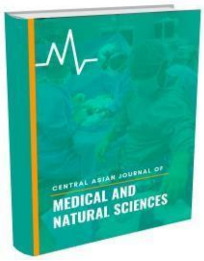

CENTRAL ASIAN JOURNAL OF MEDICAL AND NATURAL SCIENCES
https://cajmns.centralasianstudies.org/index.php/CAJMNS
Volume: 06 Issue: 03 | July 2025 ISSN: 2660-4159

.

Artic le

The Optimization of Traffic Routing Systems using A Swarm Intelligence

Ali Hassan Ahmed*[1], Shahla Uthman Umar[2]

1. University of Kirkuk \ College of Computer Science and Information Technology, Iraq\Kirkuk
2. University of Kirkuk \ College of Computer Science and Information Technology\ Software Department, Iraq\Kirkuk

* Correspondence: stcm23009@uokirkuk.edu.iq

Abstract: Route optimization is one of the significant roles in ITS, as it enables dynamic changes in routes in real-time, based on traffic information. This would reduce travel time, decrease congestion, and minimize the environmental impact due to vehicle emissions. Many algorithms, particularly those inspired by nature, like the Ant Colony Optimization, Particle Swarm Optimization, Elephant Herding Optimization, Whale Optimization Algorithm, Grey Wolf Optimization, Shark Smell Optimization algorithms, have had success in enhancing efficiency in route optimization. These are all evaluated in hybrid forms here along with their independent forms to promote optimum traffic flow selection. The algorithms about the solution time, utilization of memory, iterations at which the solution was found as optimum, and iteration time of the best-iteration are implemented by using an artificial highway network that includes 15 nodes and 33 segments. This experimentation clearly illustrates how EHO is tending towards swiftness in finding an optimum at approximately 0.1042 seconds while still consuming minimal memory. The GWO_PSO hybrid algorithm had balanced performance in route optimization, efficiently lowering computation time and memory consumption. The present study shall contribute to further insights into selecting an appropriate algorithm for each optimization goal with regard to ITS by considering system efficiency and the reduction of environmental impacts.

Citation: Ahmed, A. H., Umar¬, S. U. The Optimization of Traffic Routing Systems using A Swarm Intelligence. Central Asian Journal of Medical and Natural Science 2025, 6(3), 904-919.

Received: 27th Mar 2025
Revised: 30th Mar 2025
Accepted: 8th Apr 2025
Published: 17th Apr 2025

Keywords: Intelligent Transportation Systems (ITS), Swarm Intelligence Algorithms, Traffic Flow Optimization, Dynamic Multi-Objective Programming and Hybrid Optimization Techniques

1. Introduction

Due to the possibility of dynamic changing of route depending on traffic data in real time, route optimization has become essential for modern Intelligent Transport Systems (ITS). This system allows passengers to reduce travel time, improve productivity, and reduce the negative impact of vehicle emissions on the environment. of effectively predict traffic conditions and provide valuable guidance to drivers, it is crucial to have accurate and up-to-date traffic information. This helps to alleviate congestion, enhance road capacity, and improve overall travel comfort [1] . The goal of path optimization is to get traffic flowing through road networks better to solve urban congestion. This reduces traffic delays and pollution from vehicle emissions. Real-time dynamic path optimization, which uses data-driven decision-making to improve transportation efficiency, has become the solution to the complexity of traffic problems [2]. (ITS) have become an essential part of urban infrastructure, providing practical answers to traffic issues that hinder mobility and negatively impact human life. A key component of these systems is route optimization, which improves mobility efficiency, reduces travel time, vehicle pollution and energy consumption [3]. Intelligent transportation systems rely on advanced algorithms of artificial intelligence technologies, the most important of which are swarm intelligence technologies such as (ACO, PSO, EHO, WOA, GWO, SSO) to find ideal solutions to traffic problems [4], [5], [6]. Path optimization also addresses multiple objectives, such as reducing travel distance and time, while improving traffic distribution to avoid congestion and ensure safety [7]. There's a real and deep necessity for real-time and accurate road section information for the purpose of efficiently managing traffic and this is then coupled with trip planning and dynamic decision-making. The whole plan of dynamic multi-objective road planning models focuses on the most important parameters of the system, like traffic flow, journey time, and path length. This model calculates comprehensive weight-influencing elements by integrating AHP, entropy theory, and PSO.

In ITS, dynamic path optimization is a very important area of study for route planning by incorporating real-time traffic data into predictive analytics. Algorithm refinement in this respect has advanced considerably in 2018, Wei et al. came up with an updated version of the Dijkstra algorithm with traffic prediction models. This improved the accuracy and computational efficiency under dynamic traffic conditions. Zhang et al. (2020) developed an elaborate A* algorithm that seeks to balance the efficiency and stability of search by incorporating machine learning to adapt to real-time traffic variances. Among the evolutionary algorithms, Sharma et al. have proposed a hybrid Genetic Algorithm and Deep Reinforcement Learning for the optimization of dynamic routes in the year 2021. It has drastically reduced travel time as well as computational cost. Meanwhile, Li et al. (2022) Improved the standard Ant Colony Optimization by incorporating real-time traffic characteristics, such as signal timings and congestion levels, yielding faster convergence to better route choices. Yang et al. (2023) integrated ACO with Swarm Intelligence methods further for solving TSP in dynamic settings. The hybrid methods have also evolved considerably. For path planning with multiple vehicles, Che et al. combined (ACO-PSO) in 2020. This resulted in faster execution times and reduced energy consumption. Wang et al. (2022) consider urban traffic systems and use ACO to develop a WOA-based hierarchical optimization framework that was verified with actual data. Very recently, trajectory planning for autonomous vehicles in real-time based on Reinforcement Learning combined with ACO has been proposed by Zhao et al. (2022), with promising results in high-density traffic conditions. Chen et al. (2023) have presented a hybrid (PSO-GWO) technique that has much lower computational cost and is intended for collision avoidance in autonomous vehicle navigation. These pieces demonstrate the effective integration of artificial intelligence and conventional optimization methods, and the latest advancements in dynamic path optimization. Future research will be needed to provide more understanding of real-time performance, computational economy, and adaptability in the case of more complicated traffic situations.

