

Chaotic Hybrid Meta-Heuristic Optimization Algorithm for Construction Site Layout Planning Problem

Amirhossein Hojati ^{1*}, Davood Sedaghat Shayegan ², Aliasghar Amirkardoost ³,
Kamand Sedaghat Shayegan ⁴

¹ Department of Civil Engineering, Roudehen Branch, Islamic Azad University, Tehran, Iran.

² Department of Civil Engineering, Roudehen Branch, Islamic Azad University, Tehran, Iran.

³ Department of Civil Engineering, Roudehen Branch, Islamic Azad University, Tehran, Iran

⁴ Department of Information Technology and Electrical Engineering, Oulu University, Oulu, Finland

¹ a.hojati@iau.ir, ² al.amirkardoust@iau.ac.ir, ³ da.sedaghat@iau.ac.ir, ⁴ ksedagha23@student oulu.fi

Abstract: In recent decades, site layout has been a major challenge for researchers in construction management. Recognized as an NP-complete problem, it resists exact solutions, particularly for medium to large-scale projects. Numerous studies have explored metaheuristic approaches to tackle this issue, yet there is a demand for novel methods that promise improved accuracy within shorter computational periods. In this study, a pioneering solution is introduced: a hybrid Ant Lion Optimizer Algorithm (ALO) and Aquila Optimizer Algorithm (AO) based on chaos theory tailored specifically for optimizing construction management tasks. The combination of ALO and the Aquila Optimizer harnesses the strengths of two distinct optimization strategies. ALO mimics the trapping behavior of antlions, striking a balance between exploration and exploitation for optimal outcomes. In contrast, the Aquila Optimizer replicates the dynamic hunting tactics of eagles, facilitating swift and adaptive search methods. By merging these approaches, the hybrid algorithm adeptly navigates complex problems, dynamically adapting to environmental changes. This collaborative synergy holds promise for efficient optimization across various domains. The chaotic hybrid algorithm (CH-ALOAO) utilizes interactive memory to store optimal solutions throughout the optimization process. Its performance is benchmarked against established metaheuristic algorithms regarding search capabilities, avoidance of suboptimal solutions, and convergence speed. The results undergo rigorous statistical analysis, with experimental data showcasing CH-ALOAO's superior performance in addressing construction management optimization challenges compared to its competitors.

Keywords: Construction Management, Site Layout Planning, ALO Optimization, Aquila Optimizer, Chaotic.

1. Introduction

Construction site layout planning (CSLP) is a vital component of construction planning, aimed at optimizing the arrangement of temporary facilities while balancing multiple, often conflicting objectives within logical and resource constraints [1]. Key goals include minimizing costs related to facility interactions and reducing safety and environmental hazards, which have been the focus of recent research [2]. Traditionally, CSLP has been approached as a static problem, where temporary facilities are placed at the project's outset and remain fixed until completion [3]. However, in practice, the demand for temporary facilities changes throughout different phases of construction, influenced by the activity schedule. Zouein and Tommelein [4] emphasized the crucial need to consider the interdependence between activity scheduling and

site layout. Acknowledging its impact on result reliability, recent studies have increasingly adopted dynamic approaches to tackle the CSLP problem.

CSLP is recognized as an 'NP-hard' problem due to its complexity [5]. Recent advancements in swarm intelligence-based meta-heuristic algorithms have proven effective in solving optimization challenges [6-15], leading researchers to apply these modern techniques to CSLP models. Ant colony optimization (ACO) is a bio-inspired metaheuristic that mimics the foraging behavior of ants, as described by Dorigo et al. [16]. ACO has been applied to solve facility layout problems in hypothetical medium-sized construction sites [17]. Both Gharaie et al. [18] and Lam et al. [17] utilized ACO to address static site layout problems in construction projects. Ning et al. [19] presented a method to tackle the dynamic multi-objective CSLP

problem using the Max-Min Ant System (MMAS), a variant of the standard ACO algorithm. While previous research has addressed the CSLP problem, the introduction of new metaheuristics continues to be beneficial, offering potential improvements in solution quality.

Lien and Cheng [3] developed a hybrid particle-bee algorithm for optimizing construction site layout with a single objective function, aiming to place facilities in predetermined positions. Li and Love [20] investigated the use of Genetic Algorithms to find optimal solutions for single-objective CSLP problems, specifically focusing on facilities with unequal areas in fixed positions.

Particle swarm optimization (PSO) is a metaheuristic technique that simulates the social manners of birds flocking toward a target, as introduced by Eberhart and Kennedy [21, 22]. Additionally, Xu and Li [2] proposed a multi-objective PSO algorithm to address the multi-objective dynamic CSLP problem. Zang and Wang [23] employed a particle swarm optimization (PSO) methodology to optimize static layouts for facilities with varying sizes in predetermined locations.

The genetic algorithm (GA), which simulates the method of natural evolution, is widely used to develop effective solutions for optimization and search issues. GA employs approaches motivated by natural evolution—such as inheritance, modification, choice, and crossover—to address optimization problems. Several studies have applied GA to address the facility site layout problem, including works by Adrian et al. [24], Cheung et al. [25], and Mawdesley and Al-Jibouri [26]. Osman et al. [27] presented a hybrid CAD-based algorithm that uses GA to improve the placement of facilities with unequal areas in any available area at a construction site.

The artificial bee colony (ABC) algorithm, developed by Dervis Karaboga [28], is a recent addition to swarm intelligence-based algorithms, simulating the foraging manners of honey bees and other algorithms such as CBO and MBF [29-32]. In [28], a multi-objective ABC algorithm is employed to solve the CSLP problem. The standard algorithm is improved with Levy flight random walks to help employed bees find new food sources. The study aims to optimize the dynamic layout problem with two primary objectives: undervaluing safety hazards

and environmental concerns and reducing the total handling cost associated with interchange between facilities. The model addresses the CSLP issue as a non-linear layout challenge concerning facilities of varying sizes, which can be arranged in a horizontal or vertical orientation. Additionally, the model takes into account the existence of blocks when calculating travel distances.

The following are the paper's main contributions:

- I. Introducing a hybrid ALO and AO and assessing its performance on diverse global optimization problems.
- II. Creating a chaotic hybrid algorithm CH-ALAO to optimize construction management problem.
- III. Experimentally evaluating CH-ALAO on construction management problem with varying complexities and statistically analyzing the results.

The article is structured as follows: Section 2 covers the problem and Section 3 outlines background and preliminaries of the original algorithms. Section 4 summarizes the structure of the proposed modified optimizer, while Section 5 proposes and examines the experimental results, and evaluates CH-ALAO for optimizing the construction management problem. Section 6 concludes the paper.

2. Construction Site Layout Problem

As noted, construction site layout issues can be modeled as quadratic assignment problems, where the costs related to the flow between facilities are linearly dependent on the distance crossed and the amount of flow [33]. The objective is to uniquely assign several predetermined facilities (n) to several predetermined locations (m), with the number of places being equal to or more significant than the quantity of facilities. If the quantity of positions (m) exceeds the quantity of facilities (n), then $m - n$ dummy facilities are added to balance the numbers. By allocating both the length and frequency as zero, these dummy facilities will not impact the layout consequences.

If each predetermined location can accommodate any facility, the layout problem is modeled as an equal-area facility layout issue. However, if some locations can only accommodate specific facilities, the issue evolves into an unequal-area facility layout issue, where the predetermined locations have

varying scopes. Unequal-area layout issues are generally more challenging to solve than equal-area

layout issues due to the additional constraints introduced by the differing area requirements [20].

The objective functions of various models, as presented by [27], are summarized in Table 1 and follow a general form.

$$\text{Minimize } F = \sum_{i=1}^n \sum_{j=1}^n W_{ij} \times d_{ij} \quad (1)$$

Table 1. Various types of objective functions have been utilized in previous research studies

No.	Objective function
1	To minimize the total transport expenses of resources between facilities.[25]
2	The objective is to reduce a generalized charge function that accounts for construction costs, transportation costs, and safety concerns related to the closeness or remoteness of facilities.[34]
3	underestimate the total handling costs of interchange flows between facilities, as well as safety hazards and environmental concerns. [35]
4	To minimize the costs associated with exchanges between structures. [36]
5	To minimize the whole charge of site layout, and maximize the space between the ‘high-risk’ structures and the ‘high-protection’ structures to decrease the chance of safety or environmental mishaps. [2]
6	To minimize safety concerns and reducing construction cost. [19]

The objective function F is defined, with n representing the quantity of structures and positions. The coefficient W_{ij} signifies either the real transportation cost per unit space between structures i and j (accounting for the number of travels made) or a comparative proximity weight indicating the desired nearness between structures i and j . The term d_{ij} indicates the distance between structures i and j .

Each design choice can be described by an $n \times n$ permutation matrix, where n is the number of structures or places. In this matrix, the rows correspond to structures and the columns to places. The permutation matrix has a single entry of one in each row and column, with all other entrances being

zero. Table 2 provides an instance of a permutation matrix for five structures and five places.

A typical solution to the site layout issue, as demonstrated in Table 2, results in a sparse matrix that requires significant computing resources for large issues. A more efficient option is using a sequence of integers, where each entry illustrates a structure and the integer indicates its location, as depicted in Table 3. However, this method can lead to infeasible solutions with overlapping entrances, especially when using meta-heuristic approaches. Therefore, adjustments are needed to resolve this issue [37].

Table 2. An example of permutation matrix

Number of Facilities	Number of Locations				
	L1	L2	L3	L4	L5
F1	1	0	0	0	0
F2	0	0	1	0	0

Table 3. An example of the sequence-based illustration for CSLP

F1	F2	F3	F4	F5
2	1	3	5	4

$F3$	0	1	0	0	0
$F4$	0	0	0	0	0
$F5$	0	0	0	0	0

3. Background

We leverage an improved metaheuristic algorithm named CH-ALAO that contains ALO optimization and AO algorithm for optimizing construction management tasks. In this section, we introduce Based algorithms.

