International Journal of Applied Mathematics

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

SCUBA DIVER OPTIMIZATION ALGORITHM: A NOVEL METAHEURISTICS ALGORITHM IN SOLVING TRAVELLING SALESMAN PROBLEM

Saman M. Almufti[1,2], Amira Bibo Sallow[1]

[1] Department of Information Technology, Technical College of Duhok, Duhok Polytechnic University

[2] Department of Information Technology, Technical College of Informatics - Akre, Akre University for Applied Sciences

Email: saman.mohammed@auas.edu.krd

ABSTRACT

Over the years, Metaheuristic algorithms have proven significant efficacy in handling None-Deterministic polynomial hard (NP-hard) combinatorial optimization problems, mainly the Traveling Salesman Problem (TSP) which has been used to evaluate and test many metaheuristics algorithms such as Ant Colony optimizations. This paper contributes a new optimization algorithm to the famous-family of swarm and evolutionary techniques named Scuba Diver Optimization Algorithm (SDOA) which is a novel population-based metaheuristic optimization algorithm inspired by the behavioural and physiological dynamics of scuba divers in the open water environments. SDOA simulates main diving factors including (depth, tank pressure/oxygen, bottom time, and ascent strategies) for keeping a balance between explorations and exploitations during the search process. SDOA novelty appears in its adaptive control of solution diversity using pressure and oxygen level thresholds, enabling robust avoidance of local optima and faster convergence.

The SDOA algorithm was evaluated on a range of TSP instances from the TSPLIB library, which were categorized into three groups: Group 1, comprising instances with 1–99 cities; Group 2, including instances with 100–999 cities; and Group 3, consisting of large-scale instances with 1000 or more cities. And the results of SDOA is compared with established metaheuristics algorithms such as Ant Colony Optimization, Artificial Bee Colony, and Genetic Algorithm. The study results showing that SDOA consistently produces shorter (near optimal) tour lengths with significantly minimum computational time in larger problem sizes (Group 3). The proposed algorithm also demonstrates better scalability and stability in different initialization and population sizes.

Future directions include adapting SDOA for solving other optimization problems including multi-objective or dynamic optimization contexts.

Keywords— Scuba Diver Optimization Algorithm, Metaheuristic, Traveling Salesman Problem, Swarm Intelligence, TSPLIB.

I. INTRODUCTION

The Traveling Salesman Problem (TSP) is a classical combinatorial optimization problem, with

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

extensive applications in various fields including (transportation, circuit design, urban planning, scheduling, and resource planning). Generally, the TSP involves finding the shortest possible route that visits each city exactly once and returns to the origin city. TSP is classified as an NP-hard problem, in which there is no polynomial-time to guarantee optimal solutions across all instances. With the increase in the number of cities, the solution space expands factorially, making traditional search techniques computationally impractical for large-scale, real-world scenarios(Eido and Ibrahim, 2025).

To Handle TSP challenges, various metaheuristic algorithms has been developed including evolutionary algorithms such as Genetic Algorithms (GAs), swarm-based methods such as Artificial Bee Colony (ABC), Ant Colony Optimization (ACO), and Elephant Herding Optimization (EHO), and physics-inspired algorithms such as Simulated Annealing (SA) and Gravitational Search Algorithm (GSA). These methods have shown significant efficacy in approximating near-optimal solutions for extensive TSP instances (Ferhat et al., 2025). However, those algorithms frequently encounter specific limitations, such as premature convergence, scalability, and its sensitivity to parameter configurations. Thus, The quest for more robust, adaptable, and efficient algorithms remains a critical domain of research.

Over the past few years, there has been an increasing interest in developing algorithms inspired by real-world systems and behaviours such as Biological processes, physical phenomena, social dynamics, and animal and insects behaviour, in which all those behaviours served as best grounds for algorithmic inspiration. In the realm of natural heuristics, it provides mechanisms for distributed problem-solving and efficient exploration-exploitation strategies (Almufti, 2025). In this study, we developed a new metaheuristic inspired by the physiology and decision-making strategies of scuba divers—called the Scuba Diver Optimization Algorithm (SDOA).

In the real situations, Scuba divers operate in environments that require constant controlling of variables such as depth, tank pressure/oxygen, bottom time, and ascent strategies. Divers required to plan their movements carefully to avoid decompression sickness, optimize oxygen consumption, and safely ascend. These real-time adjustments, and safety strategies can be used to optimization mechanisms.

In this study, SDOA uses these dynamics to explore the solution space of TSP efficiently. The concepts of descent (exploration), bottom time (local search), oxygen level (constraint handling), and controlled ascent (exploitation) form the algorithm's foundational principles. Unlike existing swarm-based centralized information sharing or global best tracking algorithm, SDOA introduces a decentralized decision-making structure. In which every individual diver (agent) separately evaluates their position based on dynamic parameters "oxygen level" to preserves solution diversity and delays convergence until better solutions are found, minimizing the risk of being in local optima. In practise, SDOA is benchmarked on multiple TSPLIB instances and contrasted with established metaheuristics. We assess the algorithm's performance in terms of solution quality, execution time, and error percentage. SDOA tested on 30 instants divided into three groups: Group 1, comprising instances with 1–99 cities; Group 2, including instances with 100–999 cities; and Group 3, consisting of large-scale instances

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

with 1000 or more cities.

The main contributions of this paper are: (i) introducing SDOA, a metaheuristics algorithm inspired from scuba divers; (ii) evaluating SDOA performance compared to ACO, ABC, and GA on standard TSPLIB datasets; and (iii) demonstrating its scalability and robustness for large-scale TSP instances.

The rest of this paper is organized as follows. Section 2 presents a critical review of related works in metaheuristic optimization for TSP. Section 3 is the methodology. Section 4 illustrates the results of the study. Finally, Section 5 concludes the paper.

II. RELATED WORK

The Traveling Salesman Problem (TSP) has historically served as a benchmark for evaluating and testing the efficiency and performance of various metaheuristics optimization techniques (Almufti, 2017). TSP have an NP-hard nature which makes it a challenging ground for both classical and modern algorithms. Over the decades, different metaheuristic algorithms have been developed, aiming to generate near-optimal solutions efficiently in a minimum time. These methods can be generally categorized into evolutionary algorithms, swarm intelligence techniques, and physics-based models(M. Almufti et al., 2023).

