International Journal of Scientific World, 11 (2) (2025) 38-49

International Journal of Scientific World

Website: www.sciencepubco.com/index.php/IJSW
https://doi.org/10.14419/0hndc578
Review paper

Metaheuristic Algorithms for Engineering and Combinatorial Optimization: A Comparative Study Across Problems Categories and Benchmarks

Awaz Ahmed Shaban [1] *, Saman M. Almufti [2,3], Renas Rajab Asaad [2]

[1] Information Technology Department, Technical College of Informatics-Akre, Akre University for Applied Sciences, Duhok, Iraq
[2] Department of Computer Science, College of Science, Knowledge University, Erbil, Iraq
[3] Department of Computer Systems, Ararat Technical Institute, Duhok, Iraq
**Corresponding author E-mail: Saman.Almofty@gmail.com*

Received: June 22, 2025, Accepted: August 11, 2025, Published: August 23, 2025

Abstract

Optimization remains a cornerstone of modern engineering and computational intelligence, playing a vital role in the design, control, and allocation of limited resources across industries ranging from logistics to structural engineering. Traditional optimization methods, such as gradient-based and exact algorithms, often struggle with the nonlinear, multimodal, and constrained nature of real-world problems, necessitating the adoption of metaheuristic approaches. These biologically and physically inspired algorithms offer flexibility, scalability, and robustness in navigating complex search spaces.

This study presents a systematic categorization of optimization problems—including combinatorial, continuous, constrained, and multi-objective classes—followed by a rigorous comparative analysis of nine prominent metaheuristics: Ant Colony Optimization (ACO), Lion Algorithm (LA), Cuckoo Search (CS), Grey Wolf Optimizer (GWO), Vibrating Particles System (VPS), Social Spider Optimization (SSO), Cat Swarm Optimization (CSO), Bat Algorithm (BA), and Artificial Bee Colony (ABC). The algorithms are evaluated across five representative benchmark problems: the Traveling Salesman Problem (TSP), Welded Beam Design (WBD), Pressure Vessel Design (PVD), Tension/Compression Spring Design (TSD), and the Knapsack Problem (KP).

Key contributions include: 1)Domain-specific suitability analysis, revealing how algorithmic mechanisms align with problem structures. 2) Performance benchmarking under standardized conditions, highlighting convergence speed, solution quality, and constraint-handling efficacy. 3) Practical insights for practitioners on algorithm selection, hybridization potential, and adaptation challenges.

Results demonstrate that no single algorithm dominates universally; instead, problem characteristics dictate optimal choices. For instance, ACO excels in discrete problems (TSP, KP), while GWO and BA outperform in continuous engineering designs (WBD, PVD). The study concludes with recommendations for future research, including dynamic parameter tuning, hybrid models, and real-world scalability assessments.

Keywords: *Optimization Problem Categories; Metaheuristic Algorithms; Engineering Design Optimization; Benchmark Problems; Combinatorial Optimization; Swarm Intelligence; Constraint Handling; Algorithm Comparison; Structural Design; NP-Hard Problems.*

1. Introduction

Optimization lies at the core of modern engineering and operations research, providing systematic methods for improving performance, reducing costs, and maximizing efficiency across a wide range of industries. From the routing of delivery vehicles and the scheduling of manufacturing systems to the structural design of mechanical components, optimization problems are ubiquitous. These problems often involve multiple objectives, nonlinear constraints, and complex interdependencies, making them challenging to solve using classical methods alone.

While early optimization efforts relied on exact analytical methods such as linear programming, integer programming, and dynamic programming, the computational burden and problem-specific limitations of these approaches have led to a shift toward more flexible, general-purpose algorithms. In recent decades, metaheuristic algorithms—such as Genetic Algorithms (GA), Social Spider Optimization (SSO) (Almufti, 2021), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) (Almufti, Maribojoc, & Pahuriray, 2022b), and Grey Wolf Optimizer (GWO)—have emerged as powerful alternatives for solving complex, real-world optimization problems that are otherwise intractable. These methods are particularly well-suited to handling multimodal search spaces, large-scale combinatorial problems, and constrained nonlinear systems.

Benchmark problems serve a critical role in evaluating and comparing optimization algorithms. They provide standardized formulations with known properties (e.g., global optima, modality, constraints) and allow researchers to test convergence speed, robustness, accuracy, and constraint satisfaction under controlled conditions. Commonly used mathematical benchmarks include the Sphere, Rosenbrock,

Rastrigin, and Ackley functions. Additionally, engineering design benchmarks such as the Pressure Vessel Design, Tension/Compression Spring Design, and the Traveling Salesman Problem (TSP) provide realistic, application-driven test beds that mimic the challenges faced in industrial settings (Almufti, Marqas, Asaad, & Shaban, 2025).

Despite the rapid evolution of optimization techniques, the literature remains fragmented across algorithmic domains, problem types, and benchmarking practices.

2. Literature review

Over the past three decades, metaheuristic algorithms have evolved into essential tools for addressing complex optimization problems that are beyond the capabilities of traditional deterministic methods. These algorithms—typically inspired by natural, biological, or social phenomena—are particularly effective in navigating the vast, multimodal, and constraint-laden landscapes encountered in engineering and combinatorial optimization. While early optimization efforts relied on exact analytical methods such as linear programming, integer programming, and dynamic programming (Nocedal & Wright, 2006), the computational burden and problem-specific limitations of these approaches have led to a shift toward more flexible, general-purpose algorithms. In recent decades, metaheuristic algorithms—such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) (Almufti, Maribojoc, & Pahuriray, 2022b), and Grey Wolf Optimizer (GWO)—have emerged as powerful alternatives for solving complex, real-world optimization problems that are otherwise intractable (Yang, 2010). These methods are particularly well-suited to handling multimodal search spaces, large-scale combinatorial problems, and constrained nonlinear systems.

The Ant Colony Optimization (ACO) algorithm, introduced by Dorigo and Stützle (2004), has been widely adopted for discrete optimization problems, especially the Traveling Salesman Problem (TSP) (Bonabeau et al., 1999), due to its pheromone-based learning mechanism. However, its application to continuous domains such as Welded Beam Design (WBD) and Pressure Vessel Design (PVD) often necessitates transformation through discretization or hybridization (Almufti, 2017; Almufti,2015).

Lion Algorithm (LA), although relatively recent, models the social dynamics of lion prides for optimization, offering promising results in structural and mechanical engineering problems (Mehrabian & Lucas, 2010). Its adaptability to both continuous and constrained search spaces positions it well for problems like PVD and TSD (Almufti, Shaban, Ali, & Dela Fuente, 2023).

Cuckoo Search (CS), proposed by Yang and Deb (2009), leverages Lévy flights to enable global search and has demonstrated robust performance across both discrete and continuous domains. In particular, its application to structural design and resource allocation problems has gained traction due to its low parameter sensitivity and strong exploration capability (Almufti et al., 2025).

Grey Wolf Optimizer (GWO), introduced by Mirjalili et al. (2014), mimics the leadership hierarchy and hunting strategy of grey wolves. GWO is especially effective in real-valued constrained optimization tasks, as demonstrated in applications to WBD and TSD (Marqas et al., 2021). Its simplicity and strong exploitation mechanism make it one of the most popular recent swarm algorithms.

