JOURNAL OF SOFT COMPUTING AND DATA MINING
e-ISSN: 2716-621X

JSCDM

Vol. 6 No. 3 (2025) 271-291
https://publisher.uthm.edu.my/ojs/index.php/jscdm

Enhancing Pressure Vessel Design Optimization with a Hybrid CSA-PSO Algorithm

Ammar Adel Ahmed[1], Amera Istiqlal Badran[2], Ibrahim M. Ahmed[3], Yahya T. Qassim[4,5], Dilovan Asaad Zebari[6,7]*, Reving Masoud Abdulhakeem[8]

[1] *Department of Computer Science, College of Education for Pure Sciences, University of Mosul, Mosul, IRAQ*
[2] *Department of Computer Sciences, College of Computer Sciences and Mathematics, University of Mosul, Mosul, 41002, Nineveh, IRAQ*
[3] *Department of Cybersecurity, College of Computer Science and Mathematics, University of Mosul, Mosul, IRAQ*
[4] *Department of Medical Devices Engineering Techniques, Alnoor University, Mosul, Nineveh, IRAQ*
[5] *School of Engineering and Built Environment-Electrical and Electronic Engineering, Griffith University, Nathan Campus, Brisbane, AUSTRALIA*
[6] *Faculty of Computing and Information Technology, Sohar University, Sohar 311, OMAN*
[7] *Department of Information Technology, Technical College of Informatics, Akre University for Applied Sciences, Akre, Kurdistan Region, IRAQ*
[8] *Computer and Communication Engineering Department, Nawroz University, Duhok 42001, Kurdistan Region, IRAQ*

*Corresponding Author: dilovan.majeed@nawroz.edu.krd
DOI: https://doi.org/10.30880/jscdm.2025.06.03.018

Article Info

Received: 19 February 2025
Accepted: 6 August 2025
Available online: 30 December 2025

Keywords

Metaheuristics algorithms, constrained problem, welded beam design problem, big bang–big crunch, cuckoo search algorithm, artificial bee colony, vibrating particles system, water evaporation optimization

Abstract

The optimization of the Pressure Vessel Design Problem (PVDP) plays a crucial role in various engineering applications where structures with the most efficient material utilization are required. However, the intricate interplay between design parameters and strict constraints presents a significant barrier for classical optimization algorithms to efficiently traverse the convoluted landscape of PVDP. Applying metaheuristic algorithms has been introduced as a solution to tackle these problems. The techniques introduced have the potential to confront the difficulties associated with nonlinearity, multimodality, and complex constraints that are frequently encountered in complex engineering optimization problems. This work carries out a comprehensive and comparative analysis of the performance of well-established metaheuristic algorithms in solving the PVDP. The methods under consideration at present are Big Bang –Big Crunch (BB-BC), Cuckoo Search Algorithm (CSA), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Vibrating Particles System (VPS), and Water Evaporation Optimization (WEO). Third, we introduced the CSA-PSO algorithm. Through extensive numerical experiments, their rates of convergence, the quality of the solutions, and the amount of computation required are compared and analyzed, revealing their relative pros and cons in terms of the PVDP. As a result, the knowledge

of how metaheuristic algorithms perform on the PVDP can be exploited when selecting optimization methods.

1. Introduction

The Pressure Vessel Design Problem (PVDP) is defined as the selection of optimal design variables that guarantee the structural integrity and material economy for the pressure vessel. The coupling of design parameters and the constraints in the PVDP system creates a significant difficulty for regular optimization methods. Therefore, combinations of metaheuristic algorithms have become a good alternative in order to overcome the intricacies of this optimization problem [1]. PVDP is important in industrial fields in which high structural integrity must be maintained with low material utilization. To succeed in this balance, batch design variables based on rigorous safety conditions have to be found. This issue becomes more complicated due to non-linear constraints, which motivates the search for solutions beyond common optimization procedures.

The potential and significance of these metaheuristic algorithms as effective black-box optimization techniques have been discussed due to the intractability of many engineering optimization problems (e.g., the PDVP as considered in this study). Nature-inspired, human-inspired, and physics-inspired algorithms represent a wide range of efficient methodologies for addressing hard optimization problems [2]. Prior art associated with the PVDP has investigated a number of well-known metaheuristic algorithms, such as Particle Swarm Optimization (PSO) and Genetic Algorithms (GAs). This paper is devoted to adapting and exploring five algorithms on their performance in comparison with the some best perform ones in literature with new algorithms for Solar PV power output prediction that have different nature: Big Bang-Big Crunch (BB-BC), Cukoo Search Algorithm (CSA), Artificial Bee Colony (ABC), and Vibrating Particles System (VPS), Water Evaporation Optimization (WEO).

The main objective of this study is to conduct a thorough and comparative study on the performance of all these MHAs to solve the PVDP. The influences of different algorithm parameters on the performance of the metaheuristic are investigated, showing that the algorithms have some dynamic properties. The impact of varying parameter values systematically on convergence rate, solution accuracy, and computational cost is studied. The purpose of this detailed study of parameter sensitivity is to provide assistance in selecting the best parameter settings for each method in the context of the Pressure Vessel Design Problem. In addition, in order to put the findings in the larger setting, we compare the results obtained in this paper with those found in other related work. Such a comparison can help us identify regularities and differences, and hence contribute to the understanding of how metaheuristic algorithms behave in a variety of problem contexts.

The problem addressed in this study is to compare different meta-heuristic algorithms in solving PVDP. To further extend the scope of the problem, certain aspects are studied. The complexity of the PVDP is first analyzed in terms of the non-linear form of the design problem and its constraints. The computational complexity of different algorithms is also a factor; some are simply more efficient than others in terms of resource usage. Additionally, the scalability and robustness of these algorithms are investigated when faced with larger problem sizes and constraints, demonstrating their performance on a more complex and extensive scale. The sensitivity of the algorithm's performance concerning the parameter settings is another important aspect, where the parameter settings, population size, and mutation rate can have a significant influence on the quality of the solution and computational performance. The research also examines the gap between the application of these algorithms to real-world problems, where other practical constraints come into play, such as manufacturing tolerances, material costs, and quality requirements. Solving these practical issues may result in optimized and effective design practices in industry. The main contributions of the presented study are as follows:

- A hybrid optimization method is suggested, which consists of using the CSA and PSO. This combines the best of both worlds in that it enhances the convergence rate and accuracy of the solutions, especially when dealing with complex optimization problems, and thus, it is a more efficient way of solving the PVDP.
- This work is a detailed comparison of various metaheuristic algorithms such as CSA, Water Evaporation Optimization Algorithm (WEOA), Vibrating Particles System (VPS), and Artificial Bee Colony (ABC). Based on this, the evaluation uncovers the advantages and drawbacks of each algorithm when applied to solving the PVDP.
- Key performance figures examined in the paper are objective function value (F(x)), error rates, computational time, and convergence speed to determine the most favorable trade-offs involving solution quality and computational efficiency.
- Heuristics are given to aid in choosing the best algorithm to use in various sizes of problems, constraints, and the accuracy of the solution needed in industrial problems. 5. The performance of the algorithms is verified in experimental conditions by simulating the behavior with different problem scenarios and verifying the performance in real-world conditions and with different design constraints.

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

2. Related Works

A metaheuristic algorithm that can solve complex optimization problems with nonlinearity, high dimensions, and many local optima. These are inspired and motivated by the behavior of organisms, physical phenomena, and social phenomena [3]. A metaheuristic is the optimization strategy that greedily refines potential solutions according to its rules or procedures as the solution representations span large solution spaces. Metaheuristics are applicable to a variety of problems where analytical solutions are hard to get or too costly to calculate, because they usually have no strong assumption about the problem, and no knowledge of the gradient is needed. They are well-suited to optimization problems of unconstrained and irregular landscapes as they compromise between exploration and exploitation of search spaces. Such flexibility and the capability to search for global solutions while circumventing local ones have driven the metaheuristic algorithms to become competitive in the area of optimization research and have been receiving attention in engineering, economics, and logistics. etc [4].

Several metaheuristic algorithms have been evaluated considering the PVDP, each one possessing its own strengths and advantages, such as the Genetic Algorithms (GAs) [5]. Due to the ideas behind natural selection and evolution, genetic algorithms are well recognized for their ability to deal with complex and time-varying search spaces. The GAs were employed in [6] for optimization of mechanical property pressure vessel designs, simultaneously taking into account limitations while making the design variables better. [7] also presented a multi-objective GA based approach for pressure vessel design optimization with an emphasis on trade-offs between conflicting objectives.

PSO simulates particles in a search space, and the collective's behavior is utilized to optimize pressure vessel designs [8]. This algorithm is effective in exploring the solution space and tracking down optimal designs. In [9], the authors extended the PSO algorithm to handle the multi-objective PVDP, where safety and economic objectives are considered simultaneously. Also, since Simulated Annealing implements a simulation of the annealing of metal, it can prevent the placement from being trapped at local optima and find the global optimization. [10] Use this method to compute the optimal pressure vessel design and show that this approach is capable of handling non-linear constraints and realizing balanced designs.

The issue of improving the performance of SA in solving the PVDP has been addressed in [11] by introducing the sensitivity analysis that employs the operations of mutation, crossover, and selection for evolving solutions to the problem at hand. Reference [12] was utilized to improve the pressure vessel design; it showed that it can produce satisfactory solutions based on the complex demands of the PVDP. By using a modified DE variant, the work [13] solved design optimization problems with multiple objectives. Work [14] employed Harmony Search, an optimization method inspired by musical harmony, to optimize pressure vessel design solutions. This method used an ACO for the limitation proposed for the PVDP. [15] studied the ACO model and showed that it was capable of overcoming the limitation due to the PVDP. In the work of [16], the Firefly Algorithm (FA), inspired by the flashing behavior of fireflies seeking a mate, was proposed, demonstrating its effectiveness in solving multi-objective problems and PVDP. Work [17] employed Grey Wolf Optimizer (GWO) to obtain perfect designs of the pressure vessels [18] by simulating the hunting of grey wolves.

