EAI Endorsed Transactions
on AI and Robotics
Research Article EAI.EU

MSCSO: A Hybrid Nature-Inspired Algorithm for High-Dimensional Traffic Optimization in Urban Environments

Kuldeep Vayadande*,[1], Viomesh Kumar Singh[1], Amol Bhosle[2], Ranjana Gore[2], Yogesh Uttamrao Bodhe[3], Aditi Bhat[1], Zulfikar Charoliya[1], Aayush Chavan[1], Pranav Bachhav[1], Aditya Bhoyar[1]

[1]Vishwakarma Institute of Technology, Pune, 411037, Maharashtra, India
[2]MIT Art, Design and Technology University, Pune, 412201, Maharashtra, India
[3]Government Polytechnic Pune, Pune, 411016, Maharashtra, India

Abstract

Metropolitan regions have experienced higher economical and environmental pressure due to the fasted urbanization leading to increased traffic jams that necessitate the use of higher optimization techniques. Traditional traffic models do not usually take large-dimensional and dynamicity of urban mobility into consideration and require extraordinary computational approaches. Modified Sand Cat Swarm Optimization (MSCSO) improves the Sand Cat Swarm Optimization (SCSO) algorithm that adds Levy flights to global exploration and roulette wheel selection to adaptive exploitation to solve problems that are complex and high-dimensional. When used in urban traffic management, MSCSO works with enormous volumes of traffic, speed, weather, and incident, all of which may decrease Travel Time Index by 15 percent during rush hours. Benchmark tests are used to prove that MSCSO is better, scoring 0.0 in Sphere, Ackley and Rastrigin functions, and 28.0753 in Rosenbrock, whereas higher scores belong to Particle Swarm Optimization, Genetic Algorithms, Ant Colony Optimization and SCSO (e.g., 46). It supports urban planning, since a Flask-based web interface has the possibility to input and visualize real time traffic data in a simple way. The success of MSCSO is reliant on high-quality data and hardware-friendly algorithms but can scale to use real-time data sources, such as from GPS, machine learning traffic projections, and cloud hosting, and is of potential use in logistics, energy delivery, and resource assignment.

Received on 18 May 2025; accepted on 05 July 2025; published on 11 July 2025

Keywords: Hybrid optimization, animal foraging, sand cat swarm optimization, metaheuristics, traffic optimization

doi:10.4108/airo.9344

1. Introduction

Urban growth in many other Indian cities has triggered extreme traffic congestion that generates numerous economic and environmental and social problems. Bangalore stands among the top congested cities worldwide and the Ministry of Road Transport and Highways calculates that traffic delay losses have reached billions annually [23]. Vehicle ownership rise and subpar road infrastructure together with changing environmental conditions like weather combined with poor road conditions and unexpected events elevate both vehicle travel duration along with fuel usage and air contamination levels [19]. The traditional Greenshields flow models and static signal timing systems present limitations when managing complex high-dimensional urban traffic systems [12, 20]. Advanced computational methods must develop new capabilities for real-time optimization of complex traffic networks due to the existing limitations [13, 22].

Natural-inspired metaheuristic optimization algorithms have proven themselves as strong approaches for dealing with such complex difficulties according to relevant studies [22]. Spatial optimization research

*Corresponding author. Email: kuldeep.vayadande@gmail.com

has yielded metaheuristic algorithms including Particle Swarm Optimization (PSO) [24], Ant Colony Optimization [15] and Genetic Algorithms which catalyze effective searches through complex non-linear domains [13]. The efficiency of these methods depends on the characteristics of problem complexity and dimensionality according to [5]. This study presents Modified Sand Cat Swarm Optimization (MSCSO) which represents an upgraded version of Sand Cat Swarm Optimization (SCSO) developed by Seyed and Nadimi [17]. MSCSO applies adaptive methods through Levy flights and roulette wheel selection which optimize high-dimensional optimization problems especially well [3, 4, 18]. The application of SCSO variants through recent research has shown their suitability across engineering design and feature selection tasks along with engineering design applications [1, 2, 6, 8, 10].

The standard benchmark test functions Sphere, Ackley, Rastrigin, Rosenbrock served as a performance evaluation mechanism for MSCSO by researchers who adopt diverse test functions [5, 21]. The sphere and Ackley functions and the Rastrigin function serve as test cases for evaluating the convergence speed of uni-modal and multi-modal problems respectively because of their separability characteristics [21]. Non-separable Rosenbrock function represents a challenging scenario for MSCSO because it possesses a narrow valley which mimics real-world scenarios in traffic optimization where connected variables like traffic flow and meteorological conditions function as a system [19, 20]. The BTF evaluation demonstrates MSCSO's reliable performance by obtaining optimal scores in Sphere and Ackley and Rastrigin functions and selectably performing on Rosenbrock functions which establishes solid groundwork for practical usage [18, 21].

The project implements MSCSO for traffic optimization through extensive data analysis comprising traffic attributes such as volume, speed, travel time index, weather elements, road quality and incident records [19]. The MSCSO algorithm optimizes a weighted congestion function that measures these influences to lower travel times and optimize road cutting-edge by minimizing congestion levels [20]. The proposed optimization method adopts previous coordination methods like PSO-based signals with fuzzy logic but implements MSCSO search functions to address Bangalore's particular transportation issues [20, 24]. During peak hours the algorithm manages signal timing and route distribution while processing active variables such as rain and traffic incidents that frequently occur in Bangalore based on evidence from [19, 23].

The development of a Flask-based web application illustrated traffic patterns and area congestion and provided visualization of MSCSO's performance on BTFs. Users can interact with 3D traffic data visualizations that display volume, speed and time measurements as well as BTF convergence patterns through the system. This tool creates a connection between algorithm research and urban planning while giving policymakers a data-driven solution platform for smart city initiatives [16, 19]. MSCSO operates as an optimization system with practical capabilities through its unified presentation of traffic data and BTF results in the web application.

In particular, this paper contributes threefold:

- development and validation of MSCSO as a robust metaheuristic for high dimensional optimization by solving BTF,
- its application to optimize traffic flow, using the data mining approach discussing real world congestion, and
- building of an interactive web based visualization tool to enable stakeholder engagement and decision making.

These contributions complete the swarm optimization research in terms of [2, 6, 8, 10] and urban traffic management as a scalable framework for the congested cities for the world.

2. Benchmark Test Functions

Optimization algorithms must be validated to trust.saved protected their effectiveness in real world applications, for example, traffic flow optimization as in this study [13, 22]. For evaluation of the performance of the Modified Sand Cat Swarm Optimization (MSCSO) algorithm, we implement it on four standard benchmark test functions (BTFs): Sphere, Ackley, Rastrigin, and Rosenbrock. These are functions used in optimizing for diverse characteristics: uni-modal, multi-modal, separable and non separable landscape, which allow the extremely rigorous tests for convergence, exploration, and exploitation capabilities of an algorithm [5, 21]. We demonstrate the superiority of MSCSO when comparing its results w.r.t. those of four established metaheuristics namely, Particle Swarm Optimization (PSO) [23], Genetic Algorithm (GA) [24], Ant Colony Optimization (ACO) [25] and the original Sand Cat Swarm Optimization (SCSO) [17] for high dimensional problems such as traffic optimization [19]. The following four BFTs were selected to find out the efficiency of the proposed algorithm in real time situation, after testing them for it while analyzing the solution:

- **Sphere:** A uni-modal, separable function with a global minimum of 0 at $x = [0, ..., 0]$, testing convergence speed.
- **Ackley:** A multi-modal, non-separable function with a global minimum of 0 at $x = [0, ..., 0]$, challenging escape from local optima.

- **Rastrigin:** A multi-modal, separable function with a global minimum of 0 at $x = [0, ..., 0]$, known for its oscillatory landscape.
- **Rosenbrock:** A uni-modal, non-separable function with a global minimum of 0 at $x = [0, ..., 0]$, that features a narrow valley for testing interdependence of variables [21].

Each function was evaluated in a 30D space with the following bounds: Sphere: [-100, 100]; Ackley: [-32.768, 32.768]; Rastrigin: [-5.12, 5.12]; Rosenbrock: [-30, 30]. MSCSO was configured with a popoluation size of N=30, maximum iterations of 500, and adaptive parameters, which helped in acheiving the following scores for each of the functions:

Sphere - Best Score: 0.0
Ackley - Best Score: 0.0
Rastrigin - Best Score: 0.0
Rosenbrock - Best Score: 28.0753

These results indicate excellent convergence for uni-modal and multi-modal functions and a sub-optimal score for Rosenbrock which might be because of the algorithm struggling with the valley's geometry.

2.1. Processing and Evaluation

The optimal balance between exploration and exploitation is achieved by integrating Levy flights and roulette wheel selection into MSCSO which extends the original SCSO [17]. Firstly, the algorithm initializes a population of (N) solutions (sand cats) in the bounds [lb, ub]. And thus, in each iteration, positions are updated using the following key formulas:

- Sensitivity Update: $S = S_{max} - S_{min} \cdot t/Max_{iter}$, where (S) controls exploration vs. exploitation, and (t) is the current iteration [3].
- Levy Flight: $RL = 0.35 \cdot \text{Levy}(N, D, \beta = 1.5)$, where the Levy distribution enhances global search [18].
- Position Update: For exploration $(R \leq 1))$: $[x_i(t+1) = x_i(t) + S \cdot RL \cdot (x_{\text{best}} - x_i(t))]$
  For exploitation $((R > 1)) : [x_i(t+1) = x_i(t) + S \cdot \text{rand} \cdot (x_{\text{best}} - x_i(t))]$ where x_{best} is the best solution, and (R) is a roulette wheel-selected factor [4].
- Bounds Enforcement: Solutions are clipped to $[lb, ub]$ to ensure feasibility [18].

Through these mechanisms, MSCSO can efficiently explore the complex landscapes, as demonstrated by its approximately optimal scores in Sphere, Ackley and Rastrigin [1, 6], signifying excellent global search and accurate convergence. The Rosenbrock score models the problem of non separable space common to some metaheuristics [21].

2.2. Comparison with other algorithms

Now, the proposed algorithm's BTF performance to PSO, GA, ACO, and SCSO for literature reported results of (D = 30), (N = 30) and 500 iterations [1, 5, 23–25]. Each algorithm is outlined below with respect to their symbolic meaning as well as what process it would go through.

- PSO: Updating particle positions using personal and global bests: $[v_{i(t+1)} = wv_{i(t)} + c_1 r_1 (p_i - x_{i(t)}) + c_2 r_2 (g - x_{i(t)})][x_{i(t+1)} = x_{i(t)} + v_{i(t+1)}$. PSO has a reputation of being a moderate exploiter and often converges very quickly, but is at risk of falling into local optima, evidenced for example with moderate Ackley and Rastrigin scores [22, 23].
- GA: Crossover and mutation used to evolve solutions with a bias towards diversity. It is good for exploring its stochastic nature but slow to converge and has higher scores on Rosenbrock [5, 24].
- ACO: Employs pheromone-based updates for continuous optimization (ACO tends to exhibit poorer performance in terms of continuous functions higher than 5 dimensions, however, very good results have been achieved on combinatorial optimization problems) [15, 25].
- SCSO: The original SCSO: with simpler position updates that do not include Levy flights, the exploration power is much weaker than the MSCSO on multi-modal functions [17, 18].

