International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I5P110

Volume 74 Issue 5, 148-168, May 2026

Original Article

Incorporation of Reinforcement Learning in Ant Colony Optimization Algorithm: Mathematical Analysis

Nami Susan Kurian[1], B.Rajesh Shyamala Devi[2]

[1,2]*Department of Electronics and Communication Engineering, Hindustan Institute of Technology and Science Chennai, Tamil Nadu, India.*

[1]*Corresponding Author : namisusan7@gmail.com*

Received: 06 October 2025 Revised: 17 February 2026 Accepted: 28 February 2026 Published: 30 May 2026

Abstract - *The research investigates the incorporation of Reinforcement Learning (RL) techniques into the metaheuristic Ant Colony Optimization tuned Traveling Salesman Problem (ACO-TSP) algorithm for data collection in Wireless Sensor Networks (WSNs) using the Mobile Sink (MS), aiming to enhance adaptiveness, intelligence, and decision-making efficiency. RL is an approach to machine learning where the algorithm learns using a reward-punishment technique, and the agent makes decisions through repeated interactions with the environment. The primary constraint of WSN is its limited energy, which results in challenging implementations, and hence, competent utilization of resources is required to ensure network longevity. Traditional methods in wireless sensor networks follow scheduled sleep, predefined routes, low adaptability, and no learning capability. Reinforcement learning maximizes the network lifetime, improves data collection, learns from the environment to handle dynamic topologies, which in turn reduces human interaction. In this article, a mathematical analysis of how to use reinforcement Q-Learning in ACO to find the optimal path is presented. Additionally, an analysis on how the mobile sink traverses through the Q-learning-based scheduled best path is done, and the suggested approach, Ant Colony Optimization with Mobile Sink and Q-learning algorithm (ACOMS-Q), is compared with prior research on different metrices, and it is found to be effective. In ACOMS-Q, the reinforcement learning algorithm learns and finds the active nodes based on node behavior such as buffer occupancy and energy level over time, reducing the tour length of the mobile sink, ensuring network longevity, and reducing delay.*

Keywords - *Ant Colony Optimization, Mobile Sink, Pheromone Level, REINFORCEMENT Q-LEARNING, Traveling Salesperson Problem.*

1. Introduction

With the self-configured spatially distributed sensor nodes, wireless sensor networks are used abundantly in most of the real-time applications like environmental monitoring, industry automation, precision agriculture, security, military, and many more [1]. The motes or sensor nodes are positioned and scattered over the sensory field to obtain specific data about the environmental or physical conditions. The collected data is then aggregated and forwarded to nearby nodes based on the category of routing algorithm used. The gathered sensory information is either obtained by the individual sensor nodes and pushed to the base station, or it is collectively obtained from various sensor nodes by a clustering mechanism and sent to the predefined sink station. In both scenarios, the sink node (control station or base station) remains static, and data forwarding to the sink node results in a funneling effect. To overcome the limitation of static sink, the mobile sink concept was introduced, where a mobile node enables the collection of data directly from the nodes, which subsequently reduces communication overhead and energy consumption. However, it is a computationally challenging issue to find and locate the finest and optimal path for the mobile sink. To address this, a bio-inspired algorithm called Ant Colony Optimization (ACO), which follows a probabilistic method for path optimization in wireless sensor networks, is proposed [2]. The metaheuristic algorithm, ACO, mimics the foraging behavior of ants and uses two factors – pheromone level and heuristic information for discovering the shortest optimized path. Incorporation of the mobile sink (mobility agent) concept into ACO results in discovering the best path, resulting in less delay and higher delivery rates. ACO offers a solution for the Traveling Salesman problem (TSP) [3]. Furthermore, the reinforcement learning results in providing more accuracy and a powerful structure for adaptive decision making that maximizes long-term performance under uncertainty [4]. Reinforcement learning learns optimal behaviors from the autonomous agents (sensor nodes) through trial and error by its interaction with the environment. By combining Q-learning with ACO, nodes can be dynamically prioritized according to learned policies, and the mobile sink traversing path can be modified to prioritize high-utility or critical nodes, improving network longevity and energy

efficiency. Additionally, the hierarchical approach of clustering the sensory field by separating the network into multiple groups named clusters, and each individual cluster is assigned with a Cluster Head (CH), acts as the leader, and other nodes act as cluster members join the head, significantly improves the energy efficiency.

Recent advancements in hybrid intelligent systems have demonstrated the potential of integrating Reinforcement Learning (RL) with metaheuristic optimization algorithms to improve decision-making in dynamic environments. In this context, RL offers a prevailing framework for dynamic decision-making by learning optimal policies from environmental feedback based on the action. When fused with Ant Colony Optimization, RL can introduce adaptive behavior in the path-selection process, allowing artificial ants to dynamically refine their strategy based on cumulative rewards rather than depending only on static pheromone updates.

This synergy enables the network to better respond to real-time state variations, such as energy capacity or node energy levels, data buffer status, or environmental constraints—especially relevant in domains like wireless sensor networks, robotics, and resource-constrained optimization.

The research work proposes a mathematically grounded incorporation of reinforcement learning within the ACO structure. The hybrid model leverages state information to derive priority values that influence node selection, while Q-learning is employed for updating action values depending on the feedback obtained from the environment. The interaction between pheromone updates and reward-based learning provides a novel perspective on how learning rate or discount factor as tuning parameters affect convergence and exploration-exploitation trade-offs in ACO. Mathematical analysis of the proposed approach suggests the potential to enhance scalability, adaptability, and solution quality in complex optimization tasks. Furthermore, the proposed algorithm, ACOMS-Q, is executed and compared with prior research by taking different parameters into consideration.

The research gap identified is as follows: In a typical WSN, the sink node is static, and the nodes near the sink handle more traffic, resulting in a hotspot problem leading to reduced network lifetime. To curb this, a mobile sink was introduced that traverses the network and collects the data following the trajectory. However, finding the best trajectory is a challenging task while maintaining efficiency in clustering, scheduling, and routing to obtain improved network lifetime and reduced energy consumption.

Problem statement: There are many studies predominantly done on clustering, routing, scheduling, and mobile sink. Clustering divides the sensory network into multiple clusters for energy efficiency, but parameters chosen for cluster head selection are found to be inefficient in many research works. Even though many data aggregation schemes are presented, network delay and longevity are not enhanced to a predictable level [5]. There are numerous scheduling protocols; latency is still an issue [6, 7]. Even though mobile sink routing [8, 9] reduces the overhead on cluster heads, finding the best trajectory intelligently and dynamically to provide less network delay and reduced network lifetime is challenging.

To address the research gap, the following Research Questions (RQ) are formulated. Using the improved ACO algorithm with the incorporation of RL (ACOMS-Q), the mobile sink travels through the sensory field and collects the aggregated data.

RQ (1) How can the mobile sink trajectory be found efficiently, and how will it contribute to the efficiency in terms of network lifetime?

RQ (2) How efficiently can scheduling be done using the Q-learning-based RL algorithm?

RQ (3) How are clustering, scheduling, and routing adopted in the proposed algorithm, and how well does it outperform the traditional methods?

Taking the gap and research questions into consideration, the following are the key contributions of the research over existing methods.

- The research contributes to extending the longevity of the sensory network by reducing the energy consumption through improved clustering and duty cycling.
- The research contributes an efficient clustering with three key metrics-residual operational energy, node weight, and the distance to conserve energy.
- The research contributes a mobile sink approach that curbs the hotspot problem.
- The research contributes Q-learning RL-based scheduling to find the active cluster heads with an enhanced bio-inspired ACO algorithm for routing using a mobile sink, resulting in reduced delay and improved network lifetime.
- The research contributes to reduced tour length in collecting data by finding the active cluster heads.

The research work is further structured into the following parts. Section II highlights the background work along with the summary of prior approaches. Section III explains the methodology of incorporating Q learning in ACO with MS.

Section IV provides the mathematical analysis, followed by Section V, which discusses the experimental analysis and comparisons with various metrices. Section VI includes the conclusion, limitations, and scope of future work.

2. Related Works

Numerous research studies investigate providing improvements in energy and network lifetime for WSN. This background work section provides a description of previous works based on three processes: clustering, scheduling, and routing. Table 1 shows the summary of prior works done in enhancing the routing mechanism in WSN. The contributions, aims, and limitations of each protocol are tabulated. Table 2 provides the types of Reinforcement Learning (RL) algorithms with their merits and demerits.

Artificial Bee Colony (ABC) is competitive in solving optimization problems over other metaheuristic algorithms in WSNs. The limitation is the poor exploitation capabilities and low convergence rate due to inefficiency in the search equation. To overcome the drawback, an improved ABC algorithm, iABC, was proposed, which picks the finest cluster head followed by a scheduling task that optimizes the CH in WSN [10]. By incorporating four phases in the iABC algorithm – initial setup phase, Employed (worker) Bee Phase, probabilistic selection phase with Onlooker Bee, Scout Bee (replacement) Phase, the final evaluation cost is calculated using three metrices - remaining energy, minimum power required for transmission, and separation or path length between the cluster head and mobile sink.