Genetic Algorithms (GA) were among the first evolutionary strategies used to solve the TSP(Li, 2025). By mimicking the principles of natural selection, reproduction, and crossover, it makes a population of candidate tours toward better (near-optimal) solutions(Bolotbekova et al., 2025). Several crossover operators specific to permutation problems have been proposed, such as Order Crossover (OX), Partially Matched Crossover (PMX), and Edge Recombination. Despite their effectiveness, GAs often struggle with premature convergence and require carefully tuned genetic operators to maintain diversity in the population.

Ant Colony Optimization (ACO), introduced in 1992 by Dorigo(Jiang et al., 2025), is a well-known and widely used algorithm inspired by the foraging behaviour of ants. In ACO, artificial ants construct tours based on the amount of pheromone trails in paths and its heuristic information, updating pheromone levels according to the quality of solutions. Even though ACO has shown competitive performance on TSP instances, it may require many iterations to reach optimal solutions and is sensitive to parameter choices such as evaporation rate and pheromone influence.

Particle Swarm Optimization (PSO) (Kou et al., 2024), first formulated for continuous spaces problem solving, then it has been adapted to discrete problems like the TSP through position mapping strategies. Discrete PSO variants use swap operators or permutation encoding to represent tour solutions. Although PSO offers simplicity and fast convergence, but it frequently lacks necessary local search capabilities in complex search landscapes, leading to suboptimal solutions in large TSP instances.

Artificial Bee Colony (ABC) (Tong et al., 2025) is a population-based metaheuristic inspired by the behaviour of honeybees. Practically it divides the search-process between employed bees, onlooker bees, and scout bees, each playing a separate role in exploration and exploitation

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

process. At the beginning employed bees search for solution (food-sources), then share information with onlooker bees, who probabilistically select best-sources based on a fitness measure. Scout bees demonstrate randomization by leaving stagnant sources and exploring new areas of the search space(Shaban and Yasin, 2025). ABC's balance of intensification and diversity makes it useful for continuous and combinatorial optimization problems, such as the TSP. However, ABC have a slow convergence when the exploration controls exploitation, requiring parameter tuning or hybridization with other techniques.

In recent times, algorithms inspired by nature have seen popularity. Notable examples include the Solar System Algorithm (SSA)(Zitouni et al., 2021), Cuckoo Search (CS) (Jati et al., 2012), Water evaporation algorithm (WEA)(Saha et al., 2017), Bat Algorithm (BA) (Yahya Zebari et al., 2020), Giant Trevally Optimizer (GTO (Sadeeq and Abdulazeez, 2022),and Spotted hyena optimizer(SHO) (Dhiman and Kumar, 2017), all of which draw inspiration from the behaviours of different animals and natural events. These algorithms apply techniques such as random-walks, attraction mechanisms, and mimicry strategies to effectively balance the processes of exploration and exploitation. Although they have shown impressive results in continuous optimization challenges, their effectiveness in discrete problems, such as the Traveling Salesman Problem (TSP), remains an area of ongoing research and enhancement. Recently, studies goes towards hybrid metaheuristics and adaptive frameworks designed to address the limitations of existing algorithms(Ahmed Shaban and Mahmood Ibrahim, 2025). These approaches often involve integrating existing algorithms such as ACO with local search heuristics, or incorporating reinforcement learning techniques to dynamically adapt algorithm parameters. While these hybrid models improve convergence rates and solution diversity, they frequently lead to increased computational complexity, costs and time(Kaveh and Bakhshpoori, 2019).

Despite this propagation of methods, the persist challenges in TSP optimization includes: avoid fast convergence, preserving population diversity, achieving scalability for large TSP instances, minimizing computational time and resources and achieving a balance between exploitation (intensification) and exploration (diversification).

This paper addresses these gaps by introducing the Scuba Diver Optimization Algorithm (SDOA)—a novel, biologically inspired method that draws from the physical and behavioural principles of scuba divers. That can generate promised near-optimal solution for TSP including large-scale instants,

In summary, while existing metaheuristic algorithms offer valued contributions to the TSP problems, it remain a testbed in algorithmic design—specially for algorithms that model constrained exploration and adaptive behaviour. SDOA introduces a novel algorithm to enhance performance across a range of TSP instances.

III. METHODOLOGY

In this paper novel algorithm SDOA is compared with three other well-known metaheuristics algorithms: : Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Genetic Algorithm (GA) in solving TSP.

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A. Traveling Salesman Problem

Traveling Salesman Problem (TSP) is a well-known NP-hard problem in combinatorial optimization. In which required to find optimal tour of between cities with a condition that every city must be visited once only and return to the starting city for completing a solution among all possible solutions(tours)(Yue et al., 2025). practically, it is a complete graph G (N, A) where N is the set of all possible cities of the tour, and A (i, j) is the set of paths that connect cities together.

The length between city Ai and Aj is represented as dij .Thus the optimal tour(Btour) is calculated by (1).

$$Btour = \left(\sum_{i=1}^{n-1} d_{p(i)\,p(i+1)}\right) + d_{p(n)\,p(1)} \quad (1)$$

Where p is a probability of cities with shortest distance between city (pi and pi+1)

B. Scuba Diver Optimization Algorithm (SDOA)

Scuba Diver Optimization Algorithm (SDOA)—a novel metaheuristic proposed in this paper as a biologically inspired method that inspired by the physical and behavioural principles of scuba divers in open water environments. Unlike existing algorithms that rely on abstract metaphors or simplified dynamics, SDOA incorporates real-world physiological constraints such as depth, tank pressure/oxygen, bottom time, and ascent strategies. These elements attend as adaptive regulator parameters that control the movement of agents (diver) within the search space. determine the safety, efficiency, and success of a dive. Divers must optimize their actions to avoid decompression sickness while maximizing the time spent exploring.

The individuality of SDOA appears in its explicit modelling of dynamic decision-making under constraint, similar to the real-time trade-offs that divers must perform. Each diver agent performs a depth-regulated search, simulating exploration and intensification phases, while ascent strategies ensure escape from suboptimal regions. Moreover, SDOA‘s design is inherently modular and interpretable, making it suitable for adaptation and hybridization with local search strategies or machine learning-based guidance.

In the SDOA Algorithm, communication is a targeted, probabilistic knowledge-sharing mechanism. Divers that are at the same depth (stage) copy the solution of a "buddy", in case if the buddy has a better fitness. The probability of communication is higher for divers with higher oxygen level (divers in exploration phases). This communications process makes successful solutions found by a diver to be quickly shared with others in a similar search context. SDOA accelerates convergence by decreasing redundant effort, professionally propagating good solutions in the search space while keeping the balance between exploration and exploitation by depth-based grouping.

i.Algorithm Design

Each solution in SDOA is modelled as a diver agent. The process of optimization is divided into four major phases. See Fig. 1.