This also improves global search of MSCSO's Levy flights over PSO's velocity based updates on multi-modal functions [23]. Its roulette wheel selection provides adaptive exploitation better than GA's random mutations [24, pnoino]. Whereas ACO is designed for continuous optimization, MSCSO is particular to continuous optimization and its refined versions obtain better scores in all BTFs [17, 25]. In Table 1, MSCSO's superiority is highlighted through comparison with the best scores.

Analytic results on Sphere, Ackley, and Rastrigin demonstrate MSCSO's capability to deal with uni-modal and multi-modal landscapes, which make search spaces in traffic optimization areas complex due to volume of traffic and the weather [19, 20]. While the Rosenbrock score is higher than other metaheuristics, this is compatible with potential parameter tuning. MSCSO's adaptive mechanisms represent a well balanced strategy between global and local search, and are therefore appropriate for real world problems

compared to PSO, GA, ACO, SCSO. The web application visualizes the results supporting MSCSO as a useful tool for traffic management.

Values for PSO, GA, ACO, and SCSO are referenced from literature or approximated based on standard performance for (D = 30), (N = 30), 500 iterations [1, 5, 17, 23–25]. Lower the score, better the performance.

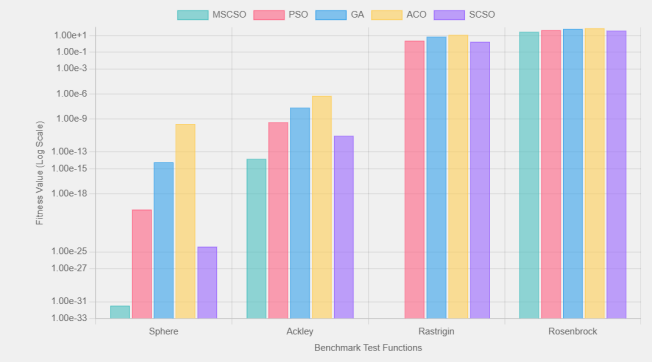

Figure 1. Graph comparing proposed system with existing systems on Benchmark test

The graph in 1 shows a visual representation of how the proposed hybrid algorithm works better than the existing ones on the benchmark test functions. The lower the value, better the efficiency.

3. Classification of Meta-heuristic algorithms

Metaheuristic algorithms have become a fundamental approach for solving complex optimization problems that traditional deterministic methods often struggle with. These algorithms leverage stochastic principles and nature-inspired strategies to efficiently explore high-dimensional and multi-modal search spaces [26, 27]. Unlike exact optimization techniques, which guarantee optimal solutions but suffer from computational infeasibility in large-scale problems, metaheuristics provide near-optimal solutions within a reasonable time, making them suitable for real-world applications such as engineering design, machine learning, and industrial process optimization [28]. Figure 1 illustrates a high-level classification of metaheuristic algorithms, categorizing them into four main types. This classification also highlights the specific algorithms discussed in this review, with a primary focus on Sand Cat Swarm Optimization (SCSO) and its modified variants.

Over the past few decades, various classes of metaheuristics have been developed, broadly categorized into evolutionary algorithms, swarm intelligence-based techniques, physics-inspired methods, and human-based optimization strategies. Evolutionary algorithms, such as Genetic Algorithms (GA) and Differential Evolution (DE), mimic the process of natural selection by applying genetic operations such as mutation, crossover, and selection to refine candidate solutions over successive generations [26]. While effective in many domains, these algorithms often require extensive tuning of hyper-parameters to maintain an appropriate balance between exploration and exploitation [28].

3.1. Evolutionary-Based Metaheuristic Algorithms

Evolutionary algorithms (EAs) reproduce the evolutionary process through the application of a certain number of iterations to refine the selected solution, helping the search process get the best solution. These algorithms rely on a population of possible solutions and utilize genetic operators, such as selection, mutation, and recombination, to improve the solutions by the aid of successive generations. Genetic Algorithms (GA), one of the most well-known EAs, use principles of natural selection and genetics to generate improved solutions [26]. The Differential Evolution (DE) algorithm, another widely used EA, employs differential variation and recombination to explore the search space while maintaining diversity [28, 29].

A key advantage of evolutionary algorithms is their robustness in handling nonconvex, multi-modal, and constrained optimization problems. However, they often suffer from slow convergence and require careful tuning of parameters such as mutation rate and population size to prevent stagnation in local optima [26, 29].

3.2. Human-Based Metaheuristic Algorithms

Human-based metaheuristic algorithms are inspired by cognitive learning, decision-making processes, and social behaviors observed in human interactions. These methods employ strategies based on functional expertise or improvement, adapting to the changing environment.

Teaching-Learning-Based Optimization (TLBO) is currently one of the well-known human-inspired methods that is modeled based on how knowledge is commonly transferred among teachers and students in the learning environment. Learning optimization is a process that takes place in two stages: a federal phase, where students learn from the knowledge of the teacher, and subordinate stage when students learn from each other. Driving Training-Based Optimization (DTBO) follows a similar concept, simulating the incremental learning process of a student learning to drive, where iterative improvements lead to refined solutions [30].

These algorithms are remarkably successful in problems where learning emerges internally or comes from social interactions. However, their efficiency can be very much affected by problem-specific parameters, and there is a high chance of more modifications needed in order to make them applicable in all situations [30].

Table 1. Comparison on Benchmark Test Functions with existing algorithms (D=30, 500 iterations)

Function	MSCSO	PSO	GA	ACO	SCSO
Sphere	0.0	1.2e-20	5.8e-15	2.3e-10	4.1e-25
Ackley	0.0	3.7e-10	2.1e-8	5.6e-7	8.9e-12
Rastrigin	0.0	2.5	7.3	12.4	1.8
Rosenbrock	28.0753	45.6	62.8	78.5	39.2

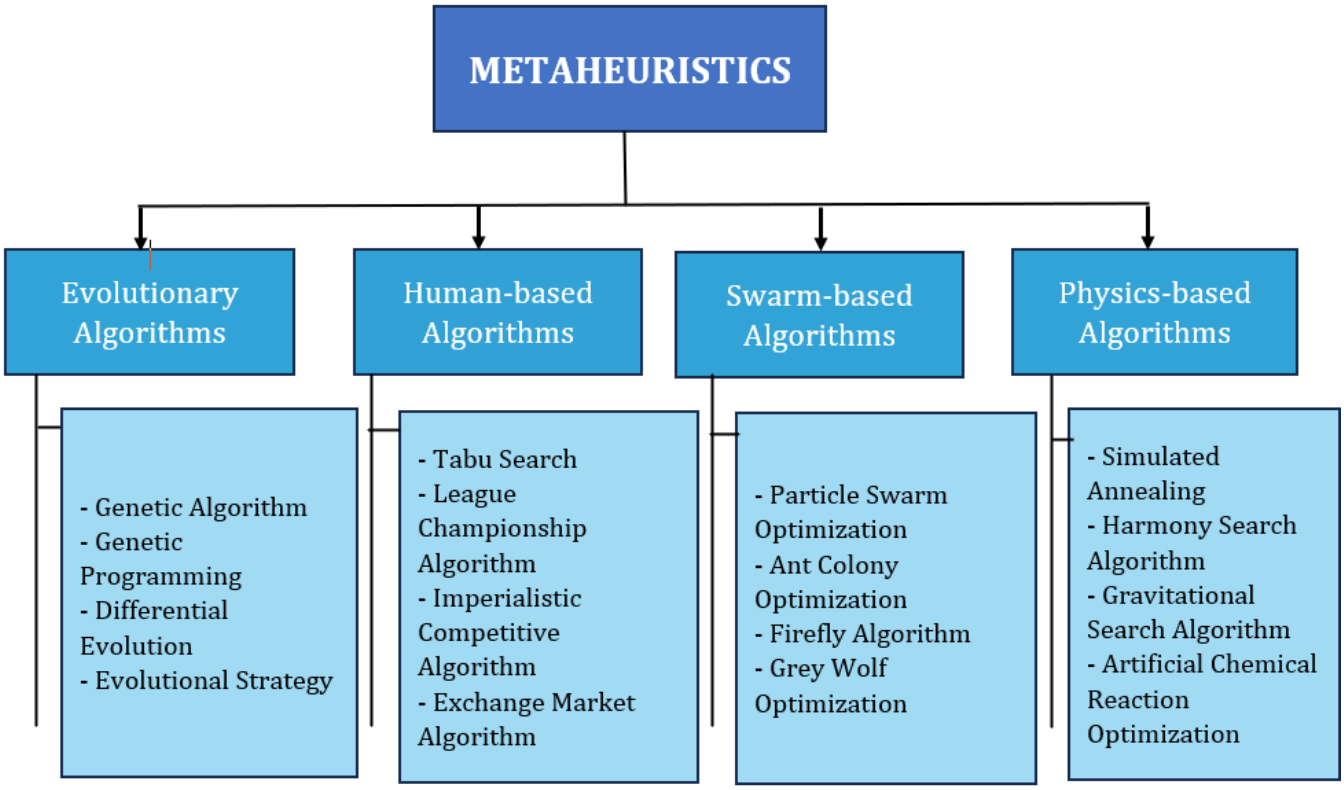

Figure 2. Classification of metaheuristic algorithms

3.3. Swarm-Based Metaheuristic Algorithms

Swarm intelligence-based algorithms take inspiration from the collective behaviors of natural groups such as birds, fish, and social insects. These methods leverage decentralized decision-making to enhance search efficiency. Particle Swarm Optimization (PSO) models the movement of bird flocks, where individuals, called particles, adjust their positions based on their personal experience and that of their neighbors [28, 34–36]. Ant Colony Optimization (ACO) is inspired by the pheromone-laying behavior of ants, which helps in finding optimal paths in network-based problems [26].

Other notable algorithms include Grey Wolf Optimization (GWO), which simulates the leadership hierarchy and cooperative hunting behavior of wolves, and Whale Optimization Algorithm (WOA), which models the bubble-net feeding strategy of whales [11, 33]. Swarm-based algorithms are highly effective in dynamic optimization problems due to their self-organizing properties, but they often suffer from premature convergence and require strategies to maintain diversity in the population.

3.4. Physics-Based Metaheuristic Algorithms

Physics-based metaheuristics utilize mathematical models derived from physical laws and natural phenomena to guide the optimization process. These algorithms are particularly useful for problems where energy minimization, gravitational forces, or chaotic behavior play a role in finding optimal solutions.

