IJE TRANSACTIONS A: Basics Vol. 39, No. 10, (October 2026) 2529-2545

International Journal of Engineering

Journal Homepage: www.ije.ir

Bermuda Weed Optimization: A Scalable Meta-heuristic for Cloud-Based Cruise Control Systems

M. Allahi Rudposhti, A. Bohlooli*, K. Jamshidi

Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran

P A P E R I N F O

Paper history:
Received 23 August 2025
Received in revised form 04 January 2026
Accepted January 7 2026

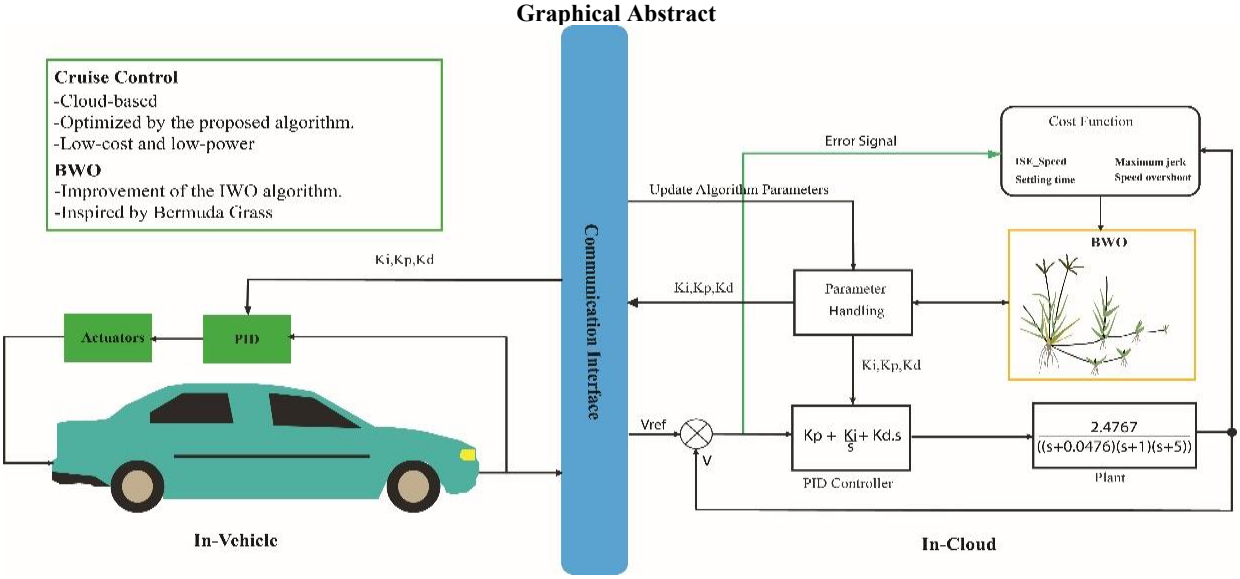
Keywords:
Stochastic Optimization
Cruise Control
Bermuda Weed Optimization
Meta-heuristic Algorithms

A B S T R A C T

This study proposes the Bermuda Weed Optimization (BWO) algorithm; a novel and scalable metaheuristic algorithm inspired by the invasive growth of Bermuda grass. This algorithm has been developed as an enhanced version of the Invasive Weed Optimization (IWO) algorithm and, by imitating the plant's robust propagation strategies, achieves a better balance between global exploration and local exploitation. The algorithm's performance was rigorously evaluated against four previous IWO-based versions, and its superior scalability was demonstrated through the lowest average error and stable performance across diverse scenarios. Furthermore, BWO was compared with the new Gray Squirrel Search Algorithm (GSFA)—which falls outside the IWO category—to assess its performance against a novel method unrelated to the IWO family; this comparison highlighted BWO's competitive superiority and achieved an average 64.43% improvement in best-cost results. The strong convergence and scalability of BWO make it highly suitable for real-time applications—particularly in automotive systems. In a practical implementation using a cloud-based cruise control (CC) framework, BWO significantly outperformed the RPO-based method (the latest approach) by reducing overshoot by 45.92%, settling time by 29.38%, ISE speed by 8.92%, and maximum jerk by 20.09%. By achieving near-optimal convergence and leveraging cloud deployment with high scalability, BWO can effectively adapt to diverse automotive system requirements and achieve high efficiency across multiple operating modes.

***doi**: 10.5829/ije.2026.39.10a.15*

Graphical Abstract



*Corresponding Author Email: bohlooli@eng.ui.ac.ir (A. Bohlooli)

Please cite this article as: Allahi Rudposhti M, Bohlooli A, Jamshidi K. Bermuda Weed Optimization: A Scalable Meta-heuristic for Cloud-Based Cruise Control Systems. International Journal of Engineering, Transactions A: Basics. 2026;39(10):2529-45.

1. INTRODUCTION

In contemporary research, meta-heuristic algorithms are increasingly inspired by biological processes and natural phenomena. These algorithms are widely employed to address complex, challenging, and occasionally intractable problems. They are characterized by their ease of understanding and implementation, adaptability to external factors influencing the problem, effective exploration of the solution space, capability to identify optimal solutions, independence from gradient information, and avoidance of local optima. Generally, meta-heuristic algorithms can be categorized into five groups (Figure 1): Evolution-Based, Physics-Based, Swarm-Based, Human-Based, and Hybrid approaches (1). Among the earliest methods for developing meta-heuristic algorithms are Evolution-Based techniques, which draw inspiration from the principles of biological evolution and natural selection. These algorithms typically incorporate mechanisms such as selection, mutation, inheritance, and natural selection. The Genetic Algorithm (GA) (2) is the most established and widely recognized Evolution-Based algorithm, operating on the principles of natural selection. Another category of meta-heuristic algorithms is the physics-based group, which utilizes physical laws and principles to solve a variety of problems and simulations. Examples of physics-based approaches include Projectiles Optimization (PRO) (3), Henry Gas Solubility Optimization (HGSO) (4), and the Newton-Raphson-Based Optimizer (NRBO) (5). Swarm-based algorithms are another category of metaheuristic algorithm, encompassing the largest group of such algorithms. They are inspired by the collective behavior of living organisms, such as birds, fish, ants, and bees, to solve complex problems. These algorithms are commonly used for optimization and searching in large, complex spaces. Examples of swarm-based algorithms include Invasive Weed Optimization (IWO) (6), Grey Wolf Optimizer (GWO) (7), Grasshopper Optimization Algorithm (GOA) (8), Crystal Structure Algorithm (CryStAl) (9), Dandelion Optimizer (DO) (10), Walrus Optimizer (WO) (11), Starfish optimization algorithm (SFOA) (12), and Gray Squirrel Foraging Algorithm (GSFA) (13). Human-based algorithms, such as Rock-Climbing Group (RCG) (14), and Gold Seekers Algorithm (GSO) (15). Meta-heuristic algorithms can be combined to improve performance and efficiency in solving complex problems, resulting in hybrid algorithms. For instance, Pérez et al. (16) introduced two meta-heuristics, including the Greedy Randomized Adaptive Search Procedure (GRASP) and a GA (2), are combined and utilized to address a real supply chain scheduling problem.

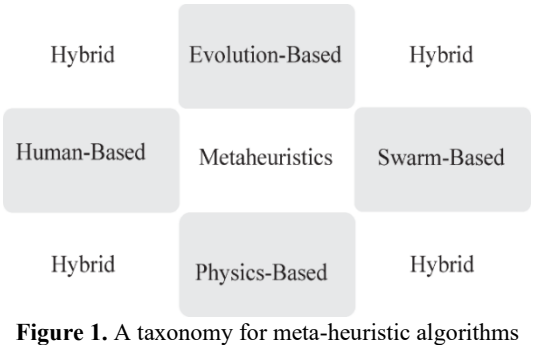


Figure 1. A taxonomy for meta-heuristic algorithms

Among the various meta-heuristic algorithms, the IWO algorithm (6) stands out as one of the most prominent due to its remarkable local and global search capabilities, ease of implementation, robustness in addressing complex real-world problems, and proven efficiency across diverse applications. Inspired by the colony behavior of invasive weeds and their natural mechanisms of growth and competition for resources, IWO has been widely adopted in numerous engineering fields (17-29).

Despite the widespread application of the IWO algorithm and its developed variants in solving various engineering problems, it faces significant limitations when dealing with objective functions that contain numerous local minima (such as the Griewank function). One of the primary challenges of IWO is its inherent tendency to get trapped in local minima that have relatively better quality than their neighboring regions. This issue becomes particularly prominent in the later iterations, where the radius of seed spread decreases, and the seed replication process becomes primarily concentrated around the parent plant. Under such conditions, if the global minimum is located far from the parent's position, the algorithm may mistakenly identify a local minimum as the global minimum. Moreover, the "best solution selection" policy at each step, which is one of the primary exploitation strategies of the algorithm, leads to the exclusion of regions with relatively lower quality but higher potential for discovering the optimal solution. This imbalance between exploration and exploitation processes is a fundamental challenge for IWO, which can limit its efficiency in optimization problems with complex and multi-modal search spaces. Another important limitation of IWO relates to the population control process. Despite the algorithm's high ability to generate diversity in the initial population, the constraints of storage resources and the population reduction policy based on the "best solution at the current time" can result in the elimination of individuals that might have significant potential to reach the optimal solution in subsequent generations. Even the improved versions of IWO based on chaos theory (19) face significantly more challenges compared to the basic algorithm. It should be noted that chaos theory studies deterministic systems with unpredictable behavior and high sensitivity to initial conditions. In optimization, it

helps algorithms escape local optima. The performance of these algorithms is highly dependent on the tuning of IWO parameters and the selected chaos map, to the extent that even a slight change in these parameters can lead to unpredictable results. Furthermore, while in most algorithms, increasing the number of iterations enhances the final solution's quality, chaos-based IWO lacks a clear rule for performance improvement with more iterations. In many cases, increasing iterations even reduces its efficiency. Moreover, the IWO algorithm based on Lévy distribution (27) performs well in limited search spaces. The Lévy distribution is a probability distribution with heavy tails and infinite variance, which enables long-range jumps in search spaces and is particularly effective in metaheuristics for global exploration. However, as the search space expands, its performance significantly declines. For instance, when the domain of the Sheffer benchmark function is extended from the range [-10, 10] to [-100, 100], the algorithm experiences a considerable drop in efficiency and loses its scalability.