LEACH [11] proposes a classical cluster-driven data aggregation path-finding mechanism in WSN. The protocol follows a tri-phase – broadcast, configuration, and operational phase. With the head of the group as cluster head, other nodes organize themselves to fit into any one cluster. The protocol introduces a randomized rotation of the cluster head with high energy, such that the battery is not drained. Provided a scheduled time by the Cluster Head (CH) to the nodes within the cluster (cluster members) for transmission, aggregated data is collected and sent to the control station by the CH. The method distributes the energy usage among the nodes and reduces energy dissipation. However, the direct long-distance transmission between the gateway as base station and CH may lead to more energy utilization, and it may affect the network longevity.

In HEED [12], the group head is elected according to the primary parameter, as available energy, and the secondary parameter, as proximity even distribution of CHs to balance the overall load in the sensory network. The primary parameter is used to set up the cluster heads, and the additional secondary parameter is used as a tiebreaker. Secondary parameter depends on cluster size and power levels. The protocol undergoes constant iterations irrespective of network density or diameter. HEED provides a multihop inter-cluster network when the density model, the range of the cluster, and the transmission range are known. Clustering overhead and poor mobility are considered limitations of the protocol. A nature-inspired metaheuristic computational optimization paradigm introduced in [13], called Ant System, is based on the foraging behavior of the ant. Solutions are constructed probabilistically upon the two factors - (a) heuristic information and (b) pheromone level. An algorithm is powerful to handle the combinatorial optimization problems, provides parallelism, routing effectiveness, and is flexible in adapting to various problem domains. The limitations include convergence speed, parameter tuning, and computational cost.

ACO-MSS overcomes the funnel effect (nodes near the sink are burdened with a large volume of sensory data) by incorporating mobile agent data collection tuned to the ACO algorithm. In [14], ACO-MSS divides the deployment region into discrete grids – (a) feasible grids, (b) infeasible grids. In all the feasible grids, the center point comprises the mobile sink data points needed for path construction by ACO. It initializes pheromone trails using a greedy solution, then employs artificial ants to construct sink routes probabilistically depending on pheromone levels and energy-aware heuristics. The pheromone matrix is updated locally after each ant's solution and globally for the best-found path, iteratively evolving towards movement strategies that balance energy consumption across sensor nodes and extend network operational duration.

There are three different phases to perform the operation of the EAPC [15] path construction algorithm. The phases include initialization, collecting point selection, and path construction. The algorithm works by finding the data acquisition points, followed by the trajectory, and data is gathered from the points that are overloaded. Prim's algorithm is used in the first phase to construct a minimum spanning tree. Mobile sink originates from the central station or base station and collects information from the selected points. Path determination is carried out using the convex polygon algorithm. The limitation is that the tour length is larger compared to ACO-MSS, which leads to data loss and delay.

ACO with clustering and a mobile sink is implemented in [16], where the network is segmented into different groups, and CHs are assigned based on leftover energy and separation distance. Improved ACO algorithm uses the metric as distance between cluster heads along with pheromone level and heuristic information for finding the next cluster head and to find the optimum mobility traversal path.

The limitation is that scheduling is inefficient and may lead to premature failure of nodes. ACO-MSPD [17] allows visiting only the finest set of locations called Rendezvous Points (RP) for gathering the sensed data to improve the network longevity. The traveling pathway of the mobile sink is found by a based algorithm. Directed Spanning Tree is constructed to neglect replication of data and also to avoid ambiguity.

The tree construction process finds the forwarding weight of the sensor node individually, which helps the algorithm to

obtain a suitable set of rendezvous points. ACO-MSPD shows an advantage over ACO-MSS as it works well with varying data rates. In [18], ACO with a reinforcement learning algorithm is used for resource utilization and load balancing. Resources are allocated in the network dynamically based on current system conditions and demand for the workload. RL approach operates closely with the environment to understand the best policy for resource allocation, but finds limitations in responding to dynamic environments.

In [19], an extended ACO with a mobile sink algorithm is proposed for aggregation of data with optimal cluster head selection to locate the minimum path. The cluster head is selected by prioritizing the leftover energy to curb the delay and energy usage. The ACO-TSP algorithm is incorporated to find the traversing route of the mobile agent at periodic intervals. But the parameters chosen for the cluster head selection are found to be inefficient and may lead to reduced network lifetime.

In [20], to avoid the data loss, inefficient data collection, and energy-hole issues, RL with a mobile sink model was proposed for dynamic collection of data by inducing learning to obtain the shortest path. The CH selection prioritizes the remaining energy as a metric, and dynamic CH selection is also induced to improve the network lifespan. RL approach locates the shortest way, and the updated route information is provided to the mobile sink, which follows the trajectory. The effectiveness of the algorithm in finding the shortest path is limited as it relies only on the RL approach.

In [21], large-scale data collection with multiple sinks is introduced, and the algorithm finds the number of mobile sinks required by combining three strategies, particularly clustering, local MS trajectory construction, and global MS route (trajectory) construction. Additionally, clustering is done using a gap static approach. Each cluster is assigned a local mobile sink, and it travels using the greedy closest insertion algorithm to get the data. The global mobile collector traverses and obtains data from each of the local mobile sinks.

The limitation is on the Complexity of the process involved in the algorithm for obtaining network longevity. Also, there are many works carried out in 3D WSN networks using unsupervised learning [22]. To curb the limitations of traditional LEACH, the LEACH-DFAIRNN algorithm was proposed with a neural network and fusion algorithm, but the algorithm is found to be complex [23].

Table 1. Comparison of existing protocols

Method	Contribution	Aim	Limitations
iABC [10]	Finest CH selection for data transfer	To find the best CH using an optimization algorithm	Lacks in communication overhead
LEACH [11]	Single-hop, randomized CH selection based on probability	To increase the network longevity through clusters	Residual energy is not considered
HEED [12]	CH election with consideration of residual energy	To distribute cluster heads evenly	Higher overhead
Ant System [13]	Solution to the Traveling Salesman Problem	To solve combinational optimization problems	Slow convergence speed
ACO-MSS [14]	Finding the best trajectory and tree construction for data gathering	To improve network lifetime using a mobile sink	Simple mobility model
ACO-MSPD [17]	Finds RPs, prevents hotspot formation, and considers energy	Finding the best traversing path for the mobile sink	Path length is large, and delay is high
ACO-Mobile Sink with Clustering [19]	Cluster head election considers leftover energy	To select the finest cluster head for mobile sink travel	Metric selection is insufficient
RL-Mobile Sink [20]	Induced learning to obtain the best path	To provide efficient data collection in WSN	Only RL is used for finding the shortest path
Multiple Mobile Sink [21]	Incorporation of local and global mobile sink collectors, routing uses ACO.	To find the number of mobile sinks, routing using ACO	Process complexity and difficulty in network management
LEACH-DFAIRNN [23]	Improved network performance with the integration of the fusion algorithm	To use a Recurrent Neural Network (RNN) for decision making	High algorithm complexity, reduced lifetime

Table 2. Types of reinforcement learning algorithms

Category	Algorithm	Description	Pros	Cons
Model-Free (Value-Based) [24]	Q-Learning	No need for a model to learn optimal action-values (Q-values)	Simple, versatile, guaranteed convergence	Does not suit for large state spaces
	SARSA	On-policy approach	Safer exploration, learning based on on-policy	Slower convergence and stuck in a suboptimal policy
Model-Based RL [25]	Dyna-Q, MCTS	The model is built in advance for planning and policy optimization	Learning efficiency, adaptability	Inefficiency in modeling accurately
Model-Free (Policy-Based) [26]	REINFORCE	Policy learned using neural networks	Good for continuous action spaces	High variance, slower convergence
Model-Free (Actor-Critic)	Actor-Critic	Combines value and policy	Outperforms REINFORCE in learning	Complexity in implementation
Deep RL [27]	Deep Q-Network (DQN)	Use of a neural network for Q value estimation	Suitable for large state spaces	Abundant data is required for computation
	Proximal Policy Optimization (PPO)	Follows the policy gradient method	State-of-the-art for many tasks	High Complexity, needs tuning

All the aforementioned algorithms contribute to clustering, scheduling, and routing in different ways. However, the novelty of the proposed protocol, ACOMS-Q, is that it incorporates an efficient clustering mechanism, RL Q-learning-based scheduling, and enhanced routing using a modified ACO-TSP algorithm to achieve a high network lifetime and less delay.

The proposed system finds the active CHs using Q-learning-based scheduling and modified ACO traverses only over the active cluster heads, reducing delay and extending network lifetime.

The efficient parameter selection for clustering, scheduling, and routing makes the algorithm superior to the existing ones.

In the proposed routing protocol,

- Optimum cluster head selection for data aggregation is done accurately by taking the effective tri-metric factors such as node weight, residual leftover energy, and separation distance. The fitness function is calculated to find CHs.
- Duty Cycling is carried out by introducing a self-learning approach – the RL Q-learning algorithm to find the active cluster heads.
- Buffer occupancy and energy level are considered for finding the active CH.
- Routing is done using an improved ACO-TSP algorithm with the incorporation of Q values to visit the active cluster heads.