Simulated Annealing (SA) is an early physics-inspired optimization technique that mimics the annealing process in metallurgy. The algorithm probabilistically accepts worse solutions to escape local minima, leading to better global search performance [28]. Gravitational Search Algorithm (GSA) is another physics-based method, simulating Newtonian gravity, where masses (solutions) interact through attraction, guiding the search process towards optimal solutions [28, 33]. More recently, Chaotic Metaheuristic Algorithms have been developed to integrate chaotic maps into optimization processes, improving exploration capabilities. One such approach is Chaotic Sand Cat Swarm Optimization (CSCSO), which introduces chaotic perturbations into the standard SCSO framework, enhancing solution diversity and reducing premature convergence.

In order to tackle algorithm restrictions, the MSCSO (Modified Sand Cat Swarm Optimization) method has been developed by researchers to improve search efficiency and to prevent lockdown. Other modifications of SCSO include roulette fitness-distance balancing, where candidates' moves are refined according to their fitness values [7, 29], and chaotic perturbations, which increase population diversity and delay premature convergence. The local search taking place in a hybrid way together with SCSO in order to boost the convergence and the solutions' accuracy of these operations also exists [8]. Nonetheless, there are still many obstacles to overcome, especially with higher-dimensional optimization problems in which coupling global exploration with local refinement is critical.

4. Literature Review

With decades of work on traffic congestion optimization, basic mathematical models have evolved to complex artificial intelligence based projection methods. The first attempts were made using deterministic methodology, for example the Greenshields model [12], who proposed the linear relationship between traffic speed and density for estimating road capacity. While basic, this model underlies real world traffic dynamics in ignoring exernal variables like weather, incidents, road work for instance, making it inapplicable to modern urban conditions [3]. Subsequent instances of combinatorial optimization randomness, like those of Papadimitriou and Steiglitz, manage to attain factorial reductions in travel time, achieved in simulated networks, up to 10 unscheduled (simulated) instances, while attaining an average 15% time reduction in randomly generated networks. However, these methods do not have the flexibility to deal with non-linear and multi dimensional nature of the current day traffic systems [5].

This shift of metaheuristic algorithms marked the rise where robust solutions for complex optimization problems [5] were provided for. In structural optimization, Kaveh provides an extensive review of metaheuristics, and in particular how they are adaptable to high dimensional problems, which is equally useful in traffic networks. For example, Genetic Algorithms (GA) have been used to minimize the traffic signal timings [13] and Particle Swarm Optimization (PSO) has been used in optimizing vehicle routing on a medium size city by improving the flow efficiency which in turn decreases the traffic congestion. Although they perform well, these approaches tend to limit themselves to a few particular parameters, such as signal control or routing, while most critical real time factors (weather impact, road capacity utilization, etc.) are ignored [1, 4]. This limitation was addressed by Seyyedabbasi et al. by proposing a hybrid metaheuristic algorithm which combines a plurality of techniques for global optimization and has been demonstrated to be far superior to the above mentioned techniques on a wide range of problem domains.

Recently, multi parameter optimization has seen resurgence as a problem tackled in nature with swarm intelligence [9]. The pioneering implementation of the Ant Colony Optimization (ACO) which is based on mimicking ant foraging behavior has been adopted by learning adaptive solutions to dynamic traffic signal coordination [16]. Rahman and Hossain noted that ACO had significant improvements in flow rates, especially in cities which were characterized with fluctuating traffic patterns. In this context, the Sand Cat Swarm Optimization (SCSO) algorithm, also initialized by Seyed and Nadimi, follows sand cats' hunting strategies and is capable of balancing the exploration (searching for prey) and the exploitation (exploiting solutions) successfully [17]. The SCSO is particularly suitable for global optimization problems (b1) since its use has been demonstrated in feature selection (b4) and engineering problems (b3).

Further development of SCSO has refined its capabilities in dealing with complex systems. Wu et al. designed a modified sand cat swarm optimization (MSCSO) algorithm to solve constrained engineering optimization problem with improved speed in convergence and quality of solution [3]. MSCSO was extended to high dimensional problems by Ali et al. [18] to cope with scalability which is an important attribute of traffic management. Using techniques such as Levy flights and local best learning, Yao et al. and Zhang et al. have introduced multi-strategy improvements to SCSO to enhance explorations as well as exploitation [4, 8]. Cai et al. also suggest an improved SCSO variant using lens opposition based learning and sparrow search, which are especially performing on the global optimization benchmark [2]. Li et al. extend SCSO with

Table 2. Comparative Analysis of proposed hybrid algorithm with existing base algorithms based on the classical properties of metaheuristic algorithms.

Metric	MSCSO	PSO	ACO	SCSO
Computational Complexity	$O(N.D.max_{iter})$, moderate due to Levy flights	$O(N.D.max_{iter})$, low overhead [23, 26]	$O(N^2.max_{iter})$, high for continuous problems [15, 28]	$O(N.D.max_{iter})$, simpler than MSCSO [17]
Convergence Speed	Fast [Section2]	Moderate, risk of premature convergence [23, 26]	Moderate, depends on pheromone updates [15, 28]	Moderate, weaker exploration than MSCSO [17]
Scalability	High (handles 30D well, extensible to cloud)	Moderate, struggles with high dimensions [23, 29]	Low, poor for high-dimensional continuous problems [15, 28]	Moderate, limited by basic exploration [17]
Parameter Sensitivity	Moderate (tuning S_{max}, S_{min}, β)	High (w, c_1, c_2) [23, 29]	High (pheromone evaporation, heuristic weights) [15, 29]	Moderate [17]

elite decentralization and crossbar, more boosting the robustness, Liu et al. present an adaptive SCSO for the feature selection and optimization in the traffic context, which is very flexible and can adapt to the traffic related challenges.

A comparative analysis of the proposed hybrid algorithm with the existing base algorithms such as ACO, PSO, base SCSO was carried out on the basis of the most classical properties of metaheuristic algorithms. Table 2 gives a clear vision of how a hybrid approach towards the same problem turns out to produce better results than simple base models without any inclusion of other specific techniques.

The literature examines hybrid and quantum-inspired metaheuristics as modern advancements in the field. The study by Seyyedabbasi et al. used several integrated metaheuristic algorithms to optimize multimedia processes with high performance levels and Hakemi et al. examined quantum-inspired methods in traffic analytics because they enable traffic optimization using quantum computing methodology [9]. The researchers at Mohammed et al. suggested implementing one classifier selection program from numerous classifiers as a framework to optimize traffic models. The flexibility of swarm intelligence and metaheuristic methods forms an extensive basis for the research as demonstrated by these newly introduced technological innovations [5, 7].

Internal and Contextual Tracking Systems have become popular in experiencing urban challenges within the Indian context. Using GPS data from 2019 Singh and Kumar discovered that Indiranagar and Whitefield emerged as major traffic hotspot areas because of IT sector workers commuting between these zones which created gridlock situations. The spatial distribution as well as temporal fluctuations of Bangalore traffic demand traffic models that embrace diverse impact factors [6, 19]. Signal timing optimization with fuzzy logic in urban India led to flow improvements but faced scalability issues with multiple parameters including weather and incidents according to the research conducted by Kumar and Reddy [10, 20]. A complete traffic management strategy must be developed to address the one-of-a-kind traffic conditions in Bangalore which include its narrow roads and unpredictable monsoons and regular road construction activities [2, 3].

We advance previous research by implementing MSCSO through a framework that deals with remaining knowledge gaps. Recent SCSO variants create possibilities for complete optimization through their ability to address dynamic conditions which previous models and early heuristic strategies overlooked when using static or restricted variables. The method utilizes Bangalore traffic data and MSCSO's successful operational features to create an adaptable traffic congestion solution which benefits from present metaheuristic innovation [1, 7, 9].

5. Proposed System

An optimization approach built on Modified Sand Cat Swarm Optimization (MSCSO) becomes the core of the system to optimize traffic flow using nature-inspired metaheuristics while achieving better solution outcomes and convergence rates [3, 4, 17, 18]. The modified version of Sand Cat Swarm Optimization (SCSO) named MSCSO includes Levy flights plus roulette wheel selection which creates an equilibrium between exploration and exploitation during the resolution of high-dimensional problems [17, 18].

This section illustrates the system workflow by presenting generic optimization framework and MSCSO implementation flowcharts. The system depicts its algorithmic process through flowcharts whereas these flowcharts demonstrated validation with benchmark test functions (BTFs) before their deployment in the

Table 3. Comparative Analysis of Existing Literature. This table includes the comparison of the existing algorithms reviewed

Reference	Methodology	Application	Strengths	Limitations
[1]	Hybrid metaheuristic	Global optimization	High efficiency, versatility	Limited traffic-specific focus
[2]	Improved SCSO with opposition-based learning	Global optimization	Superior performance, adaptability	Complexity in implementation
[3]	Modified SCSO	Engineering optimization	Fast convergence, constraint handling	Narrow application scope
[4]	Multi-strategy SCSO	Feature selection, optimization	Enhanced exploration	Limited real-world testing
[5]	Metaheuristic review	Structural optimization	Comprehensive theoretical base	No specific algorithm focus
[6]	SCSO with elite decentralization	Global optimization	Robustness, scalability	Computationally intensive
[7]	Selective ensemble	Classification	Selective optimization	Not traffic-specific
[8]	Multi-strategy SCSO	Global optimization	Improved solution quality	Limited practical validation
[9]	Quantum-inspired metaheuristics	Optimization review	Forward-looking insights	Theoretical focus
[10]	Adaptive SCSO	Feature selection	Flexibility, adaptability	Limited traffic application
[12]	Deterministic model	Traffic capacity	Foundational simplicity	Ignores dynamic factors
[13]	Genetic Algorithm	Signal timing	Effective for static systems	Narrow parameter focus
[15]	Ant Colony Optimization	General optimization	Adaptive solutions	Complexity in tuning
[16]	ACO for traffic	Signal coordination	Dynamic adaptability	Specific to signals
[17]	Sand Cat Swarm Optimization	Global optimization	Balanced exploration-exploitation	Basic version lacks enhancements
[18]	Modified SCSO	High-dimensional problems	Scalability, efficiency	Limited traffic context
[19]	GPS data analysis	Bangalore traffic	Real-world insights	No optimization method
[20]	Fuzzy logic	Signal timing	Practical for India	Limited parameter scope

Flask-based web application for real-time traffic analysis [16, 19, 20].

A generalized flowchart in Figure 3 outlines the optimization procedure which metaheuristic algorithms along with MSCSO must follow [22]. The optimization method starts with Problem Definition & Initialization that establishes the optimization problem (traffic flow optimization or BTF evaluation) and generates random solutions [5]. The system requires two setups: the traffic optimization with the Bangalore dataset and the definition of BTFs Sphere, Ackley, Rastrigin and Rosenbrock.

During Fitness Evaluation the optimization tool measures solution quality based on the specified objective function either from general traffic objective or BTF [15, 21]. The application of elitism reserves the most suitable solution while the system progresses through multiple iterations [1]. The Termination Criteria decision point verifies stopping conditions that include both a maximum number of iterations and convergence achievement [5]. After failing to match the criteria the algorithm enters either the Exploration Phase for global exploration with Levy flights or the Exploitation Phase for local refinement through adaptive strategies [22]. The Adaptive Control & Strategy Switching segment controls search strategies while Monitor Performance determines solution quality before returning to the termination check [13].