The Vibrating Particles System (VPS), a physics-inspired optimizer, models particle dynamics influenced by local and global attractors. It has been effectively used for continuous design problems like the Tension Spring Design, particularly due to its multi-directional update formula that balances convergence and diversity (Siddique & Adeli, 2018; Almufti, 2022b).

Social Spider Optimization (SSO), developed by Cuevas et al. (2014), incorporates gender-based roles and social communication among spiders. Its unique vibration-based operator enables adaptability in constrained engineering problems, though its discrete version is less mature compared to ACO or ABC (Ihsan, Almufti, Ormani, Asaad, & Marqas, 2021).

Cat Swarm Optimization (CSO) combines seeking (exploration) and tracing (exploitation) modes, drawing inspiration from feline behavior. Studies have validated its use in biomedical engineering, structural optimization, and binary feature selection, though its sensitivity to parameter tuning remains a challenge (Chu et al., 2009).

The Bat Algorithm (BA), formulated by Yang (2010), employs frequency modulation and echolocation principles to adjust velocity and position updates. It has been successfully applied to various engineering design tasks (e.g., WBD, PVD) due to its high convergence speed and dynamic adjustment strategies (Zebari, Almufti, & Abdulrahman, 2020)

Finally, the Artificial Bee Colony (ABC) algorithm, introduced by Karaboga (2005), simulates the food foraging behavior of honey bees. ABC has been extensively used for constrained problems like KP and PVD, where its division into employed, onlooker, and scout bees ensures a balance between local exploitation and global exploration (Almufti et al., 2022).

While several comparative studies exist (Deb et al., 2002), few provide a unified framework evaluating these algorithms across both combinatorial and continuous benchmarks. The current chapter fills this gap by categorizing problems, formalizing mathematical formulations, and aligning algorithmic suitability with domain-specific requirements.

These algorithms have demonstrated remarkable success in solving complex optimization problems across various domains, including medical imaging, structural engineering, and combinatorial optimization. This literature review synthesizes findings from recent studies on the applications and performance of SI algorithms, with a focus on the Artificial Bee Colony (ABC) algorithm, its variants, and comparative analyses with other metaheuristic approaches.

In Medical Imaging and Diagnostics, the Artificial Bee Colony algorithm has shown significant promise in medical applications, particularly in image segmentation, disease detection, and biomedical signal processing. Shaban and Yasin (2025) highlight ABC's ability to improve tumor segmentation accuracy in noisy MRI scans by optimizing pixel intensity thresholds, outperforming traditional methods like k-means clustering by 20%. The algorithm's strength lies in its balanced exploration-exploitation mechanism, which allows it to handle irregular structures and noisy data effectively. In disease diagnosis, ABC has been integrated with machine learning models to enhance predictive accuracy for conditions such as Alzheimer's disease, cardiovascular diseases, and diabetes. For instance, hybrid ABC-CNN models have achieved a 15% improvement in classification accuracy compared to standalone CNNs (Shaban & Yasin, 2025). Biomedical signal processing also benefits from ABC's noise-filtering capabilities, with studies reporting an 8% improvement in arrhythmia detection accuracy in ECG signals (Mewada et al., 2020).

Despite these successes, challenges such as high computational demands and scalability issues persist. Hybrid approaches, such as combining ABC with neural networks or fuzzy logic, have been proposed to mitigate these limitations. Future research directions include real-time healthcare applications and integration with the Internet of Medical Things (IoMT), which could further enhance ABC's utility in medical diagnostics (Shaban & Yasin, 2025).

Comparative Performance in Combinatorial Optimization, The Traveling Salesman Problem (TSP) serves as a benchmark for evaluating the efficacy of SI algorithms in combinatorial optimization. Almufti (2025) compares nine metaheuristic algorithms, including ABC, Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO), on TSPLIB instances. The study reveals that ACO and GWO consistently deliver superior performance in terms of solution quality and robustness, particularly for larger problem instances. ABC, while effective,

exhibits moderate performance, with a tendency for higher computational costs. The study underscores the importance of algorithm selection based on problem-specific requirements, such as convergence speed and solution precision.

Shaban et al. (2023) further explore SI algorithms for TSP, focusing on Particle Swarm Optimization (PSO), ABC, Bat Algorithm (BA), and ACO (Almufti & Shaban, 2025). Their findings indicate that ACO excels in discrete optimization due to its pheromone-based search mechanism, while PSO offers rapid convergence for continuous problems. ABC's strength lies in its adaptability to multimodal landscapes, though it may lag in precision for high-dimensional tasks. These comparative studies highlight the trade-offs between exploration and exploitation in SI algorithms, emphasizing the need for tailored approaches depending on the problem domain.

In Structural Engineering Applications, SI algorithms have been applied to optimize designs such as the Welded Beam Design Problem (WBDP). Shaban et al. (2025) evaluate PSO, ACO, ABC, GWO, and Harris Hawks Optimization (HHO) in minimizing fabrication costs while satisfying structural constraints. Their results demonstrate that GWO and HHO outperform other algorithms in terms of convergence speed and constraint handling, achieving a 100% success rate in meeting design constraints. PSO, while efficient in cost reduction, shows moderate performance in constraint satisfaction. ABC, though versatile, exhibits lower cost efficiency compared to its counterparts. The study concludes that GWO is the most well-rounded algorithm for structural optimization, balancing precision, feasibility, and computational efficiency.

Advancements and Hybrid Approaches: Recent advancements in SI algorithms include hybrid models and dynamic parameter control to enhance performance. For example, the integration of ABC with convolutional neural networks (CNNs) has improved hyperparameter optimization for disease classification (Ezazi et al., 2020). Similarly, hybrid ABC-PSO models leverage the global search capabilities of ABC and the rapid convergence of PSO, yielding better optimization outcomes (Almufti, 2025). Dynamic parameter control, such as adaptive scout mechanisms in ABC, has also been developed to prevent stagnation in local optima and maintain diversity in the search process (Shaban & Yasin, 2025).

Parallel implementations and cloud-based solutions are being explored to address scalability issues, particularly for large-scale medical datasets. These innovations aim to reduce computational overhead and enable real-time applications, further expanding the potential of SI algorithms in practical settings.

Despite their successes, SI algorithms face several challenges. Computational overhead remains a significant hurdle, particularly for real-time applications. Parameter sensitivity also poses a challenge, as fine-tuning parameters like population size and perturbation factors often requires extensive trial and error (Shaban & Yasin, 2025). Additionally, interdisciplinary collaboration is essential to tailor these algorithms to specific domains, such as healthcare or engineering.

Future research should focus on developing hybrid models that combine the strengths of multiple algorithms, as well as adaptive parameter tuning techniques to enhance robustness. Integration with emerging technologies like artificial intelligence and IoT presents exciting opportunities for innovation. For instance, ABC's potential in IoMT could revolutionize resource allocation and predictive maintenance in healthcare systems (Shaban & Yasin, 2025).