3.1. Ant Lion Optimizer Algorithm

The ALO algorithm is inspired by the hunting behavior of ant lions preying on ants [38]. First, we assume ants as particles in the search area to model their relations. Then, the ant lion is permitted to pursue. The movement of the ant, as it randomly searches for food in nature, is modeled by Equation 1.

$$X(t) = [0, (\text{cumsum}(2r(t_1 - 1)), \text{cumsum}(2r(t_2 - 1)), \dots, \text{cumsum}(2r(t_n - 1)))] \quad (1)$$

Thus, cumsum calculates the cumulative sum, where n represents the maximum iterations, t denotes the random motion phase, and $r(t)$ denotes a random function defined by Equation 2.

$$r(t) = \begin{cases} 1 & \text{if } rand > 0.5 \\ 0 & \text{if } rand \leq 0.5 \end{cases} \quad (2)$$

In Equation 2, t represents the random walk step, and $rand$ denotes a random number between 0 and 1. The positions of the ants are stored in the matrix defined by Equation 3 and used during optimization.

$$M_{Ant} = \begin{bmatrix} A_{1,1} & \cdots & A_{1,d} \\ \vdots & \ddots & \vdots \\ A_{n,1} & \cdots & A_{n,d} \end{bmatrix} \quad (3)$$

Thus, M_{Ant} specifies the position of each ant, and A_{ij} determines the j^{th} dimension of the i^{th} ant, n shows the number of ants and d the number of dimensions. A fitness function, according to Equation 4, is used during optimization for the evaluation of each ant.

Thus, M_{Ant} determines the status of each ant, with A_{ij} representing the j^{th} dimension of the i^{th} ant. Here, n denotes the count of ants, and d represents the count of dimensions. During optimization, a fitness function, defined by Equation 4, is used to evaluate each ant.

$$M_{OA} = \begin{bmatrix} f([A_{1,1}, A_{1,2}, \dots, A_{1,d}]) \\ f([A_{2,1}, A_{2,2}, \dots, A_{2,d}]) \\ \vdots \\ f([A_{n,1}, A_{n,2}, \dots, A_{n,d}]) \end{bmatrix} \quad (4)$$

M_{OA} is employed to store the fitness function values for each ant. Additionally, we consider an ant lion obscured someplace in the search area. Equations 5 and 6 are used to keep the ant lion's location and objective function.

$$M_{Antlion} = \begin{bmatrix} AL_{1,1} & \cdots & AL_{1,d} \\ \vdots & \ddots & \vdots \\ AL_{n,1} & \cdots & AL_{n,d} \end{bmatrix} \quad (5)$$

$$M_{OAL} = \begin{bmatrix} f([AL_{1,1}, AL_{1,2}, \dots, AL_{1,d}]) \\ f([AL_{2,1}, AL_{2,2}, \dots, AL_{2,d}]) \\ \vdots \\ f([AL_{n,1}, AL_{n,2}, \dots, AL_{n,d}]) \end{bmatrix} \quad (6)$$

$M_{Antlion}$ and M_{OAL} specify the location matrix and the objective function matrix of each ant lion, respectively. Additionally, AL_{ij} represents the j^{th} dimension of the i^{th} ant lion, with n indicating the number of ant lions and d representing the number of variables.

During the optimization process, ants move randomly within the search area, influenced by the traps set by ant lions. Ant lions create larger pits based on the objective function, which allows them to capture more ants. When an ant is caught by an ant lion, it is taken beneath the sand. The distance the ant travels towards the ant lion decreases. Following each capture, the ant lion shifts its position to target the next ant, forming a new pit with suitable modifications.

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i - c_i^t)}{(d_i^t - a_i)} + c_i \quad (7)$$

In Equation (7), a_i is the minimum value of the random variable for the i^{th} variable. c_i^t represents the minimum value of the i^{th} variable in the t^{th} iteration, while d_i^t is the maximum value of the i^{th} variable in

$$c_i^t = Antlion_j^t + c^t \quad (8)$$

$$d_i^t = Antlion_j^t + d^t \quad (9)$$

In Equation (7), c^t denotes the minimum value among all variables in the t^{th} iteration, while d^t represents the vector containing the maximum values of all variables in that iteration. Similarly, c_i^t indicates the minimum of all variables for the i^{th} ant, and d_i^t represents the vector containing the

The random motion of ants follows Equation 1. During optimization, ants move through a random walk. Due to the constraints of the search area, Equation 1 alone isn't sufficient for updating their positions. Therefore, the values are standardized using Equation 7 to randomize their movements within the search area.

the t^{th} iteration. This equation must be applied at every iteration. To account for the influence of ant lion traps on the ants' random motion, Equations (8) and (9) are utilized.

maximum values for the i^{th} ant. In this equation, $Antlion_j^t$ specifies the position of the j^{th} ant lion in the t^{th} iteration. The ant lion's hunting capability is modeled using the roulette wheel structure, as shown in Figure 1.

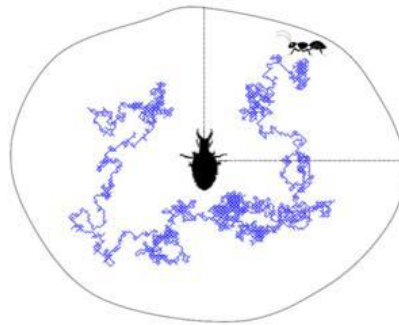


Figure 1. The random movement of an ant within the trap of an ant lion

When ants are ensnared, the ant lion tosses stones towards the trap's edges, as described by Equations 10 and 11.

$$c^t = c^t / I \quad (10)$$

$$d^t = d^t / I \quad (11)$$

Variable I represents a constant ratio, where c^t is the minimum of all variables in the t^{th} iteration. The d^t denotes the vector containing the maximum of all variables in the t^{th} iteration. The variable I is defined by Equation 12.

$$I = 10^w(t/T) \quad (12)$$

In Equation 13, t represents the current iteration, T is the maximum number of iterations, and w is a constant determined based on the current iteration, as specified by Equation 13.

$$w = \begin{cases} 2 & \text{when } t > 0.1T \\ 3 & \text{when } t > 0.5T \\ 4 & \text{when } t > 0.75T \\ 5 & \text{when } t > 0.9T \\ 6 & \text{when } t > 0.95T \end{cases} \quad (13)$$

The final stage of the hunt occurs when the prey reaches the bottom of the trap and is captured by the ant lion. At this point, the ant lion pulls the prey into the sand and consumes it. This process simulates hunting, with the ant considered trapped in the sand.

$$Antlion_j^t = Ant_i^t \text{ if } f(Ant_i^t) > f(Antlion_j^t) \quad (14)$$

In Equation 15, t represents the current iteration. $Antlion_j^t$ denotes the location of the j^{th} ant lion at iteration t , while Ant_i^t indicates the location of the i^{th} ant at the same iteration. Here, f refers to the fitness function. Throughout the algorithm, the best-
 $Ant_i^t = (R_A^t + R_E^t)/2$

Thus, R_A^t is the random position around the ant lions by the roulette wheel in iteration t . R_E^t is the random position around the elite state in t^{th} iteration. Ant_i^t shows the position of the i^{th} ant on the t^{th} iteration. This algorithm is defined as triplet $ALO(A,B,C)$.

Thus, R_A^t represents the random locations around the ant lions determined by the roulette wheel in iteration t , while R_E^t denotes the random positions around the elite state at the same iteration. Ant_i^t indicates the position of the i^{th} ant during iteration t . The algorithm is defined as the triplet $ALO(A, B, C)$.

$$\emptyset \xrightarrow{A} \{M_{Ant}, M_{OA}, M_{Antlion}, M_{OAL}\} \quad (16)$$

$$\{M_{Ant}, M_{Antlion}\} \xrightarrow{B} \{M_{Ant}, M_{Antlion}\} \quad (17)$$

$$\{M_{Ant}, M_{Antlion}\} \xrightarrow{C} \{true, false\} \quad (18)$$

In these equations, M_{Ant} denotes the matrix of ant locations, $M_{Antlion}$ represents the matrix of ant lion locations, M_{OA} includes the ants that have been hunted, and M_{OAL} lists the predator ants. Function A initializes the locations of both the ants and the ant lions. During each iteration, Function B updates the location of each ant based on a selected ant lion using Roulette Wheel Selection and elite strategies. The range for updating locations is set according to the current iteration, and this update is refined by a

Subsequently, the ant lion's position is updated to where it captured the ant, enhancing the likelihood of another successful hunt. This update is achieved using Equation 14.

performing ant lion in each iteration is designated as the elite. This elite ant lion influences all ants, guiding them towards elitism, as described in Equation 15.

Function A produces initial random solutions, and function B is the initial population updated by function A . Function C gives *true* results in satisfying conditions. These functions are shown in Equations 16, 17, and 18.