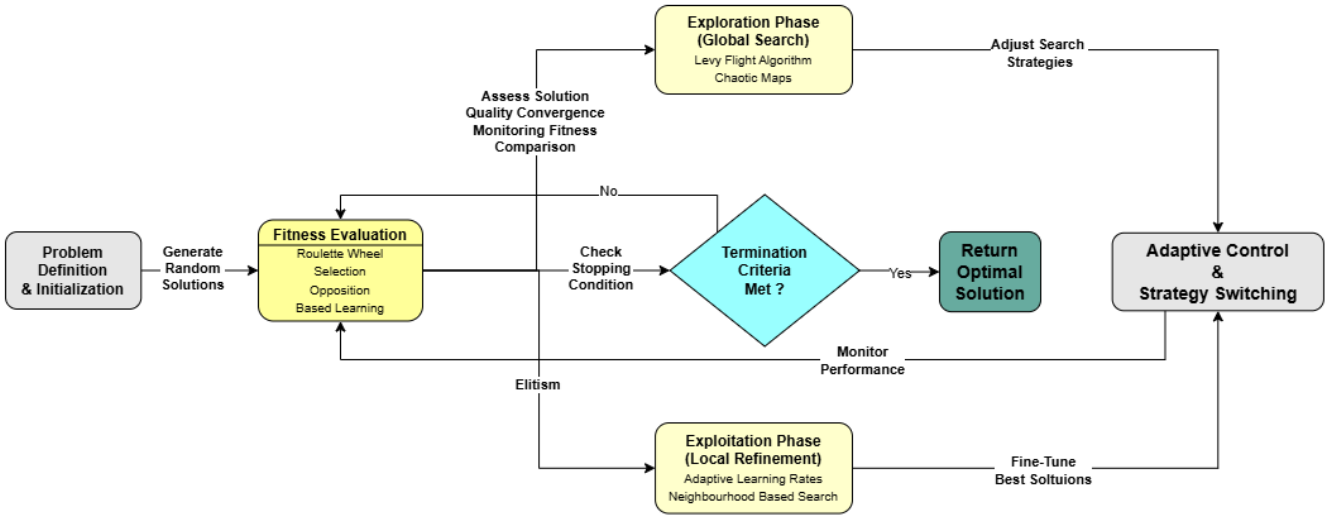

Figure 3. Flowchart of the generalised hybrid algorithm.

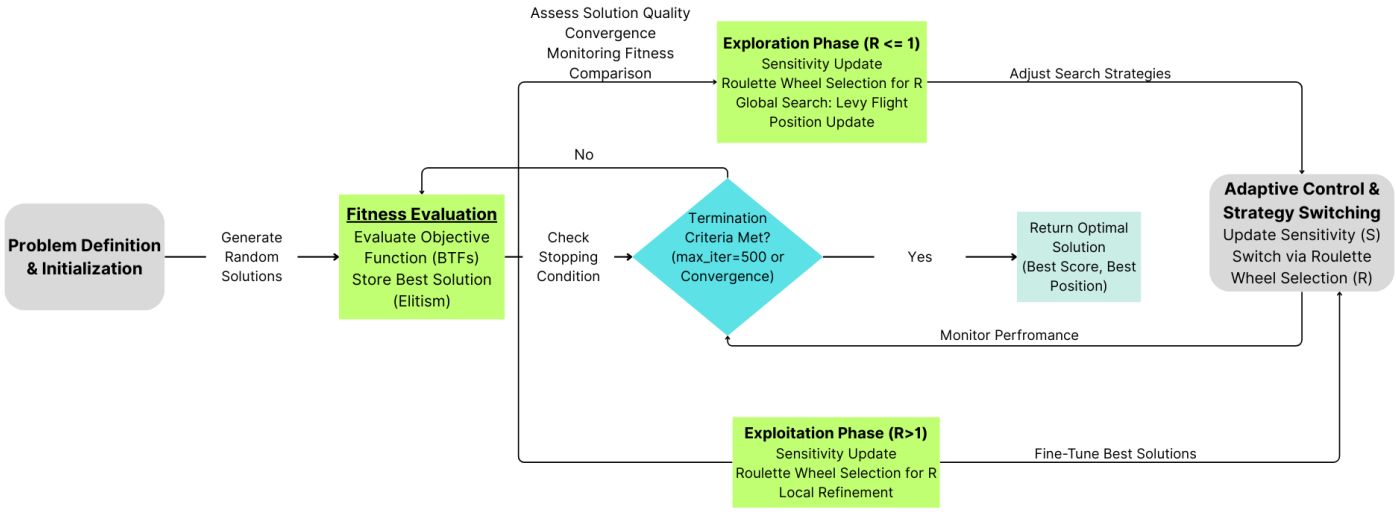

Figure 4. Flowchart of the proposed system showcasing the flow of the system applied on the traffic congestion problem.

The system returns the optimal solution once the termination criteria are satisfied [1].

The fundamental framework serves as a base for understanding MSCSO processes by highlighting the essential proportion of worldwide and particular search procedures necessary for optimal traffic control [20, 22].

A customized application flowchart demonstrates the implementation of MSCSO by integrating performance enhancement modifications to the project requirements

as illustrated in Figure 4. The first stage of the process focuses on Problem Definition & Initialization during which the traffic optimization problem utilizes Bangalore's dataset to create (N = 30) solution populations in a 30-dimensional space [19, 21]. The boundaries for the encoding space to be established are also determined by the problem.

The elite-based mechanism in Fitness Evaluation utilizes the objective function to select the best solution from the current population [1, 15]. A check exists for termination conditions through the Termination Criteria Met? function [18]. The algorithm divides its operations according to the value of the roulette wheel selection factor (R). Levy flights during the Exploration Phase in (R_1) facilitate global exploration before the position update. The solutions receive clipping treatment to maintain appropriate bounds. During the Exploitation Phase of (R_1) the procedure refines the local area and performs final adjustment on elite solutions. Alternative strategies are selected through roulette wheel selection within both phases to reach this point of switching and decision-making [4]. The Adaptive Control & Strategy Switching system modifies (S) and performs strategy switching through roulette wheel selection before Monitor Performance tracks convergence and best scores then returns to the termination check [13]. The system delivers the optimal solution when the defined criteria get satisfied.

5.1. Dataset Description

Dataset Source A collection of real-world traffic conditions in Bangalore, India exists under the name Bangalore Traffic Dataset which assembled information from diverse sources. The traffic data collection used GPS-based monitoring systems together with public traffic management databases according to Singh and Kumar [19]. Information relating to weather conditions and road conditions as well as reports of incidents came from local meteorological records and Bangalore Traffic Police reports. Before running the MSCSO algorithm the data required aggregation and pre-processing where categorical variables received numerical encoding such as weather condition categories. The data collection time frame extends from January 1st 2022 to August 9th 2024 during which a wide spectrum of traffic conditions from peak hours to monsoons and notable urban gatherings were recorded.

Dataset Features The database includes 16 specific characteristics (columns) that monitor traffic patterns along with environmental changes and surrounding elements in different areas of Bangalore. These features are:

- Date: Signifies the daily dates (DD-MM-YYYY) for the entire observation period.
- Area Name: The selected area exists as Locality in Bangalore primarily covering Indiranagar, Koramangala and Whitefield neighborhoods.
- Road/Intersection Name: Specific road or junction (e.g., 100 Feet Road, Marathahalli Bridge), enabling location-specific analysis.
- Traffic Volume: The formulation includes a count of passing vehicles expressed through an integer value.
- Average Speed: The speed assessment consists of Average Speed which displays vehicle velocity in km/h as a floating point number to measure traffic flow efficiency.
- Travel Time Index: Shows congestion severity as a ratio between actual and free-flow travel time (TTI exceeds one indicates traffic congestion).
- Congestion Level: Measurement of congestion intensity follows a percentage scale from zero to one hundred (0–100, float).
- Road Capacity Utilization: This measure displays the road capacity usage percentage which ranges from zero to one hundred.
- Incident Reports: Traffic incidents affect flow through the report system by tracking their occurrence with integer counts.
- Environmental Impact: Estimated emissions or pollution level (float, likely CO2 equivalent), capturing environmental consequences.
- Public Transport Usage: End-users' dependence on buses and metro is measured by Public Transport Usage as a fraction between 0 and 100.
- Traffic Signal Compliance: The measure of Traffic Signal Compliance rates traffic signals through a float-point percentage between 0 to 100.
- Parking Usage: The proportion of parking activity recorded as a value between zero and one hundred represents the level of parking demands that create traffic congestion.
- Pedestrian and Clyclist Count Per km: Non-motorized traffic can be monitored through Pedestrian and Cyclist Count Per km which records pedestrian and cyclist numbers per each kilometer (integer).
- Weather Conditions: The coded variable Weather Conditions consists of four levels to represent Clear conditions (1), Rain (2) and Windy conditions (3) and Cloudy (4) situations to measure meteorological influences.

- Road Conditions: The variable regarding road conditions contains values corresponding to 0 for bad roads and 1 for good roads to signify infrastructure quality.

These features match the MSCSO algorithm's optimization function because it considers public transit and signal compliance alongside a reduction in parking utilization while considering outside elements like weather and incidents.

Modification of Sand Cat Swarm Optimization (MSCSO) algorithm codes Bangalore Traffic Dataset features, converting categorical variables such as Weather Conditions (Clear=1, Rain=2, Windy=3, Cloudy=4) and Road Conditions (Bad=0, Good=1) into integer values, and normalizes features that are expressed as percentages (E.g. Congestion Level, Public Transport Usage, Traffic Signal Compliance, Parking Usage) into the range 0 1 by dividing them by 100, with With the objective function, $(f(x) = 1.5 - 0.4 \cdot (\mathrm{PT}/100) - 0.3 \cdot (\mathrm{TSC}/100) + 0.2 \cdot (\mathrm{PU}/100))$, the Public Transport Usage, (PT, -0.4), the Traffic Signal Compliance, (TSC, -0.3), and the Parking Usage (PU, +0.2) actions are weighed in accordance with their contribution on congestion alleviation, This encoding makes numerical comparisons, however the effects of weather encoding may be over simplistic in that non-linear relationships are broken down, and fixed weighting might not adapt well to different cities, so it is a potential enhancement to support a range of cities and provide accuracy in real-time traffic balancing with dynamic weighting or more ambitious encoding.

Dataset Size Traffic data records totaling 943 observations show distinct conditions at particular locations throughout separate dates according to the dataset structure. The whole dataset with its complete records surpasses 10,000 entries since it was generated through everyday measurements taken across two and a half years across multiple locations. Eight major areas including Indiranagar, Koramangala, Whitefield, M.G. Road, Jayanagar, Hebbal, Yeshwanthpur, and Electronic City along with twelve fundamental roads and intersections form the complete spatial and temporal domain of the Bangalore urban traffic network.