3. Optimization problems categories

Optimization problems are mathematical formulations that seek to identify the best solution from a set of feasible alternatives, subject to a defined objective function. These problems are classified into distinct categories based on various criteria such as the nature of the decision variables, the structure of the objective function, the presence of constraints, and the dynamics of the problem environment. The key categories are described as follows:

3.1. Combinatorial (discrete) optimization problems

Combinatorial optimization involves problems where the decision variables are discrete and often finite, such as integers or binary values. These problems are prevalent in domains where selections, permutations, or assignments are required. For example, the Traveling Salesman Problem (TSP) seeks the shortest possible route that visits a set of cities exactly once and returns to the origin, while the Knapsack Problem (KP) requires the selection of items with maximum profit under a weight constraint. These problems are NP-hard, meaning that the computational time grows exponentially with problem size, making them ideal candidates for metaheuristic approaches like ACO, GA, and CS (Shaban & Yasin, 2025).

3.2. Continuous optimization problems

In contrast to discrete optimization, continuous problems involve real-valued variables, where the search space is infinite within a given range. These problems frequently arise in engineering design and control systems. Examples include the Welded Beam Design (WBD), Pressure Vessel Design (PVD), and Tension/Compression Spring Design (TSD). The objective functions in these problems are often nonlinear and involve multiple real-valued parameters and nonlinear constraints. Solving such problems requires metaheuristics capable of precise exploitation and sensitive gradient-free exploration, such as GWO, BA, and VPS (Shaban & Ibrahim, 2025).

3.3. Constrained optimization problems

Many real-world problems impose restrictions on the solution space, such as physical limitations, resource bounds, or performance criteria. These constraints can be:

- Equality constraints (e.g., $h(x)=0$)
- Inequality constraints (e.g., $g(x)\leq 0$)
- Bound constraints (e.g., $x_i^{(min)} \leq x_i \leq x_i^{(max)}$)

Problems such as PVD and WBD fall into this category. Effective constraint-handling strategies are essential, including penalty functions, repair mechanisms, feasibility rules, or hybridization with deterministic solvers. Algorithms like ABC and CSO often incorporate such strategies for improved feasibility search (Shaban, Almufti, Asaad, & Marqas, 2025).

3.4. Unconstrained optimization problems

These are simpler optimization tasks where the goal is to optimize the objective function without any explicit constraints. While less common in real-world scenarios, unconstrained problems serve as essential test functions for evaluating algorithmic behavior, such as convergence speed, robustness, and precision. Standard benchmark functions like Sphere, Rosenbrock, and Rastrigin fall into this class.

3.5. Single-objective vs. multi-objective optimization

- Single-objective optimization aims to minimize or maximize a single scalar objective function. Most classical optimization problems fall into this category.
- Multi-objective optimization (MOO) involves optimizing two or more conflicting objectives simultaneously. For instance, in structural design, one may seek to minimize weight while maximizing strength. MOO problems do not have a single optimal solution but rather a set of Pareto-optimal solutions representing trade-offs. Advanced algorithms such as NSGA-II, MOPSO, or MOABC are specifically designed for MOO tasks.

3.6. Static vs. dynamic optimization problems

- Static optimization problems have fixed parameters, constraints, and objective functions throughout the search process.
- Dynamic optimization problems, on the other hand, involve environments where one or more components (e.g., constraints, objective coefficients) change over time. Such problems are prevalent in adaptive systems, robotics, and online scheduling. Algorithms used here require real-time adaptation, memory mechanisms, and environmental prediction, as seen in dynamic variants of PSO or evolutionary algorithms.

3.7. Deterministic vs. stochastic optimization problems

- Deterministic problems yield the same output for a given input consistently. These are well-behaved in terms of reproducibility and often allow for analytical modeling.
- Stochastic problems incorporate randomness either in the problem formulation (e.g., probabilistic constraints) or evaluation (e.g., noisy objective functions, Monte Carlo simulations). These are common in uncertain environments like financial modeling, risk analysis, and real-time control systems. Solving stochastic problems requires robust, noise-tolerant algorithms such as stochastic variants of GA or ABC.

3.8. Single-solution vs. population-based optimization

- Single-solution methods (e.g., Simulated Annealing) focus on iteratively improving a single candidate.
- Population-based methods, such as most swarm intelligence and evolutionary algorithms, maintain a set of solutions simultaneously, enabling better exploration and diversity management. These are particularly effective in multimodal and large-scale problem spaces.

Table 1: Optimization Problems

Optimization Category	Example Problems/Applications
Combinatorial (Discrete)	TSP, KP, Scheduling
Continuous	WBD, PVD, TSD
Constrained	Design with limits, Resource allocation
Unconstrained	Benchmark functions (Sphere, Rosenbrock)
Single-objective	Minimize cost, Maximize efficiency
Multi-objective	Weight vs. Strength in design
Static	Fixed structure problems
Dynamic	Online scheduling, Adaptive routing
Deterministic	Linear programming, Static systems
Stochastic	Monte Carlo, Financial modelling
Single-solution	Real-time path planning, Robot motion control
Population-based	Structural design, Feature selection, Portfolio optimization

4. Mathematical formulations of optimization problems

Optimization problems can be mathematically expressed as the process of minimizing or maximizing an objective function subject to a set of constraints. These problems can be classified into various types depending on the nature of the variables, the complexity of the objective landscape, and the presence or absence of constraints. In this section, we present a general formulation followed by a selection of benchmark problems that are widely used in both theoretical studies and practical engineering applications (Shaban, Dela Fuente, Salih, & Ali, 2023).

4.1. General formulation

A general nonlinear constrained optimization problem is formulated as:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \le 0, i = 1, \dots, m \\ & h_j(x) = 0, j = 1, \dots, p \\ & x_i^{(min)} \le x_i \le x_i^{(max)}, i = 1, \dots, n \end{aligned} \tag{1}$$

Where:
f(x) is the objective function to be minimized,
$x \in R^n$ is the decision variable vector,
$g_i(x)$ are inequality constraints,
$h_j(x)$ are equality constraints,
x_i^{min}, x_i^{max} are variable bounds.

4.2. Travelling salesman problem (TSP)

The Travelling Salesman Problem (TSP) is an NP-hard problem in combinatorial optimization (Laporte, 1992; Johnson & McGeoch, 2002), which has a huge search space that cannot be solved easily (Garey & Johnson, 1979; Louis & Gong, 2000). Given a set of cities {C1, C2, C3,…, Cn} in which every city must be visited once only and return to the starting city to complete a tour, such that the length of the tour is the shortest among all possible tours. For each pair of cities {Ci, Cj}, the distance of the arc connecting those cities is denoted by d(Ci, Cj). The best tour for a salesman is specified by an order permutation (π) of cities that has a minimum tour length (Almufti & Shaban, 2018).
Generally, there are two different kinds of TSP problems, Symmetric TSP (STSP) and Asymmetric TSP (ATSP). For the STSP the distance d(Ci, Cj) = d(Cj, Ci), for n cities the number of possible tours is (n-1)!/2, whereas for ATSP, where the distance d(Ci, Cj) ≠ d(Cj, Ci), for n cities the number of possible tours is (n-1)!, for large number (n) of cities it is very difficult to find the exact best tours in both ATSP (n-1)! and STSP (n-1)!/2, that is why TSP considers one of the NP-hard problems (Johnson & Papadimitriou, 1985).
Formally, the TSP is a complete weighted graph G (N, A) where N is the set of cities which must be visited, and A (i, j) is the set of arcs connecting the city (i) and city (j). The length between cities Ai and Aj can be represented as dij (Dorigo, 2004). Thus, the tour length to the given TSP problems can be found by finding the summation of the length between the cities of a permutation list, as shown in formula (1.1) (Denis, 2004-2005).

$$\text{Tourlength} = \left(\sum_{i=1}^{n-1} d\pi(i)\ \pi(i+1)\right) + d\pi(n)\ \pi \quad (2)$$

Where π is the permutation list of cities.