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

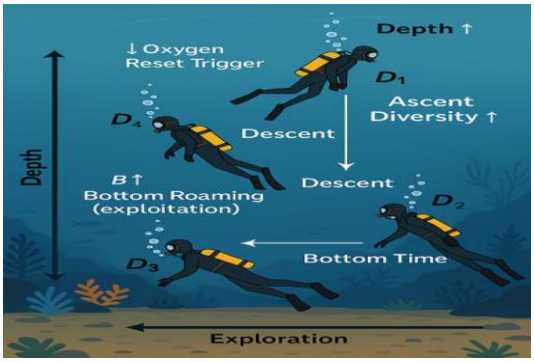


Fig. 1. Scuba Diver Optimization Algorithm Steps

- **Descent (Exploration):** in this step the Agents (Divers) explore the search-space by making large perturbations, simulating deep dives. In Fig. 1 Divers goes downward (D1 → D2 → D3 → D4), makes deeper exploration. The arrow labelled "Descent" resembles to this phase.
- **Intensive Local Search (Intensification):** Once the divers reached to the promising region, they perform local search to improve solutions. in Fig. 1, located at the seabed (D3/D4), the diver makes "Bottom Roaming (exploitation)" and "Bottom Time," which represent concentrated search activity in a confined area. At Bottom time the agents Perform an Intensive Local Search around their current position. While at the Bottom Roaming, they Undergo Mutation to avoid local optima and search the immediate vicinity differently.
- **Oxygen reset trigger (Constraint Handling):** Oxygen depletes over time and based on activity. At this step when oxygen level reaches limits it reinitialize the oxygen level, providing the agent to reexplore the solution space. In Fig.1 this step represented by the oxygen reset trigger shown near the top diver.
- **Ascent (Exploitation and Escape):** If no improvement in the solution is found, the agent ascends, reverting to safer positions. In Fig.1, the upward arrow "Ascent Diversity ↑" shows this step. Divers ascend when their search must reset or diversify, mapping to exploitation and escape.

ii.Practical Algorithm Steps

In this section we will show the practical steps, flowchart, and the incision parameters of SDOA. The algorithm works according the following 6 steps:

1. **Initialization**: Generate random tours for all agents; set initial pressure, depth, and oxygen.
2. **Exploration (Descent)**: Apply perturbations proportional to depth.
3. **Local Search (Bottom Time)**: Perform 2-opt operations.
4. **Evaluation**: Compute tour length and update pressure.
5. **Exploitation (Ascent)**: If no improvement, ascend to previous solution or global best.
6. **Termination**: Stop if max iterations reached or convergence criteria met.

Fig. 2, illustrate the SDOA flowchart

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

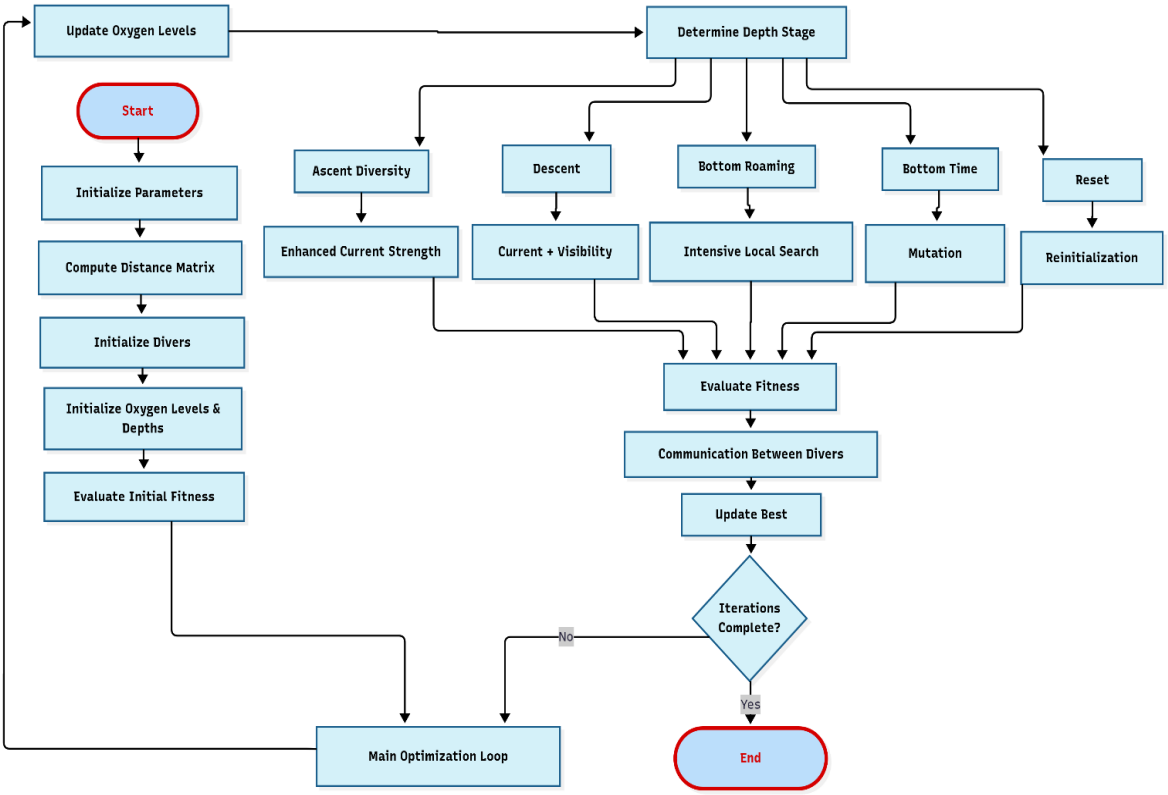


Fig. 2. Scuba Diver Optimization Algorithm (SDOA) flowchart

Table I, presents the main parameters of the SDOA including their purpose, and the impact of tuning them.

