**FALCON OPTIMIZATION ALGORITHM TO TACKLE THE TSP**

**Authors.** AACHABI Mohammed

**Department of Computer Science, Faculty of Sciences and Technologies, Tangier, Morocco.**

***Abstract.* The Traveling Salesman Problem (TSP) is a classic combinatorial optimization problem that seeks to find the shortest possible route that visits a set of cities and returns to the starting city. In this project, we apply the Falcon Optimization algorithm to tackle the TSP. Falcon Optimization is a nature-inspired metaheuristic algorithm based on the hunting behaviour of falcons. It involves the exploration and exploitation of solutions through local search, aiming to find an optimal or near-optimal solution. Our implementation of Falcon Optimization involves initializing a population of solutions represented as tours of cities. These solutions are iteratively improved through local search operations, such as swapping cities, to minimize the total distance travelled. We evaluate the performance of the algorithm on benchmark instances of the TSP, considering factors such as convergence rate, solution quality, and computational efficiency.**

***Keywords:*** *TSP . FAO. Metaheuristics . optimization.*

1. INTRODUCTION

The Traveling Salesman Problem (TSP) stands as one of the most enduring and well-known combinatorial optimization challenges in the realms of computer science, operations research, and mathematics. Its widespread prevalence stems from its simplicity in formulation yet its profound complexity in finding an optimal solution.

The TSP involves determining the shortest possible route that visits a set of cities exactly once and returns to the origin city. Despite its seemingly straightforward goal, the TSP's NP-hard nature renders it computationally challenging to solve optimally, particularly as the number of cities increases. The primary objective of the TSP is to find the optimal tour, also known as the Hamiltonian cycle, that minimizes the total distance traveled. This entails identifying a sequence of cities to visit such that the total distance covered is minimized while ensuring that each city is visited exactly once before returning to the starting city.

In recent years, the exploration of novel optimization techniques has spurred the development of new approaches to tackle the TSP effectively. One such approach is Falcon Optimization Algorithm (FOA), a relatively new population-based optimization technique inspired by the hunting behavior of falcons. FOA embodies the predatory instincts of falcons, balancing exploration and exploitation to efficiently search for optimal solutions within a solution space. This nature-inspired algorithm has shown promise in various optimization domains, offering a fresh perspective and potential for addressing complex combinatorial optimization problems like the TSP.

In this study, we delve into the application of FOA to the TSP, aiming to harness its inherent capabilities for effectively navigating the solution space and uncovering high-quality solutions. By integrating FOA's adaptive and

dynamic search mechanisms, we seek to enhance the efficiency and efficacy of solving the TSP, contributing to the ongoing quest for innovative optimization techniques in addressing real-world challenges.

1. TRAVELLING SALESMAN PROBLEM

TSP focus on reducing cost of building construction engineering and also reduces material wastages, through its principals of finding the minimum cost path of the salesman. Due to complexity in the building construction and uncertainty due to various factors, a well-designed mechanism is needed to tackle this problem [7]. The best option is to adopt TSP that is based on heuristic that emulate the human activities. TSP major on the aspect of a salesman and a set of cities. The salesman has to go or visit all the cities starting from one and return to back to the original city. The biggest challenge is how the salesman will minimize the aggregate travelling cost to visit all the cities [7]. The form definition of TSP is described as follows

T SP = { (G, f, t): G = (V, E) a complete graph,

f is a function V × V → Z,

t ∈ Z,

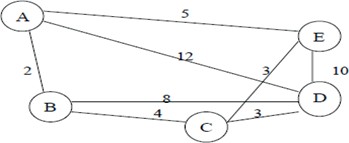
The problem is how to find a minimal route passing from all the nodes. for example, if you take path one from

Fig. 1. A Graph with Weights on Its Edges

{*A, B, C, D, EA*} and the path two that is {*A, B, C, E, D, A*} you have passed all the cities, but path one has a sum of twenty-four and path two has a sum of thirty-one. It implies that the path two has the longest distance and path one has the shortest distance. On Hamilton cycle, is a cycle in a graph that goes around in all nodes.