2. Materials and Methods

2.1 Dynamic Path Mult objective Programming Model

Most route optimization techniques only take into account one criterion, such as choosing the shortest travel distance or minimizing travel time. According to surveys conducted in places such as Munich, Paris, and London, travelers often prefer more than one criterion. For example, when choosing the best route, 71% of travelers in Munich and 42% in London and Paris consider take into account the smallest journey time and distance, while fewer travelers focus on just one factor [8]. This underscores the importance of including a variety of elements in route optimization. A dynamic multi-objective planning model has been developed with three main goals in mind:(1) Path length: To find the road network's shortest route between two places [9],2) Travel time: To guarantee a decrease in the amount of time required to get at the destination [10])3)Traffic flow: To prevent congestion by optimizing traffic distribution throughout the network[11].With 543 valid responses, we conducted a survey using conventional methods and also used QQ and WeChat for the same purpose of understanding these preferences. In making path selection, some of the major variables affecting respondents' decisions were the path length, trip time, traffic flow, road conditions, intersections, and weather. Statistical analysis also revealed that the primary factors influencing path optimization are path length, journey duration, and traffic flow. Such characteristics must be taken into account in this dynamic traffic network [12]. Based on this model, accurate traffic data should be collected in real time and analyzed quickly using a hybrid analytical model of (Analytical Hierarchy Process (AHP), Entropy Theory and PSO) [13]. These strategies help a person in picking the best option by reducing ITS variance in different goals to weighted considerations [14]. These are foreseen to raise transportation efficiency considerably and bring down the environmental impacts of traffic because of technological breakthroughs and because of the rise in need felt for intelligent solutions. [15].

2.1.1 Analytical Hierarchy Process (AHP)

The AHP was founded by Thomas Saaty in the 1970s as an organized way to tackle complicated problems by dividing them into Layers. It permits serious consideration of the factors that influence the solution and the interrelations among these factors [16]. In this study, AHP is used to assess and rank through pairwise comparisons three key variables: path length, travel duration, and traffic flow. AHP provides a rational framework for the systematic evaluation of the importance of aspects relative to one another through pairwise comparisons and weight computations.

2.1.2 Entropy Theory

Entropy is a-value based method for weighting data in multi-criteria Decision-Making. The theory of information defines the measure of disorder or randomness from a set of options, Entropy analyzes the influence of the distribution of values for many indicators on weight., The objectivity lies in the removal of subjective biases in decision-making based on already available information that has been presented objectivity. Indicators that vary greatly are usually weighted heavily, whereas those with similar values show high Entropy and suggest lower merit [17].

2.1.3 Method of Calculating Weight

Combining entropy theory and AHP is a reflexive compromise between objective data analysis and expert opinion. Entropy theory is unbiased as it allows comparison of consistency and variability of the data while AHP depends on domain knowledge to dictate the relative importance of the various elements. This integration improved the robustness and comprehensiveness of weight assignment. 30% of the weights were distributed by entropy theory and 70% by the AHP using α=0.7 as the mixing ratio. The computed weights now serve as by Equation (1)

$$\boldsymbol{W_i = a * W_i^{AHP} + (1 - a) * W_i^{\text{entropy}}} \quad (1)$$

Having optimized weights by minimizing the error through PSO and maximizing their accuracy, the final weight values are: Wfinal= [0.2096,0.2533,0.5370].

Thus, with the distinguishing features from AHP, entropy theory, and PSO, this integration gives birth to a high-performance and highly reliable weight computation scheme.

2.1.4 Measuring The Influencing Elements

The methodology looks into traffic flow, travel time, and the length of the path to ensure that a tariff structure passes these optimizations. The extremum being based on the range from 0 to 1 will ensure that dimensionless comparability exists by Equation (2)

$$\mathbf{X_i' = \frac{X_i}{max(X_i)}} \quad (2)$$

X_i^': normalized value of Xi: with max (Xi): as scaling reference.

Combining expert opinion with data analysis, the Uniformed Intelligence process estimates weights for various impacts. The full weight of each path by Equation (3):

$$S_{ij} = W_1 * D_{ij} + W_2 * T_{ij} + W_3 * N_{ij} \quad (3)$$

Sij: The comprehensive weight-influencing factor for path i and criteria j, Dij: The normalized value of the path length for path i, Tij: The normalized value of the travel time for path i, Nij: The normalized value of the traffic flow for path i, (W1, W2, W3): The respective weights for path length, travel time, and traffic flow.

Using PSO improved these weights, establishing maximum values= [0.2096,0.2533,0.5370]. Built into one, this scheme fuses together PSO optimization, data properties, and expertise info to provide a realistic path optimization frame in a properly balanced manner.

2.2 Algorithms Used

This study improves pathfinding algorithms through the use of artificial intelligence methods with special focus on both mono and hybrid models for better performance of the solution. Several strategies are employed in the study and there is evaluation strategy of efficiency, convergence time and quality of the solution provided. The evaluation of the strategies will be done through results comparison and analysis which will enable an innovative and effective solution for the optimal pathfinding problem in communication and transportation systems.

2.2.1 Ant Colony Optimization (ACO)

An example of a metaheuristic is primarily applied by combinatorial optimization problems and took its inspiration from the foraging habits of natural ant colonies. Artificial ants collaborate to select the best course in a network by simulating how actual ants find the shortest path between their colony and a food source. Each ant drops pheromones according to the quality of its solution, to direct other ants toward promising areas of the search space. This collective behavior eventually converges to near-optimal solutions [18].

Steps of The ACO Algorithm

Initialization: number of ants, pheromone levels, Pheromone influence (Controls pheromone attraction α = 0.7), Distance influence (Attractiveness based on path metrics $\frac{\mathbf{1}}{(\mathbf{w1} * \mathbf{pathLengthNorm} + \mathbf{w2} * \mathbf{travelTimeNorm} + \mathbf{w3} * \mathbf{trafficFlowNorm})}$ Pheromone evaporation rate (Determines pheromone decay ϱ = 0.5).

Solution Construction: Each ant iteratively constructs a path by choosing the next step through the action of pheromone level and attractiveness according to Equation (4)

$$\boldsymbol{P_{ij}} = \frac{\boldsymbol{T_{ij}^{a} * \eta_{ij}^{\beta}}}{\boldsymbol{\sum_{k\epsilon allowed} T_{ij}^{a} * \eta_{ij}^{\beta}}} \qquad (\mathbf{4})$$

P_{ij} = is the probability of moving from one location to another, T_{ij}^{a} = pheromone level between i and j, $\eta_{ij}^{\beta} = \frac{1}{d_{ij}}$ is the path's attractiveness, based on the inverse of the distance between locations.