3. Methodology

For the network design, assumptions are made and stated below:

- All sensor nodes are assumed to be isomorphic/homogeneous and are static in nature after deployment, except the mobile sink.
- The mobile agent is assumed to have unlimited energy with no resource constraints and can move in any trajectory in the network.
- The entire network is cluster-based, and the cluster head is tasked with data collection, aggregation, and forwarding to the mobile sink.
- Artificial ants are not entirely blind, and they have some memory.

The traditional ACO, Ant Colony Optimization, is a combinatorial optimization approach that solves combinatorial problems by replicating the behavior of ants probing for food. Artificial agents, called ants, explore possible solutions and leave virtual pheromones that influence subsequent ants' choices.

Over time, the algorithm strengthens paths that lead to better solutions by increasing their pheromone intensity, allowing it to steadily converge on an optimal or near-optimal result through a strategic trade-off between exploration (diversity in the search for paths) and exploitation (refining the solutions). The algorithm is motivated by the natural behavior of ants, i.e., their foraging behavior. With the positive feedback, the algorithm works effectively as it also provides distributed computation with the use of a greedy approach. It

provides fast discovery of good and acceptable solutions, avoiding premature convergence in early stages. ACO is applied to the Traveling Salesman Problem to solve the routing problems. The agents are the ants that imitate the characteristics of real ants. The moving ants lay the substance called pheromone trails, and the ants reinforce the edges with their pheromone after finding the high probability path. The path with high pheromone intensity will be chosen by the ant, and the best path will be found. This process provides a positive feedback loop such that the probability of selecting the optimized path length increases directly with the count of ants that follow the same path earlier. An example of the Ant algorithm is depicted in Figure 1.

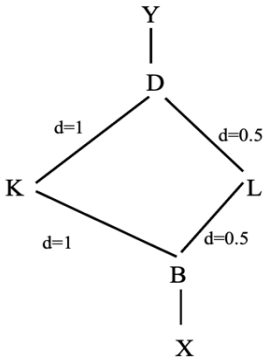

Fig. 1 Path updating using the ACO Algorithm

Consider the distance (d) from D to K and B to K as 1 each, and the distance from D to L and B to L is 0.5 each. Consider 50 ants reach B from X at time t0, and another 50 ants reach D from Y at time t0. The speed of each ant is 1 per time unit, and the intensity of the pheromone trail is 1 when the ant walks over the path. At t0, there is no trail deposited. 50 ants at D and B take random paths. On average, 25 ants go towards K, and another 25 go towards L. At t_1, consider 30 ants reached B from X, and it finds the intensity 30 on the edge BL, with 25 ants going from B, and 25 ants came from D via L. With the high intensity found on BL, the ant starts taking the path BL with a doubled ratio of 20:10 compared to the path BK. This process is repeated until all ants follow the shortest path.

The ACO algorithm follows some steps for execution, and it is then used along with Q values for the calculation of the shortest path in the proposed ACOMS-Q algorithm. The frequently used notations are declared in Table 3.

- Initialization: Pheromone levels are introduced on all edges, and the parameters α, β, and rho ρ are set.
- Solution Construction: Using the probability function derived from pheromone trails and heuristic data, each ant constructs a tour.
- Pheromone Update: All the edges are updated with new pheromone levels after all the ants complete the tour.
- Termination: Steps 2 to 4 are repeated until the algorithm reaches the predetermined number of iterations.

Table 3. Summary of frequently used notations

Notation	Meaning
$\Delta\tau_{i,j}^{m}$	Pheromone level among nodes i and j
m	Ant count
$\Delta\tau_{i,j}^{k}$	Pheromone deposition by the k^{th} ant
L^k	k^{th} ant's travel length
$P_{i,j}^{t}$	Ant transition probability at time t
η_{ij}	heuristic information
d_{ij}	Distance between nodes i and j
α	Pheromone tuning parameter
β	Heuristic tuning parameter
C_i	Cost function
Q(s,a)	Q function with state' s' and action 'a'
γ	Discount factor
r	Reward
maxQ(s′,a′)	Best future Q value
λ_m	Duration of the mobile sink
P_j	Priority value of node j
MS	Mobile Sink
ρ	Evaporation rate

The pheromone levels are updated when all ants have finished the tour. The updated shortest path will have greater pheromone deposition. Total Pheromone deposition on the edge (i,j) for a single iteration is depicted in Equations (1) and (2).

$$\Delta\tau_{i,j}^{m} = \sum_{k=1}^{m} \Delta\tau_{i,j}^{k} \quad without\ vaporization \tag{1}$$

$$\Delta\tau_{i,j}^{m} = (1-\rho)\tau_{i,j}^{m} + \sum_{k=1}^{m} \Delta\tau_{i,j}^{k} \quad with\ vaporization \tag{2}$$

The $\Delta\tau_{i,j}^{m}$ indicates the pheromone level among nodes i and j. m denotes the ant count. $\Delta\tau_{i,j}^{k}$ is the pheromone deposited by the k^{th} ant. The constant ρ controls the evaporation rate. Each ant deposits pheromone on the edges, and the amount of deposition on a particular edge by each ant can be found by using Equation (3) and is considered the pheromone deposition rule. L^k denotes the total length traveled or path length traversed by the kth ant.

$$\Delta\tau_{i,j}^{k} = \begin{cases} \frac{1}{L^k}, & if\ ant\ k\ used\ the\ edge\ i,j \\ 0, & Otherwise \end{cases} \tag{3}$$

The cumulative total pheromone $\tau_{i,j}^{m}$ updated on the edge over multiple iterations results in a global pheromone update as in Equation (4).

$$\tau_{i,j}^{m} = (1-\rho)\tau_{i,j}^{m} + \Delta\tau_{i,j}^{m} \tag{4}$$

Using Equation (4), for time t, the probability of an ant transitioning from node i to j is expressed. The selection of the next node is dependent on two factors. (a) pheromone levels (b) heuristic information. In general form, probability can be represented as in Equation (5), and the heuristic information can be found using Equation (6).

$$P_{i,j}^{t} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{k \in allowed} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}} \quad (5)$$

$$\eta_{ij} = \frac{1}{d_{ij}} \quad (6)$$

η_{ij} - the quality of the link among nodes i and j on the graph (heuristic information)
τ_{ij} - the pheromone level on the link between node i and j
d_{ij} - the distance(separation) between node i and node j
α and β: adjusting factors for the influence of pheromone and the heuristic control, respectively

In this research, an ACO-tuned TSP is used to locate the best-fit approach for a computationally difficult task. Furthermore, ACO-TSP is enhanced with clustering, Ant Colony Optimization Mobile Sink with Clustering (ACOMS-C), based on three metrics - residual energy, node degree, and the distance to the mobile sink. In ACOMS-C, nodes are assumed to be always active, and no scheduling algorithm is incorporated. ACOMS-C is further incorporated with a Scheduling Algorithm (ACOMS-SC) where the CH wakes up as per the arrival time of the MS to obtain data.

To curb the limitations of ACO-TSP, ACOMS-C, and ACOMS-SC, an algorithm called ACOMS-Q was proposed that uses the Q-learning algorithm for scheduling and improves the ACO-TSP algorithm with integration of Q values for finding the best path. The flowchart explaining the methodology is depicted in Figure 2. In the proposed algorithm, the ACOMS-Q algorithm, there are three main phases.

- Clustering
- Scheduling
- Routing

3.1. Clustering in ACOMS-Q

Apart from the traditional clustering methods, the proposed protocol, ACOMS-Q, follows a tri-metric model to find the cluster heads. The process to elect optimal CHs is challenging; a tri-metric model is implemented that takes residual energy, path length from the sink, and weight of the node as the metrics to obtain efficiency. Considering the major limitation of battery life in sensor networks, residual energy is a crucial metric in the selection of cluster heads. The leftover energy can be estimated by taking the ratio of the current available Energy (E_r) from the Maximum Energy allotted (E_{max}) for the sensory node. Nodes with maximal remaining energy resources are chosen as the head for the cluster. Distance (D_S) is considered the second metric, as energy consumption is in proportion to distance. As shorter distances consume less energy, nodes closest to the sink are given more priority to act as the cluster head. Distance can be calculated from the Euclidean distance formula as in Equation (7). The third metric is the weight of the node, which implies the number of Neighbor Nodes (N_n), and the node with maximum proximity is considered to become the CH. The Fitness Function (C_i) to elect the cluster head is calculated by assigning Weights (w_1, w_2, w_3) as shown in Equation (8). N_{nmax} is the maximum allowable number of neighbor nodes, and D_{smax} is the maximum allowable distance. (x_l,y_l) represents the coordinates of the node, and (x,y) is the coordinate of the mobile sink.

$$D_S = \sqrt{(x_l - x)^2 + (y_l - y)^2} \quad (7)$$

$$C_i = w_1 \left(\frac{E_r}{E_{max}}\right) - w_2 \left(\frac{D_s}{D_{smax}}\right) + w_3 \left(\frac{N_n}{N_{nmax}}\right) \quad (8)$$

3.2. Scheduling and Routing in ACOMS-Q: Integration of Q-Learning driven Reinforcement Learning with ACO

In the proposed protocol, ACOMS-Q, reinforcement learning can be used in scheduling the nodes and also to calculate the optimal path as per the learning. It dynamically decides when the nodes should be active and in sleep mode based on the real-time state, providing high data delivery by minimizing the energy usage. ACOMS-Q incorporates efficient clustering, scheduling based on Q learning, and mobile sink routing based on an improved ACO algorithm. The major components related to reinforcement learning involve:

Agent: The entity that learns from the environment to make decisions by interaction.