**Definition:**

P = {A, B, C, D, E} is the Hamilton cycle. The issue is identifying cycle of Hamilton in a graph is NP-complete. Theorem of travelling salesman problem is NP complete. Proof: Prove that TSP belongs to NP.

When we check a tour quality, then we check that the tour visits each vertex once. Then we have sum cost of the edges and finally we check whether the path is minimum path or has less cost. This can be termed as completed polynomial time, which implies TSP belongs to NP (The Travelling Salesman Problem, n.d).

This step is to prove or show that TSP is NP hard. To prove this is to show that Hamilton cycle ≤p TSP in the sense that Hamilton cycle problem is NP complete. Let assume that G = (V, E) to be an instance cycle of Hamilton then construct an instance. The creation of complete graph to prove is needed and is

G′ = (V, E′), where *E*′ = {(*i, j*) : *i, j* ∈ *V* and *i !=j*,

hence the cost function is defined as:



Suppose that a Hamiltonian cycle h exists in G. It is then clear that the cost of each edge is h is o in G′ because each edge belongs to E. In this regard G has a Hamilton cycle it implies g’ has o tour cost., let assume then that G′ has a tour h’ of cost of at most zero. The cost of edges in E′ is zero and one by definition. Therefore, each edge must have a cost of zero as the cost of h’ is zero. Then we can conclude that h’ contains only edges in E.

In that case we have proven that G has a Hamilton cycle if G′ has a tour of cost zero, hence TSP is NP complete.

1. FALCON OPTIMIZATION PROBLEM

Metaheuristics are nature-inspired algorithms for obtaining comparative solutions to any computationally difficult optimization problems. The swarming behaviors of animals (including firefly-BAT [35], cuckoo [9], ant, pigeon, fish, bee, etc.) have been used in metaheuristics [10]. The amazing characteristics behind the metaheuristics hold identity, illation-free tools, adaptability, and local optima eschewal ability [23].

In [31], the proposed metaheuristic algorithm depends on the falcon’s behavior of hunting. The falcon optimization algorithm (FOA) is the reliable and robust algorithm of stochastic population-based problems that requires arrangements from several parameters to its three-stage action settlement.

The motivation of the proposed method was the chase style of falcons while they are seeking their prey during flight. Falcons are recluses, and their tactics for hunting depends on their requirements. However, specific tactics arise, and amazing models hold the fundamental precepts about the flight. Based on many products of Tucker [29,30], Among birds, high-performance flyers are falcons. In various states of elevated hunting, the fitting objectives are checked for the limits of flying achievement [30].

The implementation technique of flight in the framework includes determining a standardized power about the flight, the flight average velocities, and adaptive responses to the wind [30]. One of the quickest animals in the world is a falcon; stoops have been shown to approach velocities that are faster than 300 km/h. Small thin tubercules in their beaks lead the air through high-speed stoops, allowing falcons to breath easily. The primary hunting is done throughout the day (including morning and night). They primarily feed on small and medium-sized birds, but their diets also include insects like cicadas, moths, and locusts (although such prey is rare) [12].

During flight, falcons take different routes to reach their prey. Each route has two parts: the first part is a logarithmic spiral on which a falcon continually keeps its head straight while peering at the prey with the highest visible acuity; and the second is when the falcon flies toward the prey in a straight segment – when the prey is within the falcon’s field of vision, the falcon dives. Therefore, a falcon’s achieve locomotion can be classified into three steps: the initial step (first stage) – exploring for prey; the second step (second stage) – improving its dive through a logarithmic spiral; and the third step (third stage) – the dive itself (which can result in success; i.e., acquisition of prey). Otherwise, a falcon quickly reverses its action depending on it is experience.

The quick procedure, which includes five steps for the implementation of the FOA, is given below [31].

Step 1:

Start the algorithm by adjusting the parameters for the optimization problem, including the number of falcons (NP), highest speed (V max), cognitive rate (cc), social (sc) constant, following (f c) constant, dive probability (DP), and awareness probability (AP).