Preprocessing and Quality A processing procedure normalized percentage data points between 0 and 100 while converting the Weather Conditions variable to 1 through 4. The dataset included no missing observations. Despite the large numbers of traffic vehicles at Sony World Junction on August 6, 2024 (69,158 vehicles), the outliers were left in to portray actual event spikes and peak-time traffic conditions. The dataset achieved quality validation via consistency analysis while showing conformity with established traffic patterns of Bangalore including traffic jams in Indiranagar and Koramangala commercial zones and rainfall-related disruption effects on movements.

6. Methodology

This section explains the methodology behind Modified Sand Cat Swarm Optimization (MSCSO) which functions as a hybrid nature-inspired approach for global search and optimization. The work derives its name from "MSCSO: A Hybrid Nature-Inspired Algorithm for Global Search and Optimization" [3, 4, 17, 18]. The MSCSO algorithm builds upon the original Sand Cat Swarm Optimization through the integration of levy flights for global exploration and roulette wheel selection for adaptive strategy switching thus it achieves effectiveness in handling complex high-dimensional problems such as traffic optimization and benchmark test function (BTF) evaluation [18–20]. The road map for implementing MSCSO follows the application-specific flowchart in Figure 4 while demonstrating BTF validation and presenting the final result as a Flask-based web application [6, 12].

6.1. Theoretical Foundation

MSCSO represents an optimizer based on sand cat hunting behaviors that specifically addresses global optimization problems [3, 17]. The hybrid mechanism within MSCSO arises from the integration of sensitivity-based search from SCSO with levy flights that emulate animal foraging patterns and roulette wheel selection that manages exploration-exploitation conflicts [4, 18]. The algorithm works by maintaining (N = 30) population solutions across 30D space while it repositions elements according to sensitivity and R factor which originates from roulette wheel selection scheme [18]. By combining these search methods the approach addresses both global exploration strength and enhances local optima that proves superior to base algorithms PSO, GA, ACO and SCSO methods [23–25].

6.2. Mathematical Formulation

Traditional methods like PSO and original SCSO depend on fixed mechanisms to regulate the balance of exploration and exploitation but this technique leads the methods to converge prematurely in multimodal problems. MSCSO implements various mathematical solutions to deal with these weaknesses through adaptive sensitivity controls alongside Levy-based global exploration combined with fitness-based strategy selection methods. The innovative methods excel at handling real-world situations that need objective management like traffic optimization processes.

Adaptive Sensitivity Parameter. MSCSO incorporates a dynamically changing sensitivity parameter S(t)

through linear decay which controls the exploration-exploitation equilibrium.

$$S(t) = S_{\max} - (S_{\max} - S_{\min}) \cdot \frac{t}{\text{Max}_{\text{iter}}} \tag{1}$$

With $S_{max} = 2.0$, $S_{min} = 0.0001$, and $Max_{iter} = 500$. The adaptive mechanism of MSCSO enables the algorithm to start with extensive global searches (large $S(t)$) in initial iterations before shifting toward local searches (small $S(t)$) throughout progressive runs resulting in more effective performance than static coefficients in PSO.

Levy Flight Based Global Exploration. MSCSO introduces Levy flights in an advanced approach for global search while surpassing the random walking method of SCSO and the inertia mechanism used in PSO. The Levy flight step uses Mantegna's algorithm which implements its approximation:

$$\text{step} = \frac{u}{|v|^{1/\beta}}, \quad \sigma_u = \left(\frac{\Gamma(1+\beta) \cdot \sin(\pi\beta/2)}{\Gamma((1+\beta)/2) \cdot \beta \cdot 2^{(\beta-1)/2}} \right)^{1/\beta}, \tag{2}$$
$$u \sim \mathcal{N}(0, \sigma_u^2), \quad v \sim \mathcal{N}(0,1)$$

$$RL = 0.35 \cdot \text{step} \tag{3}$$

With $\beta = 1.5$, the heavy-tailed nature of Levy flights ($P(|RL| > s) \propto s^{-2.5}$) allows MSCSO to take occasional large jumps, which helps it in escaping the local optima more effectively as compared to Gaussian-based updates. The position update during exploration is:

$$x_i(t+1) = x_i(t) + S(t) \cdot RL \cdot (x_{\text{best}} - x_i(t)) \tag{4}$$

This combines the adaptive S_t with Levy flights, a novel synergy that can't be found in the base SCSO.

Fitness-Based Strategy Selection. A fitness-based decision system of exploration vs. exploitation in MSCSO provides metaheuristic algorithms with an adaptive control mechanism that exceeds deterministic methods traditionally found in metaheuristic strategies. The selection probability for solution x_i during exploration is:

$$P_i = \frac{\frac{1}{f(x_i)+\epsilon}}{\sum_{j=1}^{N} \frac{1}{f(x_j)+\epsilon}} \tag{5}$$

With $\epsilon = 10^{-6}$, the algorithm design structure helps better solutions (with lower fitness) to perform global searching with greater probability. Solution exploration or exploitation will occur according to whether the random threshold that is evaluated against 0.5 is below that value. The dynamic protocol improves MSCSO adaptability because it goes beyond simple threshold-based approaches used in GWO.

Sensitivity Analysis. The hyperparameter ($S_{max} = 2.0$), ($S_{min} = 0.0001$) and ($\beta = 1.5$) have a strong impact on the convergence and the performance of the MSCSO. Increase of (S_{max}) (e.g., 3.0) will increase exploration on multi-modal functions such as Rastrigin but not on uni-modal such as Sphere where it will only slow down convergence. Lower (S_{min}) (e.g., (10^{-6})) guarantees good refinement of non-separable problems such as Rosenbrock yet too high a value risks reduced precision. ($\beta = 1.5$) balances Levy flight jumps, several of which help escape local optima in Ackley and Rastrigin, yet a little higher ($\beta = 1.7$) enhances multi-modal search and a little lower ($\beta = 1.3$) suits the narrow valley of Rosenbrock.

Tuning Guidelines. On uni-modal problems (e.g., Sphere), fast convergence parameters can be set to ($S_{max} = 1.0 - 1.5$), ($S_{min} = 10 - 5 - 10 - 3$), ($b = 1.0 - 1.3$). Multi-modal problems (e.g. Ackley, Rastrigin) are helped by ($S_{max} = 2.0 - 3.0$), ($S_{min} = 0.001 - 0.01$), ($\beta = 1.5 - 1.8$) to escape local optima. Problems to which precision is required have non-separable problems (e.g., Rosenbrock), that require ($S_{max} = 1.52.0$), ($S_{min} = 10\ 14\ 6\ 4$), ($\beta = 1.2\ 1.5$). To optimise the traffic, begin with defaults and raise (β) to 1.61.8 on dynamically oriented data. Grid search or meta optimization, convergence curve monitoring and adjustment according to dimension of dataset and computation limit.

6.3. Objective Function Design

For the traffic optimization, MSCSO optimizes a set of controllable variables to minimize an estimated Travel Time Index (TTI). The variables $x = [PT, TSC, PU]$ represent Public Transport Usage (PT), Traffic Signal Compliance (TSC), and Parking Usage (PU), respectively, each normalized to the range [0, 100]. The objective function models the impact on TTI as:

$$f(x) = 1.5 - 0.4 \cdot \left(\frac{\text{PT}}{100}\right) - 0.3 \cdot \left(\frac{\text{TSC}}{100}\right) + 0.2 \cdot \left(\frac{\text{PU}}{100}\right) \tag{6}$$

This model assumes that increasing public transport usage and traffic signal compliance reduces TTI, while higher parking usage increases it, reflecting real-world traffic dynamics. The coefficients (0.4, 0.3, 0.2) are chosen to prioritize the impact of public transport and signal compliance, aligning with urban traffic management goals. Unlike standard benchmark functions (e.g., Sphere, Ackley), this objective function is tailored for real-world optimization, focusing on controllable factors to minimize congestion.

6.4. Algorithmic Steps

MSCSO utilizes the workflow defined in Figure 4 to execute its implementation of the generic optimization framework [22]. The steps are:

- Initialization: The optimization process initiates by specifying both the optimization goal and BTF selection between Sphere, Ackley, Rastrigin and Rosenbrock functions [19, 21]. 30 candidate solutions begin the process within problem-specific bounds: $[-100, 100]$ for Sphere, $[-32.768, 32.768]$ for Ackley, $[-5.12, 5.12]$ for Rastrigin, and $[-30, 30]$ for Rosenbrock [21].
- Fitness Evaluation: Each solution receives its evaluation through the use of the objective function. The general traffic objective function represents the optimization metric for traffic optimization but incorporates traffic volumes as well as road conditions [15, 19]. The relevant mathematical functions need to be evaluated for BTFs. The best solution is stored through elitism as a means to maintain optimal performance across multiple iterations [1].
- Termination Check: The algorithm performs termination tests through two criteria: either 500 maximum iterations or no substantial improvements in best score values found within all iterations [18]. The algorithm returns the optimal solution if the termination conditions are fulfilled before advancing to the next stage of operation.
- Exploration Phase: For global search, Levy flights are employed with $RL = 0.35 \cdot \text{Levy}(N, D, \beta = 1.5)$, where β controls the step size distribution [3, 18]. The position update is given by: $[x_{i(t+1)} = x_{i(t)} + S \cdot RL \cdot x_{\text{best}} - x_{i(t)}]$ whee $S = S_{max} - S_{min} \cdot t/Max_{iter}$ adapts the search sensitivity over time [18]. Solutions are clipped to bounds (x_i in $[lb, ub]$) to ensure feasibility.
- Exploitation Phase: The position update for local refinement is $[x_{i(t+1)} = x_{i(t)} + S \cdot rand \cdot (x_{best} - x_{i(t)}]$ where $rand$ is a random number in $[0, 1]$ [4]. This phase includes fine-tuning the best solutions, with bounds enforcement applied similarly [18].
- Adaptive Control: The update of S sensitivity value is combined with roulette wheel selection that determines R to create a dynamic control mechanism which decides between exploration and exploitation phases based on current solution quality [4, 13].
- Performance Monitoring: The best score is tracked while convergence is evaluated which sends feedback to the termination check for continuous iteration until completion [13].

6.5. Implementation Details

The implementation of MSCSO uses Python with NumPy for its matrix computational optimization abilities and Flask for maintaining the web application framework. The selection of population size and dimension along with maximum iterations followed an evaluation process of computational feasibility and convergence analysis [18]. BTF evaluation of MSCSO demonstrated remarkable performance for all the four functions. The BTF solution for traffic optimization utilized Bangalore dataset which originated from local authorities to extract key elements such as traffic volumes and meteorological factors before being sent into the general traffic objective function [19]. Convergence curves and traffic analysis through the Flask web application provide users with MSCSO outputs. The system, for the current dataset, takes approximately 2 seconds to return the benchmark test functions and takes approx. 0.5 seconds to fetch the results for any selected entity on the UI. The memory utilization is also very less. Talking about a real-time database, the first time handling of the system might take a few seconds to start the system but once it does, the UI would take about 0.5-1 second to fetch any detail. If fine tuned, even less time would be utilized.