State(s): The inputs given to the agent from the environment

Action(a): The decision taken by the agent for the given state information

Environment: The exterior system that helps agents with the states and rewards corresponding to the action taken.

Reward(r): Feedback obtained by the agent from the environment to check the desirability of the decision.

Policy(π): The rule that makes the agent decide which action to take for a particular state.

For illustration, consider two states [residual energy and buffer size] and two actions [0: sleep and 1: active]. Positive

reward is applied for data transmitted and energy saved. Negative reward means packet loss and energy wasted. There are different choices for the algorithm. Q-learning is employed for a small-sized network that comprises a few states and action spaces, and Deep Q-Learning can be a choice for a large-scale network.

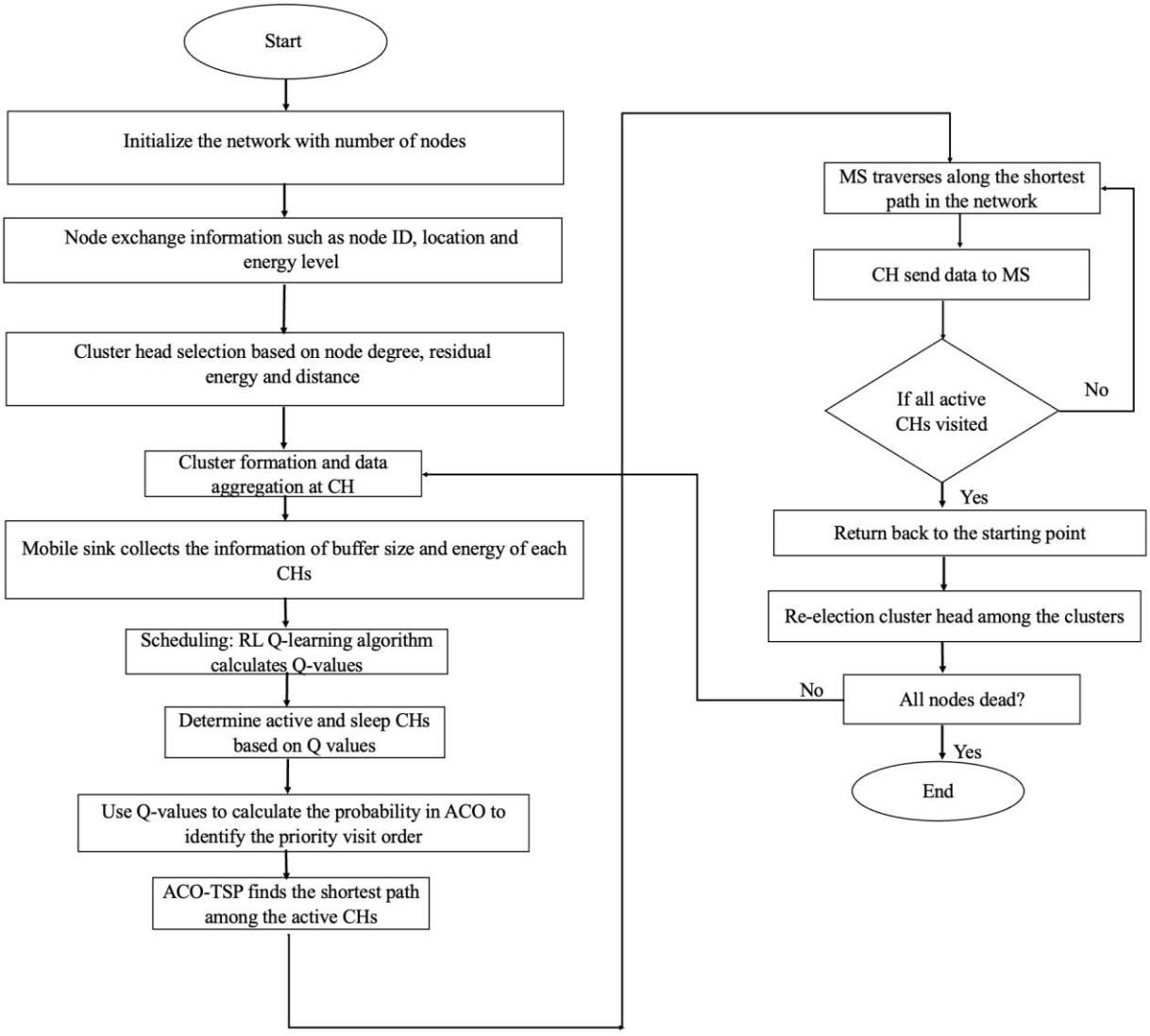

Fig. 2 Flowchart of proposed ACOMS-Q algorithm

Q-learning [28] is an RL algorithm that evaluates the quality of a particular action taken for a given state using Q-values obtained. Q-value provides a prolonged benefit of preferring action ' a ' from a specific state' s'.

Each node observes its state, and the agent selects an action based on an epsilon-greedy policy. For ACO path selection, the mobile sink uses only the active nodes to participate. After the episode, reward and Q values will be calculated. Q-values are updated using Equation (9).

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma.max_{a'}(Q(s',a'))\,Q(s,a)] \quad (9)$$

Q(s,a) - present value of Q
α - hyperparameter indicates the rate of learning
γ - discount parameter
maxQ(s′, a′) r - calculated immediate reward
- best future Q value for next state

The updated knowledge about state-action pairs is given as Q(s, a). The notation ' a ' represents the action taken in state' s' to get reward 'r', resulting in new state s′. Maximal Q-value is looked for the best action in the next state, s′, $max_{a'}Q(s', a')$.

Update the present Q value, Q(s, a), by moving it a bit towards the target $r+\gamma max_{a'}Q(s', a')$. RL Q-learning algorithm schedules the node's state, and modified ACO plans the path

with priority considerations. This can be divided into two different phases.

- Scheduling phase (using Q learning) [29, 30]
- Mobile sink path determination (using modified ACO) with the integration of RL and ACO

3.2.1. Scheduling phase (using Q learning)

Two states are taken – buffer size and residual energy, with two actions – active and sleep. Positive reward is provided for data transmission efficiency. All possible pairs of state and action are listed, and every node keeps a Q table for each possible pair. From the Q-table, the current state of the node is observed, and an action will be taken according to the epsilon-greedy strategy. Once the action is executed, the rewards will be given, and the Q-table will be updated. The Q table is initialized with a zero value in the initial step.

3.2.2. Mobile sink path determination (using Modified ACO)

Scheduling determines the active nodes, and ACO-TSP operates on the active nodes. The nodes with high Q value are given higher priority along with the pheromone and heuristic information. Nodes with high data and less energy will be visited early, keeping the sink path short and efficient. Action decision using epsilon-greedy policy is represented in Equation (10).

$$a = \begin{cases} random\ action & (exploration\ with\ probability\ (\varepsilon)) \\ arg\ max_a\ Q(s,a) & (exploitation\ with\ probability\ (1-\varepsilon)) \end{cases} \quad (10)$$

In the suggested method, ACOMS-Q, Q-learning is applied over all the sensor nodes to optimize the schedule of active and sleep states. The epsilon greedy algorithm is a strategy to balance between exploration and exploitation. Exploration means the learning stage where new actions are tried to discover better outcomes. Exploitation means sticking with the best Q values for taking the best action. ε is the exploration rate, initially set to a high value and decayed gradually. In early training episodes, a higher value of ε (e.g., 0.8) ensures sufficient exploration of the action space.