3. Constrained Optimization Problem

Constrained optimization problems include the most important subset of optimization problems for which we have to find optimal solutions in a given feasible region, but satisfying given constraints. While optimizing one particular objective function, these constraints might impose several different requirements that need to be fulfilled. Constrained optimization problems arise in numerous disciplines, such as engineering, economics, and logistics, where the sought-after solutions must not only optimize a particular objective but also adhere to particular constraints or requirements.

A constrained optimization problem seeks to find values of the decision variables that minimize or maximize an objective function subject to some constraints. There are equality constraints and inequality constraints. While inequality constraints define one or more regions of acceptable solutions, equality constraints impose one or more relationships that must be satisfied exactly [14]. A prototype of a bound-constrained optimization problem is of the form below Eq. 1:

$$\text{Minimize (or maximize) } f(x), \text{ Subject to } g(x) \leq 0, h(x) = 0, lb \leq x \leq ub \quad (1)$$

Here, f(x) is the function we wish to minimize (or maximize). g(x) is inequality constraints to be respected (g(x)≤0). h(x) = 0 are the equality constraints to satisfy. Where lb, ub are the lower bound and upper bound of the position of the decision variables.

Under conditions of constrained optimization problems, the once-for-all problems become complicated because various constraints need to be optimized simultaneously, and the constraints are generally hard to solve. Analytical solutions 40 to these problems are generally not feasible; therefore, numerical methods must be used. Metaheuristics algorithms are well known for their powerful ability to explore complex search landscapes and to

deal with non-linear functions and constraints, being particularly useful to solve difficult optimization problems. These algorithms use an iterative technique to look for optimal solutions of the objective function while keeping constraint conditions around them. This feature makes them extremely suitable for solving real-world optimization problems, for which practical solutions are crucial [14].

The Pressure Vessel Design Problem (PVDP) is one of the classical multi-objective optimization problems that must obtain the optimal combination for certain design variables in the presence of multiple complex constraints. The PVDP can be defined as follows, in a strict manner:

For a pressure vessel with design variables described by x = [x1, x2,..., xn], n is the number of design variables, it tries to minimize the weight of the pressure vessel f(x) and satisfy a series of constraints which is related to the structural integrity and operative security [19]. The limitations include the stress, buckling, displacement, and other relevant mechanical aspects that the pressure vessel has to meet (Equation 1.)

The solution of the PVDP should be subject to both inequality and equality constraints, whereby the weight of the pressure vessel is minimized. Nevertheless, aspects in relation to non-linearity of the objective function and the nature and the interaction of the constraints make closed-form, analytical solutions difficult to obtain. Metaheuristic algorithms provide an effective tool to deal with the constraints of optimization problems, such as the PVDP. Such algorithms can efficiently search for feasible solutions, gradually adapt the Bias, such that the design variables can meet the constraint conditions and maximize the objective function value. In the following sections, the use of some metaheuristic algorithms for the PVDP is presented, which have been successful in designing good pressure vessel designs that maximize the safety factor while minimizing the material used [20].

4. Pressure Vessel Design (PVD) Problem

The PVDP is an important design optimization tool applied in many fields, such as the chemical industry, power generation, and aerospace. PVDP is a process of optimizing design factors for structural strength of pressure vessels, operational safety, material efficiency, and functional effectiveness. Many industries rely on these pressure-containing tanks.

The optimization of pressure vessel design factors, such as thickness and radius, minimizes weight while meeting mechanical, thermal, and safety requirements. Mechanical restrictions, such as stress, strain, and buckling, require designs that can tolerate loads and heat stress. Safety requires resistance to pressure changes and unexpected events. This complex optimization environment necessitates intricate trade-offs between objectives and constraints, making it a nonlinear problem. Solution strategies using metaheuristic algorithms are effective. We utilize these algorithms to identify pressure vessel designs that strike a balance between competing factors [21]. Through careful algorithmic research, this paper shows how metaheuristic approaches may solve the Pressure Vessel Design Problem. This work helps engineering design optimization by expanding our grasp of complex real-world problems.

Figure 1 illustrates PVD model with four options variables Now, the total variables as (x1,x2,x3,x4) = (pressure vessel thickness Ts, head thickness Th, vessel inner radius R, vessel barreling head length L) = min total cost (material + forming + welding) So the basic unique pressure vessel design optimization idea is described as Find: X=(X1, X2, X3) for (Ts, Th, R, L) and detailed based on Eq. (2) – (6).

To minimize:

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

1-Hoop stress ≤ Allowable stress

$$g_1(x) = -x_1 + 0.0193x_3 \leq 0 \quad (3)$$

2- Longitudinal stress ≤ Allowable stress

$$g_2(x) = -x_2 + 0.00954x_3 \leq 0 \quad (4)$$

3- Volume ≤ Desired volume

$$g_3(x) = -\Pi x_3^2x_4 - \frac{4}{3}\Pi x_3^2 + 1296000 \leq 0 \quad (5)$$

4- Length of Shell:

$$g_4(x) = x_4 - 240 \leq 0 \quad (6)$$

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

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

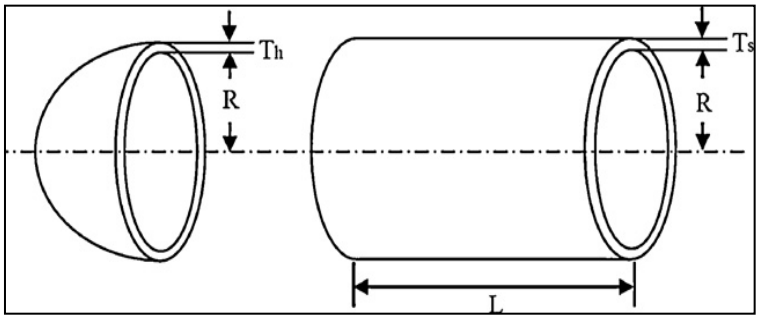

Fig. 1 *Pressure vessel design*

5. Metaheuristics Algorithms

Metaheuristics, as a single component within the optimization area, have been developed as a powerful and flexible class of algorithms. They have revolutionized how we approach problem-solving when it comes to complex and diverse problems [4]. These are inspired by natural and social systems and their dynamics to search complex search spaces and find solutions, either optimal or close to optimal. Metaheuristics are very powerful and flexible problem solvers for non-standard optimization problems, in particular when standard optimization techniques (for various reasons, such as inherent non-linearity, large dimensions, constraint-based, etc.) cannot be effectively applied.

Metaheuristics have addressed problems of vast models, and cutting-edge work has recently extended the range of solution techniques. One such example is the Cuckoo Search Algorithm (CSA) [22], which emulates the 6 ASCC 2019 reproductive strategy of cuckoo birds that lay new eggs and throw old ones with the help of Levy flight distributions. The procedure appears to have the ability to find a global optimum, in particular, in challenging terrains. GWO [10] mimics the hierarchy system and hunting mechanism of grey wolves using α, β, Δ, and ω wolves to depict different stages of the optimal solution. Applications GWO has been applied to parameter optimization of machine learning models, tweaking of control parameters in engineering systems, image classification-based feature selection, etc. The Firefly Algorithm (FA) [23], based on the flashing dynamics of fireflies, is well known for its effectiveness in solving various optimization problems due to its capability of exploring a complicated search region in an effective manner. The WEO algorithm is based on the concept of water evaporation in finding good solutions and contributes a new flavor. The novel approach offers new alternatives to tackle challenging issues. The process of natural selection and evolution is emulated using genetic algorithms (GAs) [8], allowing solutions to become better with each generation through techniques such as mutation, crossover, and selection. In engineering, economics, and AI, GA is applied to efficiently search for parameters in complex systems.

In PSO of [11], assume that the swarm size is small, and every individual in the swarm updates its position based on its historical performance and the performance of its neighbors. Besides robot path planning, PSO has also been applied to neural networks [20] and economic models [21]. ACO [15] takes the foraging behaviors of ants as a reference, and simulates ants laying pheromones to communicate while searching for optimal paths between nests and food sources. It is effective in solving routing and graph-based problems, e.g., vehicle routing and network routing. 13], motivated by the way honeybee colonies forage, in which bees search for food and communicate locations of food sources, by waggle dances, reflecting multi-directional search behaviors. It is used in image clustering and feature selection in data mining. Bat Algorithm (BA) [21] focuses on the echolocation and the flying behavior of bats that they use to hunt prey and to change their flying direction based on seismic waves. It is applicable to continuous optimization problems, such as parameter estimation and image segmentation.

The flexibility of metaheuristics is profusely verified by their application to numerous fields and industries. These algorithms have been used to advance optimization of structural design in the engineering field a great deal. This approach has been proven to be particularly useful in the design of efficient and robust solutions for a variety of applications such as pressure vessels, aircraft wings, and truss structures [24]. The use of metaheuristics in transport and logistics also has some advantages, specifically to optimize routing and scheduling so as to increase the transport process efficiency and reduce cost [25]. Energy optimization has had great novelties as metaheuristics have been employed to optimize power generation and distribution tasks, leading to a more optimal way that resources are distributed.

Metaheuristics are largely used in finance to optimize portfolios by aiming for optimal risk/return trade-offs when making investment decisions. Metaheuristics are applied to healthcare for a wide range of applications, including medical image registration, personalized treatment planning, and disease prediction modelling. Machine learning models promote their flexibility to optimize hyperparameters, ultimately improving prediction quality.

5.1 Artificial Bee Colony Algorithm

Karaboga devised the artificial bee colony algorithm (ABC) in 2005 as a swarm-based metaheuristic method for optimizing numerical problems, drawing inspiration from honey bee behavior [7]. ABC algorithm bees are categorized into three roles: workers, observers, and scouts. An employed bee travels to a food source independently and waits on the dance floor to select the best choice. A scout bee searches for everything. Artificial bees make up half of the ABC algorithm colony, while observers make up the other. Each food source has one hired bee. Thus, the colony's active population equals its food supplies. The hired bee becomes a scout when the employed bees and the spectator exhaust their food source [9]. The most crucial event in collective knowledge acquisition is the exchange of information. When surveying the entire hive, some similar elements can be found. Most hive communication occurs in the dance area. Bees debate food quality in the dance area. Waggle dance describes this movement. A person watching a dance has access to all the lucrative sources, so she may watch numerous before choosing the most lucrative one. More knowledge about sources increases the likelihood that watchers will choose the most profitable ones. The recruitment procedure depends on the profitability of the food supply.