Update of Pheromone: Once every ant has followed its route, use Equation to update the pheromone (5)

$$\boldsymbol{T_{ij} = (1 - p) * T_{ij} + \sum_{k=1}^{m} \triangle T_{ij}^{k}} \qquad (\mathbf{5})$$

Tij Pheromone level between i and j, p: Pheromone evaporation rate, $\triangle T_{ij}^{k} = \frac{Q}{L_k}$ Pheromone by ant k based on path length Lk.

Iteration: Repeat for 25 iterations or when improvements in the best solution become negligible.

Final Solution: The best path stored as globalBestPosition with its quality measured as globalBestScore is the solution.

2.2.2 Particle Swarm Optimization (PSO)

It contains set of particles , each particle is representing candidate solution in speed and a suitability function to enhance its position. PSO is a simple algorithm with no requirements on derivative information about the optimization function and just requires a few straightforward mathematical operations [19].

Steps of the PSO Algorithm:

Initialization: Represent and distribute the particles randomly and give the initial speeds.

Evaluation of Fitness: The representative and weights of a particle determine its fitness.

Velocity Update: Equation (6) uses the global speed (gBest) and the personal best speed (pBest):

$$V_i = c_2 * r_2 * (gBest - X_i) + \omega * V_i + c_1 * r_1 * (pBest_i - X_i) \quad (6)$$

ω: Inertia weight c_1, c_2: pBest and gBest coefficients, r_1, r_2: Random (0 and 1).

Position Update: Particle positions are adjusted based on their velocities by Equation (7):

$$X_i^{new} = X_i^{old} + V_i \quad (7)$$

X_i^new: Particle i updated position, X_i^old: Its old position, V_i: velocity updated using Equation (6).

Update best values: Update pBest and gBest according to the best values obtained.

Iteration: Repeat for 25 iterations or there is no improvement in the best values.

The Solution: Global best position based on global Best Score.

2.2.3 Elephant Herding Optimization (EHO)

Wang presented a swarm-based metaheuristic in 2015. It was motivated by the way elephants' herd within clans [20], [21]. A synopsis of the herding behavior is as follows:

1. Several female elephants and their offspring form clans, which are smaller groups within the swarms of elephants[20], [21].
2. Each clan is headed by an adult female, known as the matriarch [20], [21].
3. A male calf in a clan leaves the group when it reaches adulthood [20], [21].

Steps of the EHO Algorithm

Initialization: representation and position of each elephant will be done randomly.

Fitness Evaluation: Calculate the fitness score of every elephant in view of the minimum total cost weighted in path length, travel time, and traffic flow.

Clan Grouping: Segregate the elephants into subgroups, usually clans, which cooperatively work towards optimizing the solution.

Position Update: Using Equation (8), update each elephant's location in relation to the best and center of the clan.

$$X_{ij}^{t+1} = X_{best,j}^{t} + a * \left(X_{ij}^{t} - X_{center,j}^{t}\right) * r \quad (8)$$

X_{ij}^t: position of elephant i, $X_{best,j}^t$: Best elephant in the clan, $X_{center,j}^t$: clan center, a: learning rate, r: random value in [0,1].

Migration Separation Factor: Replace the worse solution by random solution in order to maintain the diversity in solution space as indicated in Equation (9).

$$X_{worst} = X_{new\ random} \quad (9)$$

X_{worst} : is the current solution of the worst elephant, i.e., the elephant with the longest tour, $X_{new\ random}$: is a newly generated random solution, created by randomly shuffling the cities to form a new tour.

Global Best Update: Update the tracked best solution.

Iteration: Repeat from step 4 until convergence or maximum 25 iterations.

Final Solution: Store the optimal path as output (Global best position).

2.2.4 Grey Wolf Optimization (GWO)

IBy initializing positions, assessing fitness, and updating according to Alpha, Beta, and Delta, it replicates the hunting and hierarchy of grey wolves. Exploration and exploitation are balanced throughout the process, and after the maximum number of repetitions, Alpha is the best answer [22].

Steps of the GWO Algorithm

Initialization: Define an objective function (e.g., maximizing traffic flow or minimizing distance) after setting a population of wolves as random solutions in the search space.

Evaluate the Wolves: In order to rank the wolves (Alpha (α): best solution, Beta (β): second-best, Delta (δ): third-best, and the remaining wolves are Omega (ω)), evaluate each wolf's fitness using scorei=f(xi) (objective function value).

Update Positions: Update wolves' positions based on Alpha, Beta, and Delta by Equation (8,9,10,11,12,13):

Distance to Alpha: Da= ∣C1 * Xa – Xi(t)∣ (8)
Distance to Beta: Dβ = ∣C2 * Xβ – Xi(t)∣ (9)
Distance to Delta: Dδ = ∣C3 * Xδ – Xi(t)∣ (10)
X1=X_a-A1* 〖D 〗 _a (11)
``aX2= 〖X 〗 _β-A2* 〖D 〗 _β (12)
X3= X_δ -A3* 〖D 〗 _δ (13)

A_1, A_2, A_3 are random factors calculated $A = 2 * a * rand\ () - a$

The updated position by Equation 14:

$$X_i(t+1) = \frac{X_1 + X_2 + X_3}{3} \qquad (14)$$

Exploration and Exploitation Balance: Balance controlled by a, Where, over iterations, drops linearly from 2 to 0. calculated by $a = 2 - \frac{2 * t}{maxIter}$ where t is the current iteration

Final Solution: After completing the iterations, the solution provided by the Alpha wolf is considered the best solution: optimal Solution=Xα

Iteration: Repeat until convergence or reaching maximum iterations (e.g., 25).

Final Solution: The Alpha wolf's position $X\alpha$ represents the best solution.

2.2.5 Whale Optimization Algorithm (WOA)

Was first presented in 2016 by Andrew Lewis and Seyedali Mirjalili, and it Inspired by natural hunting habits of whales [4]. Whale hunting characteristics like spiral update, encircling prey, and random search are the basis for the algorithm.

Steps of the WOA Algorithm

Initialization: Random assignment of the whales' positions to the solutions of the search space.

Whales Fitness Evaluation: Assess the whales for their fitness with respect to the objective function.

Update Encircling Prey: Whales converge to the optimum with by equation 15:

$$X_{s_d}(t+1) = X_{b_d}(t) - A * D(15)$$

$X_{s_d}(t+1)$: the whale's position at iteration t+1 and $X_{b_d}(t)$: he best whale's position, A controls movement, $D = |C * X_{b_d}(t) - X_{s_d}(t)|$ with C as random factor between 0 and 2.