Function A generates the initial random solutions, function B updates the initial population based on function A , and function C ensures the results meet the required conditions. These functions are outlined in Equations 16, 17, and 18.

random walk around the ant lions and elite solutions. After all ants have moved randomly, their locations are evaluated using a fitness function. If an ant's location relative to the ant lions is deemed appropriate, it becomes the new location for the next iteration. The best ant lion is compared with the best found so far and replaced if it shows improvement. This process continues until Function C returns an incorrect value. Figure 2 illustrates the ant lion algorithm [38].

```

Start by randomly generating the initial population of ants and ant lions
Evaluate the fitness levels of both the ants and the ant lions
Identify the top-performing ant lions and designate them as the elite (optimal solution)
while the termination condition is met
    for every ant
        Choose an ant lion based on the roulette wheel selection method
        Adjust the values of `c` and `d` according to Equations (10) and (11)
        Generate a random walk and normalize it using Equations (1) and (7)
        Adjust the ant's position using Equation (15)
    end for
    Evaluate the fitness of each ant
    Substitute an ant lion with its corresponding ant if the ant demonstrates improved fitness, as per Equation (14)
    Update the elite ant lion if it shows better fitness than the current elite.
end while
Return elite
    
```

Figure 2. Pseudocode of the ALO algorithm

3.2. Aquila Optimizer Algorithm

The Aquila Optimizer (AO) is a meta-heuristic algorithm that leverages swarm intelligence and a population-based approach. It draws inspiration from the hunting techniques of the Aquila, a prominent predatory bird from the northern hemisphere, and represents one of the latest advancements in this field [39]. The four distinct hunting techniques used by Aquila are simulated by the Aquila Optimizer. The following subsection

$$X_i(t+1) = X_{best}(t) \times (1 - t/T) + (X_M(t) - X_{best}(t) \times rand) \quad (19)$$

$$X_M(t) = 1/N \sum_{i=1}^N X_i(t), \quad i = 1, \dots, Dim \quad (20)$$

In Equation 19, $X_i(t+1)$ is the position of i^{th} individuals in the next iteration, and the best result in this iteration is represented by the $X_{best}(t)$. The mean locations of all individuals in the i^{th} generation are represented by $X_M(t)$. The exploration operation in search area is controlled by $(1 - t/T)$. where t is the current generation, T is the maximum number of iterations, and N illustrates the swarms' population

$$X_i(t+1) = X_{best}(t) \times Levy(D) + X_R(t) - (y - x) \times rand \quad (21)$$

expresses Aquila's four strategies to hunt prey [40-42]. The Aquila Optimizer Algorithm flowchart is illustrated in Figure 3.

I. Expanded exploration

The Aquila initially investigates any potential prey in the target region. The Aquila chooses the place to seek its prey, then stoops down vertically to capture it. The following Equations provide the mathematical formula for such a behavior:

size. $rand$ represents a randomly generated number based on a Gaussian distribution between 0 and 1.

II. Narrowed exploration

Aquila uses this phase the most often for hunting, and the hunt is conducted at contour flying with a brief glide assault. This procedure is described as:

$$Levy(D) = s \times \mu \times \sigma / |v|^{\frac{1}{\beta}} \quad (22)$$

Where $Levy(D)$ is the levy flight distribution function for the dimensionality of the problem solving D , which the calculation would be done by Equation 22. $X_R(t)$, is a position of Aquila generated randomly in the i^{th} iteration. y and x represent the

spiral shape, and the $rand$ is a random real number between 0 and 1. The s is a constant parameter equal to 0.01, and μ and v are random values between 0 and 1. σ is a dynamic adaptive coefficient, and β is a constant fixed to 1.5.

III. Expanded exploitation

In this process, the target area is specified and Aquila can vertically achieve a primary attack with a slow decent attack. Mathematically, this action is shown by Equation 23.

$$X_i(t+1) = \alpha \times [X_{best}(t) - X_M(t)] + \delta \times [(UB - LB) \times rand + LB] \quad (23)$$

α and δ are the parameters that adjust the exploitation and they are 0.1 based on tests for various benchmarks. $X_M(t)$ demonstrates the mean position in the i^{th} iteration, Also LB and UB denote the lower and upper bounds, respectively.

In this phase, the Aquila walks on the land and can easily capture the prey and hunt it by pulling. This step is mathematically modeled by Equation 24. QF is the quality value that balances the search strategies, G_1 is the different motions of the AO and G_2 is the slope of the chasing flight of the AO.

IV. Narrowed exploitation

$$X_i(t+1) = QF \times X_{best}(t) - G_1 \times X_i(t) \times rand - G_2 \times Levy(D) + rand \times G_1 \quad (24)$$

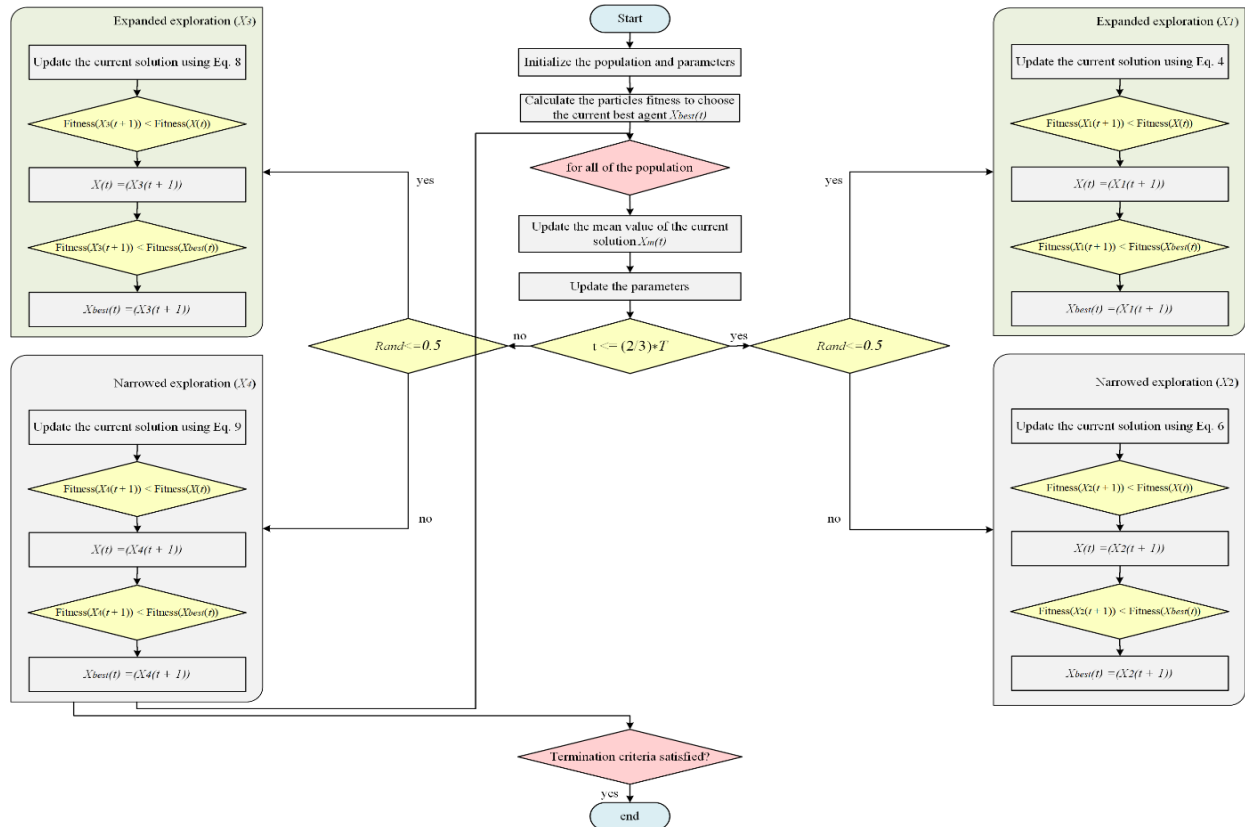


Figure 3. The flowchart of the Aquila Optimizer algorithm

4. Proposed hybrid Meta-heuristic model definition

This section will cover the ways to enhance the ALO algorithm and its integration with AO. In this section, we introduce our method that integrates ALO and AO. We apply a binary version of ALOAO to the CSLP issue. While the ALO and AO algorithms perform exceptionally well compared to other swarm intelligence-based optimization techniques, they struggle with highly complex functions due to their propensity to become trapped in local optima [43-47]. To overcome these limitations and improve search efficiency, we introduce a novel hybrid algorithm that merges ALO and AO. This new approach utilizes a multi-swarm strategy, leveraging the advantages of both algorithms to enhance exploration, exploitation, and convergence towards optimal solutions.

The ALO algorithm adjusts ant positions using a random walk influenced by antlions chosen through an elite strategy and roulette wheel selection. This method provides rapid speed, robust convergence, and high efficiency. However, it tends to converge prematurely and get trapped in local optima in complex optimization problems [48]. To address these issues, enhancements have been implemented to improve the algorithm's optimization capability and accuracy.

The Aquila Optimizer (AO) algorithm excels in its ability to effectively explore the search area and find global optima in complex and high-dimensional optimization issues. It mimics the diverse hunting strategies of the Aquila bird, which enhances its adaptability and flexibility in solving various optimization tasks. AO's ability to avoid local optima ensures it does not get trapped in suboptimal solutions, making it a robust tool for achieving high-quality results in challenging optimization scenarios.

Non-hybrid metaheuristic algorithms are often based solely on mathematical theory and might end up in a local minimum. The aim of hybridizing metaheuristic algorithms is to combine the natural processes of two different algorithms to solve various hard optimization problems. The hybrid algorithm's performance and accuracy will be enhanced by keeping the balance between exploration and exploitation when hybridizing metaheuristic algorithms. The following methods

are commonly employed to combine optimization algorithms and create novel hybrid metaheuristic techniques [49]:

- Sequential method: Each metaheuristic algorithm will execute consecutively, one after another.
- Concurrent method: Each algorithm can operate on the whole population or a part of it. In the latter case, proper attention should be given to the formation, integration, and management of sub-populations.
- Conditional method: Here, a single optimization technique is used per iteration, with its choice depending on specific conditions.

The proposed method adopts a serial approach to integrate the suggested algorithms, with the AO utilizing the ALO as its initial population to enhance accuracy and convergence speed. This hybrid method, ALOAO, combines the strengths of two meta-heuristic algorithms: the ALO and the AO. The ALO algorithm uses a random walk approach around antlions chosen through an elite strategy and roulette wheel selection. In contrast, the AO algorithm draws inspiration from the Aquila's hunting tactics and includes four distinct methods. Combining these algorithms, ALOAO effectively balances exploration and exploitation, avoiding premature convergence and stagnation.

By integrating these algorithms, ALOAO achieves high performance and robustness in solving CSLP problems. Consequently, ALOAO can find optimal or near-optimal solutions for the CSLP problem. Figure 4 presents the flowchart for the proposed hybrid optimization method. Section 5 showcases various experiments that highlight the convergence speed and accuracy of our approach.