The system instructs the hired bees to hunt for food sources based on their fitness levels and shares the results to attract other bees. The populace has as many solutions as hired bees or watchers. Solution (food source) xi is a D-dimensional vector with I = 1, 2, ..., SN. The observation bees will use this knowledge to choose a food source. Higher-quality food is preferred by observer bees.

The ideal reaction is found by following a process similar to a swarm of bees seeking, promoting, and prioritizing the best food source. The probability value pi associated with a food source determines which observer bee uses it, as computed in Equation 7 [2]:

$$Pi = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{7}$$

fit is the employed bee's fitness value of the solution *I*, which is similar to the nectar amount of the food source at position *I*, and *SN* is the total number of food sources, which are equal to the total number of employed bees. Thus, the booked bees perform pet-bite and laptop skills transfer. The ABC employs Eq. (8) to generate a candidate food position from the previous food position:

$$x_{ij}^{*} = x_{ij} + \varphi_{ij} (x_{ij} - x_{lj}) \tag{8}$$

x_{ij}^{*} is a new food source determined by comparing the d(x_{ij}) to a randomly chosen X of l and j, where l ∈ {1,2, ..., SN} and j ∈ {1,2, ..., D}, and $\varphi_{ij} \in [-1,1]$. It replaces the current food source with a new one in the next iteration. Algorithm 1 shows the ABC algorithm's pseudocode, which uses variable initialization, employed, onlooker, and scout phases to find the best answer.

Algorithm 1: The ABC algorithm

```
Initialize employed bees randomly
Evaluate the fitness of each bee's solution
Repeat for a certain number of iterations:
  For each employed bee:
    Generate a new solution, evaluate fitness
    If a new solution is better, replace the current one.
  For each onlooker bee:
    Select guide, generate new solution, evaluate fitness
    If a new solution is better, replace the current one.
  For each employed bee:
    If trials exceed the limit, abandon and generate new ones
    Evaluate fitness
  Update the best solution
Return the best solution found
```

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

5.2 Water Evaporation Optimization Algorithm (WEOA)

WEOA resolves complex engineering issues based on the observation of the movement of the water droplet and evaporation. Study [26] developed the WEOA algorithm as a special algorithm to overcome optimization problems. It is like the evaporation of water, and this inspires WEOA. There is the occurrence of a difference in vapor pressure, which naturally pushes water droplets towards evaporation. Water droplets in the algorithm are the candidate solutions that are tested on their fitness to come up with the best one (evaporation source). Through this movement, the algorithm can locate near-optimal solutions successfully [27].

The WEOA initiates water droplets that occupy diverse solutions, as shown in Algorithm 2, performs a fitness value of each water droplet, moves water droplets near the best solution, and finally, there is evaporation that enhances the step size. Local search may be used to optimize solutions around their current locations to enhance the convergence of the algorithm. The idea behind WEOA is the evaporation of water on solid materials and not on bulk surfaces. This kind of evaporation of water is commonly macroscopic, like the soil evaporation [27].

Algorithm 2: The WEOA algorithm

```
Initialize solutions randomly
Evaluate the fitness of each solution
Repeat for a certain number of iterations:
  For each solution:
    Update variables with perturbations, evaluate fitness
    If improved, update the solution
  For each solution:
    Evaporate variables, evaluate fitness
    If improved, update the solution
Return the best solution found
```

Molecular Dynamics (MD) computations were conducted by researchers to study the vaporization of water at solid surfaces of different wettability. Surface wettability can be controlled with a change in charge value (0e to 0.7e) by collecting and attaching nanoscale water to a naturally chargeable substrate. Water vaporization is called evaporation flux. The average time of sub-molecules impacting the acceleration region is within nanoseconds. In spite of the projections, the velocity of vaporization is unaffected by variation in the charges on surfaces, given that the surface changes in charged status, i.e., hydrophobic (q < 0.4 e) and hydrophilic (q >= 0.4 e). It rises and falls once it is at its peak. Such an abnormal evaporation stream can be interpreted by the synergistic influence of the concentration and escape opportunities of a water molecule on the interfacial fluid-air base, Eq. (9) and (10).

$$J(q) \propto P_{geo}\big(\theta(q)\big)P_{ener}(E) \tag{9}$$

$$P_{geo}(\theta) = p_0\left(\frac{2}{3} + \frac{cos^3\theta}{3} - cos\theta\right)^{-\frac{2}{3}}(1 - cos\theta) \tag{10}$$

Here, Pgeo(θ) in Equation 10 is the probability of a water molecule on the liquid-gas interface due to system geometry. P0 is a constant function of water molecule width and system molecule volume. Pener(E) is the probability that a water molecule will escape from the surface layer as a function of the average energy (E) of its interactions. This interaction energy (EWW + Esub(q)) is composed of a contribution of the interaction with the neighboring water molecules (EWW) and that corresponding to the interaction energy of the substrate, mainly due to the electrical charge (q) transferred to the substrate [28].

Here, P geo (θ) is the probability of a water molecule being located at the liquid-gas interface, and it depends on the geometry of the system. P0 is a constant of the order of the width of a molecule of water multiplied by the volume of the molecules of the system. Pener(E) is the probability for the escape of a surface water molecule of a water cluster with a mean interaction energy (E). EWW + Esub(q) = binding energy due to water neighboring molecules (EWW) + interaction energy due to the substrate due to the coulombic charge (q) on the substrate, Eq. 11 [29].

$$J(\theta) = J_0 P_{geo}(\theta), q < 0.4\ e \tag{11}$$

In this case (q>0.4 e), water sticks to the substrate with minimal overlap and thus creates a flat monolayer with a well-defined single-layer morphology, and little water aggregation is stable in terms of morphology with respect to q. When all of the water molecules are on the surface layer, we define Pgeo(θ)=1. In the NVT ensemble, the probability of a free molecule having kinetic energy above E0 is exp $(-\frac{E_0}{K_B T})$ where T is the room temperature, and KB is the Boltzmann constant. MD simulations indicate an almost exponential decay of Esub flow in evaporation. As a result, the evaporation flow of Eq. 9 will be modified as in Eq. (12).

$$J(q) = exp\left(-\frac{E_0}{K_B T}\right), q \geq 0.4\, e \tag{12}$$

5.3 Big Bang-Big Crunch Algorithm

The algorithm theorizes the universe's expansion and collapse, known as Big Bang-Big Crunch (BB-BC). In continuous optimization, the BB-BC algorithm has been proposed in [30] in the case of continuous optimization. It finds the best or near-best solutions through a balance between exploration and exploitation. Refinement of a set of possible solutions does this. The BB-BC method initializes a set of possible solutions, as shown in Algorithm 3. It progressively searches the space of solutions, refinements, and comes close to optimal/almost-ideal solutions [31]. The random in search space approach is referred to as the Big Bang-Big Crunch (BB-BC). The first stage of this algorithm causes a converging operator referred to as the Big Crunch. Then, step sizes close to the converging operator of phase two, the Big Bang, are used in updating the particles in the search space. A population or solution matrix (P) is taken into consideration. The matrix has nP particles. Penalized objective functions go hand in hand with individual particles. In this iteration, the best observed (bestP) particle is the one that has the minimum penalized objective function [32].

Algorithm 3: The BB-BC algorithm

```
Initialize a random population of solutions
Evaluate fitness for each solution
Repeat for a certain number of iterations:
  If the convergence condition is met, stop
    Explore phase (Big Bang):
      For each solution:
        Adjust the solution's variables randomly
        Evaluate the fitness of the adjusted solution
        If the adjusted solution is better, replace the original solution
    Exploit phase (Big Crunch):
      Sort solutions by fitness
      For each solution:
        Calculate gravitational forces from other solutions
        Update the solution's variables based on forces
        Ensure variables stay within limits
        Evaluate the fitness of the updated solution
        If the updated solution is better, replace the original solution
Return the best solution found
```

The Big Crunch phase-based convergence operator can be described by the following weighted average of the position of candidate solutions, commonly known as the center of mass (CM) or the best option (bestP). The constraint minimizing (CM) approach, which is used when minimizing an objective function, is defined as follows, as displayed in Equation 13, and was first introduced by Ghasemi-Mirzavand in his study [33].

$$CM(i) = \sum_{j=1}^{nP}\left(\frac{P(j,i)}{PFit(j)}\right) / \sum_{j=1}^{nP}\left(\frac{1}{PFit(j)}\right), \qquad i = 1, \ldots, nv \tag{13}$$

During the initial stages of the Big Bang cosmological model. In the original BB-BC, particles undergo updates based on the previously established center of mass (CM) or the location of the best particle (bestP). This update involves relocating the particles by a random fraction of the permitted step size, which is set by the upper (Ub) and lower (Lb) limits of the design variables, Eq. 14 [34].

$$newP = (CM\ or\ bestP) + \frac{rand * (Ub - Lb)}{nIT} \tag{14}$$

Let rand be a uniformly distributed random number on (0, 1). By dividing the step size by the algorithm iterations or Big Bang phases, the effective search range around the global optimum or center of mass is determined. This divide limits the algorithm's search. In all metaheuristics, the population size and the maximum number of algorithm iterations, which act as a halting condition, are critical. The efficacy of a new Big Bang formulation that added two criteria has been modified as it is given in Equation 15:

$$newP(i) = (\beta * CM + (1 - \beta) * bestP + \frac{rand * \alpha * (Ub - Lb)}{nIT},\ \ i = 1, \ldots., nP \tag{15}$$

5.4 Vibrating Particles System (VPS)

In order to promote the free vibration of single-degree-of-freedom systems with viscous dampening, Kaveh and [11] first developed and suggested the evolving metaheuristic search method, Vibrating Particles System (VPS) in 2017. VPS has been applied to various structural optimization problems and has shown both algorithm convergence and accuracy [12]. Similar to population-based metaheuristics, VPS begins with a random group of initial solutions and considers them as free vibrating single-degree-of-freedom systems with a single mode of vibration. Damped every free vibrating system with natural frequency and returns again after some time to its equilibrium with a specific formulation. We can do this easily through differential equations. During the optimization process, VPS sometimes improves the quality of the particle by the oscillation of the particles toward the equilibrium based on random values [14].