As learning progresses and the Q-values stabilize, ε is reduced (e.g., to 0.1) to prioritize exploitation of the best-known strategies. maxQ(s,a) returns the maximum Q-value. argmaxQ(s,a) returns the action 'a' at which that maximum Q-value occurs. This policy ensures that: (a) nodes with high priority (e.g., low energy but high buffer) are identified effectively, (b) the mobile sink visits such nodes earlier using the modified ACO algorithm, and (c) nodes not selected for immediate visit transition to sleep, conserving energy. Trail intensity is updated after every tour, and the duration of mobile agent m^{th} tour is represented in Equation (11).

$$\lambda_m = \frac{X_m}{V} \ \ where\ X_m \leq T_{maxl} \quad (11)$$

λ_m - Duration of the mobile sink's m^{th} tour (time taken)
X_m - Distance traveled during the m^{th} tour
V - Speed of the mobile sink
T_{maxl} - Maximum allowable distance for the m^{th} tour

From Equation (5), the basic ACO formula can be represented generally as shown in Equation (12).

$$P_{ij} = \frac{\tau_{ij}^{\alpha} . \eta_{ij}^{\beta}}{\sum_{k \in allowed} \tau_{ik}^{\alpha} . \eta_{ik}^{\beta}} \quad (12)$$

Initially, the priority calculation is done based on the two factors: the node's buffer data size and energy level. The priority value is then used by the ACO algorithm along with the heuristic and pheromone value to calculate the probability of finding the next node to visit, as in Equation (12). The heuristic information is calculated with the incorporation of priority values as represented in Equations (13) and (14).

$$\eta_{ij} = \frac{P_j}{d_{ij}} \quad (13)$$

$$\eta_{ik} = \frac{P_k}{d_{ik}} \quad (14)$$

P_j - Priority value of node j based on initial sensor data
P_k - Priority value of all the remaining nodes

After the initial round, all the nodes are updated with the Q-table, and the Q-values generated through reinforcement learning reflect a more accurate measure of how valuable each node is for future visits. So, instead of using the initial priority estimates, the corresponding Q-value can be used, and the new transition probability is shown in Equation (15).

$$P_{ij} = \frac{\tau_{ij}^{\alpha} \cdot \left[\frac{Q_j}{d_{ij}}\right]^{\beta}}{\sum_{k \in allowed} \tau_{ik}^{\alpha} \cdot \left[\frac{Q_k}{d_{ik}}\right]^{\beta}} \quad (15)$$

Q_j now stands for the learned priority of node j based on its past performance. Incorporation of Q-value adds a layer of novelty and intelligent decision-making to the traditional ACO algorithm. Reward is a numerical feedback signal that informs the learner (agent) about the action taken according to the outcome[31]. In the proposed case, the agent tries to schedule a node (either to sleep or active mode) to optimize for energy efficiency and data delivery (collect as much data as possible).

4. Mathematical Analysis and Explanation

For the mathematical analysis, consider the subsequent parameters: Learning rate (α) = 0.5, Initial Q(s, a) = 0 for all, Discount factor (γ) = 0.9, Position of sink(S): (0,0). Consider one one-node scenario that can be in one of 4 states, based on Energy: Low or High, and Buffer: Empty or Full.

Actions can be in two states: Action 1: Sleep, Action 2: Active. Table 4 defines the possible operational states of a node in the network, determined by its residual energy and buffer occupancy level. Each state represents a unique condition that influences the scheduling decision in the reinforcement learning process. For instance, a Low_Empty state (State 1) indicates minimal energy with no buffered data, while a High_Full state (State 2) reflects sufficient energy with maximum buffer occupancy, thus demanding immediate data transmission. These states are critical for prioritizing nodes during the ACO-TSP-based path optimization. Table 5 shows the rewards allotted for different states, and Table 6 represents the initial Q Table.

Table 4. Possible operational states of the node based on two parameters

State No.	Energy	Buffer	Meaning
1	Low	Empty	Low_Empty
2	High	Full	High_Full
3	High	Empty	High_Empty
4	Low	Full	Low_Full

Table 5. Rewards allotted for different states with meaning

State	Action	Reward	Remarks
Low_Empty	Active	-2	No data to send, wasted energy
High_Full	Active	+5	Safe but not urgent
High_Empty	Sleep	+2	Conserve energy and refills buffer
Low_Full	Active	+10	Critical node served, data saved

Table 6. Initial Q-Table for each state

State	Actions	
	Sleep(1)	Action(2)
Low_Empty	0	0
Low_Full	0	0
High_Empty	0	0
High_Full	0	0

Episode 1: Start at state 4: (High_Full)
Current State (S): High_Full
Random Action(A): Active (2)
Reward(r): +5
Next State(s'): 3, High_Empty
Q-Update: $Q(4,2) = Q(4,2) + \alpha\,[r+\gamma.\max(Q(3,:)) - Q(4,2)]$
$= 0+0.5\,[5 + 0.9 * 0 - 0]$
$= 2.5$

The suggested next state is High_Empty because the node had high energy before. The sink visited the node, and the buffer became empty. Energy dropped slightly but is still above threshold; hence, it is considered high. If the energy level drops below the High_Empty threshold, then the next state becomes Low_Empty.

Episode 2: Start at state 3: (High_Empty)
Current State (S): High_Empty
Random Action(A): Sleep (1)
Reward(r): +2
Next State(s'): 4, High_Full
Q-Update: $Q(3,1) = Q(3,1) + \alpha\,[r+\gamma.\max(Q(4,:)) - Q(3,1)]$
$= 0+0.5\,[2 + 0.9 * 2.5 - 0]$
$= 2.125$

Episode 3: Start at state 2: (Low_Full)
Current State (S): Low_Full
Random Action(A): Active (2)
Reward(r): +10
Next State(s'): 1, Low_Empty
Q-Update: $Q(2,2) = Q(2,2) + \alpha\,[r+\gamma.\max(Q(1,:)) - Q(2,2)]$
$= 0 + 0.5\,[10 + 0.9 * 0 - 0]$
$= 5.0$

Episode 4: Start at state 1: (Low_Empty)
Current State (S): Low_Empty
Random Action(A): Active (2)
Reward(r): -2
Next State(s'): 1, Low_Empty
Q-Update: $Q(1,2) = Q(1,2) + \alpha\,[r+\gamma.\max(Q(1,:)) - Q(1,2)]$
$= 0 + 0.5\,[-2 + 0.9 * 0 - 0]$
$= -1.0$

Table 7 shows the Q-Table after 4 episodes, and it is observed that Q(2,2) = 5.0 has the highest Q-value, meaning that the node is in low energy, high buffer, and the best action is active, and the node should be visited first in path planning. This Q-value is used in the ACO-TSP algorithm for best path finding. Now, for mathematical calculation, assume that there are 5 nodes with the following assumed active Q values as in Table 8.

Table 7. Q-Table after 4 episodes for a single node

State	Q Sleep (Action 1)	Q Active (Action 2)
1	0	-1.0
2	0	5.0
3	2.125	0
4	0	2.5

Table 8. Assumed states and active Q values for 5 nodes

Node	Q-value
A	5.0
B	2.5
C	-1.0
D	0
E	2.125

The standard ACO formula in Equation (15) is modified to include the Q value to find the modified probabilities as shown in Equations (16) and (17).

$$\eta_{ij} = \frac{max(Q_j,0)+ c}{d_{ij}} \quad (16)$$

$$P_{ij} = \frac{\tau_{ij}^{\alpha} \cdot [\frac{max(Q_j,0)+c}{d_{ij}}]^{\beta}}{\sum_{k \in allowed} \tau_{ik}^{\alpha} \cdot [\frac{max(Q_k,0)+c}{d_{ik}}]^{\beta}} \quad (17)$$

c - small constant to avoid division by 0 (c=0.1)

Table 9 shows the calculation of heuristic information incorporating the Q values obtained, and the negative Q value is clipped to 0.

Table 9. Calculation of η_j

Node	(x,y)	Distance (d_{Sj})	Q-value	η_j = (max(Q,0)+c)/d
A	(3,4)	5.00	5	$(5 + 0.1)/5 = 1.02$
B	(4,0)	4.00	2.5	$(2.5 + 0.1)/4 = 0.65$
C	(3,5)	5.83	-1.0	$(0 + 0.1)/5.83 = 0.017$
D	(5,5)	7.07	0	$(0+0.1)/7.07 = 0.014$
E	(0,4)	4.00	2.125	(2.125+0.1)/4 = 0.556

Assume all pheromones as $T_{Sj} = 1$, where T_{Sj} stands for the path selection probability from the source node(S) to the candidate node (j). Equation (18) shows the simplified equation when the current node i is implied and ranks all j in the candidate set, assuming α=1 and β=2. The heuristic information in Table 10 is used to calculate the modified probability in Table 11. The cumulative sum is 1.0404 + 0.4225 + 0.000289 + 0.000196 + 0.3091 = 1.772.

$$P_{sj} = \frac{\tau_{sj}^{\alpha} \cdot \eta_j^{\beta}}{\sum_{k=allowed} \tau_{Sk}^{\alpha} \cdot \eta_k^{\beta}} \quad (18)$$

η_j - one specific next node or the one that is currently being evaluated for selection

η_k - each of the remaining candidate nodes in the neighborhood

Table 10. Calculation of η_j^2

Node	η_j^2
A	$(1.02)^2 = 1.0404$
B	$(0.65)^2 = 0.4225$
C	$(0.017)^2 = 0.000289$
D	$(0.014)^2 = 0.000196$
E	$(0.556)^2 = 0.3091$

Table 11. Calculation of probability (P_{sj})

Node	Probability (P_{sj})
A	1.0404/1.772 = 0.587
B	0.4225/1.772 = 0.238
C	0.000289/1.772 = 0.00016
D	0.000196/1.772 = 0.00011
E	0.3091/1.772 = 0.174

Table 12. Final calculation of probability with clipped Q

Node	Q (Clipped)	Q′	d_{sj}	η_j	n_j^2	P_{sj}
A	5.0	5.1	5.00	1.02	1.0404	0.587
B	2.5	2.6	4.00	0.65	0.4225	0.238
C	0	0.1	5.83	0.017	0.000289	0.00016
D	0	0.1	7.07	0.014	0.000196	0.00011
E	2.125	2.225	4.00	0.556	0.3091	0.174

In Table 12, to find the updated probability. Node A has the highest probability (Low_Full), Q = 5.0, and has the highest selection probability of 0.587. Modified ACO with Q-learning prioritizes the critical nodes first. The final path based on these probabilities could be: A → B → E → C → D.