Table I

SDOA Algorithm Parameters

Parameter Name	Description & Purpose	Effect of Increasing the Value
nDivers	The population size; number of candidate solutions (tours) exploring the search space simultaneously.	Pros: Better exploration, less prone to getting stuck in local optima. Cons: Higher computational cost per iteration.
maxIter	The maximum number of generations/iterations for the main optimization loop. The stopping condition.	Allows the algorithm more time to converge to a better solution. Must be balanced with nDivers for total computational budget (nDivers * maxIter).
initialCurrentStrength	Controls the exploration intensity in early stages (D1, D2). Scales the number	Divers make larger, more disruptive changes to their tours,

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	of random swaps applied to a tour.	promoting exploration and diversity. May slow convergence if too high.
initialVisibilityRange	Influences the probability of applying the twoOpt local search during the D2 (descent) phase.	Increases the focus on local improvement (exploitation) during the transition from exploration to exploitation.
initialMutationRate	The probability that a diver in the D3 (bottom) stage will perform a random swap (mutation).	Helps divers escape local optima even during the exploitation phase. Increases diversity but can disrupt good solutions if too high.
initialOxygenLevel	The starting oxygen value for all divers. Primarily a scaling factor for other calculations.	A higher value slows the perceived rate of oxygen depletion, prolonging the early, exploratory phases.
alpha	The oxygen decay rate coefficient. Controls how quickly divers' oxygen levels deplete over time exp(-alpha * iter / maxIter).	Higher alpha: Oxygen depletes faster, divers transition to exploitation (B1) and reset more quickly. Lower alpha: Oxygen depletes slower, prolonging exploration (D1, D2).
Depth Thresholds	The oxygen level thresholds that determine a diver's depth stage and resulting behaviour.	Adjusting these changes the algorithm's balance between exploration and exploitation. Lowering thresholds keeps divers in exploration longer.

iii.Mathematical Modelling

Scuba Diver Optimization Algorithm (SDOA) is mathematically developed in a manner that integrates real-world diving constraints into the search dynamics of a metaheuristic optimizer for solving the Traveling Salesman Problem (TSP). the main equations are shown below:

1. Oxygen Level Equation: This is the main mechanism for controlling the change between exploration and exploitation in the search process. It ensures the algorithm starts with more global exploration and gradually focuses on local refinement, Oxygen level can be calculated by (2).

$$O_i^t = O_i^{t-1} * e^{\left(-\alpha * \frac{t}{t_{max}}\right)} \quad (2)$$

Where O_i^t is the oxygen level of diver i at iteration t, α represent oxygen decay rate, and the

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

maximum number of iterations is represented by $\boldsymbol{t_{max}}$ where *t* is current iteration.

2. Depth Stage Assignment: This parameter is uses as a behavioural switch. It maps a diver's continuous oxygen-level onto a discrete set of five distinct search strategies (D1, D2, D3, D4, Reset) according to (3). This creates a clear, structured progression for each solution in the population.

$$d_i^t = \begin{cases} 1, if\ O_i^t > 60 \\ 2, if\ O_i^t > 40 \\ 3, if\ O_i^t > 25 \\ 4, if\ O_i^t > 0.5 \\ 5, otherwise\ (reset) \end{cases} \quad (3)$$

3. Current Strength & Visibility Range Scaling: those two parameters are uses to control scaling of exploration intensity and local search for each diver based on its oxygen level. Both decrease equivalently with oxygen and can be calculated by (4) and (5):

$$CS_i^t = CS_0 * \left(\frac{O_i^t}{O_0}\right) \quad (4)$$

$$VR_i^t = VR_0 * \left(\frac{O_i^t}{O_0}\right) \quad (5)$$

Where CS_0 represent the initial current strength, $VR_0$is initial visibility range, and the O_0 represents the initial oxygen level

4. Swap Operation (Mutation): is the process of randomly switches two cities in the solution. In the tour of cities R=(C_1, C_2, C_3, ….., C_n) the mutation take place by (6)

$$R' = Swap(R, p, q) \quad (6)$$

The result of mutation (swapping) is rearrange form of the tour R′=(C_1, …, C_{p-1}, C_q, C_{p+1},…, C , C_p, C_{q+1},…, C_n)

5. Fitness Update Rule: as in all exiting metaheuristics algorithm, SDOA updates fitness If a new route improves, calculated by (6):

$$if\ D(R') < D(R) => O_i^t = \min(O_i^t + 5, O_0) \quad (6)$$

6. Communication Between Divers: in SDOA divers at the same stage (depth) share solution with buddy. If buddy j has better fitness Diver i as calculated by (7):

$$R_i^t = R_j^t if\ D\left(R_j^t\right) < D(R_i^t) \quad (7)$$

7. Reset Condition: in SDOA a diver with low oxygen level have the probability to reinitializes:

$$R_i^t \sim NearestNeighborHeuristic, \quad O_i^t = O_0 \quad (8)$$

Received: August 05 2025

International Journal of Applied Mathematics

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

C. Existing Metaheuristic algorithm

In this paper uses three well-known metaheuristics algorithm: Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Genetic Algorithm (GA) for comparing the results of the novel SDOA in solving TSP. Table II, illustrate the details of ACO, ABC, and GA. The table provide the key equations and the inspiration phenomena.

Table II

used metaheuristics algorithm

Algorithm	Key Equations	Inspiration
Ant Colony Optimization (ACO)	The Transition Probability rule: $$\boldsymbol{P_{ij}^{k}} = \frac{[\tau_{ij}]^{\alpha}.[\eta_{ij}]^{\beta}}{\sum_{\mathbf{1}\in\mathcal{N}_i^k}[\tau_{ij}]^{\alpha}.[\eta_{ij}]^{\beta}}, j \in N_i^k \quad (9)$$ Heuristic Function: $\eta_{ij} = \frac{1}{d_{ij}}$ (10) Initial Pheromone: $\tau_{ij} = \tau_0, \tau_0 > 0$ (11) Local Pheromone Update: $$\tau_{ij}^{new} = \tau_{ij}^{old} + \Delta\tau_{ij} \quad (12)$$ Global Update: $\Delta\tau_{ij} = \Sigma_{\{k=1\}}^{m}\Delta\tau_{ij}^{k}$ (13) $$\Delta\tau_{ij}^{k} = \begin{cases}\frac{Q}{L_k if} \; ant\ k\ uses\ arc\ (i,j) \\ 0\ otherwise\end{cases} \quad (14)$$ Pheromone Evaporation: $$\tau_{ij}^{new} = (1-\psi)\tau_{ij}^{old} \quad (15)$$	ACO is inspired from the Foraging behaviour of real ants, in which in the search process ants deposit pheromones along paths it takes from there colony to the food sources. The shortest paths is the path with more concentration of pheromone and ants communicate through stigmaria thus balancing exploration and exploitation.
Artificial Bee Colony (ABC)	Selection Probability (Onlooker Bee): $$p_i = \frac{fit_i}{\sum_{n-1}^{SN} fit_n} \quad (16)$$ Candidate Food Source Update:	ABC inspired from the Foraging behaviour of honeybee. Bees colony consist of three kind of bee (employed bees,