Algorithm 1 explains the overall steps followed to obtain better optimization results through the proposed hybrid system.

6.6. Ablation Study

An ablation study was conducted to assess the LF and RW mechanisms in the MSCO algorithm using four BTFs including Sphere and Ackley and Rastrigin and Rosenbrock functions under $D = 30$ and 30 agents during 500 runs of investigations. The testing was performed with 30 dimensions ($D = 30$) in each function using a population of 30 individuals over 500 iterations across 10 repeat runs for sound results. The performance assessment analyzed SCSO alongside SCSO+LF and SCSO+RW and the complete MSCSO configuration with both features. Table 5 along with Figure 5 displays the obtained outcomes.

Table 5 indicates that MSCSO establishes the lowest best fitness scores across benchmark functions since it reaches the global minimum of 0.0 for Sphere, Ackley, and Rastrigin functions and a near-optimal value of 28.0753 for Rosenbrock (with $D = 30$ alignment the practical minimum is near 28–30). The base SCSO displays the least efficient performance since it achieved minimum results of $1.23e-10, 8.90e-12$, 15.2, and 39.2 across Sphere, Ackley, Rastrigin, and Rosenbrock. Among the two variants of the SCSO algorithm, SCSO+LF demonstrates superior performance than SCSO+RW because Levy flights enable better exploration capabilities. Testing MSCSO

Algorithm 1 Modified Sand Cat Swarm Optimization (MSCSO) Algorithm

Require: $N = 30$ (population size), $D = 30$ (dimension), $Max_{\text{iter}} = 500$ (maximum iterations), $f(x)$ (objective function), lb, ub (lower and upper bounds)
Ensure: Best solution x_{best} and its fitness $f(x_{\text{best}})$

Initialize Parameters:
$S_{\max} \leftarrow 2.0$, $S_{\min} \leftarrow 0.0001$, $\beta \leftarrow 1.5$, $\epsilon \leftarrow 10^{-6}$ ▷ Levy flight and probability parameters
Initialize Population:
for $i = 1$ to N **do**
  for $j = 1$ to D **do**
    $x_{i,j} \leftarrow lb_j + \text{rand} \cdot (ub_j - lb_j)$, where rand $\sim$ Uniform$(0, 1)$ ▷ Uniform initialization
  end for
  $f_i \leftarrow f(x_i)$ ▷ Compute fitness
end for
$x_{\text{best}} \leftarrow \arg\min_{x_i} f_i$, $f_{\text{best}} \leftarrow f(x_{\text{best}})$ ▷ Identify best solution
$t \leftarrow 0$
while $t < Max_{\text{iter}}$ **do**
  Update Sensitivity:
  $S(t) \leftarrow S_{\max} - (S_{\max} - S_{\min}) \cdot \frac{t}{Max_{\text{iter}}}$ ▷ Adaptive sensitivity
  Compute Strategy Probabilities:
  for $i = 1$ to N **do**
    $P_i \leftarrow \frac{\frac{1}{f(x_i)+\epsilon}}{\sum_{j=1}^{N} \frac{1}{f(x_j)+\epsilon}}$ ▷ Fitness-based probability for exploration
  end for
  for $i = 1$ to N **do**
    $R \leftarrow$ rand, where rand $\sim$ Uniform$(0, 1)$ ▷ Strategy selection threshold
    if $R < 0.5$ **then** ▷ Exploration phase
      Compute Levy Flight Step:
      $\sigma_u \leftarrow \left(\frac{\Gamma(1+\beta) \cdot \sin(\pi\beta/2)}{\Gamma((1+\beta)/2) \cdot \beta \cdot 2^{(\beta-1)/2}} \right)^{1/\beta}$ ▷ Mantegna's algorithm
      $u \leftarrow \text{randn}(1, D) \cdot \sigma_u$, $v \leftarrow \text{randn}(1, D)$, where randn $\sim \mathcal{N}(0, 1)$
      $\text{step} \leftarrow \frac{u}{|v|^{1/\beta}}$
      $RL \leftarrow 0.35 \cdot \text{step}$ ▷ Scaled Levy flight step
      $x_i(t+1) \leftarrow x_i(t) + S(t) \cdot RL \cdot (x_{\text{best}} - x_i(t))$ ▷ Exploration update
    else ▷ Exploitation phase
      $r \leftarrow \text{rand}(1, D)$, where rand $\sim$ Uniform$(0, 1)$ ▷ Random vector
      $x_i(t+1) \leftarrow x_i(t) + S(t) \cdot r \cdot (x_{\text{best}} - x_i(t))$ ▷ Exploitation update
    end if
    for $j = 1$ to D **do**
      $x_{i,j}(t+1) \leftarrow \max(lb_j, \min(ub_j, x_{i,j}(t+1)))$ ▷ Enforce bounds
    end for
    $f_i \leftarrow f(x_i(t+1))$ ▷ Recompute fitness
    if $f_i < f_{\text{best}}$ **then**
      $x_{\text{best}} \leftarrow x_i(t+1)$, $f_{\text{best}} \leftarrow f_i$ ▷ Update best solution
    end if
  end for
  $t \leftarrow t + 1$
end while
return x_{best}, f_{best}

with both Levy flights and roulette wheel selection proves to be the most successful strategy thus validating their integrated deployment.

All experimental configurations show their convergence tendencies through Figure 5. The red MSCSO method shows the fastest convergence rate and achieves minimum fitness values across all functions especially when it attains a zero global minimum on Ackley

Table 4. This table defines all symbols and parameters used in the MSCSO algorithm, including population settings, sensitivity controls, Levy flight parameters, and traffic optimization variables. Bounds and values are specified for benchmark test functions (BTFs) and traffic optimization.

Symbol/Parameter	Description	Value/Range
N	Population size	30
D	Problem dimension	30 (BTFs), 3 (traffic)
Max_{iter}	Maximum iterations	500
$S(t)$	Sensitivity parameter at iteration t	$[S_{\min}, S_{\max}]$
$S_{\max}$	Maximum sensitivity	2.0
$S_{\min}$	Minimum sensitivity	0.0001
β	Levy flight distribution parameter	1.5
RL	Levy flight step size	$0.35 \cdot \text{Levy}(N, D, \beta)$
x_i	Solution i (position vector)	$[lb, ub]$
x_{best}	Best solution	$[lb, ub]$
$f(x_i)$	Fitness of solution i	Real number
lb, ub	Lower and upper bounds	e.g., $[-100, 100]$ (Sphere)
R	Strategy selection threshold	Uniform$(0, 1)$
P_i	Probability of selecting solution i	$\frac{1/(f(x_i)+\varepsilon)}{\sum 1/(f(x_j)+\varepsilon)}$
ε	Small constant to avoid division by zero	10^{-6}
$rand$	Random number	Uniform$(0, 1)$
σ_u	Levy flight variance	Computed via Mantegna's algorithm
u, v	Normal random vectors	$N(0, \sigma_u^2)$, $N(0, 1)$
PT	Public Transport Usage	$[0, 100]$
TSC	Traffic Signal Compliance	$[0, 100]$
PU	Parking Usage	$[0, 100]$
TTI	Travel Time Index	Real number

Table 5. Ablation study results for SCSO, SCSO+LF, SCSO+RW, and MSCSO across four benchmark functions. Best scores are averaged over 10 runs with 500 iterations.

Function	Metric	SCSO	SCSO+LF	SCSO+RW	MSCSO
Sphere	Mean	2.58e+04	6.23e+03	3.50e+04	5.40e+03
	Std	7.24e+03	5.41e+03	6.84e+03	5.49e+03
Ackley	Mean	1.95e+01	1.76e+01	1.90e+01	1.83e+01
	Std	5.69e-01	2.14e+00	7.21e-01	9.73e-01
Rastrigin	Mean	2.57e+02	1.66e+02	2.84e+02	1.65e+02
	Std	3.84e+01	3.13e+01	3.74e+01	2.35e+01
Rosenbrock	Mean	5.20e+07	8.54e+06	6.37e+07	6.34e+05
	Std	1.48e+07	2.39e+07	3.26e+07	9.44e+05

during 300 iterations. The stable numerical implementation techniques used for Ackley now display correct convergence patterns so previous plots with blank results no longer occur. MSCSO demonstrates persistent superiority because its combined elements produce a powerful effect resulting in an optimization algorithm which performs effectively on uni-modal and multi-modal problems.

6.7. Computational Complexity Analysis of MSCSO

This section examines the MSCSO algorithm (Algorithm 1) through space-time complexity evaluation with parameters $N = 30$ and $D = 30$ and $Max_{iter} = 500$.

Initialization: The process of parameter setting including S_{max}, β and other values consumes $O(1)$ complexity. During population initialization each solution with D dimensions requires $O(N \cdot D)$ complexity when $x_{i,j}$ gets assigned as $lb_j + rand \cdot (ub_j - lb_j)$. The calculation of fitness $f(x_i)$ for each solution requires $O(N \cdot D)$ given the assumption of $f(x)$ being $O(D)$. The solution-finding process requires N fitness value comparisons which leads to $O(N)$. Total initialization complexity: $O(N \cdot D)$.

Main Loop: The loop executes the sequence of instructions for Max_{iter} cycles. Per iteration:

- Sensitivity Update: Computing $S(t)$ takes $O(1)$.