The proposed algorithm, which represents an enhanced version of the IWO algorithm, draws inspiration from the intelligent behavior of a specific weed species called Bermuda weed. It successfully establishes an optimal balance between the two key components of local search (exploitation) and global search (exploration). This characteristic leads to efficient and effective convergence to the global optimum solution.

The contributions of this work in the metaheuristic field are summarized as follows:

1. An enhanced variant of the IWO algorithm, named Bermuda Weed Optimization (BWO), is proposed. The primary feature of BWO is scalability. This algorithm was deployed in a cloud-based Cruise Control (CC) system for electric and resource-constrained vehicles. The scalability of BWO allows the system to dynamically balance accuracy and speed based on driving conditions: In accuracy-sensitive scenarios (e.g., traffic), it increases the algorithm's iteration count to achieve higher solution quality .In time-critical situations (e.g., highway driving), it rapidly delivers near-optimal solutions within tightly constrained iteration limits.
2. The proposed BWO-leveraged system demonstrated superior performance in key driving and comfort metrics compared to previous methods. Simulation results demonstrated that BWO significantly outperformed the RPO-based method (the latest approach) by reducing overshoot by 45.92%, settling time by 29.38%, ISE speed by 8.92%, and maximum jerk by 20.09%. With its cloud-based architecture, the system is compatible with resource-limited vehicles while simultaneously enhancing accuracy and delivering superior outcomes in vehicular comfort measures.

The structure of this paper is organized as follows: The basic IWO is introduced in Section 2. The theory of BWO is illustrated in Section 3. Section 4 presents the experiments of benchmark functions and real-world optimization problems, respectively. Finally, the conclusions and future works are summarized in Section 5.

2. THE BASIC IWO

Observing the propagation of invasive weeds in nature reveals their remarkable efficiency in identifying and thriving in the most fertile areas. Despite various attempts to control these plants, most conventional methods have proven ineffective in preventing their spread. Inspired by this phenomenon, researchers have developed a novel approach to solving optimization problems, leading to the introduction of the IWO algorithm. IWO is a swarm-based metaheuristic that mimics the natural behavior of weeds in locating optimal sites for growth and reproduction. This algorithm leverages key characteristics of invasive weeds, including their ability to spread aggressively, high reproduction rate, wide dispersal, and competitive exclusion. The IWO process consists of four main stages: initialization, reproduction, spatial dispersal, and competitive exclusion, which are explained in detail in the following:

Initialization: An initial population is randomly distributed within the search space. Each member of this population, representing a weed, has N-dimensional coordinates and a cost value associated with its position in the search space .

Reproduction: Each agent (weed) in the population has a specific position in the search space, and its fitness is evaluated based on the objective function (or cost) value at that position. The higher the fitness of a weed, the more favorable its position is considered in the search space. The number of offspring (seeds) produced by each weed depends on its fitness and is calculated using Equation 1.

$$S_i = \left\lfloor \frac{f_{max}-f_i}{f_{max}-f_{min}} \times (S_{max} - S_{min}) + S_{min} \right\rfloor \quad (1)$$

where f_i represents the fitness value of the i-th weed, while f_{min} and f_{max} correspond to the lowest and highest fitness values within the current population, respectively. Similarly, S_{min} and S_{max} define the minimum and maximum number of seeds that a weed is permitted to generate. Equation 1 clearly indicates that weeds with higher fitness values (f_i) will produce a greater number of seeds, facilitating a more extensive search in promising regions of the solution space. These seeds are distributed around the parent's position, with a dispersion pattern designed to enhance diversity in the search space. This mechanism allows for better exploration of the problem space, increasing the chances of discovering optimal solutions and ultimately improving the algorithm's

performance. Since the number of seeds is proportional to the parent's fitness, the offspring tend to settle in regions with higher fitness potential. Over successive generations, this process gradually enhances the overall population quality and accelerates the algorithm's convergence toward optimal solutions.

Spatial dispersal: The seeds produced by each weed are distributed randomly across the D-dimensional search space according to a normal distribution with a mean of zero and a standard deviation σ that changes adaptively. This method ensures that the seeds are clustered near the parent weed's position. As the algorithm advances through its iterations, the value of σ gradually reduces from its maximum σ_{max} to its minimum σ_{min}, allowing the algorithm to shift from a broad exploration phase to a more focused exploitation phase, where the search is refined in the more promising areas of the space. The standard deviation σ_k for the k-th iteration is given by Equation 2.

$$\sigma_k = \left(\frac{iter_{max}-k}{iter_{max}}\right)^n \times (\sigma_{max} - \sigma_{min}) + \sigma_{min} \quad (2)$$

where:

- **n** is a nonlinear modulation factor.
- $\mathbf{iter_{max}}$ is the total number of iterations.
- **k** denotes the current iteration number.

Using this value of σ_k, the updated position of a seed is computed using Equation 3.

$$x'_i = x_i + N(0, \sigma_k^2) \quad (3)$$

Where:

- $\mathbf{x_i}$ is the i-th weed in the current population,
- $\boldsymbol{N(0, \sigma_k^2)}$ is a random number drawn from a normal distribution with a mean of zero and a standard deviation of σ_k.

This adaptive strategy enhances the algorithm's efficiency by initially promoting wide exploration across the search space and gradually focusing on the most promising regions to refine the solution, thus accelerating convergence towards optimal solutions.

Competitive exclusion: After the seeds are generated, the parent plants transform into weeds, and the total population of weeds consists of both the parent plants and their offspring. This new population is then sorted based on fitness, and if the number of weeds exceeds the maximum allowed colony size (P_{max}), weeds with lower fitness are eliminated to maintain the population within the allowed limit.

Repeat Until Stop: Repeat the above steps until the termination condition is met.

3. BERMUDA WEED OPTIMIZATION (BWO)

This section introduces the mimicry mechanism, the mathematical model and the procedure execution of BWO algorithm.

3. 1. Mimicking Bermuda Grass (Cynodon dactylon), a significant species of the Poaceae (grass) family, is a perennial, turf-forming, and resilient plant that plays a key role in agriculture, soil cover, and erosion control. This species, with its rapid growth and remarkable adaptability to diverse climatic conditions, particularly thrives in tropical and subtropical regions. It is recognized as one of the most important forage sources in arid and semi-arid areas. In addition to its forage utility, it is widely used in urban green spaces and sports fields due to its resistance to environmental stresses (e.g., drought and livestock grazing). However, the high invasive potential of this plant in certain ecosystems has turned it into a challenging weed. Its rapid growth, extensive reproduction through seeds and stolons, and resilience to adverse conditions have led to its swift spread in agricultural lands, orchards, and natural ecosystems, making control efforts difficult [30].

Sexual Reproduction (via seeds): Seeds produced by the plant are dispersed in proximity to the parent plant. This dispersal pattern mirrors the local search strategy in optimization algorithms, which aims to refine solutions within the vicinity of the current solution.

Vegetative Reproduction (via stolons): Stolons (horizontal stems) have nodes that, upon contact with soil, develop adventitious roots and give rise to new shoots, which grow into independent plants. This mechanism enables the plant to spread farther from the parent plant and corresponds to the global search strategy in optimization algorithms, which explores new regions within the search space.

Growth and Dispersal Pattern: Figure 2 illustrates a sample of Bermuda grass. The parent plant covers extensive areas through horizontal stolons. Each stolon can produce multiple nodes that develop into new plants after rooting. Additionally, vertical stems (culms) bear seeds which, upon dispersal, generate new plants near the parent. Figure 3 presents a conceptual model of this process, which can serve as a basis for computationally simulating the plant's growth. Bermuda grass tends to grow and reproduce more extensively in fertile soils with adequate nitrogen, phosphorus, and potassium. Under such conditions, the production of seeds, stolons, and nodes increases. This behavior is analogous to the optimal solution strategy in intelligent algorithms, where the system seeks peak efficiency under ideal conditions. However, the plant is also capable of surviving in poor soils, albeit with more limited growth.

3. 2. Mathematical Model The proposed algorithm follows the IWO algorithm in seed-based reproduction, employing the same mathematical model for this process. However, its key distinguishing feature is a node-based reproduction strategy. In this strategy, each plant produces one or more horizontal branches, with the number of stolons determined by the soil quality—

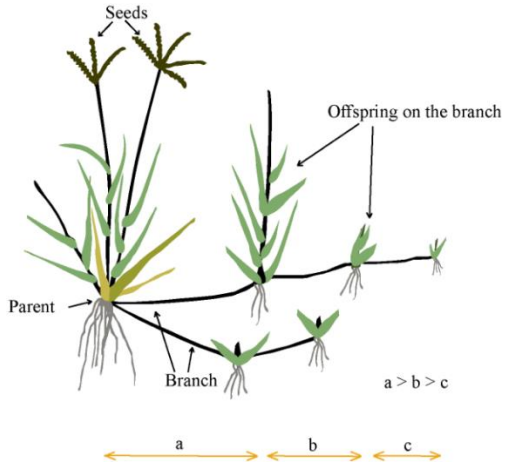


Figure 2. Morphological features of Bermuda Grass

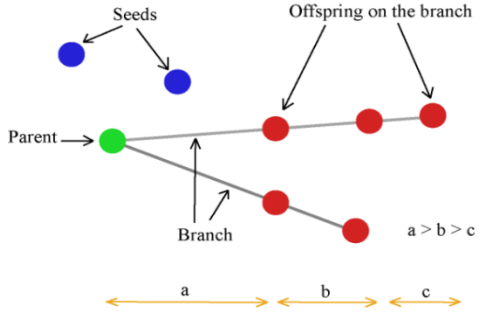


Figure 3. Conceptual model of Bermuda Grass morphology

evaluated based on the plant's fitness (i.e., its objective function value). Along each stolon, several offspring (nodes) are then generated at specific intervals from the parent plant, where they develop independently. A conceptual model of a Bermuda grass plant is illustrated in Figure 3. Based on this model, reproduction via seeds follows the mathematical framework of the IWO algorithm: the number of seeds is calculated using Equation 1, which depends on the parent plant's fitness relative to the best and worst fitness values in the population. The offspring positions are generated using Equation 3, where the standard deviation (σ) is dynamically updated via Equation 2 to balance exploration and exploitation. In contrast, reproduction through stolon-based node propagation follows a distinct mechanism, as described below:

Number of Branches: The number of horizontal branches produced by each weed, determined based on the soil quality, is given by Equation 4.

$$B_i = \left\lfloor \frac{f_{max}-f_i}{f_{max}-f_{min}} \times (b_{max} - b_{min}) + b_{min} \right\rfloor \quad (4)$$

where B_i is the number of branches of the i-th parent, and b_{min} and b_{max} define the minimum and maximum number of branch that a weed is permitted to generate.