The proposed algorithm divides the population into subpopulations assigned to different algorithms for diverse search strategies. Initially, solutions are randomly split between the algorithms, each operating independently on its designated subset. To optimize performance, the best solutions are exchanged using shared memory, where each algorithm stores its top k solutions (up to 30% of the population). To improve exploration, a mutation operator is applied to 20% of the population. Finally, the best solutions from shared memory are selected based on objective functions.

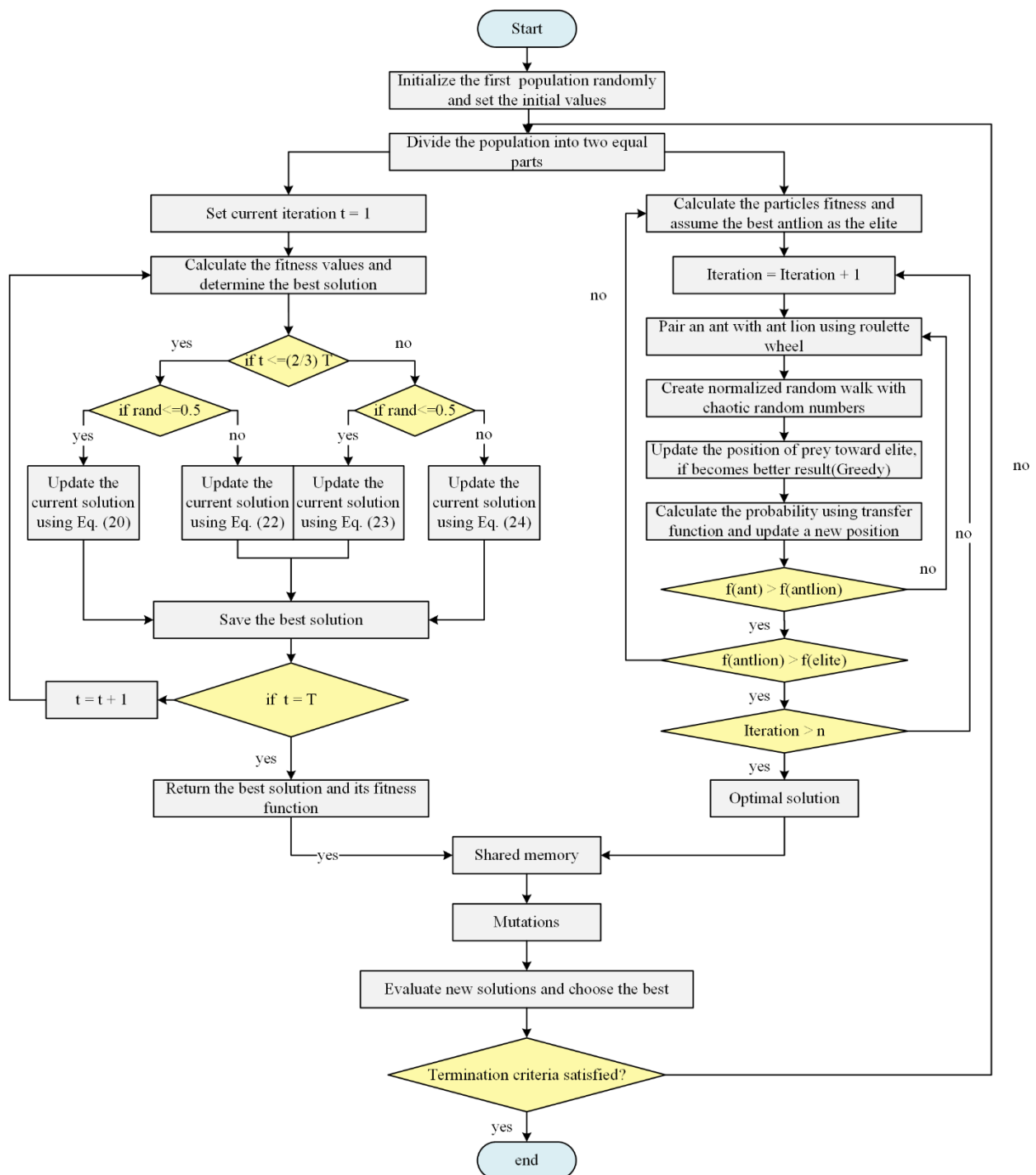


Figure 4. The ALOAO flowchart

The ALO algorithm tends to perform poorly in exploring global optimizations [50]. To mitigate this issue and enhance the algorithm's efficiency, we incorporated five functions based on chaos theory—circular, Gaussian, and logistic—instead of traditional random functions [51]. These chaos-based functions generate numbers within the range [0, 1], as demonstrated in Table 4, where all random numbers start with an initial value of 0.7 [52].

Chaos, deterministic, and quasi-random functions used in dynamic and nonlinear systems are notable for their irregular, non-repeating patterns and bounded behavior. From a mathematical perspective, chaotic functions describe deterministic systems that exhibit seemingly random behavior. Unlike conventional mathematical functions, chaos maps are utilized in optimization algorithms, particularly over the past decade, due to their

dynamic properties that facilitate discovery in complex environments. These functions have proven effective in practical applications by helping algorithms avoid local optima and improving

convergence speed. In the CH-ALOAO algorithm, a random chaotic function is included in each iteration to harness these advantages.

Table 4. Chaotic maps

No.	Name	Function
1	Chebyshev	$V_{b+1} = \cos(b \cos^{-1}(V_b))$, $b=1..100$
2	Circle	$V_{b+1} = V_b + d - \left(\frac{c}{2\pi}\right) \sin(2\pi V_b) \bmod(1)$, $C=0.5$, $d=0.2$
3	Gauss	$V_{b+1} = \begin{cases} 0 \\ 1 \\ V_b \bmod(1) \end{cases} \quad \text{otherwise } \frac{V_b = 0}{V_b \bmod(1)} = \frac{1}{V_b} - \left\lfloor \frac{1}{V_b} \right\rfloor$
4	Iterative	$V_{b+1} = \sin\left(\frac{c\pi}{V_b}\right)$, $C=0.7$
5	Logistic	$V_{b+1} = CV_b(1 - V_b)$, $C=4$

In CSLP issues, each particle is represented as a sequence of numbers that define a layout solution. Different sequences represent different layout configurations, with each variable indicating a structure and its value specifying the assigned position for that facility. Since each location can only host one facility, duplicate values within a particle result in infeasible solutions. However, in the proposed algorithm, all variables are updated independently, which can lead to multiple variables having the same value. Therefore, modifications are needed in the updating mechanism to address this issue. The updated mechanism for CH-ALOAO is detailed below:

To tackle this issue, we can utilize the discrete environment optimization algorithm. Developing a binary CH-ALOAO requires binary solutions, necessitating several conversions. A highly effective method for transforming a continuous optimizer into a binary one is to use transfer functions. These transfer functions are easy to implement, fast, and cost-efficient [53]. This paper employs four S-shaped (S1-S4) and four V-shaped (V1-V4) transfer functions to convert the continuous CH-ALOAO algorithm into a binary format. The mathematical formulations of these transfer functions are detailed in Table 5.

Table 5. The S-shaped and V-shaped transfer functions used

S-shaped family	Transfer function	V-shaped family	Transfer function
S1	$T(x) = 1/(1 + e^{-2x})$	V1	$T(x) = \left \operatorname{erf}\left(\frac{\sqrt{\pi}}{2}x\right) \right $
S2	$T(x) = 1/(1 + e^{-x})$	V2	$T(x) = \tanh(x) $
S3	$T(x) = 1/(1 + e^{(-\frac{x}{2})})$	V3	$T(x) = x/\sqrt{1+x^2} $
S4	$T(x) = 1/(1 + e^{(-\frac{x}{3})})$	V4	$T(x) = \left \frac{2}{\pi} \arctan\left(\frac{\pi}{2}x\right) \right $

The Binary Hybrid ALO and AO chaotic optimization algorithm explores binary search spaces through the use of transfer functions. This

method involves two stages for updating particle positions. First, the particles' positions are updated in continuous space, similar to non-binary

algorithms, resulting in new continuous values. In the second stage, transfer functions are applied to convert these continuous positions into binary values. The particle positions are adjusted according

$$X_i^d(t+1) = \begin{cases} 1 & \text{if } rand(0,1) < T(\Delta X_i^d(t+1)) \\ 0 & \text{otherwise} \end{cases} \quad (25)$$

In this context, $rand(0,1)$ denotes a random number in the range $[0,1]$, and $T(x)$ refers to an S-shaped transfer function. X_i^d indicates the position of the i^{th} particle in the d^{th} dimension, while t represents the current iteration. Unlike the S-shaped transfer

$$X_i^d(t+1) = \begin{cases} -X_i^d(t) & \text{if } rand(0,1) < T(\Delta X_i^d(t+1)) \\ X_i^d(t) & \text{otherwise} \end{cases} \quad (26)$$

In continue, partially mapped crossover (PMX) in genetic algorithms addresses infeasibility in permutation problems by exchanging values between two parent chromosomes at selective genes. When values are swapped, any repeated value in one parent is substituted with the corresponding mapped value from the other parent, and vice versa [54]. Inspired by PMX, this study employs an updating mechanism in CH-ALAO to generate feasible layouts. In this mechanism, particle fitness is calculated and used as a criterion to determine the update order, with particles having higher fitness values being closer to the goal and thus updated first.

5. Results and discussions

This section presents and analyzes the results of the proposed methods. The experiments were conducted using MATLAB software on a Core i7 processor with 8 GB of RAM. The following section details the case studies and their corresponding results. The performance of the presented method is assessed in three steps in this section.

We first applied the presented algorithm 10 times to the appropriate functions and recorded the maximum, minimum, median, and mean of the iterations in Tables 6 and 7 in subsection 5.1. All the outcomes in this paper follow the IEEE CEC 2005 supported format. The outcomes are highlighted by the thick pen in these tables. The algorithm performs 1000 searches each time it is fully run. The

to Equations 25 and 26, with S-shaped functions specifically updating particle positions as outlined in Equation 25.

function, the V-shaped transfer function does not limit the search agent to the $[0,1]$ interval. Equation 26 illustrates the update process for the particle's position using V-shaped transfer functions, where $\neg X$ signifies the complement of X .

population size is set to 30, with each solution comprising a set of 30 elements. To ensure a fair and accurate comparison, the population and computational capabilities of the algorithms compared are kept consistent.