Consider each particle's equilibrium position as the best/highest position (HP), a good particle (GP), and a poor particle (BP). Thus, VPS is built on three core ideas:

- Self-adaptation: particle moves towards HP.
- Particles may change their location by cooperation (GP and BP) selected by themselves.
- Competitions: GP is more influential than BP.

VPS uses harmony search memory to correct particle positions, leaving the search space. Eq. 16 shows the formula that is used to describe the free shaking of a single-degree-of-freedom (SDOF) system with too little damping.

$$X(t) = \rho e^{-\varepsilon \omega_n t} \sin(\omega_D t + \theta) \tag{16}$$

ω_n is the vibration's natural circular frequency, whereas ρ and θ are constants obtained from the vibration's initial conditions. Figure 5 shows the damped natural frequency ω_D and damping ratio (ε), which can be determined using Eq. 17 and 18.

$$\omega_D = \omega_n \sqrt{1 - \varepsilon^2} \tag{17}$$

$$\varepsilon = \frac{c}{2m\omega_n} \tag{18}$$

Eq. 19 generates the starting locations of all particles in the search space at random:

$$X_j^i = X_{min} + rand * (X_{max} - X_{min}) \tag{19}$$

The vectors X_{min} and X_{max} represent the minimum and maximum permitted values, respectively. The variable "rand" represents a random integer that follows a uniform distribution inside the interval [0, 1]. Figure 2 shows the vibrating motion of an under-damped system.

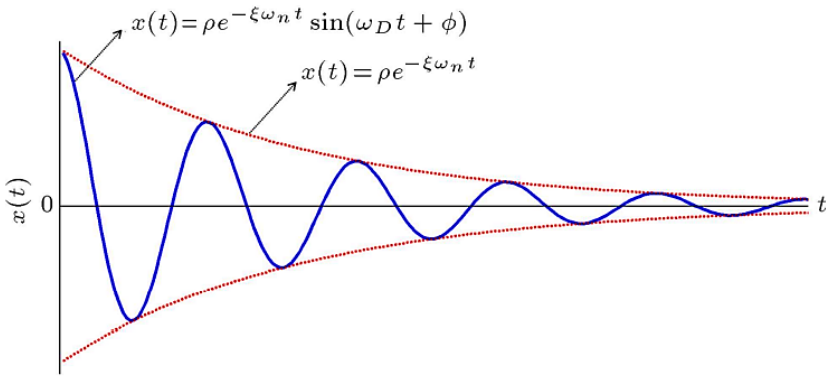

Fig. 2 *Vibrating motion of an under-damped system*

As can be seen, a single-degree-of-freedom system with under-damping exhibits free vibration and gradually approaches its equilibrium position. The concept of vibrating features serves as the inspiration for the VPS. Like other metaheuristic algorithms, the VPS starts by generating a collection of initial solutions within the search space at random. The acronym nVP represents the quantity of vibrated particles. The objective functions (Fit) and their penalized functions (PFit) are constructed after the evaluation of the items.

The VPS algorithm incorporates updates to the variables HP, GP, and BP, assigning different weights (ω_1, ω_2, and ω_3) to each particle. Once the population has been arranged in ascending order based on the values of the penalized objective function, each particle is assigned randomly chosen GP and BP values from the first and second halves of the population, excluding its own values [12].

The degree of damping has a substantial influence on the phenomenon of vibration. The magnitude of a damped free vibration diminishes with an increase in the amount of damping. In order to replicate this behavior inside the VPS environment, the descending function (D) is determined based on the number of repetitions, as computed by Eq. 20 and 21.

$$D = \left(\frac{NITs}{maxNITs}\right)^{\alpha} \tag{20}$$

$$maxNITs = \frac{\text{maxNFEs}}{nVP} \tag{21}$$

NITs denote the current number of algorithm iterations, and the maximum number of algorithm iterations is referred to as maxNITs, which is used as a stopping criterion. maxNFEs is also the maximum number of function evaluations, and α is a constant. The value of 0.05 is recommended as the optimal value. Based on the ideas presented above, the particles are updated by the following formulae, which will be called the free vibration formula for abbreviation in the sequel, where the Eq. are written as (22), (23), and (24) [2].

$$newVP_i = \omega_1(D * A * rand_1 + HP) + \omega_2(D * A * rand_2 + GP_i) + \omega_3(D * A * rand_3 + BP_i) \tag{22}$$

$$A = \omega_1(HP - VP_i) + \omega_2(GP_i - VP_i) + \omega_3(BP_i - VP_i) \tag{23}$$

$$\omega_1 + \omega_2 + \omega_3 = 1 \tag{24}$$

where VPi and newVPi are the positions of the ith particle at the present and updated coordinates, respectively. The constants ω1, ω2, and ω3 are weights and are used to check relevance among the ith particle's good, bad, and the algorithm's best so far particle. The variable rand is a random number from 0 to 1.

In Equations 22 and 23, the variables A and D have a comparable impact on the algorithmic process, akin to the influence of ρ and e^ (-εω_n t) in Equation 16. The sine function of the quantity ω_D t+θ is seen to have a value of one. An initial parameter, denoted as p and constrained to the interval (0,1), is established to determine the extent to which the impact of BP should or should not be taken into account during position updates. The variables p and random are compared in accordance with Eq. 25 [35].

$$\text{If P<rand} \rightarrow \omega_3 = 0, \omega_2 = 1 - \omega_1 \tag{25}$$

The VPS model is based on three basic principles: self-adaptivity, cooperation, and competition. When a particle is near the wished point (HP), a self-adaptation occurs. In a virtual particle system (VPS), the movement of a particle may affect where other particles will be located and promote cooperation among the particles. The GP (good particle) effect is stronger than the BP (bad particle) effects due to the existence of the p parameter, leading to competition [36]. The VPS algorithm incorporates a technique for handling particles that have exceeded the variable limits (out-of-bound particles) using harmony search. In this study, the VP memory (VP-M) saves the nVPs, along with their corresponding Fit-M AND Penalized Fit-M particles. To achieve this task, we utilize vibrating particle memory as the numerical nVP [37].

Memory and various algorithms can be utilized to improve the efficiency of metaheuristics and reduce computational cost. It is important to emphasize that the VPS can only be used to repeatedly regenerate desperately fleeing particles. The renewal of any element of the set of solutions, if it surpasses the pre-specified bounds, is possible using the proposed method by the VP-M approach. The values for this are obtained with the use of Eq. 26.

$$VP(i,j) = \begin{cases} w.p.\ vpmcr \Rightarrow & \text{select a new value for a variable from VP} - \text{M}, \\ & \text{w. p. } (1 - \text{par}) \text{do nothing}, \\ & \text{w. p. par choose a neighboring value.} \\ w.p.\ (1 - vpmcr) \Rightarrow & \text{Select a new value randomly} \end{cases} \tag{26}$$

The notation "*w.p.*" is the abbreviation for "with the probability." Here, VP(i,j) i is the jth element of the ith vibrated particle. The string vpmcr specifies the vibrating-particles memory, which ranges from 0 to 1. This rate decides the probability of choosing a value from the past values kept in VP-M. The term (1 - vpmcr) on the other hand expresses the probability of choosing by chance a value within the permitted interval. An operation for adjusting the pitch begins when the value is designated in V P-M. Parameter (1-par) is the rate of neglecting individual particles, and par is the rate based on the selected value, with part of the particles that are close to the best vibration particle and particles that are kept in memory. The random generation step size ±bw * rand in a continuous search space can be used to select a value from the particles stored in memory or the particles around the best vibrating particle. Algorithm 4 presents the VPS pseudo-code.

Algorithm 3: The BB-BC algorithm

```
Initialize a random population of solutions
Evaluate fitness for each solution
Repeat for a certain number of iterations:
  If the convergence condition is met, stop
    Explore phase (Big Bang):
      For each solution:
        Adjust the solution's variables randomly
        Evaluate the fitness of the adjusted solution
        If the adjusted solution is better, replace the original solution
    Exploit phase (Big Crunch):
      Sort solutions by fitness
      For each solution:
        Calculate gravitational forces from other solutions
        Update the solution's variables based on forces
        Ensure variables stay within limits
        Evaluate the fitness of the updated solution
        If the updated solution is better, replace the original solution
Return the best solution found
```

5.5 Cuckoo Search Algorithm

The Cuckoo Search (CS) method, first proposed by Deb and Yang [38], is a population-based metaheuristic that combines the Lévy flight theory with the behavior of a particular cuckoo species. In this algorithm, nests or eggs, whether owned by cuckoos or other host birds, are considered potential solutions. Both cuckoos and other host birds' endeavor to produce their own sets of solutions or generations [39].

The Cuckoo Search (CS) approach begins by generating a set of solutions randomly, which serve as representations of the nests or eggs belonging to the host birds. The dimensions of the Nest matrix are established by the value of the number of nests (nN), which serves as an initial parameter for the procedure. Following the evaluation of the initial population, the goal (Fit) and penalized objective function (PFit) vectors are formulated [40]. The method is characterized by a cyclic process that encompasses two successive search phases executed by both cuckoos and host birds. In the first phase, cuckoos use the Lévy flight mechanism to generate eggs, therefore directing their current solutions (Nest) towards the most optimal solution detected (bestNest).