Number of Offspring per Branch: The number of nodes on each branch is calculated using Equation 5.

$$R_{i,j} = \left\lfloor \frac{f_{max}-f_i}{f_{max}-f_{min}} \times (r_{max} - r_{min}) + r_{min} \right\rfloor \quad (5)$$

Where $R_{i,j}$ denotes the number of nodes on the j-th branch of the i-th parent, and r_{min} and r_{max} define the permissible range for the number of nodes per branch.

Node Spacing: Nodes are distributed along each branch at distances from the parent plant defined by Equation 6.

$$D_{i,j} = \sum_{j=1}^{k} \frac{d_i}{2^{(j-1)}} \quad (6)$$

where $D_{i,j}$ is the distance of the j-th node from the i-th parent, and d_i is the base distance for the i-th parent, adaptively calculated via Equation 7.

$$d_i = \left\lfloor \frac{f_{max}-f_i}{f_{max}-f_{min}} \times (d_{max} - d_{min}) + d_{min} \right\rfloor \quad (7)$$

Where d_{min} and d_{max} are the minimum and maximum values of d_i, respectively.

Node Positioning: The spatial coordinates of the nodes are determined by Equation 8. A random direction vector is first generated to define the branch orientation. Nodes are then positioned along this direction.

$$x'_{i,j} = x_i + D_{i,j} \cdot \frac{u}{\|u\|} \sqrt{\xi} \quad (8)$$

where u is a randomly generated vector with a normal distribution, i.e., u=(N(0,1), N(0,1),...,N(0,1)), $\|u\|$ denotes the norm of the vector u, ensuring its normalization for direction. Additionally, ξ is a uniformly distributed random variable in the range [0,1], x_i represents position of the i-th parent(weed), $x'_{i,j}$ represents the position of the j-th node of the i-th parent.

Weed Competitive Exclusion: In each iteration, the entire population is ranked based on fitness. Only the top-performing individuals are retained for the next generation.

3. 3. Escape from Local Optima

The proposed algorithm effectively prevents getting trapped in local optima by utilizing a node-based mechanism. In this mechanism, each solution in the population can generate several independent search branches with different random directions. Each branch, in turn, consists of a sequence of offspring with exponentially decreasing search step sizes, which creates a dynamic balance between exploration and exploitation. When the population converges towards a local optimum, the long branches open new escape routes from the current attraction basin by creating large mutations in the search space. This feature is further enhanced by an adaptive step size control mechanism, which adjusts the length and number of branches based on the quality of each solution. Consequently, even under strong convergence towards local optima, the algorithm can maintain its global exploration capability and significantly increases

the probability of finding the global optimum. In fact, in IWO-based algorithms, one reason for getting stuck in a local optimum is the convergence of the sigma parameter (σ_k) to zero over time, causing the algorithm to perform only exploitation (local search). Since this point is a local optimum, it produces more offspring, further trapping the algorithm. Furthermore, due to the selection policy favoring the best points, other positions with the potential to reach the global optimum are overlooked. This issue also exists in newer algorithms like GOA (8), where a decreasing coefficient (similar to σ_k) is reduced each iteration, influencing new positions and gradually steering the algorithm solely towards exploitation. However, in the proposed algorithm, the branches are independent of such decreasing coefficients and depend only on the quality of the current position. The higher the quality of a position, the more extensively farther points are explored in addition to the local search. This enables the algorithm to escape local optima more effectively.

Figure 4 shows a simple simulation of the proposed algorithm where a point is trapped in a local optimum. However, because this point is of high quality, it generates the maximum number of branches, and each branch produces the maximum number of points. The branches scatter in different directions, covering the entire search space. Due to the extensiveness of these branches, the probability of creating new points near the global optimum is very high. This allows the proposed algorithm to easily escape the local optimum and move towards the global one. According to the figure, the global optimum is located between two branches, a positioning which guarantees that the algorithm will move towards the global point in subsequent iterations.

3. 4. The procedure of BWO This section examines the procedure of the proposed algorithm. The pseudocode and flowchart of the algorithm are provided in Algorithm 1 and Figure 5, respectively. As shown in Algorithm 1, the algorithm begins with an initial population and then, within a main loop, each individual (weed) utilizes two reproduction mechanisms: 1) seed-based reproduction, where each weed produces a number of seeds according to its fitness, and the offspring positions are calculated using an equation that includes a random component, and 2) node-based reproduction (via runners), where the weed generates additional offspring by simulating runner stems. At the end of each iteration, the parent and offspring populations are combined and sorted according to the cost criterion; then, the top nPop individuals are selected for the next generation.

From a time complexity perspective, assuming a population of P and T iterations, and considering S seeds and B branches per weed—each producing R offspring—the computational cost in the worst case reaches O (T×P×D× (S+B× R)). Therefore, the execution time of this algorithm depends on factors such as population size, dimention (D), number of iterations, maximum number of seeds, maximum number of branches, and maximum number of nodes or roots. Compared to the IWO algorithm, the time complexity of the IWO algorithm is O(T×P×D×S). Therefore, the overhead of the proposed algorithm is 1 + (B.R)/S. This means if the product B.R is smaller than S, the overhead is negligible, and if the product is larger, the overhead increases relative to S.

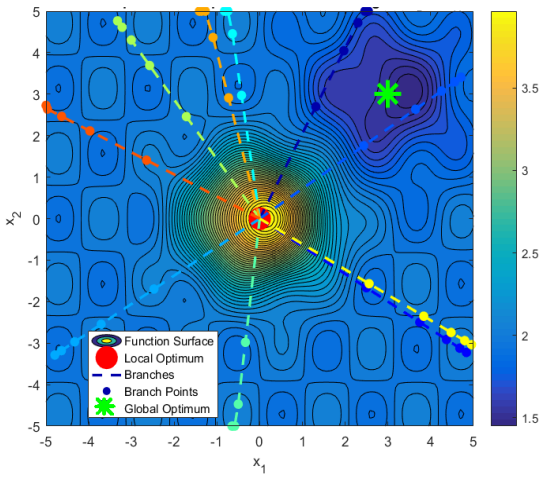


Figure 4. Simulation of the Long Branches mechanism escaping a local optimum.

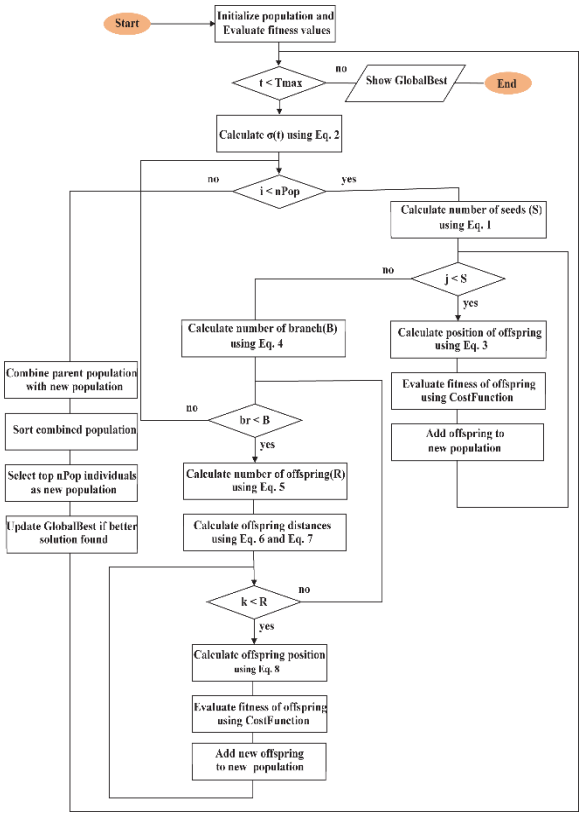


Figure 5. BWO flowchart

When B=1 and R=1, with S=5, the overhead is 1.2, indicating a 20% increase in runtime. In the case where B=2, R=5, and S=10, the overhead becomes 2.0, meaning the proposed algorithm requires twice the computational time of the standard IWO. This demonstrates that the node-based reproduction mechanism, while enhancing exploration, introduces significant overhead when the total number of node-based offspring (B×R) approaches or exceeds the number of seed-based offspring (S). In comparison with the GSFA algorithm, which has a time complexity of O(T×P×D) (equivalent to S=1 in IWO), the GSFA algorithm is S times faster than IWO and (S + B×R) times faster than the proposed algorithm. To reduce the proposed algorithm's complexity, a gradual branch reduction strategy is used. Initially, more branches enable broad exploration to avoid local optima. Later, fewer branches allow focused, faster search for the global optimum.

Algorithm: Bermuda Weed Optimization (BWO) pseudo-code
Input:
1: Algorithm parameters :
- T (Maximum iterations)
- DMin, DMax (Search space bounds)
- P (Population size)
- Other control parameters

Output:
2: GlobalBest (Best solution found)
Begin
%Initialization Phase
3: Initialize population positions randomly within [DMin, DMax]
4: Evaluate fitness values using CostFunction
5: Identify BestCost and WorstCost in population
%Main Optimization Loop
6: for t = 1 to T do
%Parameter Update
7: Calculate σ(t) using Eq. 2
%Reproduction Phase
8: new_population[] =
9: for each weed in population do
% Seed-Based Reproduction
10: Calculate number of seeds (S) using Eq. 1
11: for j = 1 to S do
12: Calculate position of offspring using Eq. 3
13: Evaluate fitness of offspring using CostFunction
14: Add offspring to new_population
15: end for
% **Node-Based Reproduction**
16: Calculate number of branches (B) using Eq. 4
17: for br = 1 to B do
18: Calculate number of offspring (R) using Eq. 5
19: Calculate offspring distances using Eq. 6 and Eq. 7
20: for k = 1 to R do
21: Calculate offspring position using Eq. 8
22: Evaluate fitness of offspring using CostFunction
25: Add selected offspring to new_population
26: end for
27: end for
% Selection Phase
28: Combine parent population with new_population
29: Sort combined population by ascending Cost
30: Select top P individuals as new population
31: Update GlobalBest if better solution found
32: end for
33: Return GlobalBest
End

3. 5. Proposed Cruise Control (CC) In recent years, CC systems have attracted growing interest as a means to partially automate the driving process, primarily to reduce driver fatigue and consequently lower the risks associated with fatigued driving. These systems are designed to regulate vehicle speed based on a reference value set by the driver. An efficiently designed CC system can substantially enhance road safety, improve traffic flow, decrease fuel consumption, and increase overall driving comfort. One of the most widely used approaches for implementing the control unit in such systems is the PID controller. Owing to its straightforward concept, ease of implementation, and reliable performance, the PID controller is commonly employed in CC systems. It ensures speed regulation by modulating the throttle, thus contributing to both comfort and safety. As illustrated in Equation 9, its transfer function integrates proportional, integral, and derivative components to minimize output error.

$$C(s) = K_p + K_d.S + K_i.S^{-1} \quad (9)$$

where K_p is the proportional gain, K_d is the derivative gain, and K_i is the integral gain. In practice, tuning the PID controller's parameters is challenging, particularly for nonlinear systems like vehicle dynamics. Metaheuristic algorithms offer an effective approach for optimizing PID parameters in CC systems, providing advantages including real-time applicability and global search capability (31). To address this challenge, the BWO algorithm is employed for tuning the PID controller in the proposed CC system.