In the second and third phases, two case studies are performed to showcase the effectiveness and applicability of the CH-ALAO metaheuristic for optimizing construction site layouts. The outcomes were compared with those obtained using the ALO, AO, and GWO algorithms. The binary version of the algorithms that are compared is used. We described the details of the second and third step experiments in subsection 5.2, and 5.3.

5.1. Benchmark functions

The effectiveness of the exploitation process can be assessed using unimodal benchmark functions. The performance of the basic algorithms, the presented algorithm, and some new algorithms on unimodal exponential functions is summarized in Table 6 using average, median, minimum, and maximum values. The presented algorithm has a better exploitation stage than the other algorithms, as shown in Table 6. The exploitation stage of the presented algorithm is improved significantly by using the ALO algorithm, compared to the basic Aquila optimizer algorithm and other compared algorithms.

Table 6. Results for the unimodal benchmark (F1-F7)

<i>F1</i>								
	CH- ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	6.4311e-43	8.2763e-06	1.4717e-27	2.5393e-33	8.8970e-35	0.002213	1.2821	5408.30
<i>Med</i>	5.7219e-43	7.0739e-06	2.5708e-28	2.3910e-34	6.1580e-35	0.003157	1.2199	348.20
<i>Worst</i>	2.3140e-41	6.0059e-05	6.3424e-27	1.0620e-32	1.9140e-34	0.005176	1.7088	6901.30
<i>Best</i>	2.3425e-43	9.0284e-06	1.9719e-28	2.0561e-35	3.2210e-36	0.000212	1.0346	21.5905
<i>F2</i>								
	CH- ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	3.1723e-26	0.06129	5.0014e-16	2.2021e-19	2.1925e-20	23.5150	0.6270	27.8018
<i>Med</i>	3.2021e-26	0.05801	4.4168e-16	2.5013e-19	2.6509e-20	22.7011	0.6023	29.0291
<i>Worst</i>	3.1223e-25	0.16602	2.2137e-15	5.5084e-19	3.8126e-20	42.4121	0.8423	29.4062
<i>Best</i>	2.2369e-26	0.00780	3.6019e-16	0.1372e-19	5.0095e-21	9.1270	0.4012	20.0113
<i>F3</i>								
	CH- ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	4.1303e-9	65.1090	3.3096e-07	6.3341e-09	3.0211e-08	3029.6101	267.5031	8840.0019
<i>Med</i>	3.6505e-9	68.0677	6.6690e-07	5.7101e-09	3.2009e-08	3299.0090	290.7091	8390.5582
<i>Worst</i>	4.5608e-9	98.2266	7.3201e-06	3.7204e-08	6.5150e-08	6772.0291	320.2920	9920.9300
<i>Best</i>	1.3918e-10	52.4072	4.0157e-08	2.0970e-10	2.0118e-09	2876.4800	201.0284	7013.3202
<i>F4</i>								
	CH- ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	2.0117e-10	4.6584	3.3228e-07	4.1985e-09	3.0174e-08	32.2081	7.1220	120.2037

<i>Med</i>	2.1104e-10	4.2895	3.1092e-07	4.2022e-09	5.0938e-08	30.0106	7.2107	122.4992
<i>Worst</i>	4.2012e-10	4.0115	4.8013e-07	5.0153e-09	5.0613e-08	42.0931	9.0056	145.0078
<i>Best</i>	2.1700e-11	2.7010	2.2430e-07	2.3120e-09	2.52326e-09	21.1004	3.1004	60.5600
<i>F5</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	10.1873	28.5121	25.7456	28.3340	29.0136	125.9000	220.1034	450.8001
<i>Med</i>	9.0293	26.5104	25.0940	28.0977	28.2348	132.3340	225.4500	429.5000
<i>Worst</i>	17.0096	32.0170	27.1101	29.0030	32.8209	230.7000	290.4560	1087.000
<i>Best</i>	9.0102	21.6110	22.0398	24.7870	24.1099	100.1128	178.5130	400.6453
<i>F6</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	2.0029e-05	0.00001	1.2029	1.1001	1.0309	0.0021	0.8902	13.0931
<i>Med</i>	4.092e-04	0.00012	1.0433	1.9100	1.9600	0.0021	1.2360	12.6234
<i>Worst</i>	0.0029	0.00029	1.9930	2.0702	2.0031	0.0017	1.9007	17.0901
<i>Best</i>	1.1008e-05	1.722e-05	0.2506	1.0015	1.2130	0.0006	0.7001	7.0909
<i>F7</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	0.0002	0.1565	0.0034	0.0022	0.0019	0.2320	0.0310	0.0170
<i>Med</i>	0.0019	0.2009	0.0025	0.0019	0.0014	0.2400	0.0290	0.0022
<i>Worst</i>	0.0022	0.2097	0.0039	0.0027	0.0027	0.5510	0.0700	2.1005
<i>Best</i>	0.0001	0.1200	0.0018	0.0005	0.0009	0.2002	0.0218	0.0020

Multimodal benchmark functions test how well the search algorithm can analyze the search area and evade from local optima. The CEC 2005 test problems include multimodal functions from F8 to F16. You can find the details of the unimodal and multimodal benchmark functions, such as their dimensions, ranges, minimum inputs, and cost functions, in [7]. Table 7 displays the statistical results for the multimodal functions. The proposed algorithm excels over other algorithms in all multimodal functions except for F15 and F16. The PSO algorithm surpasses the others specifically on the F15 function, and similarly, achieves comparable results on the F16 function.

Table 7. Results for the multimodal benchmark (F8-F15)

<i>F8</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	-9003.12	-5361.93	-5817.79	-5909.66	-5645.68	- 6247.32	- 7228.41	- 7087.49
<i>Med</i>	-8900.40	-6587.37	-5632.88	-5730.73	-5775.16	- 5518.28	- 7657.85	- 7196.89
<i>Worst</i>	-6282.91	-3143.67	-4233.70	-3260.13	-4850.06	- 3417.67	- 6047.43	- 6289.38
<i>Best</i>	-9124.62	-6600.24	-6401.59	-6768.75	-6003.93	- 6016.27	- 8132.06	- 8064.13
<i>F9</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	2.5001e-14	20.17	7.07	3.96	0.81	78.53	90.43	130.07
<i>Med</i>	2.2437e-13	19.55	0	0	0	77.66	92.18	143.14
<i>Worst</i>	4.3521e-12	73.16	20.15	8.70	4.84	91.56	103.97	150.08
<i>Best</i>	1.1019e-14	0	0	0	4.7643e-14	50.73	80.25	42.26
<i>F10</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	4.3200e-15	0.1901	1.0489e-13	1.5081e-14	5.0122e-13	5.0755	4.0012	20.37
<i>Med</i>	4.3100e-15	0.0161	1.0476e-13	1.5190e-14	7.2800e-13	3.0132	4.8170	19.82
<i>Worst</i>	4.7511e-15	0.9214	1.1810e-13	1.8400e-14	1.2810e-12	10.108	15.840	23.95
<i>Best</i>	3.0873e-15	0.0094	2.3200e-14	0.8798e-14	8.5900e-14	2.1205	2.4081	12.12
<i>F11</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	0.0002	0.0089	0.0070	0.0040	0.0077	0.0493	0.8967	12.9006
<i>Med</i>	0	0.0092	0.0062	0.0032	0.0079	0.0467	0.9100	13.0114
<i>Worst</i>	0.0068	0.0190	0.0250	0.0200	0.0194	0.0612	0.9420	90.1040
<i>Best</i>	0	1.3011e-05	0	0	0	0.0377	0.8345	10.9600
<i>F12</i>								
	CH-ALAO	PSO	AO	DA	WOA	ALO	MVO	MFO

<i>Avg</i>	4.0490e-06	0.0034	0.0422	0.2022	0.0198	12.0039	5.4905	7.0910
<i>Med</i>	4.0204e-06	4.0036e-05	0.0397	0.1248	0.0133	10.5035	5.2503	7.1995
<i>Worst</i>	6.0954e-06	0.0084	0.0690	0.6697	0.0658	20.5500	8.4505	10.059
<i>Best</i>	3.1002e-07	1.6014e-05	0.0254	0.0229	0.0115	10.1784	2.9052	2.9690
<i>F13</i>								
	CH-ALOAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	0.0063	0.1263	0.5109	2.9032	0.6334	19.4474	0.5798	12.6002
<i>Med</i>	0.0001	0.0919	0.4129	2.3042	0.6124	20.0098	0.1206	11.4670
<i>Worst</i>	0.0110	0.3909	0.8256	2.8061	0.7949	32.6724	0.1630	20.4028
<i>Best</i>	1.2010e-06	4.2200e-05	0.1504	0.8664	0.4584	3.1091	0.0350	8.7490
<i>F14</i>								
	CH-ALOAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	0.0947	4.3014	4.1815	2.2380	1.7900	2.1002	0.9980	3.1500
<i>Med</i>	0.0761	3.9021	4.9820	1.7093	0.9980	1.9902	0.9980	2.9820
<i>Worst</i>	1.0025	6.9900	6.9820	5.6903	2.9902	2.9820	0.9980	5.9280
<i>Best</i>	0.0018	0.9980	0.9980	0.9980	0.9980	0.9980	0.9980	0.9980
<i>F15</i>								
	CH-ALOAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	0.0086	0.0009	0.0046	0.0083	0.0058	0.0048	0.0046	0.0014
<i>Med</i>	0.0006	0.0009	0.0012	0.0008	0.0075	0.0034	0.0075	0.0008
<i>Worst</i>	0.0203	0.0010	0.0203	0.0303	0.0233	0.0203	0.0203	0.0016
<i>Best</i>	0.0003	0.0007	0.0003	0.0007	0.0004	0.0003	0.0005	0.0005
<i>F16</i>								
	CH-ALOAO	PSO	AO	DA	WOA	ALO	MVO	MFO
<i>Avg</i>	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
<i>Med</i>	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
<i>Worst</i>	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
<i>Best</i>	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316

5.2. Facilities Case Study 1

This case study, based on a medium-sized project from Li et al. [55], focuses on finding the best arrangement for positioning 11 facilities into 11 designated locations on the site. The facilities considered in this analysis are:

- | | |
|------------------------|---|
| 1. Site office | 7. Reinforcement steel workshop |
| 2. False work workshop | 8. Side gate |
| 3. Labor residence | 9. Electrical, water and other utilities control room |
| 4. Storeroom 1 | 10. Concrete batch workshop |
| 5. Storeroom 2 | 11. Main gate |
| 6. Carpentry workshop | |

In this case study, the construction site layout is based on two key assumptions:

1. Any facility can be placed at any of the predetermined locations.
2. The main gate and side gates are treated as fixed facilities, positioned in their designated locations.