The aforementioned recently produced eggs are afterwards introduced into the nests of avian hosts in accordance with a predetermined placement plan. Following this, the algorithm proceeds to the phase involving the host birds. In this context, the avian hosts are seen to detect a proportion (pa) of eggs that do not belong to their own species, and then modify them using a step size determined by a random permutation. The replacement technique is used to replace the existing eggs with newly created ones. The substitution of existing eggs may be achieved by the use of a probability-based update matrix, as proposed by [41].

a) Lévy Flights as Random Walks

Randomization plays a vital role in metaheuristic algorithms, making substantial contributions to both the search and usage phases. The fundamental principle behind this process of randomization is rooted in the theoretical framework of random walks. A random walk is characterized by a sequence of consecutive random steps, which gives rise to a stochastic process [6]. Let us denote SN as the cumulative total of each subsequent random step Xi. Therefore, SN represents the random walk in Eq. 27.

$$S_N = \sum_{i=1}^{N} X_i = X_1 + X_2 + \cdots + X_N = \sum_{i=1}^{N-1} X_i + X_N = S_{N-1} + X_N \tag{27}$$

In the simple random walk, the step (Xi) is determined by a random distribution. This means that the future state of the system depends only on the present state and on the transition XN from the present state to the next. The above idea can be generalized to higher dimensions by performing the above ways in an n-dimensional space. What's more, the length of the steps isn't required to be the same every time. The step size, in reality, can be drawn from a certain distribution. For instance, if the value of the step length corresponds to a Gaussian distribution, the random walk transforms to the famous Brownian motion. However, if the step length follows a Levy distribution, the corresponding stochastic process is called Lévy flight or Lévy walk [6].

The implementation of random numbers using Lévy flights has two primary stages: firstly, the random selection of a direction, and secondly, the generation of steps that conform to the selected Lévy distribution. The process of generating these stages might be complex; however, one of the more efficient and straightforward approaches is the Mantegna algorithm. The computation of the length of step (S) in Mantegna's method may be derived from the following expression, Eq. 28:

$$S = \frac{u}{|v|^{1/\beta}} \tag{28}$$

The parameter β is recommended to be within the range [1-2] and is often indicated to have a value of 1.5 according to literature guidelines. The variables u and v are derived from a normal distribution, which is represented as follows in Eq. 29 and 30:

$$u \sim N(0, \sigma_u^2), \qquad v \sim N(0, \sigma_u^2) \tag{29}$$

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta) * \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{(1+\beta)}{2}\right) * \beta * 2^{\frac{\beta-1}{2}}} \right\}, \quad \sigma_u = 1 \tag{30}$$

Research findings indicate that L´evy flights prove successful in the enhancement of resource search in dynamic, unpredictable environments. Flights like these have been spotted in feeding behavior in various animals, such as albatrosses, spider monkeys, and fruit flies. These examples highlight the natural use of Lévy flights as an evolutionary strategy to improve resource exploration and exploitation efficiency in ecologically relevant contexts.

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

b) Cuckoo Breeding

We select the nests or eggs on the basis of their quality, except the nest marked the highest quality (i.e., the bestNest). The Lévy flight mechanism is employed in the CS algorithm to generate new cuckoo eggs (newNest) and to move the current solutions (Nest) towards the global optimal solution (bestNest), as given by Eq. (31) and (32) below.

$$Stepsize = rand_{(i)(j)} * \alpha * S * (Nest - bestNest) \tag{31}$$

$$NewNest = Nest + Stepsize \tag{32}$$

During this particular stage, the process of creating new cuckoo eggs (newNest) depends on the step size parameter α, and the user should guess and make it more than zero, based on how "hard" the difficulty being considered is. The symbol rand(i)(j) describes a pseudo-random number drawn from the continuous uniform distribution in the range from -1 to 1. Furthermore, the quantity S is a representation of a stochastic process called random walks that originates in the theoretical framework of Lévy flights [1] given by Kaveh and Bakhshpoori.

By introducing this phase in the algorithm, the elitism and intensification mechanisms are found. In this stage, the bestNest (the best solution) does not change, and the other solutions are adjusted toward it instead. This mechanism helps the algorithm focus on promising areas and the quality of the solutions.

c) Alien Eggs Discovery by the Host Birds

Extraterrestrial egg identification method: The method for identifying ET eggs is based on calculating the probability of finding these eggs for each component of each event. The possible outcome, using the discovery probability matrix (P), is predicted as in Algorithm 4.

Choice of replacing the present eggs in this case depends in a mixed way on the quality of the eggs and on those obtained by random walks of step sizes determined by random permutations, as shown in Algorithm 4. The replacement operation is controlled by the probability of discovery (pa) and the random number (rand) chosen from the region [0,1]. The P matrix and Nest matrix are the same size and play an important role in replacing and updating the egg, thereby improving overall solution quality [6], as shown in Eqs. (33), (34), and (35).

$$P_{(i)(j)} = \begin{cases} 1 & if\ rand < pa \\ 0 & if\ rand \leq pa \end{cases} \tag{34}$$

$$Stepsize = rand_{(i)(j)} * \left(Nest\left[permute1_{(i)(j)} - permute2_{(i)(j)}\right]\right) \tag{35}$$

$$NewNest = Nest + Stepsize * p \tag{36}$$

In this scenario, the random permutation functions such as randperm1 and randperm2 are utilized to perform row permutation required by the nest matrix and the discovery probability matrix (P), respectively. These permutations aim to replace existing eggs with recently produced eggs, thereby improving the algorithm's ability to create diversity in its solutions. Successful performance in this stage enables the ability of the algorithm to both explore different territories in the search space and exploit a diversity of solutions.

Algorithm 4: The ET eggs algorithm

```
Initialize nests randomly
Evaluate the fitness of each nest's solution
Repeat for a certain number of iterations:
  For each nest:
    Generate a new solution with Levy flight, evaluate fitness
    If improved, update nest
  Sort nests by fitness
  For each nest except the best pa%:
    With probability pa, replace a nest with a new solution
    Generate a new solution with Levy flight, evaluate fitness
    If improved, update nest
Return the best solution found
```

6. Proposed Method

In this paper, a new hybrid approach is proposed, combining the CS with the PSO, to enhance convergence rates and improve solution accuracy in problems of complexity similar to PVDP. In this new approach, it aims to merge the strengths of the two algorithms, specifically the ability of CSA to perform a global search and the ability of PSO to perform a local search. CSA belongs to the class of meta-heuristics and is based upon the breeding behavior of the cuckoo bird. It has been proven to have good exploitation and exploration abilities, as the updated solutions exhibit a Levy flight-like characteristic. The basic models of PSO are inspired by the flocking of birds during social behavior. The search is carried out by a particle swarm in the search space; each particle explores the search space individually according to its best-known position and the best-known position in the swarm. The hybrid approach aims to integrate the advantages of CSA and PSO with respect to exploration and exploitation, respectively. The algorithm initially performs exploration using CSA and then transitions to PSO for refining local improvements of the solutions.

The proposed method consists of two main parts. In the population initialization stage (Exploration Phase), CSA is employed for the global search of the algorithm. The global search is done by use of the Levy flights, which cause the search to move over a big portion of the solution space and prevent it from being stuck at a local point. The second phase (Exploitation Phase), after a promising area is identified, this phase makes PSO explore the promising area to form optimal solutions. PSO particles are guided by their best individual positions and the global best position to fine-tune the solution. It uses CSA as the exploration step aimed at surveying the solution space in the hybrid approach. The places of all solutions are updated as Eq. (37):

$$x_i^{t+1} = x_i^t + \propto . (x_i^t - x_r^t) + \beta . L(F) \tag{37}$$

Where x_i^t is the position of the i-th solution at iteration t, x_r^t is a randomly chosen solution in the population, $\propto$ determines the exploration rate, $\propto$ is the step size, and $L(F)$ is the Levy flight step, which creates a new candidate solution. Levy flights are used to perform random large jumps in the search space and are characterized by the following distribution, Eq. (38):

$$L(\lambda) \sim t^{-\lambda} \tag{38}$$

Where λ usually falls between 1 and 3. The distance moved in the Levy flight is drawn based on this distribution that favors exploration, especially at the beginning, to prevent premature convergence of the algorithm to a local minimum. At this point, CSA has searched the solution space and found good areas of interest; the algorithm transfers to PSO, the refinement stage. The updated eq. (39) of PSO is as follows:

$$v_i^{t+1} = w . v_i^t + c_1 . r_1 . (pbest_i - x_i^t) + c_2 . r_2 . (gbest_i - x_i^t) \tag{39}$$

The v_i^t is the velocity of the i-th particle after iteration t, w is an inertia weight which regulates the influence of the previous velocity, c_1 and c_2 are acceleration constants, r_1 and r_2 are random numbers between 0 and 1, $pbest_i$ is the personal best position of particle i, and $gbest_i$ is the global best position. The updated position has been calculated with the help of the following Eq. (40):

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{40}$$

The PSO updates drive the particles to the global optimum by a temporal balance between exploration (through the inertia term) and exploitation (through the personal and global best positions). Such updates assist in the optimization of the solutions discovered at the exploration phase. The CSA-PSO hybrid method adopts a two-phase approach: exploration, then exploitation. Firstly, a swarm of candidate solutions is created randomly, and the parameters of CSA (a, b) and PSO (w, c1, c2) are set randomly. Phase 1 (exploration phase) CSA uses the Levy flight mechanism to search globally, permitting the algorithm to search the solution space widely. At this stage, the candidate solutions' positions are refreshed.

Phase 2 is the exploitation phase, whereafter CSA has determined good areas of the solution space, PSO searches the solutions by moving particles towards the best solution using the velocity and position update equations. A convergence test is carried out to verify whether the stopping criterion (e.g., maximum number of iterations or acceptable solution) is satisfied. If not, the algorithm would iterate Phase 1 or switch to Phase 2 based on refinements needed. This enables the two search features - global and local so that the algorithm can reach a good solution to difficult optimization problems like the Pressure Vessel Design Problem.

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

7. Experimental Results

In the chapter on experimental results, an extensive assessment of the merit of the Metaheuristic Algorithm in how it solves the Pressure Vessel Design Problem (PVDP). In this section, we provide an in-depth study of the performance of the algorithms, which includes the convergence speeds, quality of the solutions, and the computational costs of each algorithm we analyze. The algorithms studied are Big Bang–Big Crunch, Cuckoo Search, Artificial Bee Colony, Vibrating Particles System, and Water Evaporation Optimization. By rigorously testing and analyzing these algorithms, this section conveys the specific strengths and weaknesses of these different algorithms when the parameters of the algorithm are altered. In addition, experimental results are compared with some previous studies to provide a baseline for measuring the potential effectiveness of these algorithms in this specific context. Despite its limited use for assessing the usefulness of the selected comparative metaheuristics, this work provides useful guidelines related to their adaptability and performance for solving challenging optimization problems, like the PVDP.