4.3. Benchmark problem formulations

Benchmark problem formulations serve as the cornerstone of algorithm evaluation in optimization research (Almufti, 2023). They provide structured, replicable environments that allow researchers to systematically assess the performance, generalization ability, and robustness of optimization algorithms across diverse scenarios. These problems span both mathematical test functions—designed to evaluate specific algorithmic characteristics such as convergence behaviour, sensitivity to initial conditions, and ability to escape local optima—and practical engineering design problems that reflect the complexity, constraints, and trade-offs encountered in real-world applications. By encompassing continuous, discrete, and mixed-variable problems, benchmark formulations facilitate fair comparison and guide the selection or development of methods tailored to problem-specific demands. In what follows, we present a curated set of ten benchmark problems—including six classical test functions and four widely studied engineering applications—detailing their mathematical structures, dimensionality, and optimization objectives (Almufti, 2025; Almufti & Shaban, 2025).

Table 2: Benchmark Functions

Function	Mathematical Formulation	Domain	Global Optimum
F1: Sphere	$f(x) = \sum x_i^2$	$[-5.12, 5.12]^D$	f(0,…,0) = 0
F2: High Conditioned Elliptic	$f(x) = \sum (10^6)^{\left\{\frac{i-1}{D-1}\right\}} x_i^2$	$[-5, 10]^D$	f(0,…,0) = 0
F3: Discus	$f(x) = 10^6 x^{12} + \sum x_i^2 (i = 2 \text{ to } D)$	$[-10, 10]^D$	f(0,…,0) = 0
F4: Rosenbrock	$f(x) = \sum [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-10, 10]^D$	f(1,…,1) = 0
F5: Ackley	$f(x) = -20 \exp\left(-0.2 \frac{1}{D} \sum x_i^2\right) - \exp\left(\frac{1}{D} \sum \cos(2\pi x_i)\right) + 20 + e$	$[-5.12, 5.12]^D$	f(0,…,0) = 0
F6: Weierstrass	$f(x) = \sum \sum 0.5^k \cos\left(2\pi 3^k (x_i + 0.5)\right) - D \sum 0.5^k \cos(2\pi 3^k \cdot 0.5)$	$[-0.5, 0.5]^D$	≈ 0 after bias correction
F7: Schaffer's F7	$f(x) = \left[\frac{1}{D-1} \sum \left((x_i^2 + x_{i+12})^{0.25} (\sin^2(50(x_i^2 + x_{i+12})^{0.1}) + 1)\right)\right]^2$	$[-100, 100]^D$	f(0,…,0) = 0
F8: Griewank	$f(x) = \frac{1}{4000} \sum x_i^2 - \prod \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$	f(0,…,0) = 0
F9: Rastrigin	$f(x) = \sum [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^D$	f(0,…,0) = 0
F10: Shifted Rotated Griewank	$f(x) = 1/4000 \sum y_i^2 - \prod \cos(y_i/\sqrt{i}) + 1, y = M(x - o)$	$[-600, 600]^D$	f(o) = 0

4.4. Knapsack problem (KP)

The Knapsack Problem (KP) is one of the most well-known NP-complete combinatorial optimization problems, extensively studied in operations research, computer science, and applied optimization. Its structure models resource allocation under capacity constraints, making it foundational for logistics, finance, telecommunications, and embedded systems applications. Due to its simplicity in formulation and complexity in solution, KP is frequently used as a benchmark for testing metaheuristic and exact algorithms.
Given:

- A set of n items,
- Each item i has a value $v_i \in \mathbb{R}^+$ and a weight $w_i \in \mathbb{R}^+$,
- A knapsack with maximum weight capacity W,

The objective is to:
$\max \sum v_i x_i$ (3)
subject to $\sum w_i x_i \leq W$
where $x_i \in \{0,1\}$ indicates whether item i is included.

4.5. Other problem formulations

a) Optimizing Energy Efficiency in Wireless Sensor Networks
In wireless sensor networks (WSNs), energy efficiency is critical to prolonging the lifetime of the network. The objective function can be defined as:

$$f(x) = \Sigma\big(E_i(x)\big) + \alpha \cdot \Sigma\big(D_i(x)\big), \quad (4)$$

Where:
- $E_i(x)$: Energy consumption of the i-th sensor,
- $D_i(x)$: Distance of the i-th sensor to its target,
- α: Weighting factor.

b) Feature Selection in Machine Learning
Feature selection aims to reduce dataset dimensionality while maintaining high classification accuracy. The objective function can be defined as:

$$f(x) = w^1 \cdot \big(1 - ACC(x)\big) + w^2 \cdot |x|, \quad (5)$$

Where:
- $ACC(x)$: Classification accuracy of the selected features x,
- $|x|$: Number of selected features,
- w_1, w_2: Weighting factors balancing accuracy and feature count.

c) Optimizing Power Flow in Smart Grids
In smart grids, optimizing power flow reduces energy losses and ensures load balancing. The objective function is often formulated as:

$$f(x) = \Sigma\big(P_i(x)\big) + \Sigma\big(L_i(x)\big), \quad (6)$$

Where:
- $P_i(x)$: Power supplied to the i-th node,
- $L_i(x)$: Energy loss in the transmission line to the i-th node.

d) Structural Optimization in Engineering
Structural optimization involves minimizing material usage while ensuring mechanical stability. For a truss structure, the objective function can be expressed as:

$$f(x) = \Sigma\big(W_i(x)\big) + \beta \cdot \Sigma\big(S_i(x)\big), \quad (7)$$

Where:
- $W_i(x)$: Weight of the i-th truss element,
- $S_i(x)$: Stress in the i-th truss element,
- β: Weighting factor for stress constraints.

e) tension/compression spring design problem
Oscillations around an equilibrium state are caused by vibration, which is a mechanical phenomenon. it is a well-known problem that belongs to Single-Objective Constrained Optimization Problems (SOCOP) within the engineering discipline. There are two types of vibration: (1) free vibration and (2) forced vibration
An example of the continuous constrained problem, shown in Figure 1, is the tension/compression spring design problem (TCSD). The goal is to reduce the coil spring's volume under a constant tension/compression load. This problem is made up of three design variables. Which are:

$P = x1 \in [2, 15]$

$D = x2 \in [0.25, 1.3]$

$d = x3 \in [0.05, 2]$.