4. 2. 1. Dynamic Model of CC System The vehicle's longitudinal dynamics are derived from Newton's second law, as illustrated in Figure 6, and are mathematically expressed in Equation 10.

$$m\frac{dv(t)}{dt} = F_d - F_a - F_g \quad (10)$$

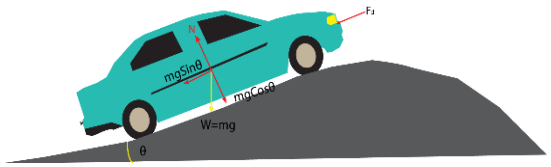


Figure 6. Newton-based longitudinal dynamic model of a CC vehicle

In this equation, F_d is the driving force, $F_a = C_a (v - v_w)^2$ represents the aerodynamic drag force, and $F_g = mgSin\theta$ denotes the gravitational component due to road inclination, often referred to as the climbing resistance. The term $m(dv/dt)$ represents the inertial force acting on the vehicle's mass. The vehicle's total mass (including passengers), aerodynamic drag coefficient, wind speed, and road slope angle are denoted by m, C_a,v_w, and θ, respectively. To simplify the controller design process and focus on the primary vehicle dynamics, a nominal model is considered under ideal operating conditions. Specifically, the vehicle is assumed to travel at a constant nominal speed of v_0=30m/s on a flat road with zero wind speed (θ=0, v_w=0). Furthermore, actuator saturation effects are neglected within the operating range of interest. Under these assumptions, the nonlinear state-space model describing the vehicle's longitudinal dynamics can be formulated as shown in Equations 11 and 12. The first equation represents the vehicle acceleration dynamics, while the second equation captures the actuator dynamics (31-35).

$$v' = \frac{1}{m}(F_d - C_a v^2) \tag{11}$$

$$F_d' = \frac{1}{T}(C_1 u(t-\tau) - F_d) \tag{12}$$

Where T is the time constant of the actuator, C_1 is the actuator gain, u(t) is the throttle control signal, and τ denotes the actuator delay.

Let $v=v_0+\delta v$ and $F_d=F_{d0}+\delta F_d$, where $F_{d0}=C_a v_0^2$ represents the steady-state (where system variables remain constant over time after transient responses have decayed) driving force required to maintain the nominal velocity v_0. Linearizing the simplified nonlinear dynamics in Equations 11 and 12 around the equilibrium operating point (v_0,Fd_0,u_0) yields the perturbation dynamics (i.e., the system response to small deviations from equilibrium), as shown in Equations 13 and 14.

$$\delta v' = -\frac{2C_a v_0}{m}\delta v + \frac{1}{m}\delta F_d \tag{13}$$

$$\delta F_d' = -\frac{1}{T}\delta F_d + \frac{C_1}{T}\delta u(t-\tau) \tag{14}$$

These linearized equations describe the small-signal variations of vehicle velocity and driving force around the steady-state condition. The coefficient $-2C_a v_0/m$ arises from differentiating the nonlinear aerodynamic drag term $-C_a v^2$ with respect to velocity at v=v0, introducing an effective damping proportional to the nominal speed. The term $\delta F_d/m$ represents the influence of driving-force fluctuations on vehicle acceleration, while the second equation captures the actuator dynamics with a time constant T, a static gain C_1/T, and a time delay τ in the control input. Here, $\delta v=v-v_0$, $\delta F_d=F_d-F_{d0}$, and $\delta u=u-u_0$ denote small perturbations from the equilibrium point. The linearized model (Equations 13–14) thus provides a tractable representation for control-system design and stability analysis near the nominal operating speed of 30 m/s. By applying the Laplace transform to the linearized (Equations 13 and 14) under the assumption of zero initial conditions, frequency-domain relationships are obtained. Eliminating $\Delta F_d(s)$ yields the input-output relationship expressed in terms of the Laplace variable. This relationship is derived by solving the Laplace-domain Equations 15 and 16.

$$(s-p_1)\Delta V(s) = \frac{1}{m}\Delta F_d(s) \tag{15}$$

$$(s-p_2)\Delta F_d(s) = \frac{C_1}{T}e^{-\tau s}\Delta U(s) \tag{16}$$

Here, the parameters are defined as $p_1 = -2C_a v_0/m$, $p_2 = -1/T$, and $C = C_1/(mT)$. By substitution and simplification, the transfer function relating the control input ΔU(s) to the velocity output ΔV(s) is derived by Equation 17. This equation represents two dynamic poles—one associated with aerodynamic damping and the other with actuator dynamics—along with a time delay manifested as the exponential term $e^{-\tau s}$.

The non-minimum phase delay ($e^{-\tau s}$), characterized by right-half-plane zeros causing initial inverse response, complicates control design. To simplify analysis, this delay is typically approximated using rational functions. For low-frequency applications, the first-order Padé approximation (Equation 17) is commonly employed, matching the Taylor series expansion to represent time delays as rational transfer functions.

$$\frac{\Delta V(s)}{\Delta U(s)} = \frac{Ce^{-\tau s}}{(s-p_1)(s-p_2)} \tag{17}$$

$$e^{-\tau s} = \frac{1}{e^{\tau s}} \approx \frac{1}{1+\tau s} \tag{18}$$

Substituting Equation 18 into Equation 17 yields the final rational transfer function, where the denominator includes the term 1+τs. Thus, the general form of the linearized and approximated model is represented by Equation 19.

$$G(s) = \frac{C}{(s-p_1)(s-p_2)(s+\tau)} \tag{19}$$

By substituting the parameter values given in Table 1 into Equation 19, the simplified Equation 20 is obtained.

$$G(s) = \frac{2.4767}{(s+0.0476)(s+1)(s+5)} \tag{20}$$

TABLE 1. Model Parameter considered

Parameter name	Value	Parameter name	Value
C_1	742	C_a	1.19
F_{dmax}	3500	m	1500
F_{dmin}	-3500	τ	0.2

3. 4. 2. The Proposed CC System Figure 7 presents the block diagram of the proposed CC system. The proposed framework is divided into two main components: in-vehicle and in-cloud modules. The in-vehicle component consists of a simple PID controller that generates the required control force based on the vehicle's instantaneous speed. Additionally, it includes a communication interface that transmits data to the cloud, such as optimization requests, reference speed, algorithm configuration parameters, and updates or queries regarding CC settings. The cloud-based component comprises several integrated modules: an optimization unit based on the proposed BWO algorithm, a cost function designed to enhance driving comfort, and virtualized models of the vehicle dynamics and the PID controller. It also includes a parameter handling unit responsible for handling data exchange between the vehicle and the cloud. The proposed framework performs the optimization process in the cloud based on the defined objective function and real-time input data. Once the optimal PID parameters are computed, they are transmitted back to the vehicle. The onboard controller then applies these parameters to regulate the vehicle speed effectively, ensuring optimal performance and improved driving comfort.

In many control and optimization engineering problems, designers are often required to satisfy multiple conflicting objectives simultaneously. For instance, in a CC system, it is necessary to strike a balance between reference speed tracking accuracy, passenger comfort, system responsiveness, and long-term stability. These objectives inherently conflict with one another—for example, enhancing system responsiveness can increase jerk levels and thereby reduce ride comfort. It is worth noting that multi-objective optimization frameworks such as NSGA-II have been extensively employed in engineering control and logistics applications to manage trade-offs among such competing goals (36, 37). Inspired by these methodologies, the present study introduces a single-objective cost formulation that integrates the aforementioned performance metrics into a unified scalar function, which is then optimized using the BWO algorithm to achieve an efficient and well-balanced control performance. The proposed cost function is defined in Equation 21.

$$J = W_1 \cdot ISE_{Speed} + W_2 . M_p^{speed} + W_3 \cdot T_S + W_4 . \max|j(t)| \quad (21)$$

where:

- $\mathbf{ISE_{Speed}} = \int_0^t (v_{ref} - v_{overshoot}(t))^2 \, dt$: Integral of Squared Error (ISE) for vehicle speed tracking.
- $\mathbf{M_p^{speed}} = \max(v_{overshoot}(t)) - v_{ref}$: Speed over-shoot penalty to ensure passenger comfort and safety.
- **Ts**: Settling time, defined as the time taken for the vehicle speed to stay within ±2% of the reference speed.
- $\mathbf{max}|\boldsymbol{j(t)}|$: Maximum absolute jerk over the simulation horizon, used to penalize abrupt accelerations and decelerations for ride comfort.

The weighting factors W_1 through W_4 are empirically selected based on the relative importance of each component. In the simulation setup, their values are chosen as: W_1=1, W_2=15, W_3=3, and W_4=8. This weighting strategy reflects practical automotive engineering requirements where comfort and safety often supersede strict reference tracking, especially in commercial vehicle applications. Utilizing this integrated cost function enables the optimization framework to perform coordinated tuning of the PID controller parameters, thereby achieving a desirable balance between precision, comfort, responsiveness, and robustness in the CC system's performance.