The algorithm executed by considering the upper and lower boundaries equal $1 * 0.0625 \leq x1 \leq 99, 1 * 0.0625 \leq x2 \leq 99, 10 \leq x3 \leq 240, 10 \leq x4 \leq 240$. All algorithms are compared with one run of maxNFEs =20000 times, using Core i7, Ram=16, and GPU =GTX 1660i. A holistic overview of the experimental results produced from the optimization of the PVDP utilizing the specified parameters is summarized in Tables 1,2,3,4,5. There is literature that shows that the performance of the algorithm and the quality of the solutions generated by the algorithm are dependent on two parameters that are fixed, and the third, which is varied and not constant. The table shows the minimum and maximum values, delivery time, and impact of a number of metrics.

For several configurations, the lowest, highest, and average values are recorded for the design variables (X1, X2, X3, X4) and for the related objective function F(X) value. These time measurements are useful in order to shed some light in the computational efficiency of the particular metaheuristic that was used. The results illustrated show drastic variations in design parameters and objective function values, suggesting the complexity behind the optimization. This is partly due to the controlled environment these experiments offer, which allows the results to be considered as a base for assessing the success of the algorithms, understanding their behavior, and comparing their efficiency as solutions to the complex PVDP.

7.1 BB-BC Algorithm

The BB-BC uses the parameters nP, Alpha, and Beta to limit the initial search space and randomize the first solutions. Analysis of the Effect of variants on the Performance of BB-BC in the PVDP Problem is shown in Table 1. Since the result is optimal, and we have chosen two parameters, we can fix all the other parameters and demonstrate how the algorithm works.

Table 1 *The obtained results of the BB-BC algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nP=50	beta=0.5 alpha=0.9	6183.6765	1.868E+14	2.219E+13	0.8996792	0.4442803	46.149784	132.37461	6183.6765	1.1258
nP=100		6124.8104	1.855E+14	1.24E+13	0.8734706	0.4318337	44.969667	144.8566	6124.8104	1.6093
nP=150		5993.7769	2.083E+14	1.489E+13	0.8285006	0.4103338	42.86194	167.64534	5993.7769	1.7578
nP=200		**5942.3012**	**1.875E+14**	**1.788E+13**	**0.8010055**	**0.3960754**	**41.412944**	**185.52637**	**5942.3012**	**1.9922**
beta=0.25	nP=200 alpha=0.9	**5921.7223**	**1.768E+14**	**2.041E+13**	**0.7857552**	**0.3888118**	**40.642407**	**196.13774**	**5921.7223**	**1.6717**
beta=0.5		5942.3012	1.875E+14	1.788E+13	0.8010055	0.3960754	41.412944	185.52637	5942.3012	1.9922
beta=0.75		6162.7706	1.553E+14	7.078E+12	0.8899198	0.4520143	46.031239	133.43027	6162.7706	1.6858
beta=1		13576.751	1.26E+15	1.108E+14	1.3146629	1.5795531	54.398058	68.030093	13576.751	1.7213
alpha=0.3	**nP=200 beta=0.25**	**5913.2528**	**1.83E+14**	**2.854E+13**	**0.7917452**	**0.3908324**	**41.000243**	**190.86712**	**5913.2528**	**1.7285**
alpha=0.6		5917.118	1.662E+14	1.949E+13	0.7820333	0.3907901	40.485982	198.07975	5917.118	1.6827
alpha=0.9		5921.7223	1.768E+14	2.041E+13	0.7857552	0.3888118	40.642407	196.13774	5921.7223	1.6717

7.2 ABC Algorithm

The nHB, A, and MR parameters of the ABC were used to limit the initial search space and randomly create first solutions. The effect of different variables on the performance of the ABC algorithm in solving the PVDP problem is shown in Table 2. An ideal output from the algorithm, for 2 parameters, we select and keep the rest as Maxmin.

Table 2 *The obtained results of the ABC algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nHB=50	A=300 mr=0.9	6382.1653	6382.1653	6382.1653	1.0019113	0.4932877	51.9125	83.938434	6381.6653	2.3127
nHB=100		6394.6809	6394.6809	6394.6809	1.0062224	0.4952908	52.135877	82.357392	6394.1809	1.6652
nHB=150		5984.635	5984.635	5984.635	0.7867224	0.3850714	40.533827	200	5984.135	1.0711
nHB=200		**5884.3774**	**5884.4649**	**5884.4228**	**0.7784179**	**0.3831581**	**40.332387**	**199.95133**	**5883.8774**	**0.8887**
A=200	nHB=200 mr=0.9	5931.8539	5931.8564	5931.855	0.8055288	0.3965039	41.737244	181.28459	5931.3539	1.6469
A=250		5887.1828	5889.0542	5887.9663	0.7793262	0.3835205	40.368318	199.45258	5886.6828	1.363
A=300		**5884.3774**	**5884.4649**	**5884.4228**	**0.7784179**	**0.3831581**	**40.332387**	**199.95133**	**5883.8774**	**0.8887**
mr=0.3	nHB=200 A=300	6731.8588	7.06E+03	6992.6486	1.0966938	0.5452675	56.51387	53.942724	6731.3588	1.5994
mr=0.6		6283.1522	6283.4817	6283.2693	0.9654695	0.4752379	50.024191	98.237503	6282.6522	1.6789
mr=0.9		**5884.3774**	**5884.4649**	**5884.4228**	**0.7784179**	**0.3831581**	**40.332387**	**199.95133**	**5883.8774**	**0.8887**

7.3 VPS Algorithm

The VPS way of working uses the nVP, P, and Alpha that limit the range of the search space at first, generating random first solutions. Table 3 shows the effects of various features of the VPS algorithm on its performance in PVDP-related problems. The best result for the algorithm is obtained by choosing two parameters and varying all other parameters.

Table 3 *The obtained results of the VPS algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nVP=50	p=0.05 alpha=0.9	**5887.2963**	**5887.5291**	**5887.3196**	**0.7799179**	**0.3838974**	**40.410253**	**198.87028**	**5886.3963**	**1.8791**
nVP=100		5889.3998	5889.3998	5889.3998	0.7811434	0.3845007	40.473626	197.99493	5888.4998	3.6827
nVP=150		5928.1424	5928.1424	5928.1424	0.8032543	0.3953843	41.619396	182.78539	5927.2424	4.4851
nVP=200		6050.8982	6050.8982	6050.8982	0.8662931	0.4264139	44.885653	145.01392	6049.9982	4.8007
p=0.025	nVP=50 alpha=0.9	5893.2531	5893.9516	5893.4208	0.7834084	0.3856155	40.591107	196.38189	5892.3531	2.7535
p=0.05		5887.2963	5887.5291	5887.3196	0.7799179	0.3838974	40.410253	198.87028	5886.3963	1.9791
p=0.075		**5884.691**	**5885.1567**	**5884.8074**	**0.7783814**	**0.3831411**	**40.330642**	**199.97524**	**5883.791**	**1.9358**
alpha=0.3	nVP=50 p=0.75	5892.8492	5.89E+03	5893.0169	0.7831727	0.3854995	40.578896	196.54896	5891.9492	2.2162
alpha=0.6		5885.857	5887.254	5886.0526	0.7790698	0.3834799	40.36631	199.47946	5884.957	1.9904
alpha=0.9		**5884.691**	**5885.1567**	**5884.8074**	**0.7783814**	**0.3831411**	**40.330642**	**199.97524**	**5883.791**	**1.9358**

7.4 WEOA Algorithm

WEOA employs nWM, Tetamax, Tetamin, Emax, and Emin values to limit the initial search space and produce first solutions through a random selection process. The effectiveness of different variables on the efficacy of the WEOA algorithm in addressing the PVDP problem is presented in Table 4. By choosing two parameters and varying the other parameters, we show the best outcome of the algorithm.

J. of Soft Computing and Data Mining Vol. 6 No. 3 (2025) p. 271-291

Table 4 *The obtained results of the WEOA algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nWM=50	Tx=-20; Tn=-50; Ex=-.5; En=-3.5;	**5884.6331**	**5884.6374**	**5884.6342**	**0.7783471**	**0.3831242**	**40.328864**	**200**	**5883.73**	**2.572**
nWM= 100		5884.2352	5897.0317	5884.8617	0.7783472	0.3831244	40.328867	200	5883.735	2.952
nWM= 150		5884.2359	6141.9955	5889.3098	0.7783475	0.3831249	40.328882	199.9997	5883.735	4.455
nWM= 200		5884.2377	6304.0241	5894.7561	0.7783479	0.3831251	40.328904	199.9994	5883.737	4.251
Tx=-30; Tn=-60;	nWM= 50; Ex=-.5; En=-3.5;	5884.2414	5887.884	5884.6785	0.7783487	0.3831264	40.328946	199.9989	5883.741	2.905
Tx=-20; Tn=-50;		5884.6331	5884.6374	5884.6342	0.7783471	0.3831242	40.328864	200	5883.733	2.572
Tx=-10; Tn=-40;		**5884.2329**	**5884.2331**	**5884.2329**	**0.7783471**	**0.3831242**	**40.328863**	**200**	**5883.73**	**2.425**
Ex=-0.25; En=-2;	Tx=-10; Tn=-40; nWM=50	5884.2329	5884.233	5884.2329	0.7783471	0.3831242	40.328863	200	5883.732	2.475
Ex=-0.5; En=-3.5;		5884.2329	5884.2331	5884.2329	0.7783471	0.3831242	40.328863	200	5883.732	2.425
Ex=-0.75; En=-4;		5884.2329	5884.2331	5884.2329	0.7783470	0.3831242	40.328863	200	5883.732	2.344

Tx = Tetamax; Tn = Tetamin; Ex = Emax; En = Emin

7.5 CSA Algorithm

The CSA method uses the nN, Pa, and Alpha parameters to limit the initial query areas and randomly create first solutions. Table 5 shows the effect of various variables on the efficacy of the CSA algorithm in solving the PVDP problem. From this optimal outcome of the algorithm, we can choose 2 parameters and fix the other parameters, and vary.