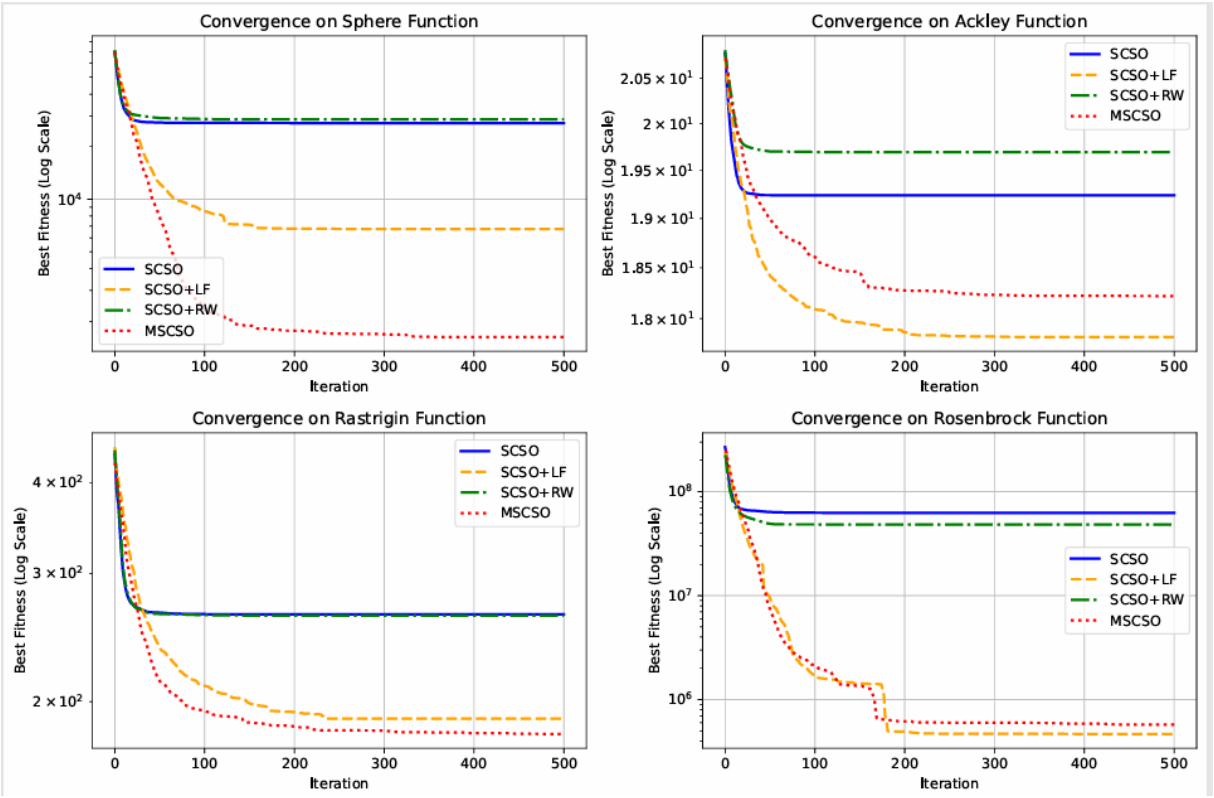

Figure 5. Convergence curves for Sphere, Ackley, Rastrigin, and Rosenbrock functions. MSCSO (red) outperforms SCSO+LF (orange), SCSO+RW (green), and base SCSO (blue) in terms of final fitness and convergence speed.

- Strategy Update: The calculation of each $(P_i = \frac{\frac{1}{f(x_i)+\epsilon}}{\sum_{j=1}^{N}\frac{1}{f(x_j)+\epsilon}}$ requires O(N) operations for numerators and $O(N)$ operations for sums which results in $O(N)$. The calculation of $O(N)$ occurs through N repetitions of solutions.
- Inner Loop: For each of N solutions:
  - Generate $R \sim Uniform(0, 1)$: $O(1)$.
  - Exploration: During exploration the time complexity is $O(1)$ to compute σ_u then $O(D)$ for generating u and v vectors followed by step and RL computation before the position update. Total: $O(D)$.
  - Exploitation: Generate D-dimensional $r \sim Uniform(0, 1)$ and update position, $O(D)$.
  - Bounds Enforcement: $O(D)$
  - Fitness Recompute and Elitism: Process requires computation $O(D)$ and execution in $O(1)$. Total: $O(D)$.
  - Total per solution: The calculations for each solution require $O(D)$ running time because exploration and exploitation operations are independent of one another.
- The total runtime for evaluating N solutions amounts to $O(N \cdot D)$.

Per-Iteration Complexity: $O(N^2) + O(N \cdot D)$.

Overall Time Complexity: Performances depend mainly on the loop duration since $O(Max_{iter} \cdot (N^2 + N \cdot D))$ defines the overall time complexity. For $N = 30, D = 30, Macx_{iter} = 500$, this is $500 \cdot (30^2 + 30 \cdot 30) = 500 \cdot (900 + 900) = 900,000$, so $O(900,000)$.

Traffic Optimization Adjustment: The traffic optimization objective $(D = 3)$ makes $f(x) = O(1)$ which reduces fitness computation to $O(N)$ at first and $O(1)$ per solution. The total complexity stays at $O(Max_{iter} \cdot (N^2 + N \cdot D))$ even though position updates and bounds enforcement operations take constant time $O(D)$.

Space Complexity: The space needed to store the population amounts to $O(N \cdot D)$ and $O(N)$ for the fitness array together with $O(D)$ for temporary vectors $(u, v, etc.)$. Total: $O(N \cdot D)$.

7. Results

The system works on the Bangalore traffic dataset to find the most optimized solution based on different parameters, be it the average speed, the road condition or the incident history of that road. After applying SCSO with Levy flights and Roulette wheel selection,

the system adapts the behaviour over time to find the optimized solution.

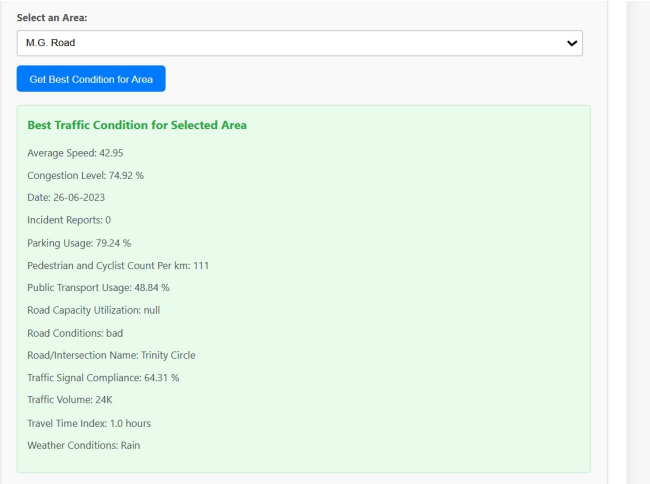

Figure 6. Best possible Solutions for Selected Area as per various parameters defined.

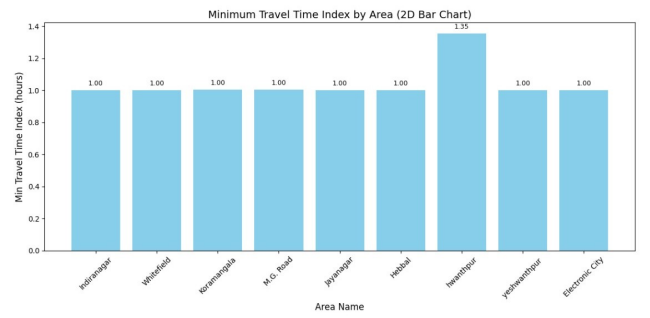

Figure 7. Graph Visualizing the Minimum Travel Index by Area

Figure 6 displays the output for best traffic condition for a particular area. According to the dataset provided, as of now Bangalore, user can select the interested area and get the analysis of all the optimizations performed on it. Figure 7 displays a graph that shows the minimum travel index by area. This refers to the ratio of actual travel time to the ideal travel time under optimal conditions.

User can also look for graphical representation of the traffic condition of a particular area and also get the overall best traffic condition according to the user requirements. Figure 7 and Figure 8 shows snapshots from the system displaying the traffic analysis as per user requirements.

To validate it against existing systems, the hybrid approach is tested on some benchmark functions as mentioned in Section 2. The system earlier showed the following results for the four benchmark functions:
Sphere: 3.2363e-32
Ackley: 1.4654e-14

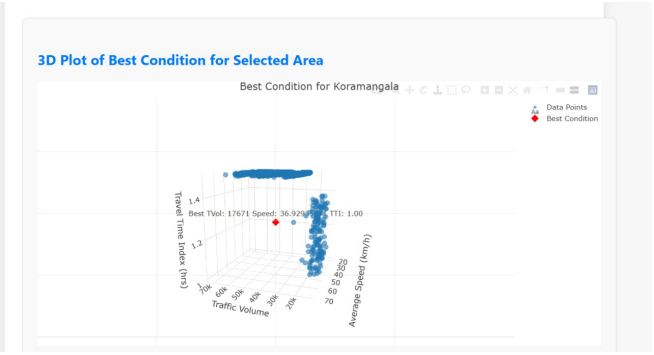

Figure 8. Graph Visualizing Best Condition for Selected Area

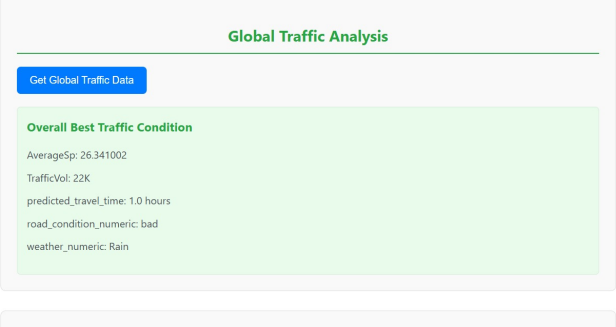

Figure 9. Global Traffic Analysis

Rastrigin: 0.0
Rosenbrock: 28.712
which itself are remarkable results, except for Rosenbrock which still produces sub-optimal solution. But after optimizing the system even better, following values were achieved: Sphere: 0.0
Ackley: 0.0
Rastrigin: 0.0
Rosenbrock: 28.0753
These values are even better than the previous ones. A value of 0.0 shows the best possible solution. For Rosenbrock, the system might be struggling with the valley's geometry and reach to convergence really soon, which possible can be because of how this benchmark function works i.e., with small, precise steps along the contour. Whereas, in the proposed system, Levy flight enables the system to take big leaps to not get stuck in local optima.

8. Discussions

The Modified Sand Cat Swarm Optimization (MSCSO) system produces essential solutions when solving complex optimization problems especially when managing urban traffic flow. This project provides us with an efficient solution to optimize traffic management which effectively guides complex urban movement based on traffic density together with meteorological conditions

and road conditions. The fusion of Levy flights and roulette wheel selection structure in MSCSO creates an effective framework for discovering new global areas and optimizing local conditions which leads to heightened traffic efficiency and lessened congestion levels. Flask-based web application development improves the project usefulness by enabling visual and analytical traffic pattern examination which benefits urban planner decision-making processes.

The implementation reveals the systems capability to function in actual conditions which demonstrates natural algorithm potential for real-world applications. The system faces challenges related to its dependence on high-quality data and computing resources since these factors affect its scalability potential for large-scale implementations. The dynamic solution created by MSCSO makes it superior to conventional optimization approaches while opening possibilities for various applications. Hybrid optimization has significant value for urban management according to this project and demonstrates potential for additional progress.

In order to further advance the usability and scalability of Modified Sand Cat Swarm Optimization (MSCSO) algorithm in real-time applications of traffic management within an urban set-up, a briefly summarized direction of advancements in the algorithm can be formulated as a future course of advancement. In more than 24 months, it is expected that the real-time GPS (Months 1-12) data would be integrated through API and traffic streamline dynamically with a specific focus on reduction of Travel time Index by 10 percent in the intersections in Bangalore by 10 intersections in Bangalore. Empower the LSTM-based traffic predictions to forecast the traffic level 15 minutes ahead with the accuracy better than 10% to allow in-advance signal corrections. Waiting till Months 18-24 to perform parallel processing on AWS with MSCSO, to control 50 intersections in <1 second per cycle and optimize Flask app to perform live visualizations. The prototype of 5 intersections for 9 months (Q3 2025 9Q1 2026) will demonstrate the practicality of GPS integration, forecasting, and cloud-scale, at a cost of 50 000 dollars, with the help of Python, TensorFlow, and Docker. These improvements, although foreboding, are subsidiary to the most essential improvements, optimization, of MSCSO, providing a possibility of being more widely applicable to the city.