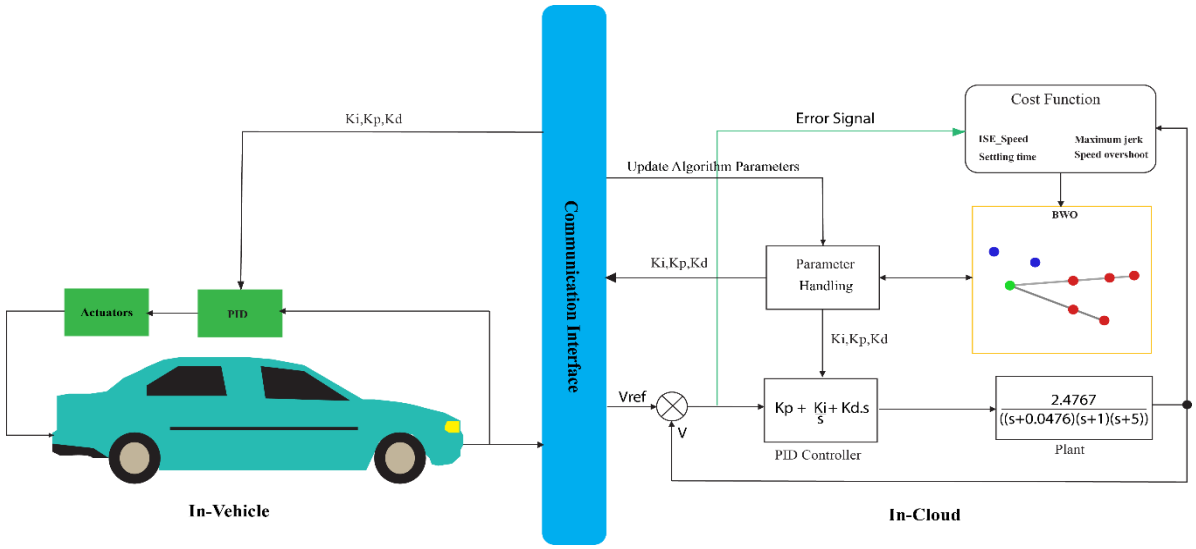


Figure 7. The block diagram of the proposed cloud-based CC system

4. RESULTS AND DISCUSSION

In this section, several simulation studies are conducted to demonstrate the advantages of the proposed optimization algorithm. Initially, we will assess the algorithm's capability to determine the global minimum of 16 standard benchmark functions commonly utilized in the literature. Following this, we simulate the proposed cloud-based CC equipped with the BWO algorithm and compare it with conventional methods. It should be noted that similar methods are not cloud-based, and aside from the functional performance results of the proposed scheme, its cloud framework distinguishes it from similar approaches.

4. 1. Benchmark Test In this section, the proposed BWO algorithm has been evaluated on 16 standard and well-known benchmark functions in the field of optimization. These classical functions, which have been widely utilized in scientific research, provide a suitable basis for comparing the performance of the proposed algorithm with other metaheuristic methods. Table 2 presents the specifications of these functions, where: Dim represents the number of dimensions of the function, Range denotes the search space boundaries and f_{min} indicates the optimal value of the function.

We utilized MATLAB version 2016b for the simulations. In this study, the proposed algorithm, the basic IWO algorithm, and three improved versions of

TABLE 2. Unimodal and multimodal benchmark functions

Functions	Dim	Range	f_{min}
$\mathbf{F_1(x)} = \sum_{i=1}^{n} x_i^2$	3,10,30	[-100,100] [-10,10]	0
$\mathbf{F_2(x)} = \sum_{i=1}^{n}\|x_i\| + \prod_{i=1}^{n}\|x_i\|$	3,10,30	[-10,10] [-1,1]	0
$\mathbf{F_3(x)} = \sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	3,10,30	[-100,100] [-10,10]	0
$\mathbf{F_4(x)} = \max_i\{\|x_i\|, 1 \le i \le n\}$	3,10,30	[-100,100] [-10,10]	0
$\mathbf{F_5(x)} = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	3,10,30	[-30,30] [-3,3]	0
$\mathbf{F_6(x)} = \sum_{i=1}^{n}(\|x_i + 0.5\|)^2$	3,10,30	[-100,100] [-10,10]	0
$\mathbf{F_7(x)} = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	3,10,30	[-1.28,1.28]	0
$\mathbf{F_8(x)} = \sum_{i=1}^{n} -x_i \sin(\sqrt{\|x_i\|})$	3,10,30	[-500,500] [-50,50]	-418.9829×Dim
$\mathbf{F_9(x)} = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	3,10,30	[-5.12,5.12]	0
$\mathbf{F_{10}(x)} = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	3,10,30	[-32,32]	0
$\mathbf{F_{11}(x)} = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	3,10,30	[-600,600] [-60,60]	0
$\mathbf{F_{12}(x)} = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_i)] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$ *	3,10,30	[-50,50] [-5,5]	0
$\mathbf{F_{13}(x)} = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	3,10,30	[-50,50] [-5,5]	0
$\mathbf{F_{14}(x)} = -\sum_{i=1}^{n}\sin(x_i).\left(\sin\left(\frac{ix_i^2}{\pi}\right)\right)^{2m}, m = 10$	3,10,30	$[0,\pi]$	-4.687
$\mathbf{F_{15}(x)} = \sum_{i=1}^{n}\|x_i\|^{i+1}$	3,10,30	[-20,20] [-2,2]	0
$F_{16}(x) = \{[\sum_{i=1}^{n}\sin^2(x_i)] - \exp(-\sum_{i=1}^{n} x_i^2)\}.\exp[-\sum_{i=1}^{n}\sin^2(\sqrt{\|x_i\|})$	3,10,30	[-10,10] [-1,1]	-1

IWO were implemented and evaluated on 16 benchmark functions (Table 2) under three different scenarios. In the first scenario, the search space is limited, the problem has low dimensionality (3 dimensions), and the number of iterations is set to 10. In the second scenario, the search space remains limited (focused region), the problem has medium dimensionality (10 dimensions), and the number of iterations is increased to 100. In the third scenario, the search space is broad, the dimensionality is high (30 dimensions), and the number of iterations is set to 1000. Each algorithm was executed five times for every benchmark function in each scenario, and the average best cost of the five runs is reported in the tables (Tables 3 to 5). In these tables, D represents the problem dimensionality and I indicates the number of algorithm iterations.

Based on the results presented in Tables 3 to 5, the normalized error for each benchmark function is calculated, and statistical indicators including mean, standard deviation, maximum, and minimum errors are displayed in various graphs. These results effectively demonstrate the scalability of the proposed algorithm compared to other IWO-based methods. After confirming the efficiency of the proposed algorithm within the IWO category, a comparison is made with the novel GSFA algorithm to determine the position of the proposed algorithm outside this category and in comparison with newer algorithms. The simulation results confirm the high efficiency of the proposed algorithm.

In all three scenarios, the average best cost values indicate that the proposed algorithm achieves significantly superior results for most benchmark functions. Overall, the proposed algorithm demonstrates exceptional performance in benchmark functions F1, F2, F3, F4, F6, F11, F13, and F14, while yielding comparable or marginally different results in other benchmarks compared to peer algorithms in its category.

TABLE 3. Average best cost comparison over 5 runs per algorithm (Scenario 1)

Function	IWO [2006] I=10, D=3	CIWO [2012] I=10, D=3	MCIWO [2019] I=10, D=3	LF-IWO [2021] I=10, D=3	Proposed BWO I=10, D=3
F1	1.70e+00	3.34E+01	2.49E+01	4.11e+01	2.45e-03
F2	5.01e-02	6.39E-01	3.16E-01	6.68e-01	3.04e-02
F3	1.59e+00	3.76E+01	3.32E+01	2.28e+01	3.42e-03
F4	1.24e-01	3.27E+00	3.34E+00	4.24e+00	2.86e-02
F5	3.08e+01	1.29E+02	1.57E+02	1.68e+02	3.03e+01
F6	1.45e+00	3.92E+01	2.35E+01	2.53e+01	9.12e-04
F7	1.06e-02	4.89E-01	2.97E-01	9.64e-02	2.09e-02
F8	-4.16e+01	-1.58E+01	-4.46E-01	-3.95e+01	-8.64e+01
F9	3.27e+00	2.67E+01	2.12E+01	2.49e+01	1.67e+00
F10	1.30e-01	3.37E+00	4.13E+00	4.03e+00	1.07e-01
F11	2.62e-01	1.06E+00	1.02E+00	7.05e-01	5.48e-02
F12	1.98e+01	2.28E+01	2.38E+01	2.53e+01	1.98e+01
F13	6.36e-03	9.00E-01	9.64E-01	1.06e+00	1.66e-03
F14	-2.59e-10	-1.46E-10	-1.96E-10	-1.71e-10	-2.88e-10
F15	1.71e-05	6.88E-01	4.42E-01	5.50e-01	3.13e-05
F16	-9.56e-01	-3.04E-01	-5.15E-01	-4.97e-01	-9.69e-01

TABLE 4. Average best cost comparison over 5 runs per algorithm (Scenario 2)

Function	IWO [2006] I=100, D=10	CIWO [2012] I=100, D=10	MCIWO [2019] I=100, D=10	LF-IWO [2021] I=100, D=10	Proposed BWO I=100, D=10
F1	3.16e-04	2.76E+02	2.64E+02	1.30e+02	3.20e-04
F2	4.47e-02	4.60E+00	3.79E+00	2.37e-01	5.61e-02
F3	7.76e-03	3.29E+02	2.29E+02	1.35e+02	1.71e-03
F4	1.63e-02	8.54E+00	8.02E+00	6.03e+00	1.20e-02
F5	3.76e+01	9.21E+03	1.06E+04	8.86e+02	3.72e+01
F6	2.43e-04	2.61E+02	2.45E+02	1.40e+02	4.03e-04
F7	2.90e-02	2.26E+01	1.13E+01	4.17e-01	1.93e-02
F8	-2.25e+02	-3.33E+00	-1.43E+02	-9.71e+01	-2.29e+02
F9	1.38e+01	1.45E+02	1.35E+02	5.23e+01	9.20e+00
F10	2.47e-02	6.54E+00	6.80E+00	3.98e+00	2.62e-02
F11	1.39e+00	3.39E+00	3.57E+00	2.35e+00	6.09e-01
F12	2.69e+01	3.27E+01	3.39E+01	2.83e+01	2.69e+01
F13	4.50e-04	1.158E+01	9.84E+00	5.59e+00	4.52e-04
F14	-9.03e-10	-3.21E-10	-3.67E-10	-7.63e-10	-8.41e-10
F15	1.49e-06	6.24E+01	4.40E+01	8.22e-01	1.84e-06
F16	5.96e-03	8.02E-03	8.58E-03	6.16e-03	5.95e-03

Based on the results presented in Tables 3, 4, and 5, which display the mean best cost values obtained from five independent runs of each algorithm across three different scenarios, a comprehensive comparison of the algorithms can be conducted from multiple perspectives. To ensure statistical reliability and comprehensive assessment, we compute multiple performance indicators including the mean relative error, standard deviation of relative errors, and minimum and maximum relative error values for each algorithm in each scenario.

