Conceptualization, Investigation, Methodology, Writing—review, Software, Original draft & editing.

Declaration of Competing Interest

The author declares no conflict of interest.

Funding / Acknowledgements

This study was not funded by any institution.

Data Availability

Data will be made available on request.

References

[1] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM computing surveys (CSUR),* vol. 35, no. 3, pp. 268-308, 2003.

[2] S. Vaudenay, *A classical introduction to cryptography: Applications for communications security*. Springer, 2006.

[3] M. Abdel-Basset, R. Mohamed, I. M. Hezam, and K. M. Sallam, "Performance Optimization and Comprehensive Analysis of Binary Nutcracker Optimization Algorithm: A Case Study of Feature Selection and Merkle–Hellman Knapsack Cryptosystem," *Complexity,* vol. 2023, 2023.

[4] H. Grari, S. Lamzabi, A. Azouaoui, and K. Zine-Dine, "Cryptanalysis of Merkle-Hellman cipher using ant colony optimization," *Int J Artif Intell ISSN,* vol. 2252, no. 8938, p. 8938, 2021.

[5] N. Kantour and S. Bouroubi, "Cryptanalysis of merkle-hellman cipher using parallel genetic algorithm," *Mobile Networks and Applications,* vol. 25, no. 1, pp. 211-222, 2020.

[6] S. Sikdar, J. Biswas, and M. Kule, "Cryptanalysis of Markle Hellman Knapsack Cipher Using Cuckoo Search Algorithm," in *International Conference on Frontiers in Computing and Systems*, 2022: Springer, pp. 147-160.

[7] M. Abdel-Basset, R. Mohamed, and O. M. Elkomy, "Knapsack Cipher-based metaheuristic optimization algorithms for cryptanalysis in blockchain-enabled internet of things systems," *Ad Hoc Networks,* vol. 128, p. 102798, 2022/04/01/ 2022, doi: https://doi.org/10.1016/j.adhoc.2022.102798.

[8] M. Abdel-Basset, D. El-Shahat, I. El-henawy, A. K. Sangaiah, and S. H. Ahmed, "A Novel Whale Optimization Algorithm for Cryptanalysis in Merkle-Hellman Cryptosystem," *Mobile Networks and Applications,* vol. 23, no. 4, pp. 723-733, 2018/08/01 2018, doi: 10.1007/s11036-018-1005-3.

[9] S. Sikdar, J. Biswas, and M. Kule, "Cryptanalysis of Markle-Hellman knapsack cipher using nature inspired algorithms," *International Journal of Applied Cryptography,* vol. 5, no. 1, pp. 41-56, 2024.

[10] M. Abdel-Basset, R. Mohamed, I. M. Hezam, K. M. Sallam, and I. A. Hameed, "An efficient binary hybrid equilibrium algorithm for binary optimization problems: analysis, validation, and case studies," *International Journal of Computational Intelligence Systems,* vol. 17, no. 1, p. 98, 2024.

[11] M. Abdel-Basset, R. Mohamed, S. Saber, I. M. Hezam, K. M. Sallam, and I. A. Hameed, "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,* vol. 36, no. 6, p. 102093, 2024.

[12] M. Abdel-Basset, D. El-Shahat, I. El-Henawy, A. K. Sangaiah, and S. H. Ahmed, "A novel whale optimization algorithm for cryptanalysis in Merkle-Hellman cryptosystem," *Mobile Networks and Applications,* vol. 23, pp. 723-733, 2018.

[13] A. Agarwal, "Encrypting messages using the Merkle-Hellman knapsack cryptosystem," *IJCSNS,* vol. 11, no. 5, p. 12, 2011.

[14] R. Merkle and M. Hellman, "Hiding information and signatures in trapdoor knapsacks," *IEEE Transactions on Information Theory,* vol. 24, no. 5, pp. 525-530, 1978, doi: 10.1109/TIT.1978.1055927.

[15] S. Mahajan, L. Abualigah, A. K. Pandit, and M. Altalhi, "Hybrid Aquila optimizer with arithmetic optimization algorithm for global optimization tasks," *Soft Computing,* vol. 26, no. 10, pp. 4863-4881, 2022.

[16] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. Al-Qaness, and A. H. Gandomi, "Aquila optimizer: a novel meta-heuristic optimization algorithm," *Computers & Industrial Engineering,* vol. 157, p. 107250, 2021.

[17] G. Yildizdan and E. Bas, "A new binary coati optimization algorithm for binary optimization problems," *Neural Computing and Applications,* vol. 36, no. 6, pp. 2797-2834, 2024.

[18] Y. He, F. Zhang, S. Mirjalili, and T. Zhang, "Novel binary differential evolution algorithm based on Taper-shaped transfer functions for binary optimization problems," *Swarm and Evolutionary Computation,* vol. 69, p. 101022, 2022.

[19] B. Crawford, R. Soto, G. Astorga, J. García, C. Castro, and F. Paredes, "Putting continuous metaheuristics to work in binary

search spaces," *Complexity,* vol. 2017, no. 1, p. 8404231, 2017.

[20] A. Jain and N. S. Chaudhari, "Cryptanalytic results on knapsack cryptosystem using binary particle swarm optimization," in *International Joint Conference SOCO'14-CISIS'14-ICEUTE'14: Bilbao, Spain, June 25th-27th, 2014, Proceedings*, 2014: Springer, pp. 375-384.

[21] S. Palit, S. N. Sinha, M. A. Molla, A. Khanra, and M. Kule, "A cryptanalytic attack on the knapsack cryptosystem using binary firefly algorithm," in *2011 2nd International conference on computer and communication technology (ICCCT-2011)*, 2011: IEEE, pp. 428-432.

[22] S. N. Sinha, S. Palit, M. A. Molla, A. Khanra, and M. Kule, "A cryptanalytic attack on Knapsack cipher using differential evolution algorithm," in *2011 IEEE Recent Advances in Intelligent Computational Systems*, 2011: IEEE, pp. 317-320.