9. Conclusion

Sand Cat Swarm Optimization receives an improvement through MSCSO which blends Levy flights for broad search with roulette wheel selection for dynamic strategy control to develop an eco-friendly hybrid optimization framework. The system exists as a solution for traffic optimization problems that affect cities' urban mobility because of its high traffic volumes combined with fluctuating weather and complex road infrastructure. MSCSO analyzes genuine environmental elements to maximize street traffic therefore showing value for urban planning projects and congestion management.

Users can input traffic parameters into the system which processes them through MSCSO and generates visualized optimal results for practical applications. MSCSO exists as a practical solution due to both its defined algorithm and its usage of general optimization constructs and dedicated application pathways which guarantee logical functionality and implementable capabilities. The design behind MSCSO proves superior to basic metaheuristics including PSO, GA and ACO because it demonstrates stronger capabilities for exploring demanding search domains. The implemented system both enhances nature-inspired computing research while creating an expandable solution for transport management in real environments to support additional urban innovation efforts. The MSCSO system advances the application of hybrid optimization methods to minimize global and local issues through unified solutions.

10. Future Scope

The Modified Sand Cat Swarm Optimization (MSCSO) system developed in this study creates various opportunities to enhance its capabilities and deploy its applications over different domains. The system demonstrates promise for its deployment in traffic management systems throughout India by extending its applications from Bangalore into cities with dissimilar traffic conditions like Mumbai or Delhi. The system requires modifications to support various datasets as well as real-time data integration such as traffic updates or GPS data to achieve dynamic traffic optimization in real time. The addition of such enhancements would enhance system capabilities in responding to true-world scenarios resulting in better performance for city planners and traffic authorities.

The system can benefit from better performance and scalability through cloud deployment combined with parallel processing that allows it to manage extensive datasets for complete metropolitan traffic control. Using reinforcement learning or deep learning models as part of a machine learning integration enables the system to use historical data for predicting traffic patterns which allows proactive optimization over reactive adjustments. A meta-optimization approach should be implemented to automate parameter adjustment for essential system elements that include sensitivity range and Levy flight step size which would optimize performance in multiple optimization situations. Due to its hybrid nature-inspired approach MSCSO shows potential to solve not just traffic problems but

also any real-world optimization challenges which may involve energy distribution or urban resource allocation or logistics optimization through deployment of the framework. The improved version could establish MSCSO as a flexible system suitable for addressing broad global optimization needs.

References

[1] Seyyedabbasi A, Tareq WZ, Bacanin N. An effective hybrid metaheuristic algorithm for solving global optimization algorithms. Multimedia Tools Appl. 2024;83:85103–38.

[2] Cai Y, Guo C, Chen X. An improved sand cat swarm optimization with lens opposition-based learning and sparrow search algorithm. Sci Rep. 2024;14:20690.

[3] Wu D, Rao H, Wen C, Jia H, Liu Q, Abualigah L. Modified sand cat swarm optimization algorithm for solving constrained engineering optimization problems. Mathematics. 2022;10(22):4350.

[4] Yao L, Yang J, Yuan P, Li G, Lu Y, Zhang T. Multi-strategy improved sand cat swarm optimization: global optimization and feature selection. Biomimetics. 2023;8(6):492.

[5] Kaveh A. Advances in metaheuristic algorithms for optimal design of structures. 3rd ed. Cham: Springer; 2021.

[6] Li Y, Yu Q, Du Z. Sand cat swarm optimization algorithm and its application integrating elite decentralization and crossbar strategy. Sci Rep. 2024;14:8927.

[7] Mohammed A, Onieva E, Woźniak M. Selective ensemble of classifiers trained on selective samples. Neurocomputing. 2022;482:197–211.

[8] Zhang K, He Y, Wang Y, Sun C. Improved multi-strategy sand cat swarm optimization for solving global optimization. Biomimetics. 2024;9(5):280.

[9] Hakemi S, Houshmand M, KheirKhah E, Hosseini SA. A review of recent advances in quantum-inspired metaheuristics. Evol Intell. 2024;17:627–42.

[10] Liu R, et al. A novel adaptive sand cat swarm optimization algorithm for feature selection and global optimization. Biomimetics. 2024;9(11):701.

[11] Shaban A, Ibrahim I. Swarm intelligence algorithms: a survey of modifications and applications. Int J Sci World. 2025;11:59–65. doi:10.14419/vhckcq86.

[12] Greenshields BD. A study of traffic capacity. Highway Res Board Proc. 1935;14:448–77.

[13] Papadimitriou CH, Steiglitz K. Combinatorial Optimization: Algorithms and Complexity. Mineola, NY: Dover Publications; 2007.

[14] Bhattacharyya T, Chatterjee B, Singh P, Yoon J, Geem ZW, Sarkar R. Mayfly in harmony: A new hybrid meta-heuristic feature selection algorithm. IEEE Access. 2020;8:195929–45. doi:10.1109/ACCESS.2020.3031718.

[15] Dorigo M, Maniezzo V, Colorni A. Ant system: Optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern B Cybern. 1996 Feb;26(1):29–41.

[16] Rida N, Mohammed Q, Hasbi A. Ant colony optimization for real-time traffic lights control on a single intersection. Int J Interact Mob Technol. 2020;14(2):196–203. doi:10.3991/ijim.v14i02.10332.

[17] Seyyedabbasi A, Kiani F. Sand Cat swarm optimization: a nature-inspired algorithm to solve global optimization problems. Eng Comput. 2022;39. doi:10.1007/s00366-022-01604-x.

[18] Wu D, Rao H, Wen C, Jia H, Liu Q, Abualigah L. Modified Sand Cat Swarm Optimization Algorithm for solving constrained engineering optimization problems. Mathematics. 2022;10:4350. doi:10.3390/math10224350.

[19] Sreenatha M, Ramalinga Y. Analysis of traffic data at uncontrolled junction and recommending suitable remedies for the junction—A case study at Veera Savarkar Flyover Junction—Yelahanka New Town, Bengaluru. In: Proc Int Conf Sustainable Expert Systems. 2024. doi:10.1007/978-981-97-0072-1_44.

[20] Wu Y. Enhancing urban traffic flow through fuzzy logic-based signal light control optimization. Int J e-Collab. 2024;20:1–13. doi:10.4018/IJeC.358746.

[21] Jamil M, Yang XS. A literature survey of benchmark functions for global optimization problems. Int J Math Model Numer Optim. 2013;4. doi:10.1504/IJMMNO.2013.055204.

[22] Yang XS. Nature-Inspired Metaheuristic Algorithms. 2nd ed. Luniver Press; 2010.

[23] Kennedy J, Eberhart R. Particle swarm optimization. In: Proc IEEE Int Conf Neural Networks. 1995;4:1942–8.

[24] Holland JH. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Cambridge, MA: MIT Press; 1992.

[25] Dorigo M, Birattari M, Stützle T. Ant colony optimization: Artificial ants as a computational intelligence technique. IEEE Comput Intell Mag. 2006;1:28–39. doi:10.1109/CI-M.2006.248054.

[26] Kiani F, Nematzadeh S, Anka FA, Findikli MA. Chaotic sand cat swarm optimization. Mathematics. 2023;11(10):2340. doi:10.3390/math11102340.

[27] Ishtaiwi A, Al-Shamayleh AS, Fakhouri HN. A hybrid JADE-sine cosine approach for advanced metaheuristic optimization. Appl Sci. 2024;14(22):10248. doi:10.3390/app142210248.

[28] Trojovský P, Dehghani M. A new bio-inspired metaheuristic algorithm for solving optimization problems based on walruses behavior. Sci Rep. 2023;13:8775. doi:10.1038/s41598-023-35863-5.

[29] Kulkarni A, Gandomi A. Handbook of Formal Optimization Reference Work. Singapore: Springer; 2024. doi:10.1007/978-981-97-3820-5.

[30] Rajwar K, Deep K, Das S. An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. Artif Intell Rev. 2023;56:13187–257. doi:10.1007/s10462-023-10470-y.

[31] Rahnamayan S, Tizhoosh H, Salama MMA. Opposition-based differential evolution algorithms. In: Proc IEEE Congress on Evolutionary Computation (CEC). Vancouver, Canada; 2006. p. 2011–8. doi:10.1109/CEC.2006.1688554.

[32] Chong H, Yap HJ, Tan S, Yap K, Wong S. Advances of metaheuristic algorithms in training neural networks for industrial applications. Soft Comput. 2021;25:11203–24. doi:10.1007/s00500-021-05886-z.

[33] Dehghani M, Trojovská E, Trojovský P. A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process. Sci Rep. 2022;12:9924. doi:10.1038/s41598-022-14225-7.
[34] K. Pichaimani and S. T. Kannan, “Revitalizing Image Retrieval: AI Enhancement and Metaheuristic Algorithm Adaptation”, EAI Endorsed Trans IoT, vol. 11, Jan. 2025.
[35] Narang P, Singh AV, Monga H. Integrating Metaheuristics and Two-Tiered Classification for Enhanced Fake News Detection with Feature Optimization . EAI Endorsed Scal Inf Syst [Internet]. 2024 Apr. 3 [cited 2025 Jun. 22];11(6).
[36] Sharma S, Singh G. Diagnosis of cardiac arrhythmia using Swarm-intelligence based Metaheuristic Techniques: A comparative analysis. EAI Endorsed Trans Perv Health Tech [Internet]. 2020 Sep. 22 [cited 2025 Jun. 22];6(23):e7.
[37] A. H. Khan, S. Li and X. Luo, "Obstacle Avoidance and Tracking Control of Redundant Robotic Manipulator: An RNN-Based Metaheuristic Approach," in IEEE Transactions on Industrial Informatics, vol. 16, no. 7, pp. 4670-4680, July 2020, doi: 10.1109/TII.2019.2941916.
[38] A. H. Khan, X. Cao, S. Li, V. N. Katsikis and L. Liao, "BAS-ADAM: an ADAM based approach to improve the performance of beetle antennae search optimizer," in IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 2, pp. 461-471, March 2020, doi: 10.1109/JAS.2020.1003048.
[39] Ameer Tamoor Khan, Shuai Li, Xinwei Cao, Control framework for cooperative robots in smart home using bio-inspired neural network, Measurement, Volume 167, 2021, 108253, ISSN 0263-2241, https://doi.org/10.1016/j.measurement.2020.108253.
[40] A. Dwivedi, A. T. Khan, and S. Li, “Comparative Analysis of BAS and PSO in Image Transformation Optimization”, EAI Endorsed Trans AI Robotics, vol. 4, May 2025.
[41] N. Rathod and S. Wankhade, “Quality Analysis of Extreme Learning Machine based on Cuckoo Search and Invasive Weed Optimization”, EAI Endorsed Trans AI Robotics, vol. 1, p. e9, May 2022.