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	$x_{ij}^{*} = x_{ij} + \varphi_{ij}(x_{ij} - x_{lj}), \varphi_{ij} \in [-1,1]$ (17)	onlooker bees, and scout bees) employed bees is responsible of exploiting known food sources, where the onlooker bees are selecting based on waggle-dance probability, and finally the scout bees randomly explore for new food sources).
Genetic Algorithm (GA)	Optimization Problem: min f(x), x ∈ S or max f(x), x ∈ S (18) Fitness Function (example): $F(x_i^t) = \frac{1}{(1 + f(x_i^t))}$ (19) Selection Probability: $P(x_i^t) = \frac{F(x_i^t)}{\sum_{j=1}^{m} F(x_j^x)}$ (20) Crossover (Recombination): $Offspring = \alpha x_p + (1-\alpha)x_q, \alpha \in [0,1]$ (21) Mutation: $x_i^{t+1} = x_i^t + \delta, \delta \sim D(0,\sigma^2)$ (22)	GA is a stochastic algorithm inspired from the natural selection and Darwin's survival of the fittest principle. GA works on a group of candidate solutions encoded as chromosomes, that evolve in iterative processes of selection, crossover, and mutation. The algorithm have the ability to balance between exploration and exploitation.

IV. RESULTS

The novel Scuba Diver Optimization Algorithm (SDOA) is compared with Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Genetic Algorithm (GA) in solving 30 instants of TSP that we grouped it in three different groups according to the number of cities that an instance contain:

- Group 1 (Small size instants): this group contains 6 instants that contains less that 100 cities.
- Group 2 (medium size instants): this group contains 18 instants that have 100>= N

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

<1000.

- Group 3 (large size instants): in this group also we tested 6 instants of size N>=1000.

The results are shown in different tables, illustrating: best fitness, execution time, error percentage, and also statistical average and standard deviation of each group. All proposed algorithms executed 30 iterations in a laptop (processor : Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, and RAM 16 GB). Table III, present the initialization parameters of all 4 proposed algorithms.

Table III

Proposed algorithms parameter initialization

	SDOA	ACO	ABC	GA
Initial Parameter	• nDivers=10 • maxIter=30 • initialCurrentStrength=0.5 • initialVisibilityRange=3 • initialMutationRate=0.5 • initialOxygenLevel=100 • alpha=5 • Depth Thresholds=[60, 40, 25, 0.5]	• nAnt=nCity • max_it=30 • alpha=1 • beta=2 • rho=0.5 • tau0=1 • eta=1/D	• nFoodSources=10 • maxIter=30 • limit=100	• popsize=20 • maxIter=30 • nparents=10 • mutation_rate=0.01 • tournamentSize=3

Table IV, presents the execution results of SDOA, ACO, ABC, and GA with their execution elapsed time.

Table IV

SDOA results compared with ACO, ABC, and GA

#City	TSP pro.	Opt.	SDOA	SDOA time	ACO	ACO time	ABC	ABC time	GA	GA Time	
148	att48	10628	11181	0.4	11753	4.966512	12525	0.11	11936	0.17	Group 1
251	eil51	426	441.43	0.32	472	14.423435	483.78	0.14	510.97	0.11	
352	berlin52	7542	7713.03	0.46	8092	4.405582	8824.87	0.14	8902.3	0.18	
470	st70	675	710.36	0.68	750	37.711194	1016.75	0.18	1071.61	0.26	
576	eil76	538	575.64	0.61	580	11.378319	889.75	0.19	830.07	0.3	

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

676	pr76	108159	110208.5	10.81	123916	9.172275	184350.4	0.22	165955.95	0.21	
7100	kroA100	21282	21395.37	1.6	24698	15.940223	52519.06	0.37	60007.45	0.48	Group 2
8100	kroB100	22141	22916.45	1.9	25796	32.430596	53391.38	0.39	54930.71	0.47	
9101	eil101	629	657.19	1.05	746	15.87946	1313.43	0.39	1303.1	0.39	
1130	ch130	6110	6367.74	2.71	7034	16.311682	20094.47	0.66	20194.29	0.73	
1150	kroA150	26524	28171.94	3.29	30669	32.632293	115214.1	0.97	112418.64	0.94	
1150	kroB150	26130	26813.96	4.01	31611	29.543956	109662.9	0.95	107289.92	0.94	
1150	ch150	6528	6656.22	3.62	6871	22.784681	25297.84	0.9	25556.48	1.01	
1200	kroA200	29368	29833.38	6.95	34543	51.051143	163716.6	1.79	175860.51	1.96	
1200	kroB200	29437	31062.11	6.84	35327	49.465509	172604.7	1.76	178148.22	2.02	
1225	tsp225	3916	3970.57	8.64	4578	78.175064	24484.2	2.4	22595.18	2.6	
1280	a280	2579	2754.76	13.89	3008	104.098122	21584.1	0.09	21237.35	4.5	
1318	lin318	42029	44280.85	18.71	48162	150.90364	389937.9	5.54	391576.71	6.01	
1417	fl417	11861	12509.64	36.04	13239	320.876971	353063	11.17	346786.86	11.56	
2442	pcb442	50778	53782.37	40.63	57920	395.51451	580013.8	12.68	569899.42	12.52	
2532	att532	27686	30012	64.55	32191	758.08409	366355	21.12	365482	21.45	
2575	rat575	6773	7375.53	77.08	7864	823.76911	87496.3	925.65	87715.66	25.54	
2654	p654	34643	36836.08	120.37	42296	1198.0203	1649936	35.52	1625978.6	235.38	
2783	rat783	8806	9864.72	186.44	10481	2235.7565	148124.1	57.98	146308.35	56.59	
21002	pr1002	259045	291672.6	3363.52	309316	4455.887	5215547906	117.375	559297.26	114.91	Group 3
21060	u1060	224094	254626.4	7473.61	271833	5627.894	01 5740784	150.915	697832.45	165.63	
21173	pcb1173	56892	65775.55	657.37	69482	8755.270	2321252962	212.311	239344.19	205.02	
21432	u1432	152970	176941.4	61340.35	183321	17265.41	4913582954	412.923	535023.16	398.53	
21817	u1817	57201	63575.91	2582.226	61361	84354	1938273	879.741	923074.23	883.43	
32152	u2152	64253	72900.78	7090.747	51111	87285.87	3822354225	1612.62	334713.8	1618.94	

the execution results of all algorithm are compared with the optimal known solution of each instants ant the error rate is calculated and shown in Table V.