Table 5 *The obtained results of the CSA algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nN=50	pa=0.2 alpha=0.5	**5880.87**	**6195.00**	**5913.76**	**0.77816**	**0.38303**	**40.31963**	**200**	**5880.675**	**1.18**
nN=100		5881.27	7274.179	6082.723	0.778328	0.383124	40.32734	199.8936	5881.07	1.43
nN=150		5882.471	6744.356	6092.668	0.778203	0.383495	40.31972	200	5882.271	1.96
nN=200		5884.632	7121.926	6147.708	0.778347	0.383124	40.3288	200	5883.732	2.89
pa=0.2	nN=50 alpha=0.5	**5880.87**	**6195.00**	**5913.76**	**0.77816**	**0.38303**	**40.3196**	**200**	**5880.67**	**1.18**
pa=0.5		5891.109	7312.350	6378.525	0.778760	0.383742	40.33918	200	5890.209	1.20
pa=0.7		6108.931	7314.010	6661.307	0.892606	0.439355	46.23900	131.3964	6108.031	1.07
pa=1		5944.459	7.49E+03	7117.244	0.804244	0.394564	41.52929	183.9490	5943.559	1.10
alpha=0.3	nN=50 pa=0.2	6004.879	7.32E+03	6573.276	0.841631	0.414941	43.60311	159.0082	6003.979	1.17
alpha=0.5		**5880.87**	**6195.00**	**5913.76**	**0.77816**	**0.38303**	**40.3196**	**200**	**5880.67**	**1.18**
alpha=0.9		5904.884	7315.173	6611.042	0.785436	0.386685	40.6958	195.315	5903.984	1.11

7.6 CSA-PSO Algorithm

The CSA-PSO hybrid algorithm demonstrated better solution quality and computational efficiency compared to others. The CSA-PSO hybrid algorithm has been proven to be superior in terms of solution quality and computational efficiency. The highest outcome was achieved at nNest = 100, pa = 0.1, and w = 0.8, with the optimal objective value F(x) = 5878.7877, which is a good compromise between exploration and exploitation. The robustness of the solutions can also be observed, as the range between the smallest and largest objective values is not too wide. The hybrid strategy also exhibits considerably lower variance than pure CSA or PSO-based methods and yields better results than other standalone metaheuristics, such as BB-BC and ABC, in the majority of cases. These outcomes highlight the strength of the algorithm and its suitability for complex, constrained engineering tasks, such as the PVDP. Table 6 shows the obtained Results of the CSA-PSO algorithm.

Table 6 *The obtained results of the CSA-PSO algorithm*

Parameter	Fixed par.	Min	Max	Mean	X1	X2	X3	X4	F(x)	Time
nNest=25	Pa=0.25 w=0.8	5883.1287	5879.1642	5881.5532	0.77913	0.38321	40.3143	198.9889	5882.6419	1.5384
nNest=50		5881.9143	5884.8844	5882.6732	0.77854	0.38365	40.2884	200.0000	5881.0268	1.6837
nNest=100		5878.8757	5883.6744	5882.2437	0.77832	0.38322	40.2803	199.8943	5879.9925	1.8365
pa=0.1	nNest=100 w=0.8	5889.6872	5881.1112	5879.8519	0.77821	0.38321	40.2701	199.8992	5878.7877	1.6164
Pa=0.25		5880.8913	5879.6777	5881.2842	0.77882	0.38364	40.2812	199.9912	5879.9938	1.8354
pa=0.5		5886.1351	5884.3327	5886.5503	0.77889	0.38501	40.4634	198.4752	5885.1376	2.0433
w=0.4	nNest=100 pa=0.25	5890.2200	5883.9921	5883.5881	0.77889	0.38309	40.3101	199.8925	5881.2244	1.7679
w=0.6		5882.4164	5881.9555	5882.6623	0.77821	0.38316	40.2900	199.9809	5880.3368	1.8498
w=0.8		5878.9732	5882.6712	5881.2321	0.77921	0.38412	40.2801	199.9922	5879.9958	1.8382

The best results of all adapted algorithms and others from the literature are compared in Table 7. From Table 7, it can be seen that CSA was the most suitable algorithm among the five tested algorithms.

Table 7 *Comparison among results obtained in this study and some previous studies*

Study	Algorithm	F(x)	Error
Belkourchia et al. [16]	GA-PSO-SQP	5798.7989	0.000096
This work	CSA	5880.6759	81.876904
This work	WEOA	5883.7329	84.933904
This work	VPS	5883.791	84.992004
This work	ABC	5883.8774	85.078404
Cagnina et al. [17]	PSO	5885.4032	86.604204
Coello and Montes [44]	GA	6288.7445	489.9455
Lee and Geem [45]	HS	7198.433	1399.634
	Optimal solution		5798.798996

8. Discussion

The findings of this research contribute to a better understanding and comparison of different metaheuristic methods for solving the PVDP. This highly nonlinear, mixed integer constrained optimization problem has practical significance in structural and mechanical engineering. Each of the algorithms being considered in this analysis, BB-BC, ABC, VPS, WEOA, CSA, and the proposed hybrid CSA-PSO, has points of strength regarding convergence behavior, solution quality, robustness, and computational efficiency.

Among the single metaheuristics, the CSA demonstrated the most consistent performance, providing the best trade-off between solution quality and runtime. The solution obtained was near-optimal with the objective value of 5880.6759 (only slightly worse than the known global optimum (5798.798996), but outperformed the classical methods PSO and GA as well as previously reported hybrid algorithms such as GA-PSO-SQP (et al., 2019). Due to the adaptive step size and elite selection mechanism of the CSA, CSA also appears to be a robust and stable optimization method in tackling the complex design space of the PVDP.

Likewise, the WEOA and VPS were slightly weaker (5883.7329 and 5883.791, respectively, as fitness values). On the other hand, both methods were more variable in their results and slightly longer in computation time compared with CSA, suggesting possible convergence instability for certain parameter settings. The WEOA and VPS are promising for exploration but may require further refinement or combination with other approaches to be more effective in exploitation.

In contrast, the ABC algorithm, despite giving relatively good solutions, encountered premature convergence in specific cases, especially when parameter mr (modification rate) was configured to lower values. ABC was limited by an inability to adapt to the multimodal landscape of the PVDP, which caused it to plateau more often than the other methods.

The Big Bang–Big Crunch (BB–BC) algorithm performed well when the population size and beta parameters were tuned. However, it converged generally more slowly and yielded more scattered results, suggesting higher sensitivity regarding parameter initialization and a lower overall robustness. This aligns with previous criticisms of BB–BC as being extremely stochastic and unlikely to be useful for fine-grained optimization problems.

The hybrid CSA-PSO algorithm outperformed all standalone methods. Achieving a value of 5878.7877 for the objective function for the best solution, it overcame not only CSA and PSO alone, but outperformed other hybrid models found in the literature (except for GA-PSO-SQP). The hybrid model was found to achieve a better exploration-exploitation balance by utilizing a highly aggressive search mechanism of CSA, combined with swarm-based velocity updates from PSO. Besides, the variance of results obtained by CSA-PSO was smaller, and the convergence speed was higher among different parameter values as specified in the paper. The closeness of the minimum and maximum objective values also proves the reliability of the CSA-PSO algorithm.

This comparison, presented in Table 7, further strengthens the importance of hybridization techniques of modern metaheuristics. Even though the GA-PSO-SQP (Genetic Algorithm-Particle Swarm Optimization-Sequential Quadratic Programming) method by Belkourchia et al. [16] is still the best-known solution, the simplicity, adaptability, and efficiency of CSA−PSO provide a very good alternative. In addition, the gap between CSA-PSO and traditional approaches, such as GA (Coello & Montes) or SA-based approaches (Lee & Geem), reflects advances in algorithm design for solving engineering optimization problems.

Results discussion: In general, the results show that even standalone metaheuristics can achieve competitive results. Hybrid approaches, such as CSA-PSO, yield better results in terms of accuracy, robustness, and computational efficiency. These findings further strengthen the emerging conclusion within the optimization community that customized hybridization presents an effective approach to large-scale, non-linear problems, such as the PVDP, common to many engineering applications.

9. Conclusion

This study provides a detailed examination of several optimization techniques applied to the Pressure Vessel Design Problem. The primary objective of this research was to evaluate the effectiveness of various algorithms in achieving optimal or near-optimal solutions. The findings demonstrate the efficacy of the algorithm in terms of both the quality of solutions generated and the computing efficiency achieved. Notably, the optimal solution was achieved with a solution value of 5,798.798996. In line with previous research, Belkourchia et al. [16] observed that the GA-PSO-SQP method yielded a solution that closely approximated the ideal output, hence substantiating its efficacy. The research examined many algorithms, including the Cuckoo Search Algorithm (CSA), Water Evaporation Optimization Algorithm (WEOA), Vibrating Particles System (VPS), and Artificial Bee Colony (ABC). These algorithms demonstrated competitive performance; however, they exhibited some degree of inaccuracy when compared to the optimal solution. The research also includes comparative analyses with other algorithms, such as Particle Swarm Optimization (PSO), as proposed by Garg, and Genetic Algorithm (GA), as introduced by Coello, among other methodologies presented by various authors. In general, the results of this study enhance our comprehension of the algorithmic aspects involved in addressing the Pressure Vessel Design Problem. These findings provide valuable insights that may inform future research and be applied practically in the fields of engineering and optimization.

Acknowledgement

The authors thank Alnoor University for supporting this project through the Research Supporting Fund (ANUI2025M111).

Conflict of Interest

The authors declare that they have no conflict of interest regarding the publication of this paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Ammar Adel Ahmed, Amera Istiqlal Badran, Ibrahim M. Ahmed;* ***data collection:*** *Yahya T. Qassim, Dilovan Asaad Zebari, Reving Masoud Abdulhakeem;* ***analysis and interpretation of results:*** *Yahya T. Qassim, Dilovan Asaad Zebari, Reving Masoud Abdulhakeem;* ***draft manuscript preparation:*** *Dilovan Asaad Zebari, Ammar Adel Ahmed, Amera Istiqlal Badran, Ibrahim M. Ahmed. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] Ahmad, H. B., Odeesho, D. W., Abdulhakeem, R. M., Salih, M. S., & Zebari, D. (2025). Teaching-Learning-Based Optimization Algorithm for Pressure Vessel Design Problem. *The Indonesian Journal of Computer Science*, *14*(5).