This multi-metric approach provides insights into not only the accuracy but also the stability and scalability of

TABLE 5. Average best cost comparison over 5 runs per algorithm (Scenario 3)

Function	IWO [2006] I=1000, D=30	CIWO [2012] I=1000, D=30	MCIWO [2019] I=1000, D=30	LF-IWO [2021] I=1000, D=30	Proposed BWO I=1000, D=30
F1	3.63e-05	1.12E+03	1.04E+05	5.86e+04	3.01e-05
F2	3.58e-02	5.35E+15	7.77E+16	1.05e+02	2.85e-02
F3	8.23e+01	3.54E+03	1.86E+05	1.06e+05	6.36e-02
F4	1.99e+01	9.77E+00	9.78E+01	8.07e+01	6.89e-03
F5	5.79e+01	5.89E+08	5.37E+08	4.66e+07	5.39e+02
F6	3.65e-05	1.12E+03	1.09E+05	5.77e+04	2.44e-05
F7	3.00e-02	2.67E+02	2.64E+02	5.57e-02	3.71e-02
F8	-7.08e+03	-4.77E+03	-2.16E+03	-2.55e+03	-7.20e+03
F9	6.21e+01	5.30E+02	5.30E+02	1.35e+02	9.61e+01
F10	1.53e+01	2.05E+01	2.09E+01	1.90e+01	4.35e-03
F11	2.85e+02	1.02E+03	1.00E+03	6.83e+02	1.57e-02
F12	3.32e+01	1.43E+09	1.53E+09	1.85e+08	3.50e+01
F13	2.74e-05	2.51E+09	2.65E+09	3.30e+08	3.81e-03
F14	-2.20e-09	-6.90E-10	-8.50E-10	-2.25e-09	-2.34e-09
F15	1.45e-07	3.74E+34	4.52E+33	1.57e+22	2.10e-07
F16	1.53e-17	1.28E-05	6.97E-06	7.95e-20	1.667e-17

each algorithm. To facilitate meaningful comparisons and enhance visual interpretability in graphical representations, all error values undergo min-max normalization. This process scales the absolute error values to a consistent range, enabling direct comparison across different algorithms and scenarios. Specifically, each absolute error value is normalized using the global range of variation:

$$E_{norm} = \frac{|x_i - f_i|}{|x_{worst} - x_{best}|} \quad (22)$$

where x_i represents the solution found by the algorithm for benchmark function i, f_i denotes the known global optimum of the function, and x_{max} and x_{min} represent the maximum and minimum values obtained across all algorithms and scenarios, respectively. The Mean Absolute Error in liner scale (MAE_{liner}) is then calculated by using Equation 23.

$$MAE_{Liner} = \frac{1}{N}\sum_{i=1}^{N} E_{normalized} \quad (23)$$

Where N=16 represents the total number of benchmark functions in our test suite.

Given that normalized error values may still span multiple orders of magnitude, we employ logarithmic transformation to improve interpretability and visualization clarity by Equation 24.

$$MAE_{Log} = \log\left(MLiE_{norm} + (1E-100)\right) \quad (24)$$

The addition of the 10^{-100} term ensures numerical stability when handling extremely small error values approaching zero. This transformation effectively compresses the dynamic range of error values, making subtle performance differences more discernible.

Using both linear and logarithmic error metrics, we compute comprehensive statistical measures including mean, standard deviation, minimum, and maximum values. These metrics form the foundation for our analytical visualizations and provide quantitative evidence for algorithm performance comparisons.

Figure 8 illustrates the mean relative error in linear scale across different scenarios. The proposed BWO algorithm demonstrates superior performance, maintaining the lowest error rates while exhibiting remarkable stability across diverse problem conditions. This consistent performance underscores the algorithm's robustness and adaptability.

Figure 9 presents the mean relative error in logarithmic scale, revealing nuanced performance characteristics that might be obscured in linear representations. The logarithmic transformation accentuates relative performance differences, particularly for algorithms with very small error values.

Figure 10 depicts the standard deviation of relative errors in logarithmic scale, providing insights into algorithm stability. The proposed algorithm shows remarkable consistency in standard deviation values across all scenarios, with an overall lower variability

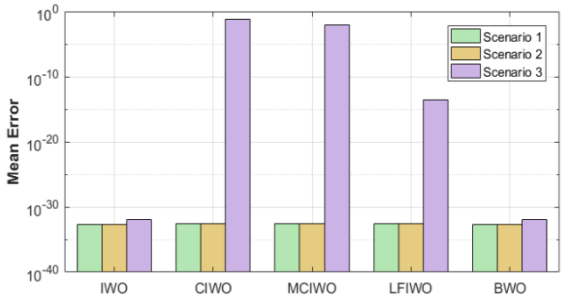


Figure 8. The mean error in linear scale across different scenarios

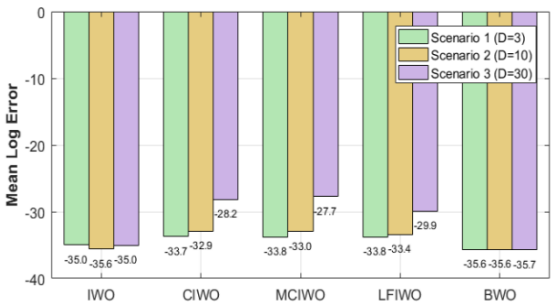


Figure 9. The mean error in logarithmic scale across different scenarios

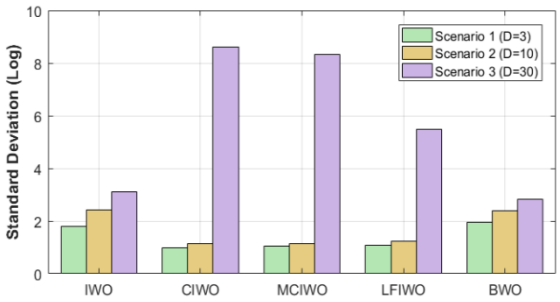


Figure 10. The standard deviation of relative errors in logarithmic scale

compared to competing algorithms. This finding indicates exceptional reliability and predictable performance. The slightly elevated standard deviation in initial scenarios actually reflects the algorithm's adaptive nature, where it achieves extraordinary performance on specific problem types while maintaining competitive performance on others.

Figure 11 examines algorithmic scalability by plotting mean logarithmic errors against increasing problem dimensions. The proposed BWO algorithm maintains a stable performance trend across all dimensionalities, demonstrating excellent scalability properties. In stark contrast, MCIWO, CIWO, and LFIWO algorithms exhibit significant performance degradation as problem dimensionality increases, revealing limitations in handling high-dimensional optimization landscapes.

Figures 12-14 provide detailed analysis of extreme performance characteristics through minimum and maximum best cost trends across the three scenarios. The proposed algorithm consistently achieves the best minimum error values in all scenarios, indicating superior exploitation capabilities. Furthermore, it maintains the most favorable maximum error values, demonstrating robust exploration and reliable worst-case performance.

This comprehensive evaluation demonstrates the proposed BWO algorithm's superior performance,

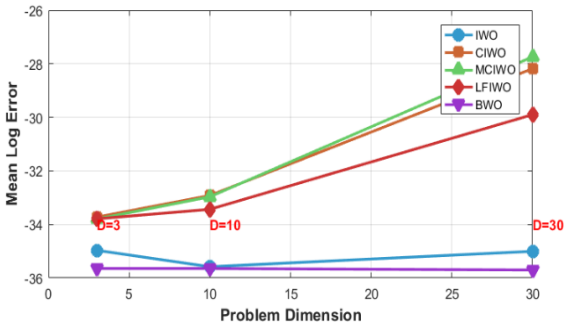


Figure 12. Performance evolution with increasing problem dimensionality

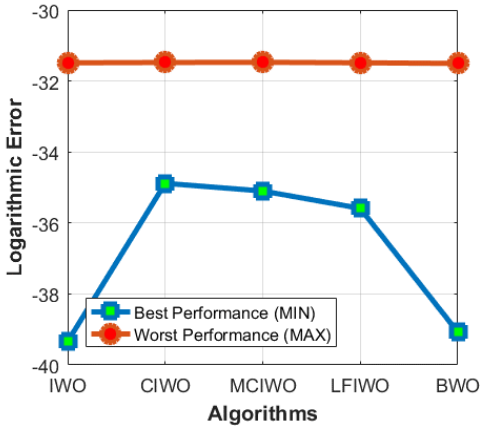


Figure 12. Error bounds in Scenario 1

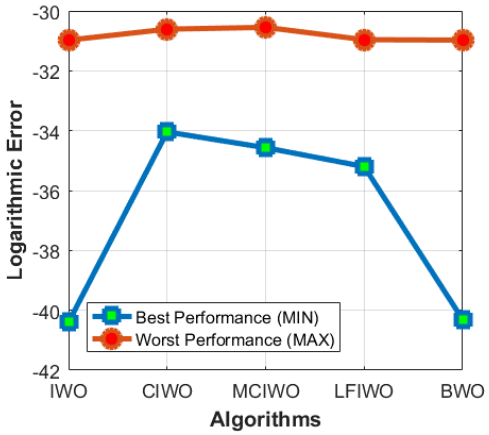


Figure 13. Error bounds in Scenario 2

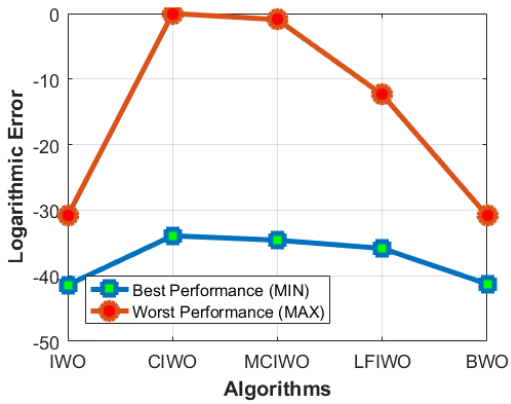


Figure 14. Error bounds in Scenario 3

robustness, and scalability compared to IWO-based algorithms. To further validate the proposed algorithm's competitiveness beyond IWO-based approaches, we performed comparative evaluation with the recently developed GSFA algorithm. The comprehensive comparison results are presented in Table 6.