Spiral Movement: If close to the optimal solution, whales spiral around it by equation 16:

$$X_{s_d}(t+1) = X_{b_d}(t) + D_b * exp(\beta l) * \cos(2\pi l) \qquad (16)$$

D_b: distance to the best solution, β defines the spiral, l: random number between -1 and 1.

Random Search: Whales randomly explore using equation 17:

$$X_{s_d}(t+1) = X_{rand_d}(t) - A * D \qquad (17)$$

$X_{rand_d}(t)$: random whale's position, D is the distance to it.

Iteration: Repeat until convergence or maximum iterations are reached (25).

Final Solution: The position of the best whale represents the global optimum solution

2.2.6 Shark Smell Optimization (SSO):

Is based on the natural foraging habits of sharks. Underlying idea of SSO is motivated by the tracking and sniffing methods employed by Sharks to reach their prey. One variety of swarm intelligence-based algorithms, SSO has been applied to find the global optimization problems effectively [7],Important Actions 1-Sharks randomly sniff to find stronger scents and better solutions.2) A shark tracks by swimming towards increasingly powerful odors in order to home in on that food. 3) These sharks, when attacking, precisely position themselves close to the prey.

Steps of the SSO Algorithm

Initialization: number of sharks(N), positions (Xi) and the objective function (F(Xi))

Sniffing (Exploration): Sharks explore randomly by equation 18:

$$X_i(t+1) = X_i(t) + a * rand().d \qquad (18)$$

α= Exploration rate and d: Direction of random movement.
Tracking: Sharks move toward the best solution by following equation 19:
$$X_i(t+1) = X_i(t) + \beta * \left(X_{best} - x_i(t)\right) \quad (19)$$ β: Tracking rate.
Attacking (Fine Adjustment): Sharks refine their position by equation 20:
$$X_i(t+1) = X_i(t) + \gamma \, * \, \nabla F(X_i(t)) \quad (20)$$ γ: Attack rate, ∇F(X_i (t)): gradient at the current position.
Iteration: Repeat steps until convergence or maximum iterations (25).
Final Solution: Store the global best position as the optimal solution.

2.3 Hybrid Optimization Models

Hybrid optimization models address individual limits by incorporating the strengths of various methods. They combine exploration and exploitation skills to yield powerful responses to tough problems. Their application domains include multi-objective problems, energy management, and traffic planning. For example, EHO [20], [23], pays more attention to the refinement of the local search, while GWO [24], has performed well in global search due to its hierarchical structure and hunting strategy. As shown from Table 1, the hybrid of both techniques' amends both the local optimization and global exploration: GWO_EHO.

Table 1. Weaknesses and Limitations of Algorithms.

Algorithm	Weaknesses and Limitations
ACO	Rapid pheromone evaporation causes slow convergence and decreased performance in large-scale issues [25], [26].
PSO	Complex multimodal problems with premature convergence that is extremely sensitive to parameter tweaking[2], [27]
EHO	Restricted investigation; in high-dimensional spaces, prone to local optima [21], [28]
WOA	exploitation is difficult to balance, and performance suffers in dynamic situations [4]
GWO	Later iterations will have less diversity, there is a chance of premature convergence [24].
SSO	High computational cost in vast search areas; vulnerable to local optima during deterministic exploitation[29], [30]

2.3.1 GWO_EHO

In order to preserve the equilibrium between exploration and exploitation, a mix of EHO's improved exploration and GWO's collective leadership. This enhances diversity, decreases premature convergence, and improves the quality of solutions for high-dimensional, difficult problems.
Integration of GWO and EHO:

1. Division Clans (EHO): Based on social behavior, agents are grouped into clans for local optimization. There are a certain number of agents in each clan by equation 21: agentsPerClan =numAgents/numClans (21)
2. Updating Positions Using GWO As described in Section 2.2.4 positions are updated using influence factors (A and C) in addition to alpha, beta, and delta agents.
3. Updating Positions Using EHO: Equation 22 states that the position of the clan center is established by averaging the positions of the agents within the clan, and that each agent's location is updated based on the center and a random impact factor

$$\mathbf{ClanCenter} = \frac{\sum_{i=start}^{end} \mathbf{positions(i)}}{\boldsymbol{agentsPerClan}}$$

$$\text{NewPositionEHO} = \begin{cases} \text{ClanCenter} + r * (\text{ClanCenter} - \text{positions(i)}) & \text{if } r < 0.5 \\ \text{ClanCenter} - r * (\text{ClanCenter} - \text{positions(i)}) & \text{if } r > 0.5 \end{cases} \quad (22)$$

4. Combining Updates from GWO and EHO: Each agent's ultimate position is determined by averaging the positions determined by (GWO_EHO). This guarantees that exploration (GWO) and exploitation (EHO) are balanced by equation 23.

$$\mathbf{NewPosition} = \frac{\mathbf{NewPositionGWO + NewPositionEHO}}{\mathbf{2}} \quad (\mathbf{23})$$

1) **Initialize:** Define (numAgents, numClans, maxIterations, lowerBound, upperBound) and randomly initialize agent positions within boundaries.
2) **Divide agents into clans:** Calculate agentsPerClan: using equation (21)
3) **Evaluate initial fitness:** Identify Alpha, Beta, and Delta leaders.
4) **Main loop (t = 1 to maxIterations):**
   - For each agent "Update positions using GWO": Compute new position influenced by Alpha, Beta using equation (14), and Delta.Store as NewPositionGWO.
   - For each clan "Update positions using EHO": Calculate clan center as the mean of agent positions using equation (22), Update positions relative to clan center with random influence and Store as NewPositionEHO
5) **For each agent "Combine GWO and EHO updates":** Calculate New Position using equation (23) and Ensure New Position is within boundaries
6) **Evaluate fitness:** Update Alpha, Beta, Delta leaders if better solutions are found.
7) **Return:** Best solution (Alpha position) and corresponding fitness value.

2.3.2GWO_PSO

PSO and GWO are combined to strike a balance between exploration and exploitation. Swarm intelligence improves convergence in PSO, whereas in GWO, convergence depends on social hierarchy for guiding the solution. This hybrid approach avoids early convergence and provides more reliable and accurate solutions to complex problems.

Integration of GWO and PSO:

Updating Positions Using GWO As described in Section 2.2.4 positions are updated using influence factors (A and C) in addition to alpha, beta, and delta agents.

Updating Positions using PSO (Particle Update): are performed based on the methodology described in Section 2.2.2.