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Table V

Proposed algorithm Error rate

#	nCity	TSP pro.	SDOA	ACO	ABC	GA	
1.	48	att48	5.20%	10.59%	17.85%	12.31%	Group 1
2.	51	eil51	3.62%	10.80%	13.56%	19.95%	
3.	52	berlin52	2.27%	7.29%	17.01%	18.04%	
4.	70	st70	5.24%	11.11%	50.63%	58.76%	
5.	76	eil76	7.00%	7.81%	65.38%	54.29%	
6.	76	pr76	1.89%	14.57%	70.44%	53.44%	
7.	100	kroA100	0.53%	16.05%	146.78%	181.96%	Group 2
8.	100	kroB100	3.50%	16.51%	141.14%	148.09%	
9.	101	eil101	4.48%	18.60%	108.81%	107.17%	
10.	130	ch130	4.22%	15.12%	228.88%	230.51%	
11.	150	kroA150	6.21%	15.63%	334.38%	323.84%	
12.	150	kroB150	2.62%	20.98%	319.68%	310.60%	
13.	150	ch150	1.96%	5.25%	287.53%	291.49%	
14.	200	kroA200	1.58%	17.62%	457.47%	498.82%	
15.	200	kroB200	5.52%	20.01%	486.35%	505.18%	
16.	225	tsp225	1.39%	16.91%	525.23%	477.00%	
17.	280	a280	6.82%	16.63%	736.92%	723.47%	
18.	318	lin318	5.36%	14.59%	827.78%	831.68%	
19.	417	fl417	5.47%	11.62%	2876.67%	2823.76%	
20.	442	pcb442	5.92%	14.07%	1042.25%	1022.34%	
21.	532	att532	8.40%	16.27%	1223.25%	1220.10%	
22.	575	rat575	8.90%	16.11%	1191.84%	1195.08%	
23.	654	p654	6.33%	22.09%	4662.68%	4593.53%	
24.	783	rat783	12.02%	19.02%	1582.08%	1561.46%	
25.	1002	pr1002	12.60%	19.41%	2041.68%	2046.07%	Group 3
26.	1060	u1060	13.62%	21.30%	2461.78%	2442.61%	

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

27.1173	pcb1173	15.61%	22.13%	2102.35%	2078.42%
28.1432	u1432	15.67%	19.84%	2242.26%	2210.93%
29.1817	u1817	11.14%	15.62%	3288.53%	3261.96%
30.2152	u2152	13.46%	16.90%	3563.99%	3533.63%

After statistical analyse of the results, the Average, Minimum, Maximum, and Standard Deviation of error rate and execution time of each group are presented in tables VI, VII, VIII, IX, X, and XI.

Table VI

Group 1 Error rate statistic

Statistic (Error %)	SDOA	ACO	ABC	GA
Average	**4.20%**	**11.36%**	**44.26%**	**39.77%**
Minimum	1.89%	7.29%	13.56%	12.31%
Maximum	7.00%	14.57%	70.44%	58.76%
Std Deviation	1.97%	2.87%	25.92%	22.95%

Table VII

Group 1 execution time statistics

Statistic (Time s)	SDOA	ACO	ABC	GA
Average	**0.56**	**13.53**	**0.17**	**0.20**
Minimum	0.32	4.41	0.11	0.11
Maximum	0.81	37.71	0.22	0.30

Table VIII

Group 2 Error rate statistics

Statistic (Error %)	SDOA	ACO	ABC	GA
Average	**5.24%**	**16.85%**	**1012.60%**	**1008.85%**
Minimum	0.53%	5.25%	141.14%	148.09%
Maximum	13.62%	22.13%	4662.68%	4593.53%
Std Deviation	4.04%	4.34%	1312.31%	1288.05%

Table IX

Group 2 execution time statistics

Statistic (Time s)	SDOA	ACO	ABC	GA

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Average	**86.41**	**1289.18**	**20.87**	**20.72**
Minimum	1.05	15.88	0.37	0.39
Maximum	657.37	8755.27	212.31	205.02

Table X

Group 3 Error rate statistics

Statistic (Error %)	**SDOA**	**ACO**	**ABC**	**GA**
Average	**13.82%**	**19.68%**	**2465.92%**	**2445.95%**
Minimum	11.14%	15.62%	2041.68%	2046.07%
Maximum	15.67%	22.13%	3563.99%	3533.63%
Std Deviation	1.75%	2.57%	626.87%	611.56%

Table XI

Group 3 execution time statistics

Statistic (Time s)	**SDOA**	**ACO**	**ABC**	**GA**
Average	**1642.13**	**43231.57**	**472.42**	**466.50**
Minimum	363.52	4455.89	117.37	114.91
Maximum	7090.74	87285.87	1612.60	1618.94

Table XII compares evaluating algorithmic convergence of SDOA, ACO, PSO, and GA on a sample of 3 TSP instants each from different group. The left column clearly identify that SDOA and ACO as the better performers compared to GA and ABC. The right column provides a focus on SDOA and ACO algorithms. This focused comparison clearly showed SDOA's superiority convergence rate, higher initial solution quality, and significantly better final solution.