The goal of this case study is to minimize the total travel distance for site personnel moving between facilities. According to [55], the total travel distance is calculated as:

$$\begin{aligned} \text{Minimize } TD &= \sum_{i=1}^n \sum_{j=1}^n \sum_{l=1}^n \sum_{k=1}^n x_{ik} \times x_{jl} \times f_{ij} \times d_{kl} \\ \text{Subjected to } \sum_{i=1}^n x_{ij} &= 1, \sum_{j=1}^n x_{ij} = 1 \end{aligned} \quad (27)$$

Here, n represents the number of facilities. The variable $x_{ik} = 1$ equals 1 if facility i is assigned to location k ; otherwise, it is 0. Similarly, x_{jl} follows the same principle. The coefficient f_{ij} indicates the number of trips made daily by construction personnel between facilities i and j , while d_{kl} represents the distance between locations k and l . Consequently, TD denotes the total daily travel distance for construction personnel. The distances between the predetermined locations are recorded and detailed in Table 8. The frequency of trips between facilities plays a crucial role in site layout planning and the positioning of site facilities. Therefore, the daily trip frequencies between facilities are summarized in Table 9.

Table 8. Travel distance between predetermined locations

Distance		Location										
		1	2	3	4	5	6	7	8	9	10	11
Location	1	0	15	25	33	40	42	47	55	35	30	20
	2	15	0	10	18	25	27	32	42	50	45	35
	3	25	10	0	8	15	17	22	32	52	55	45
	4	33	18	8	0	7	9	14	24	44	49	53
	5	40	25	15	7	0	2	7	17	37	42	52
	6	42	27	17	9	2	0	5	15	35	40	50
	7	47	32	22	14	7	5	0	10	30	35	40
	8	55	42	32	24	17	15	10	0	20	25	35
	9	35	50	52	44	37	35	30	20	0	5	15
	10	30	45	55	49	42	40	35	25	5	0	10
	11	20	35	45	53	52	50	40	35	15	10	0

	11	20	35	45	53	52	50	40	35	15	10	0
Table 9. Trip frequency between structures												
Trip frequency		Facility										
		1	2	3	4	5	6	7	8	9	10	11
Facility	1	0	5	2	2	1	1	4	1	2	9	1
	2	5	0	2	5	1	2	7	8	2	3	8
	3	2	2	0	7	4	4	9	4	5	6	5
	4	2	5	7	0	8	7	8	1	8	5	1
	5	1	1	4	8	0	3	4	1	3	3	6
	6	1	2	4	7	3	0	5	8	4	7	5
	7	4	7	9	8	4	5	0	7	6	3	2
	8	1	8	4	1	1	8	7	0	9	4	8
	9	2	2	5	8	3	4	6	9	0	5	3
	10	9	3	6	5	3	7	3	4	5	0	5
	11	1	8	5	1	6	5	2	8	3	5	0

This example was evaluated through 50 independent optimization runs, each consisting of 500 iterations, to obtain statistically significant results with various algorithms. The statistical results from these runs are summarized in Table 10. According to Table 10, the proposed method achieves an average result of 12,480, a worst-case result of 13,110, and a standard deviation of 62.5, outperforming other algorithms. The table also highlights that the best outcome in this case study is 12,460, which is better than the results from all compared algorithms. This demonstrates that the CH-ALOAO method not only finds a superior optimal solution but also offers enhanced stability. Figure 5 displays the convergence curves of the different algorithms over the course of the iterations.

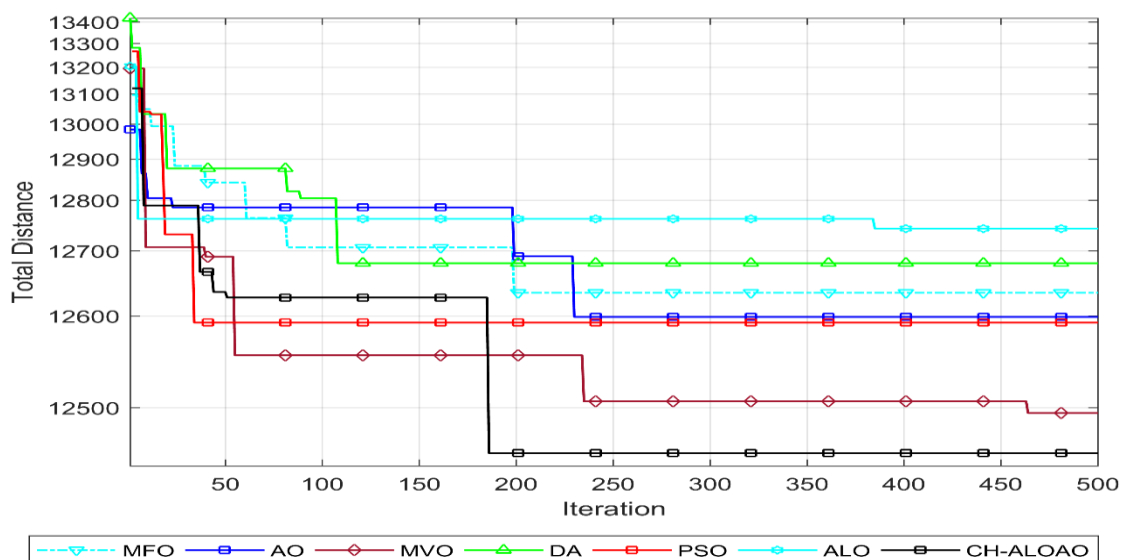


Figure 5. Convergence curve

Table 10. Comparison of the results of 50 independent runs

Algorithms	CH-ALOAO	PSO	AO	DA	ALO	MVO	MFO
Best	12460	12600	12615	12680	12750	12485	12650
Average	12480	12630	12705	12710	12752	12550	12675
Worst	13110	13290	12990	13400	13200	13198	13100
STD	62.5	172.5	93.75	180	112.5	178.25	112.5

5.3. Facilities Case Study 2

The necessity of relocating temporary project facilities to reduce transportation expenses becomes especially critical in projects with significant spatial and temporal dispersion, like road construction projects. This case study focuses on a project with highly dynamic development and extensive spatial dispersion [56].

The operational plan entails positioning ten major facilities across 11 designated locations within the site. Additionally, three key work areas have been

identified to represent actual work zones along the road. These work areas, which handle substantial material transportation from other facilities, are considered essential for the construction site. Table 11 details the facilities, including their estimated initial placement and potential relocation costs. Among these, the three work areas and the quarry are fixed in their positions. Table 12 shows the types of movements, rated from 1 (primarily human resources) to 5 (primarily heavy vehicles), while Table 13 provides a comparative scale of trip frequencies between facilities, ranging from 1 to 5.

Table 11. Costs associated with constructing and relocating facilities.

Facility	Construction cost	Relocation cost
1. Quarry area (fixed)	90000	0
2. Stone crusher	100000	25000
3. Concrete batch plant	150000	25000
4. Asphalt mixing plant	120000	25000
5. Concrete and aggregates depot	10000	5000
6. Asphalt and aggregates depot	10000	5000
7. Sub-base and aggregates depot	10000	5000
8. Work field 1 (fixed)	0	0
9. Work field 2 (fixed)	0	0
10. Work field 3 (fixed)	0	0
11. Asphalt storage	15000	0
12. Site office	10000	0
13. Labor rest area	10000	0
14. Concrete (cement) storage	15000	0
15. -	0	0

Table 12. Type of movements between facilities

Facility

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Facility	1	0	5	1	1	1	1	1	1	1	1	1	1	1	1	0
	2	5	0	1	1	5	5	5	1	1	1	1	1	1	1	0
	3	1	1	0	1	4	1	1	4	1	1	1	1	1	3	0
	4	1	1	1	0	1	4	1	1	1	4	3	1	1	1	0
	5	1	5	4	1	0	1	1	1	1	1	1	1	1	2	0
	6	1	5	1	4	1	0	1	1	1	1	2	1	1	1	0
	7	1	5	1	1	1	1	0	1	4	1	1	1	1	1	0
	8	1	1	4	1	1	1	1	0	1	1	1	2	1	1	0
	9	1	1	1	1	1	1	4	1	0	1	1	2	1	1	0
	10	1	1	1	4	1	1	1	1	1	0	1	2	1	1	0
	11	1	1	1	3	1	2	1	1	1	1	0	1	1	1	0
	12	1	1	1	1	1	1	1	2	2	2	1	0	3	1	0
	13	1	1	1	1	1	1	1	1	1	1	1	3	0	1	0
	14	1	1	3	1	2	1	1	1	1	1	1	1	1	0	0
	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 13. Number of trips between facilities

		Facility														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Facility	1	0	5	1	1	1	1	1	1	1	1	1	1	1	0	0
	2	5	0	1	1	3	3	3	1	1	1	1	1	1	1	0
	3	1	1	0	1	3	1	1	3	1	1	1	1	1	2	0
	4	1	1	1	0	1	3	1	1	1	3	2	1	1	1	0
	5	1	3	3	1	0	1	1	1	1	1	1	1	1	2	0
	6	1	3	1	3	1	0	1	1	1	1	2	1	1	1	0
	7	1	3	1	1	1	1	0	1	3	1	1	1	1	1	0
	8	1	1	3	1	1	1	1	0	1	1	1	1	1	1	0
	9	1	1	1	1	1	1	3	1	0	1	1	1	1	1	0
	10	1	1	1	3	1	1	1	1	1	0	1	1	1	1	0
	11	1	1	1	2	1	2	1	1	1	1	0	1	1	1	0
	12	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0
	13	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0
	14	1	1	2	1	2	1	1	1	1	1	1	1	1	0	0

15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

In this situation, it's highly advantageous to place the site office and labor rest area as far from the construction activities as possible. The most significant deviations from optimal solutions occur when the cost of relocating facilities outweighs the savings from reduced transportation costs, making rearrangement less appealing. In such cases, the model needs to dynamically address the static problem, with any deviation from the static solution leading to significant differences in the objective function value. Conversely, when low relocation costs incentivize facility rearrangement, multiple solutions with minimal deviations from the optimal can be identified, enhancing accuracy. However, as the problem size increases, the model's efficiency in tackling larger CSLP issues noticeably declines.