Merging Updates between GWO and PSO: After updating positions using both GWO and PSO, the improved positions are merged by calculating the average of the positions obtained from both algorithms using the following equation 24

$$\mathbf{NewPosition} = \frac{\mathbf{NewPosition}_{GWO} + \mathbf{NewPosition}_{PSO}}{\mathbf{2}} \quad (\mathbf{24})$$

1) **Initialize**: Define (num_particles, num_iterations, w, c1, c2, a, lowerBound, upperBound), Randomly initialize particle positions and velocities within boundaries and evaluate initial fitness: Identify Alpha, Beta, and Delta leaders.
2) **Main loop (t = 1 to maxIterations):**
   - For each particle "Update positions using GWO": Compute new position influenced by Alpha, Beta using equation (14), and Delta.Store as $NewPosition_{GWO}$.
   - For each particle "Update positions using PSO": are performed based on the methodology described in Section 2.2.2.and Store as $NewPosition_{PSO}$
3) **For each particle "Combine GWO and PSO updates":** Calculate New Position using equation (24) and Ensure New Position is within boundaries
4) **Evaluate fitness**: Compute fitness for the updated position, Update pBest and gBest for PSO if a better solution is found. Update Alpha, Beta, Delta leaders if better solutions are found.
5) **Return:** Best solution (Alpha position) and corresponding fitness value.

2.3.3 SSO_GWO

The integration of SSO and GWO combines the benefits of both algorithms. Considering that GWO drives the search through a social hierarchy to refine locally, SSO makes sure of powerful global exploration. This hybrid, therefore, balances both exploration and exploitation in complicated dynamic problems such as traffic flow optimization. It minimizes premature convergence while improving the precision of solutions

Integration of GWO and SSO:

Updating Positions using SSO): are performed based on the methodology described in Section 2.2.6.

Updating Positions Using GWO As described in Section 2.2.4 positions are updated using influence factors (A and C) in addition to alpha, beta, and delta agents.

Merging Updates between GWO and SSO: After updating positions using both GWO and SSO, the improved positions are merged by calculating the average of the positions obtained from both algorithms using the following equation 25

$$\mathbf{NewPosition} = \frac{\mathbf{NewPosition}_{GWO} + \mathbf{NewPosition}_{SSO}}{\mathbf{2}} \quad (\mathbf{25})$$

Initialize: Define (numSharks, numWolves, maxIter, numLocations), Randomly initialize shark positions and wolf positions within boundaries and evaluate initial fitness: Identify Alpha, Beta, and Delta leaders.

1) **Main loop (t = 1 to maxIterations):**
   - For each shark "Update positions using SSO": are performed based on the methodology described in Section 2.2.6.and Store as $NewPosition_{SSO}$
   - For each wolf "Update positions using GWO": Compute new position influenced by Alpha, Beta using equation (14), and Delta.Store as $NewPosition_{GWO}$.
2) **For each shark (or wolf) "Combine GWO and SSO updates"**: Calculate New Position using equation (25) and Ensure New Position is within boundaries
3) **Evaluate fitness**: Compute fitness for the updated position, Update the best positions (pBest, gBest) and Alpha, Beta, Delta leaders if found solution.
4) **Return:** Alpha position and corresponding fitness value.

2.3.4 WOA_GWO

Hybrid WOA/GWO benefits from both methods: while spiral encirclement in WOA favors exploration, the social hierarchy in GWO favors maximization of exploitation. Such an approach will provide more accurate and reliable solutions to complicated, multidimensional optimization problems without early convergence.

Integration of GWO and WOA:

Updating Positions using WOA): are performed based on the methodology described in Section 2.2.5.

Updating Positions Using GWO As described in Section 2.2.4 positions are updated using influence factors (A and C) in addition to alpha, beta, and delta agents.

Merging Updates between GWO and WOA: Positions are updated using a probabilistic mechanism based on a random factor p:

1. If $p<0.5$ the WOA spiral update is applied.
2. Otherwise, the GWO hierarchical guidance is used.

1) **Initialize:** Define (numAgents, maxIter, numLocations, a, b, etc.), randomly initialize positions agents (representing whales and wolves) within boundaries and evaluate initial fitness: Identify Alpha, Beta, and Delta leaders.
2) **Main loop (t = 1 to maxIterations):**
   - For each agent "Update positions using WOA": are performed based on the methodology described in Section 2.2.5.and Store as $NewPosition_{WOA}$
   - For each agent "Update positions using GWO": Compute new position influenced by Alpha, Beta using equation (14), and Delta.Store as $NewPosition_{GWO}$.
3) For each agent "**Combine GWO and PSO updates**": Combine the positions from WOA and GWO updates probabilistically using

$$\text{NewPosition} = \begin{cases} \text{NewPositionWOA}, & \text{IF } P < 0.5 \\ \text{NewPositionGWO}, & \text{otherwise} \end{cases}$$

4) **Evaluate fitness**: Compute fitness for the updated position, Update the best positions (pBest, gBest) and Alpha, Beta, Delta leaders if a better solution is found, Store the best scores and update convergence metrics if required.
5) **Return:** Best solution (Alpha position) and corresponding fitness value.

2.3.5 SSO_PSO

SSO balances exploration and exploitation by the combination of SSO and PSO. SSO investigates many regions that can promise an accurate and reliable solution for the traffic flow optimization problem, while PSO improves the solutions to accelerate the convergence.

Integration of SSO and PSO:

Updating Positions using SSO: are carried out using the approach outlined in Section 2.2.6.

Updating Positions using PSO (Particle Update): are carried out using the approach outlined in Section 2.2.2.

Merging Updates between SSO and PSO: Merge updates between PSO and SSO: After updating positions using both PSO and SSO, the optimized positions are merged using the following equation 26

$$\mathbf{newPosition} = \mathbf{alpha} * \mathbf{position}_{\mathbf{SSO}} + (\mathbf{1} - \mathbf{alpha}) * \mathbf{position}_{\mathbf{PSO}} \quad (\mathbf{26})$$

1) **Initialize:** Define (numSharks, num_particles, num_iterations, w, c1, c2, maxIter, numLocations) and randomly initialize shark positions and particle positions and velocities within boundaries
2) **Main loop (t = 1 to maxIterations):**
   - For each shark "Update positions using SSO": are carried out using the approach outlined in Section 2.2.6.and Store as $NewPosition_{SSO}$
   - For each particle "Update positions using PSO": are carried out using the approach outlined in Section 2.2.2.and Store as $NewPosition_{PSO}$
3) F**or each shark (or particle) "Combine GWO and PSO updates**": Calculate New Position using equation (26) and Ensure New Position is within boundaries
4) **Compute fitness for the updated position, Update the best positions (pBest, gBest) "Evaluate fitness"**
5) **Return:** Best solution (best position) and corresponding fitness value.