The proposed BWO algorithm demonstrates superior overall performance compared to the GSFA algorithm, achieving significant improvements in 14 out of 16 benchmark functions. With an average performance enhancement of 64.43% and exceptional precision exceeding 99% on nine key functions, BWO proves to be more robust and reliable across diverse problem types. Although GSAFA excels specifically on function F8, the consistent dominance of BWO establishes it as a more versatile and effective optimization approach.

4. 2. Simulation Results of the Proposed CC To evaluate the performance of the proposed CC system after its implementation in MATLAB, several performance metrics—such as overshoot (Mp), settling time (Ts), speed tracking error (ISESpeed), maximum jerk (max|j(t)|))— were compared against three state-of-the-art methods from the literature (33-35). Notably, all these studies adopted the Integral of Time-weighted Squared Error (ITSE) as their cost function. In contrast, the present study not only introduces an enhanced version of the IWO algorithm but also proposes a new cost function tailored to improve driving comfort. To provide a fair comparison, the previous works were re-evaluated using the proposed cost function. As evidenced in Table 7, our method results in a noticeable reduction in error metrics, highlighting its superior capability in enhancing CC performance while prioritizing passenger comfort. The proposed algorithm reduces overshoot by 45.92%, settling time by 29.38%, ISEspeed by 8.92%, and maximum jerk by 20.09% compared to the RPO-Based method (the newest). Figure 15 presents the cost function convergence plot across iterations along with the corresponding step response using the optimized parameters.

It should be noted that the parameters of the proposed algorithm for the practical problem include: decision variable bounds of [-10, 10], P = 30, initial P= 10, S_{min} = 0, S_{max} = 5, b_{min} = 1, b_{max} = 10, r_{min} = 1, r_{max} = 10, $\sigma_{initial}$ = 1, σ_{final} = 1e-7, and n = 2.

TABLE 6. Average best cost and standard deviation comparison over 5 runs per algorithm

Function	GSAFA [2026] I=100, D=3		Proposed BWO I=100, D=3	
	Best Cost	Std.	Best Cost	Std.
F1	4.102E-06	3.636E-06	2.881E-08	2.869E-08
F2	4.392E-04	3.730E-04	3.767E-04	2.525E-04
F3	4.981E-05	2.751E-05	3.274E-08	2.965E-08
F4	4.865E-03	3.125E-03	1.376E-04	4.330E-05
F5	3.026E+01	4.921E-01	3.000E+01	2.326E-03
F6	1.532E-06	1.618E-06	1.012E-07	5.906E-08
F7	6.582E-03	4.895E-03	1.361E-03	5.691E-04
F8	-1.148E+03	9.579E+01	-8.133E+01	5.084E+00
F9	1.325E+00	1.060E+00	1.529E-01	3.418E-01
F10	7.851E-02	4.872E-02	5.944E-04	2.110E-04
F11	9.925E-02	3.625E-02	4.931E-03	4.613E-03
F12	1.979E+01	3.636E-03	1.9784E+01	1.055E-07
F13	1.103E-04	8.483E-05	4.964E-07	7.180E-07
F14	-3.140E-10	1.625E-11	-3.313E-10	6.961E-12
F15	-7.558E-01	4.248E-01	3.174E-10	2.605E-10
F16	-1.478E-01	2.100E-01	-9.996E-01	1.978E-04

TABLE 7. Comparison of performance parameters for different algorithms

Algorithm	Year	M_P	T_S	ISE_{Speed}	max|j(t)|
ObAOANM	2021	4.20	6.58	622.78	69.41
CS	2022	6.58	5.83	479.36	127.03
Fierfly	2022	10.34	8.04	513.44	157.63
RPO	2024	3.31	6.40	778.33	63.57
Proposed	-	1.79	4.52	708.91	50.80

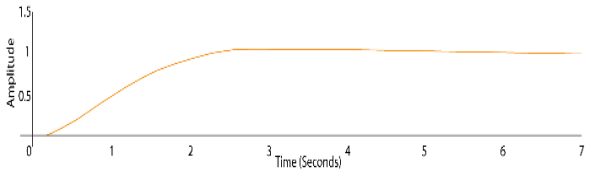


Figure 15. Optimal step response of the BWO

5. CONCLUSION AND FUTURE WORK

This study introduced the BWO algorithm—a novel, scalable metaheuristic inspired by the propagation behavior of Bermuda grass—and successfully applied it within a cloud-based CC system designed for electric and resource-constrained automotive platforms. The primary strength of the BWO algorithm lies in its scalability, which is particularly advantageous in automotive systems for two main reasons:

1. In accuracy-sensitive scenarios (e.g., traffic jams or stationary conditions), the algorithm can increase iteration count for enhanced solution quality.
2. In time-critical driving situations (e.g., highway cruising or varying road conditions), it delivers near-optimal solutions within tightly constrained iteration limits.

This flexibility allows the system to dynamically balance computational effort and real-time performance according to driving scenarios. To assess its effectiveness, BWO was rigorously evaluated using 16 well-known benchmark optimization functions. The

simulation results demonstrate that BWO outperforms the original IWO and three state-of-the-art variants in most scenarios, exhibiting superior scalability and solution quality. Additionally, BWO shows high robustness with respect to parameter sensitivity, achieving consistent performance across various problems using a single set of parameters.

The cloud-based BWO-leveraged CC system was benchmarked against four previously published approaches. Compared to the GA-based method—the strongest baseline—BWO reduced overshoot by 45.92%, settling time by 29.38%, ISEspeed by 8.92%, and maximum jerk by 20.09% compared to the RPO-Based method (the newest). Thus, the proposed CC framework offers scalability unlike conventional methods, enabling adaptive performance under diverse driving conditions. Its cloud-based architecture ensures compatibility with resource-limited and electric vehicles, while simultaneously enhancing accuracy and superior outcomes in vehicular comfort measures.

For future work, we propose an Adaptive Cruise Control (ACC) system based on Digital Twin (DT) technology, in which the DT is equipped with an optimization unit powered by the BWO algorithm. A Digital Twin is defined as a virtual replica of a physical system that synchronizes with its real-world counterpart through real-time data exchange, enabling simulation, analysis, and optimization without physical intervention.

Acknowledgements

The authors gratefully acknowledge the Cyber-Physical Systems Research Laboratory at the University of Isfahan for providing the facilities and resources essential to this research.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Ethics Approval and Consent to Participate

This article does not involve any studies with human participants or animals performed by any of the authors. Therefore, ethics approval and consent to participate are not applicable.

Competing Interests

The author declares no financial or organizational conflicts of interest.

Data Availability

The implementation codes for the proposed algorithm in this study are fully available via the following GitHub repository: [https://github.com/Cyber-Physical-Laboratory-FCE-UI/BWO].

Declaration of Generative AI and AI-Assisted Technologies in the Writing Process

During the preparation of this manuscript, the authors used DeepSeek exclusively for minor language editing to improve readability. After using this tool, the authors carefully reviewed and edited the content as needed and take full responsibility for the content of the published article.

Authors Biosketch

Maytham Allahi Rudposhti received his B.Sc. and M.Sc. degrees in Computer Engineering from the University of Science and Technology of Mazandaran and Babol Noshirvani University of Technology, Iran, respectively. He is currently a Ph.D. candidate in Computer Engineering at the University of Isfahan, focusing on Autonomous Vehicles (AV) and Digital Twin (DT) technology.

Ali Bohlooli received the B.Sc. and M.Sc. degrees (Hons.) in computer engineering from the Department of Electrical and Computer Engineering, Isfahan University of Technology, Iran, in 2001 and 2003, respectively, and the Ph.D. degree in computer engineering from the University of Isfahan, Iran, in 2011. He is currently an Associate Professor with the Faculty of Computer Engineering, University of Isfahan. His research interests include cyber-physical systems, embedded artificial intelligence, and the Internet of Things (IoT).

Kamal Jamshidi received his Ph.D. in Electrical Engineering from the Indian Institute of Technology, India. He has been a faculty member of the Faculty of Computer Engineering at the University of Isfahan, Iran, since 1995. His academic background is in electrical engineering, and his teaching and research activities are carried out within the field of computer engineering.