Mobile sink path planning is determined using ACO-TSP with Q-values. With the speed of MS known, travel time can be estimated by analyzing the distance between the nodes. The addition of guard time (time before arrival to wake up the node) helps to deal with the delay and to avoid the loss of packets. Mobile sink stops at each of the cluster heads to collect data, and the collection time depends on buffer occupancy and transmission rate. So, the time at which the node should wake up is the difference between the actual arrival time and the guard time. For the calculation of travel time, arrival time, wakeup time, and data collection time, the following assumptions are made. Mobile sink speed v=1m/s, Guard time = 2s, Transmission Rate (R) =10,000 bits/s, Conversion: Data=Buffer (KB)×8×1024 and D_j – data in bits. Position of the nodes A, B, C, D, E can be (3,4), (4,0), (3,5), (5,5), and (0,4) respectively. The final path based on these probabilities could be: A → B → E → C → D as obtained from Table 12. The separation between two nodes i and j is computed using the Euclidean distance formula, as given in Equation (19). This metric determines the spatial separation between nodes, which directly impacts the travel time of MS. For instance, the distance from point S to point A (d_{SA}) is evaluated as:

$$d_s = \sqrt{(x_i - x)^2 + (y_i - y)^2} \quad (19)$$

$$d_{SA} = \sqrt{(3-0)^2 + (4-0)^2}$$

The travel time is then obtained by dividing the inter-node distance d_{ij} by the velocity v of the mobile sink, as shown in Equation (20). Once the mobile sink reaches a node, it requires a certain duration to collect the buffered data from that node. This data collection time is obtained by taking the ratio of total data size D_j stored at the node to the transmission rate R, as expressed in Equation (21). For mathematical analysis, assume time in seconds and distance in meters.

$$t_{travel,j} = \frac{d_{ij}}{v} \quad (20)$$

$$T_{data,j} = \frac{D_j}{R} \quad (21)$$

The cumulative arrival time at nodes can be calculated by summing the travel times to all previously visited nodes and the data collection times from those nodes, as defined in Equation (22). This parameter is essential for synchronizing the sleep–wake cycles of nodes. The wakeup interval for node i is then derived by subtracting the guard time from its arrival time, as shown in Equation (23). This ensures that the node is active and ready to transmit data just before the mobile sink arrives, thereby minimizing idle energy consumption.

$$t_{arrival,j} = \sum_{k=1}^{j} t_{travel,k} + \sum_{k=1}^{j-1} t_{data,k} \quad (22)$$

$$T_{wake,j} = t_{arrival,j} - t_{guard} \quad (23)$$

Arrival time at a particular node is calculated by taking the sum of the previous node arrival time, the previous node data time, and the current node travel time. Mobile sink follows the best path, and it is assumed that nodes wake up 2 seconds before the sink arrives, and this is called guard time. Data collection time is calculated based on the buffer size. The total time plan is used to broadcast the wakeup schedule to all the nodes. Tables 13 and 14 show the travel time and data collection time calculation. Table 15 shows the arrival time and wakeup time calculations.

Table 13. Calculation of travel time to the next node

Segment	From → To	Coordinates	Distance (m)
S → A	(0,0) → (3,4)	5.00	5.00
A → B	(3,4) → (4,0)	4.12	4.12
B → E	(4,0) → (0,4)	5.66	5.66
E → C	(0,4) → (3,5)	3.16	3.16
C → D	(3,5) → (5,5)	2.00	2.00

Table 14. Calculation of the data time taken at each node

Node	Buffer (KB)	Bits	Data time(s)
A	5	5*8*1024 = 40,960	40960/10000 = 4.10
B	3	3*8*1024 = 24,576	24576/10000 = 2.46
E	4	4*8*1024 = 32,768	32768/10000 = 3.28
C	2	2*8*1024 = 16,384	16384/10000 = 1.64
D	1	1*8*1024 = 8,192	8196/10000 = 0.82

Table 15. Arrival and wakeup time calculation for each node

Node	Travel Time (s)	Arrival Time (s)	Data Time (s)	Guard Time (s)	Wakeup Time (s)
A	5.00	0 + 5.00 = 5.00	4.10	2	3.00
B	4.12	5.00 + 4.10 + 4.12 = 13.22	2.46	2	11.22
E	5.66	13.22 + 2.46 + 5.66 = 21.34	3.28	2	19.34
C	3.16	21.34 + 3.28 + 3.16 = 27.78	1.64	2	25.78
D	2.00	27.78 + 1.64 + 2.00 = 31.42	0.82	2	29.42

5. Performance Analysis and Results

The performance analysis is divided into two main parts:

(a) Mathematical analysis using 5-node scenarios

(b) Validation through comparison of simulation results with 100 nodes based on metrices with existing protocols using MATLAB software

5.1. Mathematical Model Demonstration (5 nodes)

The mathematical analysis using five nodes with the step-by-step procedure for the ACOMS-Q algorithm is explained in this section. The results obtained from the mathematical model are plotted to verify the accuracy of the mathematical analysis and to study the process involved in making decisions. The suggested reinforcement learning–enhanced ACO framework algorithm was evaluated based on its ability to optimize node wakeup scheduling and mobility-enhanced routing in a sensory network. The results highlight the algorithm's effectiveness in minimizing idle energy utilization while ensuring timely data collection.

Figure 3 represents the timing summary of each node related to arrival time, wakeup time, and end time. Each node is allowed to wake up just before the arrival of the mobile agent to minimize the energy utilization, while still avoiding communication latency. The end time is calculated according to the summation of the arrival time and the data collection

time. End time varies depending on the amount of data each node holds in the buffer. Based on the path selected by the modified ACO algorithm, the mobile sink first visits node A that has the earliest wakeup and arrival time. Nodes C and D exhibit higher end times due to both longer travel distances and larger buffer sizes, resulting in longer data transmission times. Node E, though visited later, has a smaller buffer, leading to a lower end time compared to C and D. The proposed algorithm synchronizes the node's wakeup time with the mobile sink arrival time.

Figure 4 represents the cumulative time taken by MS to traverse all the active nodes based on the trajectory found. The arrival time and the period taken for data collection result in the cumulative time. As the mobile sink visits node A first and the data is gathered, the time taken by the mobile sink at node A is 9.10s. Next node to be visited is B, and the cumulative time is 15.68s. The cumulative time increases as the mobile sink covers more nodes based on the best path found. The cumulative trajectory period acquired by the mobile sink to cover A → B → E → C → D is 32.24s.

Figure 5 represents the sleep time before the scheduled wakeup time. The algorithm effectiveness is observed, showing that an efficient sleep-wake-up scheduling mechanism is adopted in such a way that nodes wake up only during the scheduled wakeup time and remain in low-power or sleep state for the rest of the time. Node D has the maximum sleep time as it is visited last, while node A has the least sleep time as it is the first node visited after the deployment. This adaptive sleep strategy reduces energy consumption by keeping the node in a sleep state until the scheduled time.

Figure 6 represents the data collection time per node. Data collection time varies with the amount of data the node holds in the buffer and the rate at which data transfer takes place. The buffer size of nodes A, B, E, C, D is 5,3,4,2,1(in KB) respectively. The graph indicates that node A has the largest amount of data, followed by node E, and so forth.

Figure 7 shows scheduling across all participating nodes. The Gantt chart shows two segments – dark blue and light blue. The dark blue segment represents the time period between wakeup time and arrival time, and the light blue segment represents the data collection time. Initially, node A wakes up first and transmits the data, followed by nodes B, E, C, and D. The distance between nodes and their data transmission time, which is proportional to buffer size, are both reflected in the distance between bars along the time axis. This schedule demonstrates effective synchronization, reducing unnecessary awake time and ensuring that nodes consume energy only when required. Figure 8 represents the energy consumption by each node during the data transfer with the mobile sink. Energy consumption is calculated in regard to buffer size and the travel distance to the mobile agent when data is transmitted [32]. Figure 9 shows the best path selected by the sink for data collection.