[2] Shaban, A. A., Fuente, J. A. D., Salih, M. S., & Ali, R. I. (2023). Review of swarm intelligence for solving symmetric traveling salesman problem. *Qubahan Academic Journal*, *3*(2), 10-27.

[3] Sadeeq, H. T., Abdulazeez, A. M., Kako, N. A., Zebari, D. A., & Zeebaree, D. Q. (2021, February). A New Hybrid Method for Global Optimization Based on the Bird Mating Optimizer and the Differential Evolution. In *2021 7th International Engineering Conference "Research & Innovation amid Global Pandemic"(IEC)* (pp. 54-60). IEEE.

[4] Aighuraibawi, A. H. B., Manickam, S., Abdullah, R., Alyasseri, Z. A. A., Al-Ani, A. K. I., Zebari, D. A., ... & Arif, Z. H. (2023). Feature Selection for Detecting ICMPv6-Based DDoS Attacks Using Binary Flower Pollination Algorithm. Comput. Syst. Sci. Eng., 47(1), 553-574.

[5] Lemonge, A. C. C., & Barbosa, H. J. C. (2004). An adaptive penalty scheme for genetic algorithms in structural optimization. *International Journal for Numerical Methods in Engineering*, *59*(5), 703–736.

[6] Yang, X.-S., & Suash Deb. (2009). Cuckoo Search via Levy flights. *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, 210–214.

[7] Abu-Mouti, F. S., & El-Hawary, M. E. (2012). Overview of Artificial Bee Colony (ABC) algorithm and its applications. *2012 IEEE International Systems Conference SysCon 2012*, 1–6.

[8] Acan, A., Altincay, H., Tekol, Y., & Unveren, A. (n.d.). A genetic algorithm with multiple crossover operators for optimal frequency assignment problem. *The 2003 Congress on Evolutionary Computation, 2003. CEC '03.*, 256–263.

[9] Adham, M. T., & Bentley, P. J. (2014). An artificial ecosystem algorithm applied to the travelling salesman problem. *GECCO 2014 - Companion Publication of the 2014 Genetic and Evolutionary Computation Conference*, 155–156.

[10] Zainal, N., Sithambranathan, M., Khattak, U. F., Zain, A. M., Mostafa, S. A., & Deris, A. M. (2024). Optimization of Electrical Discharge Machining Process by Metaheuristic Algorithms. *Qubahan Academic Journal*, *4*(1), 277-289.

[11] Almufti, S. (2017). Using Swarm Intelligence for solving NPHard Problems. *Academic Journal of Nawroz University*, *6*(3), 46–50.

[12] Almufti, S. (2022). Vibrating Particles System Algorithm: Overview, Modifications and Applications. *ICONTECH INTERNATIONAL JOURNAL*, *6*(3), 1–11.

[13] Almufti, S. M. (2022a). Artificial Bee Colony Algorithm performances in solving Welded Beam Design problem. *Computer Integrated Manufacturing Systems*, *28*(12).

[14] Almufti, S. M. (2022b). Vibrating Particles System Algorithm performance in solving Constrained Optimization Problem. *Academic Journal of Nawroz University*, *11*(3), 231–242.

[15] Almufti, S. M., Alkurdi, A. A. H., & Khoursheed, E. A. (n.d.). *Artificial Bee Colony Algorithm Performances in Solving Constraint-Based Optimization Problem*. *21*, 2022.

[16] Belkourchia, Y., Azrar, L., & Zeriab, E.-S. M. (2019). A Hybrid Optimization Algorithm for Solving Constrained Engineering Design Problems. *2019 5th International Conference on Optimization and Applications (ICOA)*, 1–7.

[17] Cagnina, L., Esquivel, S. C., Coello, C. A. C., Cagnina, L. C., & Esquivel, S. C. (2008). Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer. In *Informatica* (Vol. 32).

[18] Dehghani, M., Trojovská, E., & Trojovský, P. (2022). A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process. *Scientific Reports*, *12*(1).

[19] Liqun Gao, Dexuan Zou, Yanfeng Ge, & Wenjing Jin. (2010). Solving pressure vessel design problems by an effective global harmony search algorithm. *2010 Chinese Control and Decision Conference*, 4031–4035.

[20] Salih, S. Q., Alsewari, A. A., & Yaseen, Z. M. (2019). Pressure Vessel Design Simulation. *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, 120–124.

[21] Yahya Zebari, A., M. Almufti, S., & Mohammed Abdulrahman, C. (2020). Bat algorithm (BA): review, applications and modifications. *International Journal of Scientific World*, *8*(1), 1.

[22] Sharma, A., Sharma, A., Chowdary, V., Srivastava, A., & Joshi, P. (2021). *Cuckoo Search Algorithm: A Review of Recent Variants and Engineering Applications* (pp. 177–194).

[23] Zebari, D. A., Zeebaree, D. Q., Saeed, J. N., Zebari, N. A., & Adel, A. Z. (2020). Image steganography based on swarm intelligence algorithms: A survey. *people*, *7*(8), 9.

[24] Kaveh, A., & Mahdavi, V. R. (2016). Optimal design of truss structures using a new optimization algorithm based on global sensitivity analysis. *Structural Engineering and Mechanics*, *60*(6), 1093–1117.

[25] Junyue, C., Zeebaree, D. Q., Qingfeng, C., & Zebari, D. A. (2023). Breast cancer diagnosis using hybrid AlexNet-ELM and chimp optimization algorithm evolved by Nelder-mead simplex approach. *Biomedical Signal Processing and Control*, *85*, 105053.

[26] Kaveh, A. (2017). Water Evaporation Optimization Algorithm. In *Advances in Metaheuristic Algorithms for Optimal Design of Structures* (pp. 489–509). Springer International Publishing.

[27] Kaveh, A., & Bakhshpoori, T. (2016b). Water Evaporation Optimization: A novel physically inspired optimization algorithm. *Computers & Structures*, *167*, 69–85.

[28] Saha, A., Das, P., & Chakraborty, A. K. (2017). Water evaporation algorithm: A new metaheuristic algorithm towards the solution of optimal power flow. *Engineering Science and Technology, an International Journal*, *20*(6), 1540–1552.

[29] Kaveh, A., & Bakhshpoori, T. (2016a). A new metaheuristic for continuous structural optimization: water evaporation optimization. *Structural and Multidisciplinary Optimization*, *54*(1), 23–43.

[30] O. K. Erol, O. Hasançebi, & S. Kazemzadeh Azad. (2011). Evaluating efficiency of big-bang big-crunch algorithm in benchmark engineering optimization problems. *INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING*, *3*, 495–505.

[31] Rathore, V. S., & Khandelwal, N. P. (2020). Portfolio optimization using Big Bang-Big Crunch algorithm. *2020 International Conference on Emerging Trends in Information Technology and Engineering (Ic-ETITE).*

[32] Tang, H., Zhou, J., Xue, S., & Xie, L. (2010). Big Bang-Big Crunch optimization for parameter estimation in structural systems. *Mechanical Systems and Signal Processing*, *24*(8), 2888–2897.

[33] Ghasemi, A., & Mirzavand, M. (2014). Robot path planning using Big Bang–Big Crunch algorithm. *Robotics and Autonomous Systems*, *62*(3), 390–399.

[34] Ghasemi, A. , & Alimohammadi, A. R. (2019). High-level synthesis and optimization of VLSI circuits using Big Bang–Big Crunch algorithm. *Integration, the VLSI Journal, 65*, 234–244.

[35] Wedyan, M., Elshaweesh, O., Ramadan, E., & Alturki, R. (2022). Vibrating Particles System Algorithm for Solving Classification Problems. *Computer Systems Science and Engineering*, *43*(3), 1189–1206.

[36] Gnetchejo, P. J., Essiane, S. N., Ele, P., Wamkeue, R., Wapet, D. M., & Ngoffe, S. P. (2019). Enhanced Vibrating Particles System Algorithm for Parameters Estimation of Photovoltaic System. *Journal of Power and Energy Engineering*, *07*(08), 1–26.

[37] Kaveh, A., Hoseini Vaez, S. R., & Hosseini, P. (2018). MATLAB CODE FOR AN ENHANCED VIBRATING PARTICLES SYSTEM ALGORITHM. In *INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING Int. J. Optim. Civil Eng* (Vol. 8, Issue 3).

[38] Xin-She Yang, & Suash Deb. (2010). Engineering optimisation by cuckoo search. *Int. J. Mathematical Modelling and Numerical Optimisation*, *1*(4), 330–343.

[39] Nguyen, T. T., Vo, D. N., & Dinh, B. H. (2016). Cuckoo search algorithm for combined heat and power economic dispatch. *International Journal of Electrical Power & Energy Systems*, *81*, 204–214.

[40] Fister, I., Yang, X.-S., Fister, I., Brest, J., & Fister, D. (2013). *A Brief Review of Nature-Inspired Algorithms for Optimization*.

[41] Soneji, H., & Sanghvi, R. C. (2012). Towards the improvement of Cuckoo search algorithm. *2012 World Congress on Information and Communication Technologies*, 878–883.

[42] Shojaei, B., Naserabadi, H. D., & Amiri, M. J. T. (2024). Optimizing Competency-Based Human Resource Allocation in Construction Project Scheduling: A Multi-Objective Meta-Heuristic Approach. *Qubahan Academic Journal*, *4*(3), 861-881.

[43] Yab, L. Y., Wahid, N., & Hamid, R. A. (2024). Improved Ozone Level Detection through Feature Selection with Modified Whale Optimization Algorithm. *Qubahan Academic Journal*, *4*(1), 265-276.

[44] Coello, C. A. C., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Advanced Engineering Informatics, 16(3), 193-203.

[45] Lee, K. S., & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer methods in applied mechanics and engineering*, *194*(36-38), 3902-3933.