Table XII

Convergence charts

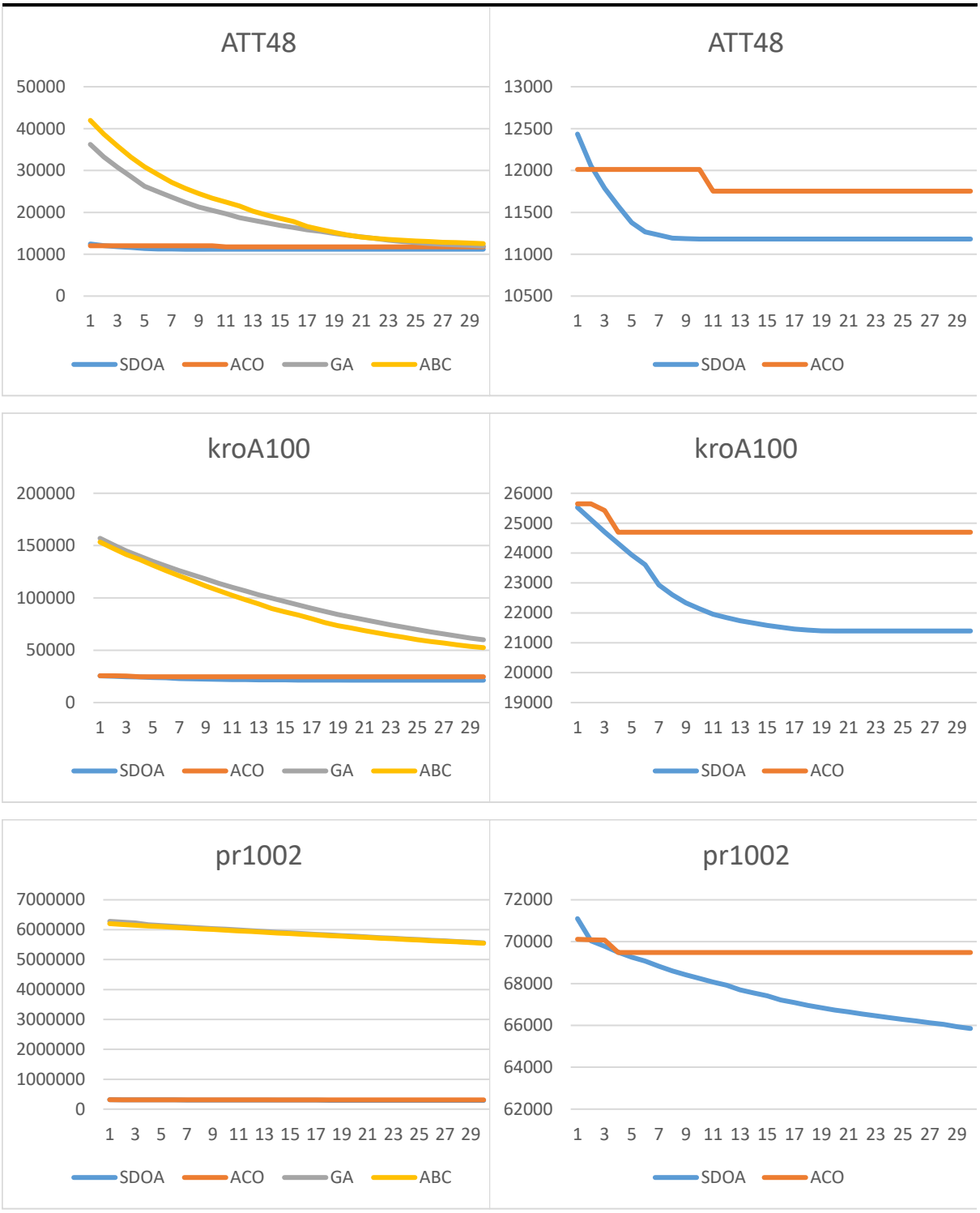
SDOA, ACO, PSO, and GA convergence	**SDOA and ACO convergence**

Received: August 05 2025

International Journal of Applied Mathematics

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)



The experimental results of the paper shows that the new Scuba Diver Optimization Algorithm (SDOA) achieves a higher balance of solution accuracy and computational efficiency compared to Ant ACO, ABC, and GA in solving various instances of the Traveling Salesman Problem (TSP).

1. Solution Quality and Accuracy

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

SDOA found near-optimal solution for all small instants problems in Group 1, with average error rate of 4.20%, which is lower than ACO (11.36%), ABC (44.26%), and GA (39.77%). In Group 2, the novel SDOA's average error is (5.24%) is less than ACO's (16.85%) and extremely lower than the large errors produced by ABC and GA. Aldo in the largest Group 3, SDOA performs an acceptable average error of 13.82%, whereas ABC and GA again failed with a very large error rate.

2. Efficiency and Execution time

SDOA demonstrated remarkable computational efficiency, consistently outperforming ACO in terms of speed across all tested groups. For instance, when tackling the pcb442 problem, which involves 442 cities, SDOA achieved a best solution in just 40.63 seconds, while ACO took a 395.51 seconds to solve it. And also for largest instance, u2152, SDOA completed the task in 7,090 seconds, ACO's execution takes 87,285 seconds. Although ABC and GA boasted quicker runtimes, the quality of their solutions was significantly not good, as evidenced by their high error rates, making their speed largely irrelevant. In comparison, SDOA strikes an impressive balance by delivering high-quality solutions without the prohibitive computational demands associated with ACO.

3. Scalability

The key strength of SDOA is the scalability. Although the performance of all algorithms naturally decreases with increasing instants size, SDOA's was the most controlled and gradual algorithm as compared to ACO, ABC, and GA. Its error rate gradually increases from 4% for small size to 14% for very large instants size. In stark contrast, the error rates for ABC and GA become unmanageably high, soaring to between 2000% and 3500% for problems involving more than 200 cities, highlighting a critical inability to scale effectively. While ACO shows greater resilience compared to ABC and GA, it still faces an increase in error rates, reaching nearly 20% for larger instances. The adaptive mechanisms of SDOA play a crucial role in effectively navigating the exponential growth of the search space.

4. Robustness and Reliability

Statistical analysis of Tables VI, VIII, and X confirms the robustness of the novel SDOA's. It had the lowest standard deviation in error rates within all groups 1, 2, and 3, indicating highly reliable and stable performance. It also have the lowest minimum error, indicating that it is able to achieve very high-quality solutions.

V. CONCLUSION

This paper presented the Scuba Diver Optimization Algorithm (SDOA), a novel metaheuristic algorithm inspired by the physiological constraints and adaptive decision-making strategies of scuba divers. By metaphorically translating key diving elements—depth, oxygen management, bottom time, and controlled ascent—into search operators, SDOA effectively balances global exploration with local exploitation. Its decentralized, physiology-driven mechanism inherently preserves population diversity and mitigates the risk of premature convergence, a common pitfall in many population-based algorithms.