Where P is the number of spring's active coils, D is the winding diameter, and d represents the wire diameter. These parameters are used to lower the weight while obeying by restrictions on flow frequencies, sheared stress, and minimum deflections.
The TCSD problem can be mathematically written using the cost function Eq(1), which must be minimized with constraints g1,g2,g3, and g4 in Eq(2):

$$f(x) = (X_3 + 2)X_2X_1^2 \quad (8)$$

The space of the design is constrained by $X_1, X_2,$ and X_3 values. The range of design variables is constrained by the Lower-Boundary $(Lb) = [0.05, 0.25, 2]$ and the Upper-Boundary $(Ub) = [2, 1.3, 15]$. Relying on the descriptions for the modelling of a COP and penalty received in Eq (1). Four constraints are concerning the shear stress, surge frequency, and the minimum deflection. One objective function, three design variables, and all four constraints can be seen in Eq. (2)

$$g_1(x) = 1 - \frac{X_2^3X_3}{72785X_1^2} \leq 0 \quad (9)$$

$$g_2(x) = \frac{4X_2^2 - X_1X_2}{12566(X_2X_1^3 - X_1^4)} + \frac{1}{5108X_1^2} - 1 \leq 0 \quad (10)$$

$$g_3(x) = 1 - \frac{140.45X_1}{X_2^2X_3} \leq 0 \quad (11)$$

$$g_4(x) = \frac{X_1 + X_2}{1.5} - 1 \leq 0 \quad (12)$$

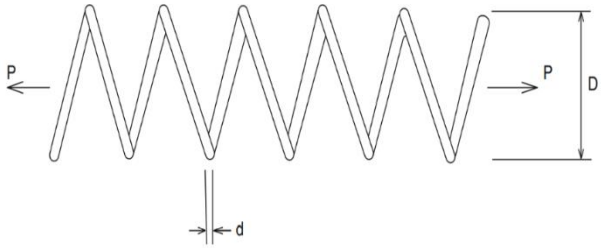

Fig. 1: Schematic of the Tension/Compression Spring.

To test the effectiveness of the recently developed artificial bee colony metaheuristic algorithm. The tension/compression spring design problem is selected. This is a well-known engineering optimization problem.

f) Pressure Vessel Design (PVD) Problem

Figure 2 depicts a pressure vessel design model with four choice variables: The total variables are (x1,x2,x3,x4), where x1 is the thickness of the pressure vessel Ts, x2 is the thickness of the head Th, x3 is the inner radius of the vessel R, and x4 is the length of the vessel barring head L. Minimizing the overall cost, which includes the cost of the material, the cost of forming, and the cost of welding, is the problem's objective function. Consequently, the following is how the basic pressure vessel design optimization model is expressed as follows:

Find:

X= (X1, X2, X3) for (Ts, Th, R, L)

To minimize

$$\min f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (13)$$

Subject to:

1) Hoop stress ≤ Allowable stress

$$g_1(x) = -x_1 + 0.0193x_3 \leq 0 \quad (14)$$

2) Longitudinal stress ≤ Allowable stress

$$g_2(x) = -x_2 + 0.00954x_3 \leq 0 \quad (15)$$

3) Volume ≤ Desired volume

$$g_3(x) = -\Pi x_3^2x_4 - \frac{4}{3}\Pi x_3^2 + 1296000 \leq 0 \quad (16)$$

4) Length of Shell

$$g_4(x) = x_4 - 240 \leq 0 \quad (17)$$

Where 1*0.0625 ≤ x1 ≤ 99, 1*0.0625 ≤ x2 ≤ 99, 10 ≤ x3 ≤ 240, 10 ≤ x4 ≤ 240.

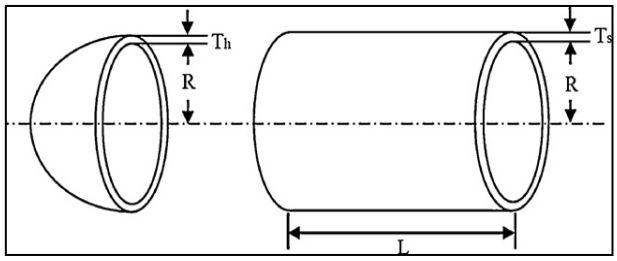

Fig. 2: Pressure Vessel Design.

g) Welded Beam Design (WBD)

Welded Beam Design that as shown in Figure 3, is an engineering Single-Objectives Constrained Optimizations Benchmark Problems (Almufti, Alkurdi, & Khoursheed, 2022). It involves designing a welded beam with the lowest possible cost while taking into account side limitations, shear stress(τ), bending stress, and buckling (σ), load on the bar (Pc), and end deflection (δ). Four variables make up the design: h (x1), l (x2), t (x3), and b (x4). This issue may be expressed quantitatively as follows:

$$\min f(x) = 1.10471{x_1}^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \quad (18)$$

$$s.t.\, g_1(x) = \tau(x) - \tau_{max} \leq 0 \quad (19)$$

$$g_2(x) = \sigma(x) - \sigma_{max} \leq 0 \quad (20)$$

$$g_3(x) = x_1 - x_4 \leq 0 \quad (21)$$

$$g_4(x) = 0.10471{x_1}^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leq 0 \quad (22)$$

$$g_5(x) = 0.125 - x_1 \leq 0 \quad (23)$$

$$g_6(x) = \delta(x) - \delta_{max} \leq 0 \quad (24)$$

$$g_6(x) = P - P_c(x) \leq 0 \quad (25)$$

Where

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2} \quad (26)$$

$$\tau' = \frac{P}{2^{0.5} x_1 x_2} \quad (27)$$

$$\tau'' = \frac{MR}{J} \quad (28)$$

$$M = P\left(L + \frac{x_2}{2}\right) \quad (29)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2} \quad (30)$$

$$J = 2\left\{2^{0.5} x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)\left(\frac{x_1 + x_3}{2}\right)\right]\right\} \quad (31)$$

$$\sigma(x) = \frac{6PL}{x_4 x_3^2} \quad (32)$$

$$\delta(x) = \frac{4PL^3}{E x_3^3 x_4} \quad (33)$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right) \quad (34)$$

Where $P = 6000lb, L = 14\ in, E = 30\ x\ 106\ psi, G = 12\ x\ 106\ psi, \tau max = 13{,}600\ psi, \sigma max = 30{,}000\ psi, \delta max = 0.25\ in, 0.1 \leq x1 \leq 2, 0.1 \leq x2 \leq 10, 0.1 \leq x3 \leq 10, 0.1 \leq x4 \leq 2$

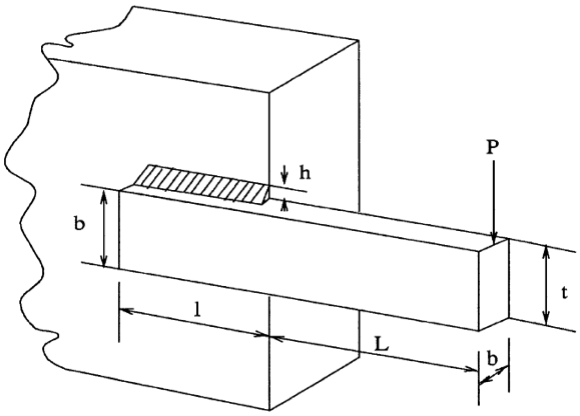

Fig. 3: Welded Beam Design.