2.3.6 SSO_WOA

It strikes Finding equilibrium between exploration and exploitation. WOA quickly discovers optimal solutions, while SSO will explore different areas, thus resulting in accurate and reliable traffic flow optimization.

Integration of SSO and WOA:

Updating Positions using SSO: are carried out using the approach outlined in Section 2.2.6.

Updating Positions using WOA: are carried out using the approach outlined in Section 2.2.5.

Merging Updates between SSO and PSO: Merge updates between PSO and SSO: After updating positions using both PSO and SSO, the optimized positions are merged using the following equation 27

$$\mathbf{newPosition} = \mathbf{alpha} * \mathbf{position}_{\mathbf{SSO}} + (\mathbf{1} - \mathbf{alpha}) * \mathbf{position}_{\mathbf{WOA}} \quad (\mathbf{27})$$

1) **Initialize:** Define (numAgents, maxIter, numLocations) and Randomly initialize Agents positions within boundaries
2) **Main loop (t = 1 to maxIterations):**
   - For each agent "Update positions using SSO": are carried out using the approach outlined in Section 2.2.6.and Store as $\text{NewPosition}_{\text{SSO}}$
   - For each agent "Update positions using WOA": are carried out using the approach outlined in Section 2.2.5.and Store as $\text{NewPosition}_{\text{WOA}}$
3) **For each shark (or particle) "Combine GWO and PSO updates":** Calculate New Position using equation (27) and Ensure New Position is within boundaries
4) **Compute fitness for the updated position "Evaluate fitness"**
5) **Return:** Best solution (best position) and corresponding fitness value.

2.3.7 WOA_PSO

In contrast, the hybridization of PSO and WOA balances exploration, involving the discovery of new sections of the solution space, and exploitation concerning quicker refinement to solution. Based on incorporating the advantages in both PSO and WOA, the optimization of traffic flow improves and ensures the acquisition of more precise and trustworthy findings.

Integration of WOA and PSO:

Updating Positions using WOA: are carried out using the approach outlined in Section 2.2.5.

Updating Positions using PSO (Particle Update): are carried out using the approach outlined in Section 2.2.2.

Merging Updates between WOA and PSO: Merge updates between PSO and WOA: After updating positions using both PSO and WOA, the optimized positions are merged using the following equation 28

$$\text{newPosition} = \text{alpha} * \text{position}_{\text{WOA}} + (1 - \text{alpha}) * \text{position}_{\text{PSO}} \quad (28)$$

1) **Initialize:** Define (num_agent, num_particles, num_iterations, w, c1, c2, maxIter, numLocations) and randomly initialize agent positions and particle positions and velocities within boundaries
2) **Main loop (t = 1 to maxIterations):**
   - **For each particle "Update positions using PSO":** are carried out using the approach outlined in Section 2.2.2.and Store as $NewPosition_{PSO}$
   - **For each agent "Update positions using WOA":** are carried out using the approach outlined in Section 2.2.5.and Store as $NewPosition_{WOA}$
3) **For each agent (or particle) "Combine WOA and PSO updates":** Calculate New Position using equation (28) and Ensure New Position is within boundaries
4) **Compute fitness for the updated position: "Evaluate fitness"**
5) **Return:** Best solution (best position) and corresponding fitness value.

3. Results

The Using all the algorithms mentioned in the above section, the path optimization of instance network and the simulated road network is performed.

3.1 Road Network Simulated:

3.1.1 Data from Road Networks

The test case applied a road network consisting of 33 segments and 15 nodes, represented by Figure 1. It can be considered that the source node is number 1 (the origin point-O) and the destination number 15-D, and there needs to determine an optimum path since traffic flow cannot always be at saturation point, assumed by normal distribution arrivals. The Depth-First Search (DFS) technique was used to find 9,161 distinct routes. Road characteristics, including segment lengths, travel durations, traffic flow rates and classifications (1 = intra-urban, 2 = main roads, and 3 = highways), are shown in Table 2. Consistent evaluation across algorithms is ensured via data normalization.

Table 2. Road Network Attributes.

From	To	Road Type	Distance	Time	Traffic
1	2	3	92.943	1.032	227
1	4	2	45.544	0.911	889
1	3	1	23.680	1.185	758
2	7	3	92.943	1.032	227
2	5	2	45.544	0.911	889
2	3	2	45.544	0.911	889
3	5	1	23.680	1.185	758
3	6	1	23.680	1.185	758
3	4	2	45.544	0.911	889
4	6	1	23.680	1.185	758
4	13	3	92.943	1.032	227
5	6	1	23.680	1.185	758
5	7	2	45.544	0.911	889
5	8	1	23.680	1.185	758
5	9	1	23.680	1.185	758
6	9	2	45.544	0.911	889
6	13	2	45.544	0.911	889

7	8	2	45.544	0.911	889
7	10	2	45.544	0.911	889
8	9	1	23.680	1.185	758
8	10	2	45.544	0.911	889
8	11	1	23.680	1.185	758
8	12	1	23.680	1.185	758
9	12	2	45.544	0.911	889
9	13	2	45.544	0.911	889
9	14	2	45.544	0.911	889
10	11	3	92.943	1.032	227
11	12	1	23.680	1.185	758
11	15	2	45.544	0.911	889
12	14	2	45.544	0.911	889
12	15	3	92.943	1.032	227
13	14	3	92.943	1.032	227
14	15	3	92.943	1.032	227

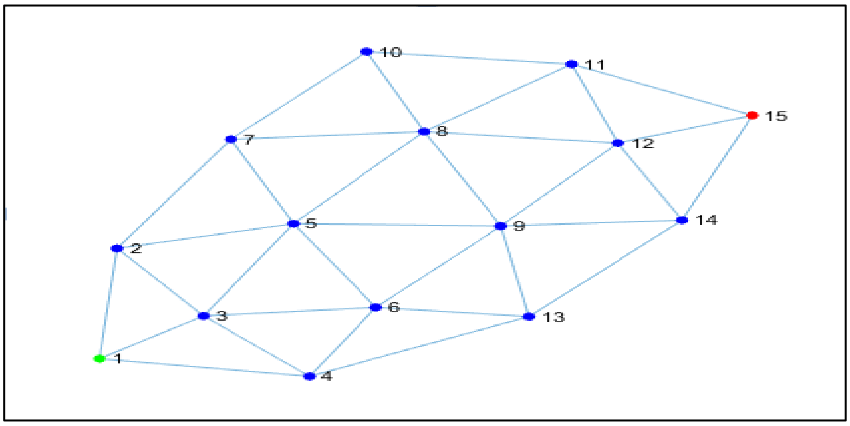

Figure 1. Road Network Diagram.