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The performance of SDOA was rigorously evaluated against three established metaheuristics—Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Genetic Algorithm (GA)—across a comprehensive set of TSPLIB instances, categorized by size. The empirical results unequivocally demonstrate SDOA's superiority in both solution quality and computational efficiency. Notably, SDOA achieved significantly lower error rates across all problem sizes, from small (Group 1) to very large instances (Group 3), while its execution time remained orders of magnitude faster than ACO for large-scale problems. Crucially, unlike ABC and GA, whose solution quality deteriorated catastrophically as problem size increased, SDOA exhibited remarkable scalability and robustness, maintaining viable solution quality even for instances exceeding 2000 cities.

The primary contributions of this work are twofold:

1. Algorithmic Innovation: The introduction of SDOA, which offers a unique and effective biologically inspired model for combinatorial optimization.
2. Empirical Validation: A demonstration that SDOA consistently outperforms classic metaheuristics in solving the TSP, particularly in large-scale scenarios where efficiency and accuracy are paramount.

Beyond its immediate performance, SDOA's modular and interpretable design provides a fertile ground for future extensions. Promising research directions include:

- Hybridizing SDOA with local search heuristics or machine learning techniques for intensified exploitation;
- Adapting the framework for multi-objective, dynamic, and constrained optimization problems where its adaptive mechanics are highly relevant;
- Exploring its application to continuous optimization domains to test its generalizability.

In summary, the Scuba Diver Optimization Algorithm represents a valuable addition to the metaheuristics landscape. Its demonstrated efficacy, efficiency, and scalability position it as a highly competitive and promising tool for tackling complex NP-hard problems in fields such as logistics, scheduling, and network design. This work underscores the potential of drawing inspiration from nuanced human-in-the-loop systems to advance the state-of-the-art in optimization.

VI. REFERENCES

Ahmed Shaban, A., Mahmood Ibrahim, I., 2025. Swarm intelligence algorithms: a survey of modifications and applications, International Journal of Scientific World.

Almufti, S., 2017. Using Swarm Intelligence for solving NPHard Problems. Academic Journal of Nawroz University 6, 46–50. https://doi.org/10.25007/ajnu.v6n3a78

Almufti, S.M., 2025. Metaheuristics Algorithms: Overview, Applications, and Modifications, 1st ed. Deep Science Publishing. https://doi.org/10.70593/978-93-7185-454-2

Received: August 05 2025

Volume 38 No. 9s , 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Bolotbekova, A., Hakli, H., Beskirli, A., 2025. Trip route optimization based on bus transit using genetic algorithm with different crossover techniques: a case study in Konya/Türkiye. Sci Rep 15. https://doi.org/10.1038/s41598-025-86695-4

Dhiman, G., Kumar, V., 2017. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Advances in Engineering Software 114, 48–70. https://doi.org/10.1016/j.advengsoft.2017.05.014

Eido, W.M., Ibrahim, I.M., 2025. Ant Colony Optimization (ACO) for Traveling Salesman Problem: A Review. Asian Journal of Research in Computer Science 18, 20–45. https://doi.org/10.9734/ajrcos/2025/v18i2559

Ferhat, A., Zitouni, F., Lakbichi, R., Limane, A., Harous, S., Kazar, O., 2025. Impact of Problem Size on Exploration-Exploitation Balance in Metaheuristics for the Travelling Salesman Problem, in: 2025 7th International Conference on Pattern Analysis and Intelligent Systems (PAIS). IEEE, pp. 1–8. https://doi.org/10.1109/PAIS66004.2025.11126504

Jati, G.K., Manurung, H.M., Suyanto, 2012. Discrete cuckoo search for traveling salesman problem, in: 2012 7th International Conference on Computing and Convergence Technology (ICCCT). pp. 993–997.

Jiang, Z.-H., Yang, Q., Duan, D.-T., Xu, P.-L., Qu, C.-Z., Lu, Z.-Y., Zhang, J., 2025. Clustering-Assisted Ant Colony Optimization for Large-Scale Travelling Salesman Problem, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion. ACM, New York, NY, USA, pp. 227–230. https://doi.org/10.1145/3712255.3726580

Kaveh, A., Bakhshpoori, T., 2019. Metaheuristics: Outlines, MATLAB Codes and Examples, Metaheuristics: Outlines, MATLAB Codes and Examples. Springer International Publishing. https://doi.org/10.1007/978-3-030-04067-3

Kou, L., Wan, J., Liu, H., Ke, W., Li, H., Chen, J., Yu, Z., Yuan, Q., 2024. Optimized design of patrol path for offshore wind farms based on genetic algorithm and particle swarm optimization with traveling salesman problem. Concurr Comput 36. https://doi.org/10.1002/cpe.7907

Li, H., 2025. Solving the TSP Problem Based on Improved Genetic Algorithm. https://doi.org/10.54254/2755-2721/133/2025.20603

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

Sadeeq, H.T., Abdulazeez, A.M., 2022. Giant Trevally Optimizer (GTO): A Novel Metaheuristic Algorithm for Global Optimization and Challenging Engineering Problems. IEEE Access 10, 121615–121640. https://doi.org/10.1109/ACCESS.2022.3223388

Received: August 05 2025

Volume 38 No. 9s , 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Saha, A., Das, P., Chakraborty, A.K., 2017. Water evaporation algorithm: A new metaheuristic algorithm towards the solution of optimal power flow. Engineering Science and Technology, an International Journal 20, 1540–1552. https://doi.org/10.1016/j.jestch.2017.12.009

Shaban, A.A., Yasin, M., 2025. Applications of the artificial bee colony algorithm in medical imaging and diagnostics: a review, International Journal of Scientific World.

Tong, M., Peng, Z., Wang, Q., 2025. A hybrid artificial bee colony algorithm with high robustness for the multiple traveling salesman problem with multiple depots. Expert Syst Appl 260, 125446. https://doi.org/10.1016/j.eswa.2024.125446

Yahya Zebari, A., M. Almufti, S., Mohammed Abdulrahman, C., 2020. Bat algorithm (BA): review, applications and modifications. International Journal of Scientific World 8, 1. https://doi.org/10.14419/ijsw.v8i1.30120

Yue, C., Shen, Y., Liang, J., Yu, K., Li, M., Ma, T., Song, H., 2025. Hierarchical Genetic Algorithm for the Multi-Solution Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation 1–1. https://doi.org/10.1109/TEVC.2025.3565947

Zitouni, F., Harous, S., Maamri, R., 2021. The Solar System Algorithm: A Novel Metaheuristic Method for Global Optimization. IEEE Access 9, 4542–4565. https://doi.org/10.1109/ACCESS.2020.3047912

Received: August 05 2025