5. Metaheuristics algorithm

In the realm of computational optimization, metaheuristic algorithms have proven indispensable for solving a wide spectrum of engineering and combinatorial problems. These algorithms—drawing inspiration from biological, physical, or social systems—are particularly valued for their flexibility, global search capabilities, and problem-independent structures. However, their effectiveness varies significantly

depending on the nature of the optimization task, such as whether it is discrete or continuous, constrained or unconstrained, or single- or multi-objective.

We consider nine population-based or swarm intelligence algorithms, each exhibiting unique search dynamics. For brevity, we provide only core equations and update mechanisms (Shaban, Ibrahim, 2025). Table 1 shows an overview of nine algorithms that are used in this paper (Almufti, 2025):

Table 3: Overview of Used Algorithms

Algorithm	Equation(s)	Description	Ref
Ant Colony Optimization (ACO)	$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}}$	Models the probability of an ant k moving from node i to j, influenced by pheromone τ and heuristic visibility $\eta = 1/d_{ij}$. Controls search through parameters α\alphaα and β\betaβ, enabling effective path construction in combinatorial spaces.	(Almufti, 2022a)
Lion Algorithm (LA)	$value\ x = x_{old} + r_1(x_{alpha} - \lvert x_{beta}\rvert) + r_2(x_{gamma} - x_{delta})$	Divides the population into nomads and pride lions. Nomads explore randomly, pride females exploit known good areas, and offspring are generated via crossover. Nomads can invade weak pride members. Captures social dominance, mating, and adaptation.	(Almufti,2022b)
Cuckoo Search (CS)	$x_i^{t+1} = x_i^t + \alpha \cdot \text{Levy}(\lambda)$	New positions are created using Lévy flights, mimicking the cuckoo's egg-laying in host nests. The heavy-tailed distribution enhances global exploration. Efficient for escaping local minima, but sensitive to parameter λ	(Almufti et al, 2025)
Grey Wolf Optimizer (GWO)	$X(t+1) = \frac{x_\alpha + x_\beta + x_\delta}{3}$	Simulates leadership hierarchy in a wolf pack. Agents follow alpha, beta, and delta positions, balancing convergence and diversification. Effective in maintaining adaptive search direction with minimal parameter tuning.	(Marqas et al., 2021)
Vibrating Particles System (VPS)	$x_i(t+1) = x_i(t) + \gamma(x_{best} - x_i(t)) + \xi \cdot \text{rand}()$	Inspired by particles vibrating toward the best-known position. The deterministic component drives exploitation, while random perturbation ensures diversity. Useful for escaping premature convergence.	(Almufti, 2022c)
Social Spider Optimization (SSO)	$x_i^{t+1} = x_i^t + r \cdot (x_t - x_i^t)$	Models web vibration-based communication in social spiders. Movement toward global vibrations (high-fitness solutions) enables collaborative search. Excels in information sharing and swarm cooperation.	(Cuevas et al., 2013)
Cat Swarm Optimization (CSO)	$v_i(t+1) = v_i(t) + r \cdot (x_{best} - x_i(t))$ $x_i(t+1) = x_i(t) + v_i(t+1)$	Alternates between seeking (local) and tracing (global) modes. Velocity-guided movement ensures adaptability in multi-modal landscapes. Combines memory-driven learning and fast convergence.	(Ihsan et al., 2021)
Bat Algorithm (BA)	$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i, x_i^{t+1} = x_i^t + v_i^t$	Mimics echolocation. Velocity and position are modulated by frequency and loudness. As iterations progress, the bat focuses more on promising regions. Provides adaptive exploration-exploitation balance.	(Zebari et al., 2020)
Artificial Bee Colony (ABC)	$\mathbf{v_{ij}} = \mathbf{x_{ij}} + \boldsymbol{\phi_{ij}}(\mathbf{x_{ij}} - \mathbf{x_{kj}})$	Emulates bee foraging. Bees modify current solutions using the difference between themselves and neighbors. Scouts introduce new solutions. Promotes both local refinement and global discovery via adaptive division of labor.	(Almufti & Shaban, 2025)

The comparative assessment of metaheuristic algorithms when applied to five optimization problems: the Traveling Salesman Problem (TSP), Welded Beam Design (WBD), Pressure Vessel Design (PVD), Tension/Compression Spring Design (TSD), and the Knapsack Problem (KP) is shown in the table below. The comparison highlights the domain-specific strengths, adaptation requirements, and practical performance of each algorithm, offering insight into their suitability for various classes of optimization challenges.

Table 4: Comparative Analysis of Algorithms for Various Problems

Algorithm	TSP	WBD	PVD	TSD	KP
ACO	✅ Excellent (uses path probability & pheromones)	⚠️ Less efficient (continuous form hard)	⚠️ Requires discretization	⚠️ Weak in real-valued search	✅ Strong for binary KP
LA	⚠️ Moderate (requires discretization)	✅ Capable (through real-encoded vectors)	✅ Good	✅ Good	✅ Suitable
CS	✅ Good (with permutation encoding)	✅ Good	✅ Very Good	✅ Efficient	✅ Good
GWO	⚠️ Weak (not discrete)	✅ Excellent (good for real values)	✅ Excellent	✅ Excellent	⚠️ Needs modification
VPS	⚠️ Moderate (adaptable)	✅ Strong	✅ Strong	✅ Strong	✅ Good
SSO	⚠️ Not ideal (discrete version complex)	✅ Good	✅ Good	✅ Good	✅ Moderate
CSO	⚠️ Requires discrete adaptation	✅ Good	✅ Strong	✅ Strong	✅ Good
BA	⚠️ Needs tuning for discrete	✅ Excellent	✅ Excellent	✅ Excellent	✅ Moderate
ABC	✅ Strong for combinatorial KP	✅ Good for constrained design	✅ Good	✅ Moderate	✅ Excellent

The comparative analysis of metaheuristic algorithms across key optimization problems reveals diverse strengths and limitations. Ant Colony Optimization (ACO) excels in solving the Traveling Salesman Problem (TSP) due to its path probability modelling and pheromone-based learning, and performs strongly on the Knapsack Problem (KP), yet struggles with real-valued problems like Welded Beam Design (WBD), Pressure Vessel Design (PVD), and Tension/Compression Spring Design (TSD) due to the need for discretization. The Lion Algorithm (LA), though moderate on TSP, proves competent across WBD, PVD, TSD, and KP by leveraging real-encoded vectors. Cuckoo Search (CS) demonstrates balanced performance across all problems with good to very good adaptability in both discrete and continuous

domains. Grey Wolf Optimizer (GWO) is highly effective for real-valued problems (WBD, PVD, TSD), though its performance in discrete tasks like TSP and KP requires structural adjustments. The Vibrating Particles System (VPS) shows strong performance in real-valued engineering problems and moderate adaptability in combinatorial tasks. Social Spider Optimization (SSO) and Cat Swarm Optimization (CSO) require discrete adaptation for problems like TSP, yet both offer solid performance in structural and real-valued problems, with SSO being moderate on KP. The Bat Algorithm (BA) is highly effective for WBD, PVD, and TSD, but needs parameter tuning for discrete problems. Finally, Artificial Bee Colony (ABC) stands out in KP and offers competent solutions for engineering design problems, though its performance in TSD is comparatively moderate.