3.1.2 Parameter Settings

The parameters of the proposed standalone and hybrid algorithms are optimized to fit the study's traffic flow optimization requirements and are outlined as follows:

General:

a. Iterations (maxiter): capped to 25 for fast convergence.

b. Population (pop size): Set to 9,161, equal to the total unique paths, ensuring full solution space coverage.

c. Early Stopping: When result changes drop below $1{\times}10^{-6}$, trigger the operation.

Specific for Algorithms

1. ACO: alpha = 0.7, evaporationRate = 0.5, `pheromoneEnhancement = 100,` visibility = $\frac{1}{(w1 * pathLengthNorm + w2 * travelTimeNorm + w3 * trafficFlowNorm)}$.
2. PSO: w_{min} = 0.4, w_{max} = 0.9, w = w_{max} - ((w_{max} - w_{min}) / maxIter) * itr, c1 = 2 - itr * (0.5 / maxIter), c2 = 2 - itr * (0.5 / maxIter); r1, r2 = rand ();
3. EHO: nClans = 5, nci = floor (nElephants / nClans), alpha = 0.5, beta = 0.25;
4. GWO: a = 2, alphaScore = inf, alphaPosition = 0, betaScore = inf, betaPosition = 0, deltaScore = inf, deltaPosition = 0;
5. WOA: a = 2;

Traffic-Specific: This dataset contains 33 road segments and 15 nodes, generated to minimize travel time and congestion. In all the algorithms, the normal distribution of vehicle arrivals is guiding the changes in the fitness function.

3.2 Performance Analysis of Algorithms for Optimal Path Selection:

This section, the outcomes attained after implementing the suggested and hybrid algorithms will be analyzed based on 4 criteria: (total execution time, memory consumption, number of iterations required to reach the optimal solution, and execution time of the iteration in which the best solution was discovered), as shown in Table 3.

Table 3. Performance of Algorithms.

Algorithms	Execution Time(sec)	Memory Used (KB)	Time of itr(sec)	itr
PSO	2.1073	4139.105469	0.2481144	3
GWO	2.3308	4751.988281	0.1947889	2
SSO	2.3558	4759.371094	0.4114885	4
WOA	2.4549	4787.203125	0.2137896	2
EHO	2.5816	4175.871094	0.1042191	1
GWO_PSO	2.3916	4729.402344	0.1316194	1
SSO_WOA	2.9738	3990.191406	0.3130373	2
WOA_GWO	2.6187	4830.679688	0.2247135	2
SSO_PSO	2.4390	3896.160156	0.3922216	3
WOA_PSO	2.9979	4007.574219	0.2546612	2
GWO_EHO	2.6549	4546.804688	0.2438471	2
SSO_GWO	2.83445	4613.742188	0.9261023	7
ACO	128.9039	4019.707031	1.131649	1

4. Discussion

After observing the performance of all the proposed single and hybrid algorithms in this study, each criterion will be analyzed separately and we will explain each criterion in the following points:

1. Total execution time: It was found that the single algorithm (PSO in 2.1073 seconds) and the hybrid algorithm (GWO_PSO in 2.3916 seconds) achieved the fastest performance, while (ACO in 128.9039 seconds) is the worst proposed algorithm.
2. Memory consumption: It was found that the single (ACO in 4019.707031 KB) and hybrid (SSO_PSO in 3896.160156 KB) algorithms achieved the highest memory savings, while (WOA_GWO in 4830.679688 KB) achieved the least memory savings.
3. And execution time of the iteration in which the best solution was discovered: We found that the single algorithms (EHO in 0.1042191 seconds) and the hybrid (GWO_PSO in 0.1316194 seconds) achieved the lowest time while (ACO in 1.131649 seconds) achieved the highest time.
4. Number of iterations needed to arrive at the best answer: It found that algorithms (EHO, GWO_PSO, and ACO) achieved the best path through one iteration, while (SSO_GWO) achieved it through seven iterations.

The product of all the criteria determines which algorithm is most suited to tackle the problem., where individual (EHO = 1123.526589) and hybrid (GWO_PSO = 11488.725796) are found to be the best choice, and the worst is (ACO = 86370.621).

5. Conclusion

The following can be understood by research and analysis: EHO is the best self-executing algorithm and got the optimal solution in a record time of 0.1042 seconds. It consumes a moderate amount of memory: 4175.871 KB and uses one iteration only, thus making the problem divisible into a sum of a very small one, with an outcome giving the best result. Among hybrid algorithms, GWO_PSO performed with the best balance; it reached the optimal solution within 2.3916 seconds, memory consumption of 4729.402344 KB, while the time it took to find the solution was 0.1316194 seconds, and after just one iteration, where (PSO) convergence improvement is based on swarm intelligence, whereas in (GWO) the convergence is dependent on social hierarchy that guides the solution. It also explored the application of a multi-objective dynamic programming model for path optimization, with Traffic flow, travel duration, and segment length are transformed into weighted complete components utilizing a hybrid model that combines AHP, Entropy, and PSO. Simulations were used to confirm the algorithm's efficacy. on typical road network. A comparison of the optimum travel times of the route showed the accuracy of the solutions obtained. Because of time and availability of real data, simulation data were used to validate the model. To increase the model's credibility for future research, the author intends to confirm the model's correctness using actual data.

REFERENCES

[1] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996, doi: 10.1109/3477.484436.

[2] J. Kennedy, R. Eberhart, and bls gov, "Particle Swarm Optimization."

[3] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015, doi: 10.1016/j.advengsoft.2015.01.010.

[4] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, May 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[5] T. A. Rashid *et al.*, "An Improved BAT Algorithm for Solving Job Scheduling Problems in Hotels and Restaurants," in *Artificial Intelligence: Theory and Applications*, E. Pap, Ed., Cham: Springer International Publishing, 2021, pp. 155–171. doi: 10.1007/978-3-030-72711-6_9.

[6] T. A. R. A. M. A. B. A. H. M. R. B. Shahla U. Umar, "Modified Bat Algorithm: A Newly Proposed Approach for Solving Complex and Real-World Problems," Jul. 2024.