In this example, 20 independent optimization runs of 500 iterations each were conducted to achieve statistically significant results using various algorithms. The statistical outcomes from these runs are compared in Table 14. The proposed method shows superior performance, with an average result of 14,540, a worst-case result of 16,280, and a standard deviation of 107.45, outperforming other algorithms. Furthermore, Table 14 shows that the best result achieved is 14,500, surpassing the outcomes of all other algorithms. This highlights that CH-ALAO not only secures a more optimal solution but also exhibits improved stability. Figure 6 displays the convergence curves of different algorithms across iterations.

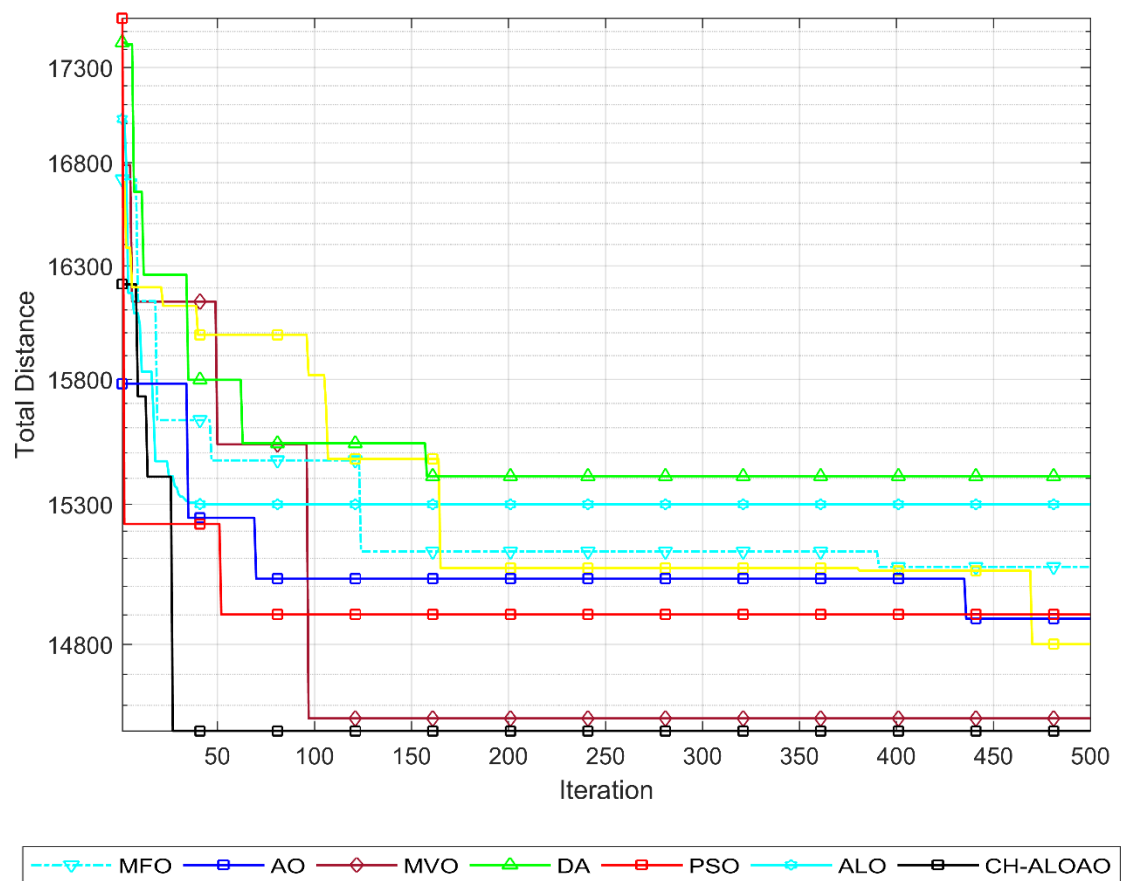


Figure 6. Convergence curve of algorithms

Table 14. Results comparison from 50 independent runs

Algorithms	CH-ALAO	PSO	AO	DA	ALO	MVO	MFO
Best	14500	14900	14890	15400	15300	14550	15050
Average	14540	15050	15210	15540	15350	15000	15120
Worst	16280	17600	15800	17400	17000	16800	16700
STD	107.45	309.5	94.23	227	194	243	190.46

6. Conclusion and Future Work

In this study, we introduce a pioneering approach: the Hybrid Ant Lion Optimizer Algorithm (ALO) and Aquila Optimizer Algorithm (AO) based on chaos theory, specifically tailored for optimizing construction management tasks. This hybrid algorithm integrates ALO and AO, leveraging their respective strengths. ALO mimics the trapping behavior of antlions, balancing exploration and exploitation for optimal outcomes. Conversely, AO emulates the dynamic hunting tactics of eagles, facilitating agile and adaptive search strategies. By combining these approaches, the hybrid algorithm adeptly navigates complex optimization problems, dynamically adapting to environmental changes. The CH-ALAO algorithm utilizes interactive memory to store optimal solutions, enhancing computational efficiency. Experimental findings reveal that CH-ALAO excels in addressing construction site layout planning (CSLP) problems, surpassing conventional metaheuristic algorithms in both solution quality and convergence speed.

To validate the models, two case studies were conducted. The results demonstrate that the proposed CH-ALAO approach not only achieved better results but also required fewer evaluations to find the optimal solution. A comparative analysis with existing metaheuristic methods highlights the effectiveness and efficiency of the CH-ALAO algorithm in tackling construction site layout planning (CSLP) problems. CH-ALAO stands out for its high-quality solutions and rapid convergence to optimal outcomes, showcasing its robustness and effectiveness for complex optimization tasks in construction management.

Declarations:

Ethical Approval

not applicable.

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Authors' contributions

Amirhosein Hojati, Aliasghar Amirkardoust, Davoud Sedaghat Shayegan and Kamand Sedaghat Shayegan contributed to the design and implementation of the research, to the analysis of the results and the writing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Availability of data and materials

Data generated during the current study are available from the corresponding author upon reasonable request.

References

- [1] X. Ning, K.-C. Lam, and M. C.-K. Lam, "A decision-making system for construction site layout planning," *Automation in construction*, vol. 20, no. 4, pp. 459-473, 2011.
- [2] J. Xu and Z. Li, "Multi-Objective Dynamic Construction Site Layout Planning in Fuzzy Random Environment," *Automation in Construction*, vol. 27, pp. 155-169, 2012/11/01/ 2012.
- [3] L.-C. Lien and M.-Y. Cheng, "A hybrid swarm intelligence based particle-bee algorithm for construction site layout optimization," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9642-9650, 2012/08/01/ 2012.