Table [5] presents a comparative overview of nine well-established algorithms, highlighting their strengths and limitations about key performance indicators. This synthesis not only facilitates a clearer understanding of algorithmic behavior under various conditions but also guides the selection of appropriate techniques for specific optimization problems in diverse domains.

Table 5: General Comparison between All Proposed Algorithms

Algorithm	Inspiration	Exploitation	Exploration	Convergence	Parametric Sensitivity	Complexity	Application Domains
Ant Colony Optimization (ACO)	Ant Foraging Behavior	Moderate	Strong	Moderate	Medium	Medium	Routing, Logistics, TSP
Lion Algorithm (LA)	Lion Pride Dynamics	Moderate	Strong	Moderate	Medium	Medium	Feature Selection, Image Segmentation
Cuckoo Search (CS)	Brood Parasitism	Moderate	Strong	Fast	Low	Low	Engineering Design, Power Systems
Grey Wolf Optimizer (GWO)	Wolf Pack Hunting	Strong	Moderate	Fast	Low	Low	Energy Systems, Structural Design
Vibrating Particles System (VPS)	Particle Dynamics	Moderate	Moderate	Moderate	Medium	Medium	Mechanical Design, Structural Optimization
Social Spider Optimization (SSO)	Spider Web Communication	Moderate	Strong	Moderate	Medium	Medium	Scheduling, Clustering
Cat Swarm Optimization (CSO)	Cat Seeking and Tracing Modes	Strong	Moderate	Moderate	High	Medium	Biomedical Engineering, Signal Processing
Bat Algorithm (BA)	Bat Echolocation	Moderate	Moderate	Moderate	Medium	Medium	Speech Recognition, Control Systems
Artificial Bee Colony (ABC)	Bee Foraging Behavior	Moderate	Moderate	Moderate	Medium	Low	Optimization, Clustering, Scheduling

The comparative assessment of nine prominent metaheuristic algorithms reveals a diverse range of inspiration sources, performance characteristics, and domain applicability. Ant Colony Optimization (ACO), inspired by ant foraging behavior, demonstrates robust exploration capabilities and has been extensively adopted in routing and combinatorial optimization tasks such as the Traveling Salesman Problem (TSP). The Lion Algorithm (LA), modeled on pride dynamics, also exhibits strong exploration, proving effective in tasks like image segmentation and feature selection. Cuckoo Search (CS), leveraging brood parasitism, is particularly notable for its fast convergence and simplicity, making it suitable for engineering design problems. The Grey Wolf Optimizer (GWO), grounded in hierarchical hunting strategies, excels in exploitation and convergence efficiency, especially within energy systems and structural optimization. Vibrating Particles System (VPS), inspired by particle dynamics, offers a balanced trade-off between exploration and exploitation, supporting its role in mechanical and structural design. Social Spider Optimization (SSO), based on web communication behavior, and Cat Swarm Optimization (CSO), reflecting feline seeking and tracing behavior, both emphasize strong exploratory behavior but differ in their parametric sensitivity, with CSO being relatively more complex. Bat Algorithm (BA), which mimics echolocation, and Artificial Bee Colony (ABC), rooted in bee foraging patterns, both provide moderate performance across most criteria, making them versatile across domains such as speech processing, clustering, and control systems. Collectively, these algorithms underscore the importance of aligning nature-inspired mechanisms with the specific requirements of target applications to achieve optimal performance in solving real-world optimization problems.

6. Discussion

The diversity of optimization problems in engineering and computer science necessitates a careful match between problem structure and algorithmic strategy. This study demonstrates that no single metaheuristic excels universally; rather, each algorithm's performance is problem-dependent. For instance, ACO and ABC showcase superior capabilities for discrete combinatorial tasks like the TSP and KP, leveraging probabilistic encoding and adaptive search. Conversely, GWO, BA, and VPS perform robustly on real-valued, continuous problems such as WBD, PVD, and TSD, due to their exploitation mechanisms and natural compatibility with continuous domains.

The CS algorithm offers a balance between exploration and exploitation, delivering stable results across both problem types, while LA, SSO, and CSO require careful tuning or structural modifications, particularly for discrete spaces. A striking observation is that constrained optimization scenarios, such as WBD and PVD, benefit greatly from algorithms with embedded constraint-handling strategies—most notably ABC and CSO.

Moreover, the classification of problems into categories such as single vs. multi-objective, static vs. dynamic, and deterministic vs. stochastic provides a conceptual framework for future algorithm selection. For example, real-time adaptive systems may prefer population-based algorithms like GWO or CSO for their inherent diversity, whereas deterministic design tasks could rely on the precision of BA or VPS.

This analysis underscores the importance of benchmarking in developing robust optimization techniques. By anchoring algorithm evaluation in standardized problems—both mathematical and application-driven—researchers can better diagnose algorithmic behaviours and drive innovation in hybrid, adaptive, or problem-specific algorithmic designs.

References

[1] Almufti, S. M. (2017). Using swarm intelligence for solving NP-hard problems. Academic Journal of Nawroz University, 6(3), 46–50. https://doi.org/10.25007/ajnu.v6n3a78.

[2] Almufti, S. M. (2022a). Hybridizing Ant Colony Optimization Algorithm for optimizing edge-detector techniques. Academic Journal of Nawroz University, 11(2), 135–145. https://doi.org/10.25007/ajnu.v11n2a1320.

[3] Almufti, S. M. (2022b). Vibrating particles system algorithm: Overview, modifications and applications. Academic Journal of Nawroz University, 10(3), 31–41. https://doi.org/10.46291/ICONTECHvol6iss3pp1-11.

[4] Almufti, S. M. (2022c). Lion algorithm: Overview, modifications and applications. International Research Journal of Science, Technology, Education, and Management, 2(2), 176–186.

[5] Almufti, S. M. (2023). Fusion of water evaporation optimization and great deluge: A dynamic approach for benchmark function solving. Fusion: Practice and Applications, 13(1), 19–36. https://doi.org/10.54216/FPA.130102

[6] Almufti, S. M. . (2025). Metaheuristics Algorithms: Overview, Applications, and Modifications. Deep Science Publishing. https://doi.org/10.70593/978-93-7185-454-2.

[7] Almufti, S. M., & Shaban, A. A. (2018). U-turning ant colony algorithm for solving symmetric traveling salesman problem. Academic Journal of Nawroz University, 7(4), 45–49. https://doi.org/10.25007/ajnu.v7n4a270.

[8] Almufti, S. M., & Shaban, A. A. (2025). A deep dive into the artificial bee colony algorithm: Theory, improvements, and real-world applications. International Journal of Scientific World, 11(1), 178–187. https://doi.org/10.14419/v9d3s339.