[7] I. M. Nasir, M. Raza, J. H. Shah, M. A. Khan, Y. C. Nam, and Y. Nam, "Improved Shark Smell Optimization Algorithm for Human Action Recognition," *Computers, Materials and Continua*, vol. 76, no. 3, pp. 2667–2684, 2023, doi: 10.32604/cmc.2023.035214.

[8] G. K. H. Pang, K. Takahashi, T. Yokota, and H. Takenaga, "Adaptive route selection for dynamic route guidance system based on fuzzy-neural approaches," *IEEE Trans Veh Technol*, vol. 48, no. 6, pp. 2028–2041, 1999, doi: 10.1109/25.806795.

[9] A. Ermagun and D. Levinson, "Spatiotemporal short-term traffic forecasting using the network weight matrix and systematic detrending," *Transp Res Part C Emerg Technol*, vol. 104, pp. 38–52, Jul. 2019, doi: 10.1016/j.trc.2019.04.014.

[10] J. Teng, T. Chen, and W. "David" Fan, "Integrated Approach to Vehicle Scheduling and Bus Timetabling for an Electric Bus Line," *J Transp Eng A Syst*, vol. 146, no. 2, Feb. 2020, doi: 10.1061/jtepbs.0000306.

[11] B. Chen, R. Zhang, S. Long, and R. Sakdanuphab, "A Multi-Objective Multi-Period Low-Carbon Location-Routing Problem: Improved NSGA-II Approach," *IEEE Access*, vol. 12, pp. 51590–51605, 2024, doi: 10.1109/ACCESS.2024.3386584.

[12] J. Cheng, "Dynamic Path Optimization Based on Improved Ant Colony Algorithm," *J Adv Transp*, vol. 2023, 2023, doi: 10.1155/2023/7651100.

[13] S. Dikshit, A. Atiq, M. Shahid, V. Dwivedi, and A. Thusu, "The Use of Artificial Intelligence to Optimize the Routing of Vehicles and Reduce Traffic Congestion in Urban Areas," *EAI Endorsed Transactions on Energy Web*, vol. 10, pp. 1–13, 2023, doi: 10.4108/EW.4613.

[14] S. Li and H. S. Yoon, "Enhancing Camera Calibration for Traffic Surveillance with an Integrated Approach of Genetic Algorithm and Particle Swarm Optimization," *Sensors*, vol. 24, no. 5, Mar. 2024, doi: 10.3390/s24051456.

[15] N. K. Thakre, D. Nimma, A. V Turukmane, A. K. Singh, D. Rohatgi, and B. Bangaru, "Dynamic Path Planning for Autonomous Robots in Forest Fire Scenarios Using Hybrid Deep Reinforcement Learning and Particle Swarm Optimization.," *International Journal of Advanced Computer Science & Applications*, vol. 15, no. 9, 2024.

[16] M. Yu, G. Yue, Z. Lu, and X. Pang, "Logistics Terminal Distribution Mode and Path Optimization Based on Ant Colony Algorithm," *Wirel Pers Commun*, vol. 102, no. 4, pp. 2969–2985, Oct. 2018, doi: 10.1007/s11277-018-5319-z.

[17] H. Yu, Z. Yu, and X. Zhang, “Condition Information Entropy and Rough Set Method Based on Particle Swarm Optimization Applied in the Natural Quality Evaluation of Cultivated Land,” *Sustainability (Switzerland)* , vol. 16, no. 8, Apr. 2024, doi: 10.3390/su16083484.

[18] I. M. Anwar, K. M. Salama, and A. M. Abdelbar, “Instance Selection with Ant Colony Optimization,” in *Procedia Computer Science*, Elsevier B.V., 2015, pp. 248–256. doi: 10.1016/j.procs.2015.07.301.

[19] Y. Yang, Y. Deng, B. Xiao, and X. Zhao, “The Method to Integrate Species Explode and Deracinate Algorithm with Particle Swarm Optimization Algorithm,” *IEEE Access*, vol. 12, pp. 52439–52451, 2024, doi: 10.1109/ACCESS.2024.3387308.

[20] S. M. Almufti, R. Boya Marqas, and R. R. Asaad, “Comparative study between elephant herding optimization (EHO) and U-turning ant colony optimization (U-TACO) in solving symmetric traveling salesman problem (STSP),” 2019.

[21] G.-G. Wang, S. Deb, X.-Z. Gao, and L. D. S. Coelho, “A new metaheuristic optimisation algorithm motivated by elephant herding behaviour,” *International Journal of Bio-Inspired Computation*, vol. 8, no. 6, pp. 394–409, 2016.

[22] M. Shehab, R. Taherdangkoo, and C. Butscher, “Towards Reliable Barrier Systems: A Constrained XGBoost Model Coupled with Gray Wolf Optimization for Maximum Swelling Pressure of Bentonite,” *Comput Geotech*, vol. 168, Apr. 2024, doi: 10.1016/j.compgeo.2024.106132.

[23] J. Li, H. Lei, A. H. Alavi, and G. G. Wang, “Elephant herding optimization: Variants, hybrids, and applications,” *Mathematics*, vol. 8, no. 9, Sep. 2020, doi: 10.3390/MATH8091415.

[24] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey Wolf Optimizer,” *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[25] M. Dorigo and T. Stützle, “Ant Colony Optimization,” 2004.

[26] K. Socha and M. Dorigo, “Ant colony optimization for continuous domains,” *Eur J Oper Res*, vol. 185, no. 3, pp. 1155–1173, Mar. 2008, doi: 10.1016/j.ejor.2006.06.046.

[27] Y. Shi and R. Eberhart, “A Modified Particle Swarm Optimizer.”

[28] M. F. El-Naggar, M. I. Mosaad, H. M. Hasanien, T. A. AbdulFattah, and A. F. Bendary, “Elephant herding algorithm-based optimal PI controller for LVRT enhancement of wind energy conversion systems,” *Ain Shams Engineering Journal*, vol. 12, no. 1, pp. 599–608, Mar. 2021, doi: 10.1016/j.asej.2020.07.013.

[29] I. M. Nasir, M. Raza, J. H. Shah, M. A. Khan, Y. C. Nam, and Y. Nam, “Improved Shark Smell Optimization Algorithm for Human Action Recognition,” *Computers, Materials and Continua*, vol. 76, no. 3, pp. 2667–2684, 2023, doi: 10.32604/cmc.2023.035214.

[30] A. Ghasemi-Marzbali, “A novel nature-inspired meta-heuristic algorithm for optimization: bear smell search algorithm,” *Soft comput*, vol. 24, no. 17, pp. 13003–13035, 2020.