Using the modified ACO algorithm with the incorporation of Q-values, the best path is selected, and nodes are visited based on the scheduled time [17]. Figure 10 shows how buffer size affects the delay at each node. The time taken at each node is obtained by subtracting the wakeup time from the end time. For node A, the buffer size is 5KB, and the total time taken at node A for collecting the data is 6.10s.

Total delay at node A = (End-time) – (Wakeup time)
= (Arrival time + Data collection time) – (Wakeup time)
= (5 + 4.10) - 3 = 6.10s at Node A

Figure 11 shows the relationship between the buffer occupancy of a node and its scheduled wakeup time. The nodes with high buffer size are prioritized first to avoid data loss and transmission delays.

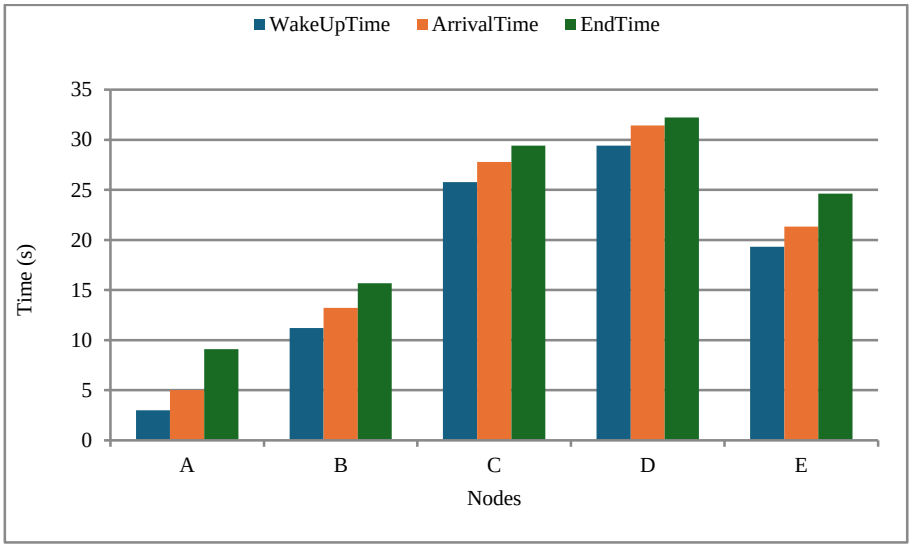

Fig. 3 Node timing summary

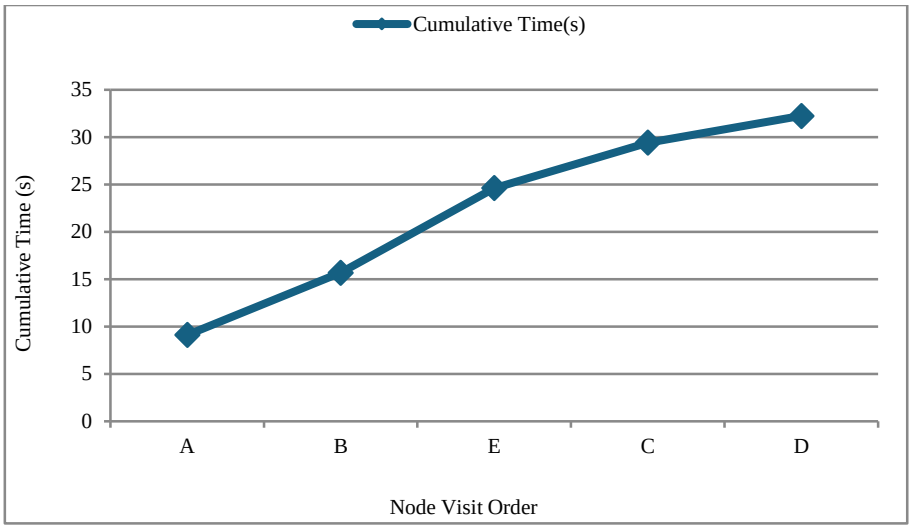

Fig. 4 Cumulative time taken by the mobile sink for the tour

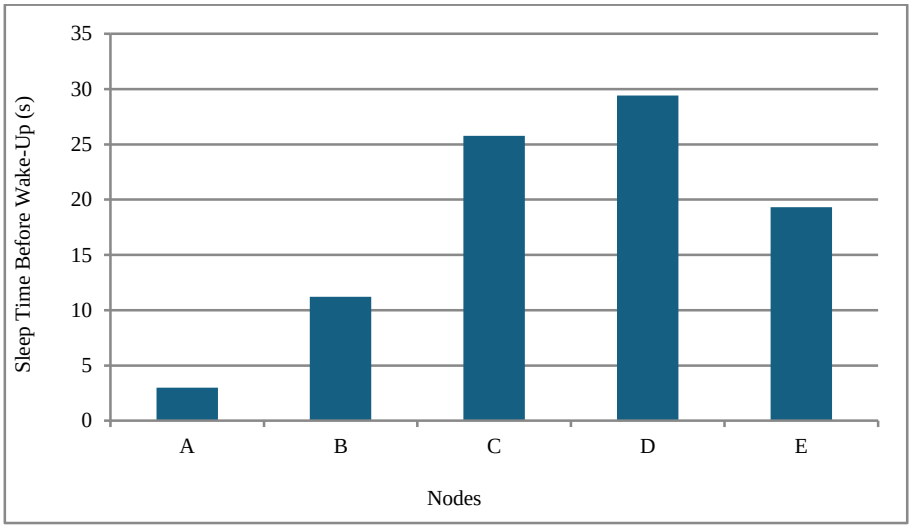

Fig. 5 Sleep time before scheduled wakeup for each node.