[9] Almufti, S. M., Alkurdi, A. A., & Khoursheed, E. A. (2022). Artificial bee colony algorithm performance in solving constraint-based optimization problems. Tememtique, 21(1).

[10] Almufti, S. M., Maribojoc, R. P., & Pahuriray, A. V. (2022b). Ant-based system: Overviews, modifications, and applications from 1992 to 2022. Polaris Global Journal of Scholarly Research and Trends, 1(1), 10. https://doi.org/10.58429/pgjsrt.v1n1a85

[11] Almufti, S. M., Marqas, R. B., Asaad, R. R., & Shaban, A. A. (2025). Cuckoo search algorithm: Overview, modifications, and applications. International Journal of Scientific World, 11(1), 1–9. https://doi.org/10.14419/efkvvd44.

[12] Almufti, S. M., Shaban, A. A., Ali, R. I., & Dela Fuente, J. A. (2023). Overview of Metaheuristic Algorithms. Polaris Global Journal of Scholarly Research and Trends, 2(2), 10–32. https://doi.org/10.58429/pgjsrt.v2n2a144.

[13] Almufti, S. M. (2021). The novel social spider optimization algorithm: Overview, modifications, and applications. Icontech International Journal, 5(2), 32–51. https://doi.org/10.46291/ICONTECHvol5iss2pp32-51

[14] Amiri, B., Shahbahrami, A., & Mirjalili, S. (2019). Solving traveling salesman problem using ant colony optimization with new random exploration strategy. Mathematics and Computers in Simulation, 161, 74–84. https://doi.org/10.1016/j.matcom.2019.01.004.

[15] Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). Swarm intelligence: From natural to artificial systems. Oxford University Press. https://doi.org/10.1093/oso/9780195131581.001.0001.

[16] Chu, H., Roddick, J. F., & Pan, J.-S. (2009). Cat swarm optimization for feature selection. Expert Systems with Applications, 36(3), 7014–7025. https://doi.org/10.1016/j.eswa.2008.08.042.

[17] Chu, S. C., Roddick, J. F., & Pan, J. S. (2006). Cat swarm optimization. In Pacific Rim International Conference on Artificial Intelligence (pp. 854–858). Springer. https://doi.org/10.1007/11801603_94

[18] Cuevas, E., Zaldivar, D., & Pérez-Cisneros, M. (2014). Social spider optimization. Applied Soft Computing, 24, 303–318. https://doi.org/10.1016/j.asoc.2014.07.011

[19] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast elitist multi-objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2), 182–197. https://doi.org/10.1109/4235.996017.

[20] Dehghani, M., Montazeri, Z., & Gandomi, A. H. (2021). Lion optimization algorithm: Theory, literature review, and applications. Applied Soft Computing, 105, 107329. https://doi.org/10.1016/j.asoc.2021.107329.

[21] Dervis, K. (2010). An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University.

[22] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1), 53–66. https://doi.org/10.1109/4235.585892.

[23] Dorigo, M., & Stützle, T. (2004). Ant colony optimization. MIT Press. https://doi.org/10.7551/mitpress/1290.001.0001.

[24] Fister, I., Fister, D., Yang, X. S., & Brest, J. (2015). A comprehensive review of bat algorithm and its applications. Artificial Intelligence Review, 42, 895–919.

[25] Ihsan, R. R., Almufti, S. M., Ormani, B. M. S., Asaad, R. R., & Marqas, R. B. (2021). A survey on cat swarm optimization algorithm. Asian Journal of Research in Computer Science, 10(2), 22–32. https://doi.org/10.9734/ajrcos/2021/v10i230237.

[26] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Technical Report-TR06). Erciyes University.

[27] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39(3), 459–471. https://doi.org/10.1007/s10898-007-9149-x.

[28] Marqas, R. B., Almufti, S. M., Ahmed, H. B., & Asaad, R. R. (2021). Grey wolf optimizer: Overview, modifications and applications. International Research Journal of Science, Technology, Education, and Management, 1(1), 44–56. https://doi.org/10.14419/efkvvd44.

[29] Marqas, R. B., Almufti, S. M., Othman, P. S., & Abdulrahman, C. M. (2020). Evaluation of EHO, U-TACO and TS metaheuristics algorithms in solving TSP. Journal of Xi'an University of Architecture & Technology, 12(4), 3245–3246.

[30] Mehrabian, A. R., & Lucas, C. (2010). A novel numerical optimization algorithm inspired from the behavior of lions. Scientia Iranica, 17, 424–433.

[31] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in Engineering Software, 69, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007.

[32] Sahoo, G., & Tripathy, R. (2020). Comparative study of nature-inspired algorithms for TSP. International Journal of Computer Applications, 175(6), 1–6.

[33] Shaban, A. A., & Ibrahim, I. M. (2025). Swarm intelligence algorithms: A survey of modifications and applications. International Journal of Scientific World.

[34] Shaban, A. A., & Yasin, H. M. (2025). Applications of the artificial bee colony algorithm in medical imaging and diagnostics: A review. International Journal of Scientific World, 11(1), 21–29. https://doi.org/10.14419/yszxm607

[35] Shaban, A. A., Almufti, S. M., Asaad, R. R., & Marqas, R. B. (2025). Swarm-based optimisation strategies for structural engineering: A case study on welded beam design. FMDB Transactions on Sustainable Computer Letters, 3(1), 1–11. https://doi.org/10.69888/FTSCL.2025.000355

[36] Shaban, A. A., Dela Fuente, J. A., Salih, M. S., & Ali, R. I. (2023). Review of swarm intelligence for solving symmetric traveling salesman problem. Qubahan Academic Journal, 3(2), 10–27. https://doi.org/10.48161/qaj.v3n2a141.

[37] Siddique, B., & Adeli, H. (2018). Vibrating particle system algorithm for optimization. Journal of Computing in Civil Engineering, 32(1), 04017080. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000715.

[38] Wang, G. G., Deb, S., & Coelho, L. D. S. (2015). Elephant herding optimization. In Proceedings of the 2015 International Symposium on Computational Intelligence and Design (pp. 1–6). https://doi.org/10.1109/ISCID.2015.134.

[39] Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In J. R. González et al. (Eds.), Nature inspired cooperative strategies for optimization (NICSO 2010) (pp. 65–74). Springer. https://doi.org/10.1007/978-3-642-12538-6_6

[40] Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In Proceedings of the World Congress on Nature & Biologically Inspired Computing (pp. 210–214). https://doi.org/10.1109/NABIC.2009.5393690

[41] Zebari, A. Y., Almufti, S. M., & Abdulrahman, C. M. (2020). Bat algorithm (BA): Review, applications and modifications. International Journal of Scientific World, 8(1), 1–7. Science Publishing Corporation. https://doi.org/10.14419/ijsw.v8i1.30120
[42] Almufti, S. M. (2015). U-Turning Ant Colony Algorithm powered by Great Deluge Algorithm for the solution of TSP Problem.
[43] Almufti, S. M., & Shaban, A. A. (2025). Comparative analysis of metaheuristic algorithms for solving the travelling salesman problems. *International Journal of Scientific World, 11*(2), 26–30.