REFERENCES

1. Mirjalili S, Lewis A. The whale optimization algorithm. Advances in Engineering Software, 2016;95:51-67. https://doi.org/10.1016/j.advengsoft.2016.01.008
2. Holland JH. Genetic algorithms. Scientific American, 1992;267(1):66-72. https://doi.org/10.1038/scientificamerican0792-66
3. Kahrizi MR, Kabudian SJ. Projectiles optimization: A novel metaheuristic algorithm for global optimization. International Journal of Engineering Transactions A: Basics, 2020;33(10):1924-38. https://doi.org/10.5829/ije.2020.33.10a.11
4. El-Shorbagy MA, Bouaouda A, Nabwey HA, Abualigah L, Hashim FA. Advances in Henry gas solubility optimization: A physics-inspired metaheuristic algorithm with its variants and

applications. IEEE Access, 2024;12:26062-95. https://doi.org/10.1109/ACCESS.2024.3365700

5. Sowmya R, Premkumar M, Jangir P. Newton–Raphson-based optimizer: A new population-based metaheuristic algorithm for continuous optimization problems. Engineering Applications of Artificial Intelligence, 2024;128:107532. https://doi.org/10.1016/j.engappai.2023.107532
6. Mehrabian AR, Lucas C. A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics, 2006;1(4):355-66. https://doi.org/10.1016/j.ecoinf.2006.07.003
7. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Advances in Engineering Software, 2014;69:46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007
8. Saremi S, Mirjalili S, Lewis A. Grasshopper optimisation algorithm: Theory and application. Advances in Engineering Software, 2017;105:30-47. https://doi.org/10.1016/j.advengsoft.2017.01.004
9. Talatahari S, Azizi M, Tolouei M, Talatahari B, Sareh P. Crystal structure algorithm (CryStAl): A metaheuristic optimization method. IEEE Access, 2021;9:71244-61. https://doi.org/10.1109/ACCESS.2021.3079161
10. Zhao S, Zhang T, Ma S, Chen M. Dandelion optimizer: A nature-inspired metaheuristic algorithm for engineering applications. Engineering Applications of Artificial Intelligence, 2022;114:105075. https://doi.org/10.1016/j.engappai.2022.105075
11. Han M, Du Z, Yuen KF, Zhu H, Li Y, Yuan Q. Walrus optimizer: A novel nature-inspired metaheuristic algorithm. Expert Systems with Applications, 2024;239:122413. https://doi.org/10.1016/j.eswa.2023.122413
12. Zhong C, Li G, Meng Z, Li H, Yildiz AR, Mirjalili S. Starfish optimization algorithm (SFOA): A bio-inspired metaheuristic algorithm for global optimization compared with 100 optimizers. Neural Computing and Applications, 2025;37(5):3641-83. https://doi.org/10.1007/s00521-024-10694-1
13. Amani B, Nouri M, Mousavi Ghasemi SA. Gray squirrel foraging algorithm for function optimization. International Journal of Engineering, 2026;39(7):1644-56. https://doi.org/10.5829/ije.2026.39.07a.09
14. Ehsaeyan E. Rock-climbing group: An innovative meta-heuristic approach for efficiently tackling optimization problems. International Journal of Engineering Transactions B: Applications, 2025;38(11):2796-818. https://doi.org/10.5829/ije.2025.38.11b.24
15. Ehsaeyan E. Gold seekers algorithm: An innovative metaheuristic approach for global optimization and its application in image segmentation. International Journal of Engineering Transactions C: Aspects, 2025;38(9):2114-29. https://doi.org/10.5829/ije.2025.38.09c.09
16. Perez C, Climent L, Nicolo G, Arbelaez A, Salido MA. A hybrid metaheuristic with learning for a real supply chain scheduling problem. Engineering Applications of Artificial Intelligence, 2023;126:107188. https://doi.org/10.1016/j.engappai.2023.107188
17. Mallahzadeh A, Eshaghi S, Hassani H. Compact U-array MIMO antenna designs using IWO algorithm. International Journal of RF and Microwave Computer-Aided Engineering, 2009;19(5):568-76. https://doi.org/10.1002/mmce.20379
18. Pahlavani P, Delavar MR, Frank AU. Using a modified invasive weed optimization algorithm for a personalized urban multi-criteria path optimization problem. International Journal of Applied Earth Observation and Geoinformation, 2012;18:313-28. https://doi.org/10.1016/j.jag.2012.03.004
19. Ahmadi M, Mojallali H. Chaotic invasive weed optimization algorithm with application to parameter estimation of chaotic systems. Chaos, Solitons & Fractals, 2012;45(9):1108-20. https://doi.org/10.1016/j.chaos.2012.05.010
20. Nikoofard AH, Hajimirsadeghi H, Rahimi-Kian A, Lucas C. Multiobjective invasive weed optimization: Application to analysis of Pareto improvement models in electricity markets. Applied Soft Computing, 2012;12(1):100-12. https://doi.org/10.1016/j.asoc.2011.09.005
21. Ghasemi M, Ghavidel S, Akbari E, Vahed AA. Solving non-linear, non-smooth and non-convex optimal power flow problems using chaotic invasive weed optimization algorithms based on chaos. Energy, 2014;73:340-53. https://doi.org/10.1016/j.energy.2014.06.026
22. Mishra SK, Bose PSC, Rao CSP. An invasive weed optimization approach for job shop scheduling problems. International Journal of Advanced Manufacturing Technology, 2017;91(9):4233-41. https://doi.org/10.1007/s00170-017-0091-x
23. Mandava RK, Vundavilli PR. Implementation of modified chaotic invasive weed optimization algorithm for optimizing the PID controller of the biped robot. Sadhana, 2018;43(5):66. https://doi.org/10.1007/s12046-018-0851-9
24. Abdelkader EM, Moselhi O, Marzouk M, Zayed T. A multi-objective invasive weed optimization method for segmentation of distress images. Intelligent Automation & Soft Computing, 2020;26(4):643-61. https://doi.org/10.32604/iasc.2020.010100
25. Kashyap AK, Parhi D, Pandey A. Improved modified chaotic invasive weed optimization approach to solve multi-target assignment for humanoid robot. Journal of Robotics and Control, 2021;2(3):194-9. https://doi.org/10.18196/jrc.2377
26. Ibrahim A, Anayi F, Packianather M, Alomari OA. New hybrid invasive weed optimization and machine learning approach for fault detection. Energies, 2022;15(4):1488. https://doi.org/10.3390/en15041488
27. Beskirli M. A novel invasive weed optimization with levy flight for optimization problems: The case of forecasting energy demand. Energy Reports, 2022;8:1102-11. https://doi.org/10.1016/j.egyr.2021.11.108
28. Kalhori M, Ashofteh PS, Moghadam SH. Development of the multi-objective invasive weed optimization algorithm in the integrated water resources allocation problem. Water Resources Management, 2023;37(11):4433-58. https://doi.org/10.1007/s11269-023-03564-3
29. Chauhan U, Chhabra H, Jain P, Dev A, Chauhan N, Kumar B. Chaos inspired invasive weed optimization algorithm for parameter estimation of solar PV models. IFAC Journal of Systems and Control, 2024;27:100239. https://doi.org/10.1016/j.ifacsc.2023.100239
30. Taliaferro CM, Rouquette FM, Mislevy P. Bermudagrass and stargrass. Agronomy Monographs, 2004;45:417-75. https://doi.org/10.2134/agronmonogr45.c12
31. Heybetli F, Danayiyen Y, Tasdemir AB, Senyigit S. Comparative analysis of metaheuristic algorithms in PID-based vehicle cruise control systems. Verus Journal, 2025;25(1):1-16. https://doi.org/10.5152/electrica.2025.25051
32. Pradhan R, Majhi SK, Pradhan JK, Pati BB. Antlion optimizer tuned PID controller based on bode ideal transfer function for automobile cruise control system. Journal of Industrial Information Integration, 2018;9:45-52. https://doi.org/10.1016/j.jii.2018.01.002
33. Izci D, Ekinci S, Kayri M, Eker E. A novel enhanced metaheuristic algorithm for automobile cruise control system. Electrica, 2021;21(3):283-97. https://doi.org/10.5152/electrica.2021.21016

34. Prasanna V, Nelson A, Hnaumanthakari S, Kumar VK, Kirubakaran S, Kumar MJ. Metaheuristic algorithm for automatic cruise control system. Proceedings of the 3rd International Conference on Smart Electronics and Communication, 2022:206-11. https://doi.org/10.1109/ICOSEC54921.2022.9952117

35. Saravanan G, Pazhanimuthu C, Sathish Kumar D, Lalitha B, Senthilkumar M, Kannan E. Red panda optimization algorithm-based PID controller design for automobile cruise control system. Proceedings of the International Conference on Smart Systems for Electrical, Electronics, Communication and Computer Engineering, 2024:33-37. https://doi.org/10.1109/ICSSEECC61126.2024.10649422

36. Afshar M, Hadji Molana SY, Rahmani Parchicolaie B. A multi-objective optimization model for multi-commodity closed-loop supply chain network considering disruption risk. International Journal of Engineering Transactions A: Basics, 2024;37(4):646-61. https://doi.org/10.5829/ije.2024.37.04a.07

37. Ebrahimi Gouraji R, Soleimani H, Afshar Najafi B. Optimization of sustainable vehicle routing problem taking into account social utility and employing a strategy with multiple objectives. International Journal of Engineering Transactions A: Basics, 2025;38(7):1631-58. https://doi.org/10.5829/ije.2025.38.07a.15

Persian Abstract

چکیده

این پژوهش الگوریتم بهینه‌سازی علف برمودا (BWO) را معرفی می‌کند؛ یک الگوریتم فراابتکاری جدید و مقیاس‌پذیر که از رشد تهاجمی علف برمودا الهام گرفته شده است. این الگوریتم به عنوان نسخه بهبودیافته‌ای از الگوریتم بهینه‌سازی علف هرز مهاجم (IWO) توسعه یافته و با تقلید از راهبردهای قوی تکثیر گیاه، تعادل بهتری میان کاوش جهانی و بهره‌برداری محلی برقرار می‌کند. عملکرد الگوریتم با ارزیابی دقیق در برابر چهار نسخه قبلی مبتنی بر IWO سنجیده شد و مقیاس‌پذیری آن از طریق میانگین کمترین نرخ خطا و عملکرد پایدار در سناریوهای متنوع اثبات گردید. علاوه بر این، BWO با الگوریتم جدید جستجوی سنجاب خاکستری (GSFA)—که خارج از دسته IWO قرار دارد—مقایسه شد تا عملکرد آن در برابر روشی جدید و خارج از خانواده IWO نیز ارزیابی شود؛ این مقایسه برتری رقابتی BWO را نشان داد و به طور متوسط ٦٤/٤٣ درصد بهبود در نتایج بهترین هزینه به همراه داشت. همگرایی قوی و مقیاس‌پذیری BWO، آن را برای کاربردهای بلادرنگ—به ویژه در سیستم‌های خودرویی—بسیار مناسب می‌سازد. در یک پیاده‌سازی عملی با استفاده از چارچوب کنترل کروز مبتنی بر ابر، BWO به طور قابل توجهی از روش مبتنی بر بهینه‌سازی پاندای قرمز (RPO) پیشی گرفت و فراجهش را ٤٥/٩٢ درصد، زمان نشست را ٢٩/٣٨درصد، میانگین مربعات خطای سرعت (ISE) را ٨/٩٢ درصد و حداکثر جرک ٢٠/٠٩ درصد کاهش داد. با دستیابی به همگرایی نزدیک به بهینه و بهره‌گیری از استقرار ابری با قابلیت مقیاس‌پذیری بالا، BWO می‌تواند به طور مؤثر با الزامات متنوع سیستم‌های خودرویی سازگار شود و در حالت‌های کاری متعدد، به کارایی بالایی دست یابد.