- [4] P. P. Zouein and I. D. Tommelein, "Improvement algorithm for limited space scheduling," *Journal of construction engineering and management*, vol. 127, no. 2, pp. 116-124, 2001.
- [5] M. Yahya and M. Saka, "Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights," *Automation in construction*, vol. 38, pp. 14-29, 2014.
- [6] A. Mohammadzadeh, M. Masdari, and F. S. Gharehchopogh, "Energy and Cost-Aware Workflow Scheduling in Cloud Computing Data Centers Using a Multi-objective Optimization Algorithm," *Journal of Network and Systems Management*, vol. 29, no. 3, p. 31, 2021/04/05 2021.
- [7] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "Improved chaotic binary grey wolf optimization algorithm for workflow scheduling in green cloud computing," *Evolutionary Intelligence*, vol. 14, no. 4, pp. 1997-2025, 2021/12/01 2021.
- [8] A. Mohammadzadeh and M. Masdari, "Scientific workflow scheduling in multi-cloud computing using a hybrid multi-objective optimization algorithm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, pp. 3509-3529, 2023/04/01 2023.
- [9] A. Chhabra, S. K. Sahana, N. S. Sani, A. Mohammadzadeh, and H. A. Omar, "Energy-Aware Bag-of-Tasks Scheduling in the Cloud Computing System Using Hybrid Oppositional Differential Evolution-Enabled Whale Optimization Algorithm," *Energies*, vol. 15, no. 13, p. 4571, 2022.
- [10] A. Mohammadzadeh, M. Akbari Zarkesh, P. Haji Shahmohamd, J. Akhavan, and A. Chhabra, "Energy-aware workflow scheduling in fog computing using a hybrid chaotic algorithm," *The Journal of Supercomputing*, vol. 79, no. 16, pp. 18569-18604, 2023/11/01 2023.
- [11] A. Mohammadzadeh, D. Javaheri, and J. Artin, "Chaotic hybrid multi-objective optimization algorithm for scientific workflow scheduling in multisite clouds," *Journal of the Operational Research Society*, vol. 75, no. 2, pp. 314-335, 2024/02/01 2024.
- [12] A. Mohammadzadeh, A. Chhabra, S. Mirjalili, and A. Faraji, "Chapter 4 - Use of whale optimization algorithm and its variants for cloud task scheduling: a review," in *Handbook of Whale Optimization Algorithm*, S. Mirjalili, Ed.: Academic Press, 2024, pp. 47-68.
- [13] A. Ghadiri, D. Sedaghat Shayegan, and A. Amirkardoust, "Multi-objective firefly optimization algorithm for construction site layout planning," *Iranian Journal of Optimization*, vol. 14, no. 4, pp. 245-260, 2022.
- [14] S. Shahebrahimi, A. Lork, D. S. Shayegan, and A. Amir, "Solving the Problem of Multi-Stakeholder Construction Site Layout Using Metaheuristic Algorithms."
- [15] S. Shahebrahimi, A. Lork, D. Sedaghat Shayegan, and A. Kardoust, "IMPACT OF CONSTRUCTION SITE LAYOUT PLANNING OPTIMIZATION ON CONSTRUCTION PROJECT MANAGEMENT (CASE STUDY OF LAUNCHER/RECEIVER STATIONS IN PIPELINE PROJECTS IN KHORASAN PROVINCE)," *Int. J. Optim. Civil Eng*, vol. 14, no. 1, pp. 83-93, 2024.
- [16] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29-41, 1996.
- [17] K. C. Lam, X. Ning, and T. Ng, "The application of the ant colony optimization algorithm to the construction site layout planning problem," *Construction Management and Economics*, vol. 25, no. 4, pp. 359-374, 2007/04/01 2007.
- [18] E. Gharai, A. Afshar, and M. R. Jalali, "Site layout optimization with ACO algorithm," in *Proceedings of the 5th WSEAS international conference on artificial intelligence, knowledge engineering and data bases*, 2006, pp. 90-94.
- [19] X. Ning, K.-C. Lam, and M. C.-K. Lam, "Dynamic construction site layout planning

- using max-min ant system," *Automation in Construction*, vol. 19, no. 1, pp. 55-65, 2010/01/01/ 2010.
- [20] H. Li and P. E. D. Love, "Genetic search for solving construction site-level unequal-area facility layout problems," *Automation in Construction*, vol. 9, no. 2, pp. 217-226, 2000/03/01/ 2000.
- [21] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39-43.
- [22] A. Hojati, a. Amirkardoust, and D. Sedaghat Shaygan, "An overview of the factors related to building information modeling and evaluating its effects on the performance of construction projects," *Civil and Project*, vol. 5, no. 12, pp. 29-41, 2024.
- [23] H. Zhang and J. Y. Wang, "Particle swarm optimization for construction site unequal-area layout," *Journal of construction engineering and management*, vol. 134, no. 9, pp. 739-748, 2008.
- [24] A. M. Adrian, A. Utamima, and K.-J. Wang, "A comparative study of GA, PSO and ACO for solving construction site layout optimization," *KSCE Journal of Civil Engineering*, vol. 19, no. 3, pp. 520-527, 2015/03/01 2015.
- [25] S.-O. Cheung, T. K.-L. Tong, and C.-M. Tam, "Site pre-cast yard layout arrangement through genetic algorithms," *Automation in Construction*, vol. 11, no. 1, pp. 35-46, 2002/01/01/ 2002.
- [26] M. J. Mawdesley and S. H. Al-Jibouri, "Proposed genetic algorithms for construction site layout," *Engineering Applications of Artificial Intelligence*, vol. 16, no. 5, pp. 501-509, 2003/08/01/ 2003.
- [27] H. M. Osman, M. E. Georgy, and M. E. Ibrahim, "A hybrid CAD-based construction site layout planning system using genetic algorithms," *Automation in Construction*, vol. 12, no. 6, pp. 749-764, 2003/11/01/ 2003.
- [28] D. Karaboğa, "AN IDEA BASED ON HONEY BEE SWARM FOR NUMERICAL OPTIMIZATION," 2005.
- [29] D. S. Shayegan, "Optimum cost design of reinforced concrete slabs using a metaheuristic algorithm," *International Journal of Optimization in Civil Engineering*, vol. 12, no. 4, pp. 545-555, 2022.
- [30] A. Saberi and D. Sedaghat Shayegan, "Optimization of Haraz dam reservoir operation using CBO metaheuristic algorithm," *Int J Optim Civil Eng*, vol. 11, no. 4, pp. 581-92, 2021.
- [31] A. Saberi, H. Ahmadi, D. S. Shayegan, and A. Amirkardoust, "Prediction of electricity consumption using three meta-heuristic algorithms," *Int. J. Optim. Civil Eng*, vol. 13, no. 1, pp. 111-125, 2023.
- [32] D. Sedaghat Shayegan, A. Lork, and S. A. H. Hashemi, "Optimum Cost Design of Reinforced Concrete Slabs Using Mouth Brooding Fish Algorithm," *Journal of Applied Engineering Sciences*, vol. 10, no. 1, 2020.
- [33] D. M. Tate* and A. E. Smith, "Unequal-area facility layout by genetic search," *IIE Transactions*, vol. 27, no. 4, pp. 465-472, 1995/08/01 1995.
- [34] I. N. Papadaki and A. P. Chassiakos, "Multi-objective Construction Site Layout Planning Using Genetic Algorithms," *Procedia Engineering*, vol. 164, pp. 20-27, 2016/01/01/ 2016.
- [35] M. Yahya and M. P. Saka, "Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights," *Automation in Construction*, vol. 38, pp. 14-29, 2014/03/01/ 2014.
- [36] I. C. Yeh, "Architectural layout optimization using annealed neural network," *Automation in Construction*, vol. 15, no. 4, pp. 531-539, 2006/07/01/ 2006.
- [37] H. Zhang and Y. Wang Jia, "Particle Swarm Optimization for Construction Site Unequal-Area Layout," *Journal of Construction Engineering and Management*, vol. 134, no. 9, pp. 739-748, 2008/09/01 2008.
- [38] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80-98, 2015.
- [39] 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.
- [40] H. Yu, H. Jia, J. Zhou, and A. Hussien, "Enhanced Aquila optimizer algorithm for global optimization and constrained engineering problems," *Mathematical Biosciences and Engineering*, vol. 19, no. 12, pp. 14173-14211, 2022.
- [41] A. A. Abou El-Ela, R. A. El-Sehiemy, A. M. Shaheen, and A. S. Shalaby, "Aquila Optimization Algorithm for Wind Energy Potential Assessment Relying on Weibull Parameters Estimation," *Wind*, vol. 2, no. 4, pp. 617-635, 2022.
- [42] J. Zhao, Z.-M. Gao, and H.-F. Chen, "The simplified aquila optimization algorithm," *IEEE Access*, vol. 10, pp. 22487-22515, 2022.
- [43] M. Wang, A. A. Heidari, M. Chen, H. Chen, X. Zhao, and X. Cai, "Exploratory Differential Ant Lion-based Optimization," *Expert Systems with Applications*, p. 113548, 2020.
- [44] W.-y. Guo, Y. Wang, F. Dai, and P. Xu, "Improved sine cosine algorithm combined with optimal neighborhood and quadratic interpolation strategy," *Engineering Applications of Artificial Intelligence*, vol. 94, p. 103779, 2020.
- [45] S. Gupta, K. Deep, and A. P. Engelbrecht, "A memory guided sine cosine algorithm for global optimization," *Engineering Applications of Artificial Intelligence*, vol. 93, p. 103718, 2020.
- [46] Y. Fan *et al.*, "Rationalized Fruit Fly Optimization with Sine Cosine Algorithm: A Comprehensive Analysis," *Expert Systems with Applications*, p. 113486, 2020.
- [47] S. Gupta, K. Deep, S. Mirjalili, and J. H. Kim, "A Modified Sine Cosine Algorithm with Novel Transition Parameter and Mutation Operator for Global Optimization," *Expert Systems with Applications*, p. 113395, 2020.
- [48] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "A hybrid multi-objective metaheuristic optimization algorithm for scientific workflow scheduling," *Cluster Computing*, vol. 24, no. 2, pp. 1479-1503, 2021/06/01 2021.
- [49] A. Mohammadzadeh, M. Akbari Zarkesh, P. Haji Shahmohamd, J. Akhavan, and A. Chhabra, "Energy-aware workflow scheduling in fog computing using a hybrid chaotic algorithm," *The Journal of Supercomputing*, 2023/05/16 2023.
- [50] M. Kohli and S. Arora, "Chaotic grey wolf optimization algorithm for constrained optimization problems," *Journal of computational design and engineering*, vol. 5, no. 4, pp. 458-472, 2018.
- [51] A. Mukherjee and V. Mukherjee, "Chaotic krill herd algorithm for optimal reactive power dispatch considering FACTS devices," *Applied Soft Computing*, vol. 44, pp. 163-190, 2016.
- [52] S. Saremi, S. Mirjalili, and A. Lewis, "Biogeography-based optimisation with chaos," *Neural Computing and Applications*, vol. 25, no. 5, pp. 1077-1097, 2014.
- [53] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 9, pp. 1-14, 2013.
- [54] A. Kaveh, M. Khanzadi, M. Alipour, and M. R. Moghaddam, "Construction Site Layout Planning Problem Using Two New Meta-heuristic Algorithms," *Iranian Journal of Science and Technology, Transactions of Civil Engineering*, vol. 40, no. 4, pp. 263-275, 2016/12/01 2016.
- [55] H. Li and E. D. Love Peter, "Site-Level Facilities Layout Using Genetic Algorithms," *Journal of Computing in Civil Engineering*, vol. 12, no. 4, pp. 227-231, 1998/10/01 1998.
- [56] P. M. Farmakis, "Genetic algorithm optimization for dynamic construction site layout planning," *Organization, technology & management in construction: an international journal*, vol. 10, no. 1, pp. 1655-1664, 2018.