Step 2:

Set the velocity and position of the falcons randomly in a D-dimensional space based on the boundary conditions, where the position of each falcon is defined in consideration of the number of NP applicants within all of its D dimensions. The speeds are arbitrarily produced among the V max and V min limitations, where both are respectively determined as follows:

V *max* = 0.1 · *ub*

V *min* = −V *max*

where *ub* denotes the upper bound (the boundary area concerning each dimension). In the beginning, generate the pairs of numbers randomly (pAP, pDP) for each falcon for correspondence among the dive and awareness probabilities.

Step 3:

Calculate the fitness value and select the best (xbest) and global (gbest) sites. The selected positions will be used to produce new positions considering the logic that rules the move behind the dive and awareness probabilities.

Step 4:

New locations are produced, including updating the location of the falcon. Then, compare pAP with the probability of awareness AP; if AP is bigger than pAP, the falcon moves from seeking for prey based on its activity (including some different experiences of the other falcons):



where Viter is the current velocity and Xiter is the current position of the falcon. If pAP is bigger than AP, formerly compare a dive likelihood DP among pDP. If DP is less than pDP, then one of the targets is chosen as prey by the falcon (Xchosen), and it completes its fundamental step toward hunting. A logarithmic spiral is provided through

where b is a fixed number that determines the

state of the spiral logarithm (equal to 1), and t is an arbitrary number within range (−1, 1) that determines the next location of the falcon with respect to its exact destination. If AD is bigger than pAP, formerly compare the score function of the preferred prey and the score function of the falcon. Wherever the prey is most appropriate, it will be followed through by the falcon related to a dive step:

 otherwise, falcon continues to fly based in its best position:

The new location that is evaluated later concerns the velocities and location boundaries. Next, its new score function is computed, and the new values of Xbest and gbest are determined.

Step 5:

Last, subsequent evaluations of Step 4 are continued until the highest number of iterations (itermax) is reached.

1. ALGORITHM: TSP BASED ON FALCON OPTIMIZATION ALGORITHM

**INPUT:** - Number of cities, N - Population size, NP - Maximum speed, Vmax - Values of cognitive (Cc), social (Sc), and following (Fc) constants - Awareness probability (AP) and dive probability (DP) - Maximum number of iterations, tmax - Upper boundary (Xmax) and lower boundary (Xmin) for position initialization

**OUTPUT:** - Optimal route for the Traveling Salesman Problem

1. Initialize an empty structure and initialize parameters for FOA: - Dimension space D = N (number of cities) - Population size NP - Constant values: Cc, Sc, Fc, AP, and DP Number of iterations = tmax - Upper boundary (Xmax) and lower boundary (Xmin) for position initialization - Initialize global best position (Gt\_best, i, d)

2. Randomly initialize velocity and position for all falcons. Compare each falcon by the objective function (total distance of the route) and find the best position (Pt\_best, i, d) in the current population.

3. Repeat for tmax iterations:

* Repeat for each falcon in the population:
  + Generate random values pAP and pDP.
  + Select a new best position by comparing the objective function (total distance) of each falcon.
  + Update falcon velocity (Vi, d) based on probabilities AP and DP: - If pAP < AP, update velocity. - Else, if pDP < DP, update velocity - Otherwise, compare the objective function of the current and previous positions. If the current position is better, update velocity;
  + Update falcon position (Xi).
* b. Evaluate the objective function (total distance) of the new position (Xt\_i, d) for each falcon.
* c. Update the best position (Pt\_best, i, d) for each falcon: - If the current position (Xt\_i, d) is better than the best position, update Pt\_best.
* d. Update the global best solution (Gt\_best, i, d) if the new position is better than the current global best.
* e. Save the best score value and solution. f. If Xmin >= Xmax, stop the iteration process and present the results. Otherwise, continue to the next iteration.

4. Return the optimal route for the Traveling Salesman Problem based on the maximum objective function value achieved.