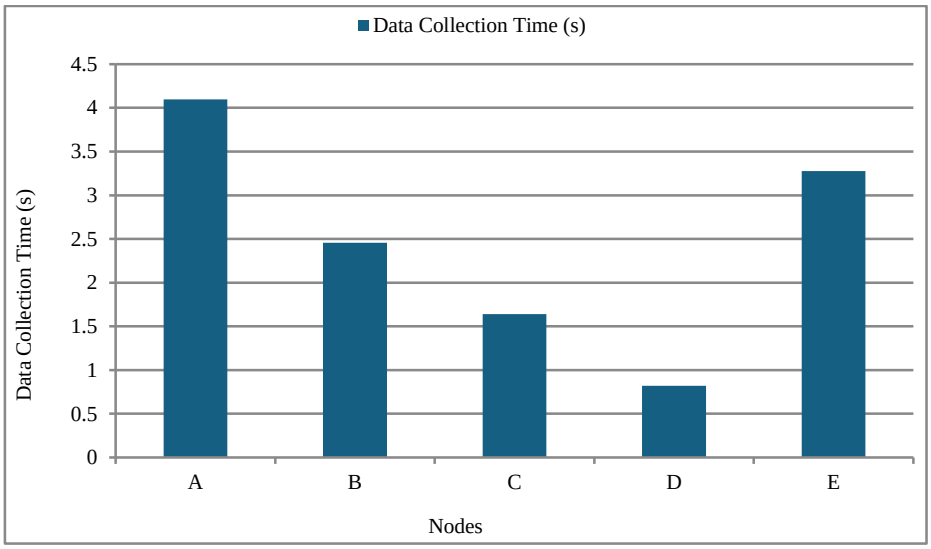

Fig. 6 Data collection time per node based on buffer size

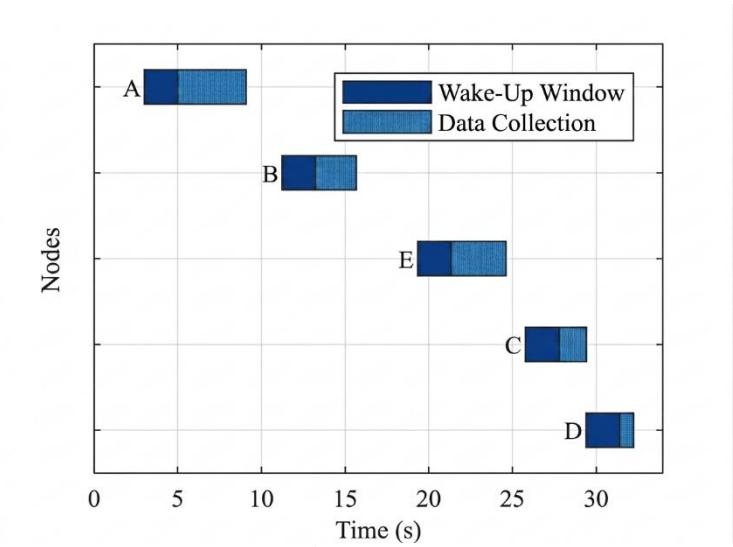

Fig. 7 Gantt chart for wakeup and data collection

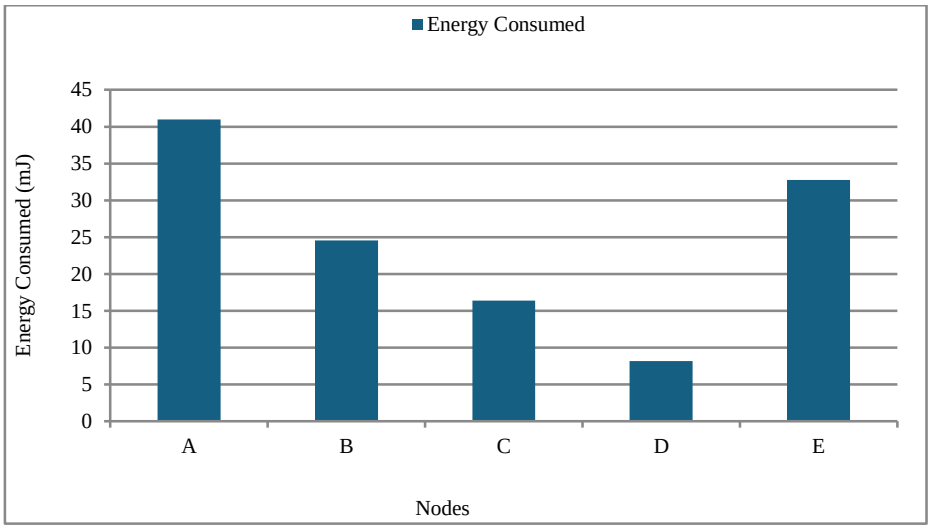

Fig. 8 Energy consumption per node

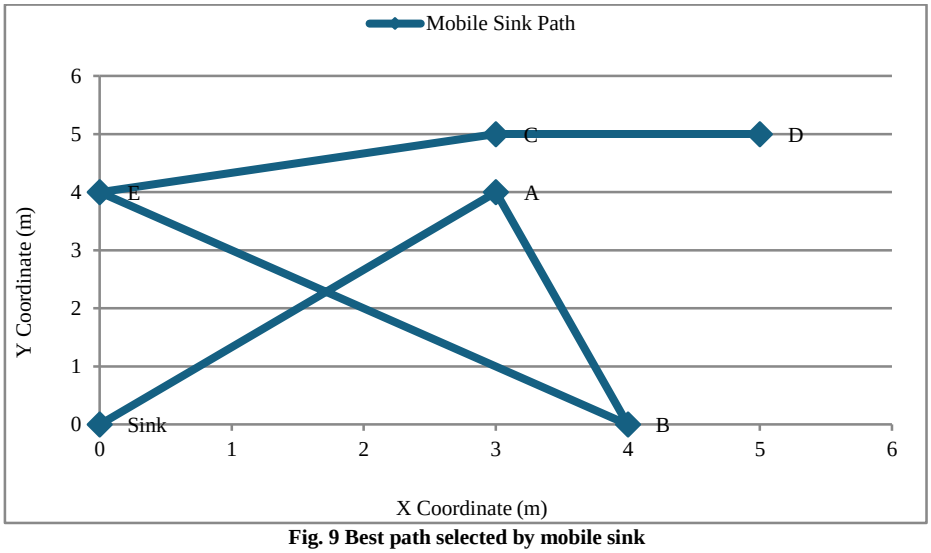

Fig. 9 Best path selected by mobile sink

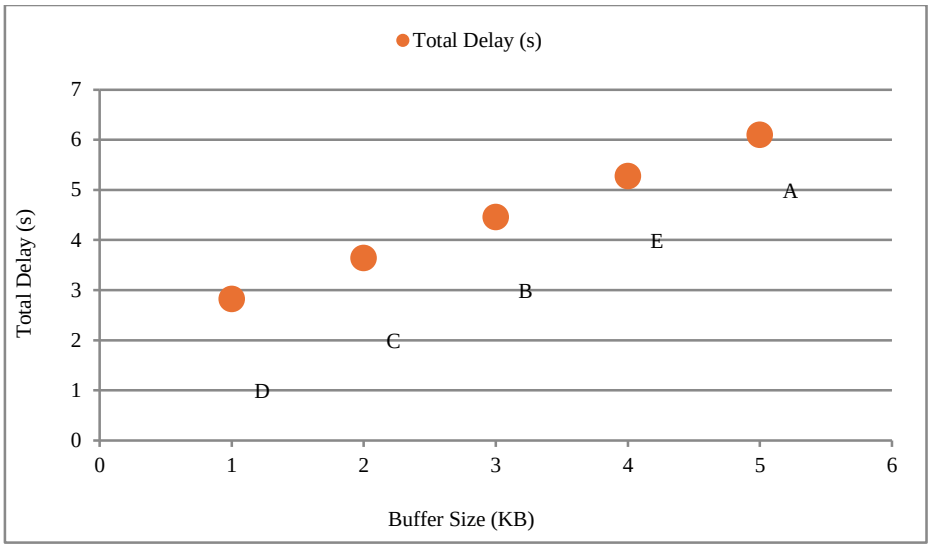

Fig. 10 Delay Vs Buffer Occupancy

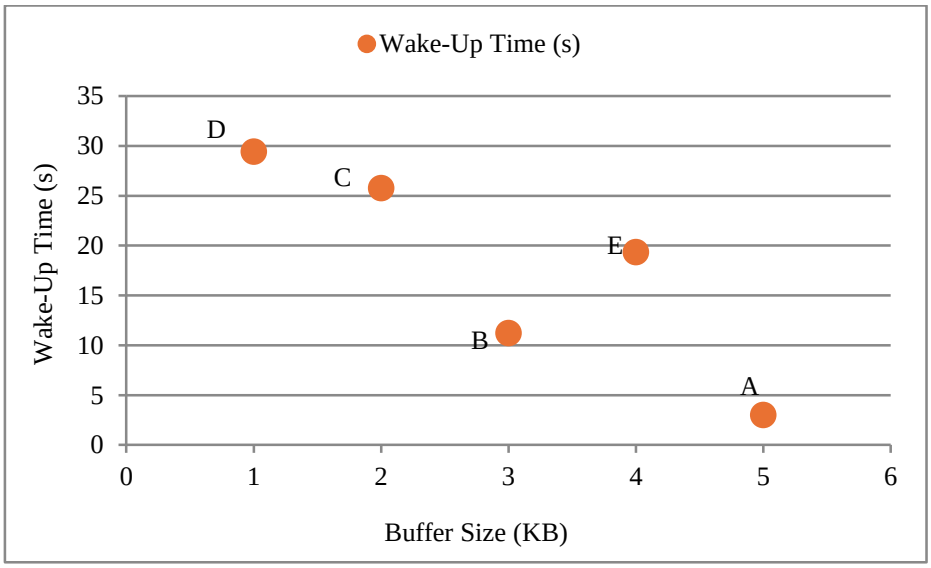

Fig. 11 Wakeup time Vs Buffer occupancy

5.2. Simulation Results with 100 nodes: Detailed Result Analysis and Comparison based on Metrics

Further, the above methodology was implemented on 100 nodes in a 100x100 network in MATLAB software, and the following performance metrices are analyzed: delay, tour length, energy utilization, network longevity, and packet delivery ratio.

The suggested ACOMS-Q protocol is compared with the classical LEACH protocol, traditional ACO algorithm with TSP and mobile sink (ACO-TSP), ACO with Clustering and MS (ACOMS-C), and ACO with clustering and scheduling based on the arrival time of the mobile sink (ACOMS-CS).

Table 16 represents the parameters chosen for the implementation of the proposed algorithm, ACOMS-Q.

Table 16. Network parameters

Parameters	Values
Number of nodes	100
Network Area	100x100
Number of clusters	13
E_elec, Electron's energy	50 nJ/bit
E_amp, Amplifier energy	100 pJ/bit/m^2
E_agg, Data aggregation energy	5 nJ/bit
E_move, MS movement energy	0.001 J/m
E_sleep, Energy consumption in sleep mode	1 nJ/bit
Guard Time	0.5
Aggregation Efficiency	0.1
Confidence Level	0.95

The LEACH protocol divides the network into different segments, and the CH sends the data to the static sink directly. In ACO with the integration of TSP, the shortest path is found by evaluating the pheromone level and heuristic information. The mobile sink follows the shortest path, individually visiting all the nodes and returning to the same position. Improved ACO algorithms are also implemented. ACOMS-C divides the entire field into groups, taking three parameters: leftover energy, weight of the node, and distance. MS transit only to CHs to get the sensory information and return to the actual position following the shortest path. In ACO and ACOMS-C nodes remain active throughout the transmission. With the incorporation of scheduling in ACOMS-CS, CHs are activated upon the time at which the mobile sink arrives. In ACOMS-Q, Q-learning is used to find the active nodes and schedule the data collection of only the active nodes through the shortest path found using the improved ACO algorithm.

Delay: Delay defines the cumulative time taken by the mobile agent to gather data and finish the tour [33]. In LEACH, multihop propagation is followed, in which the group members send information to the cluster head in their timeslot, thereby cluster heads transmit aggregated data to the static sink in one hop. This includes long-distance transmission, TDMA frame overhead, and data takes a long time to reach the sink. In ACO-TSP, the mobile sink transits across the sensory field and picks the data individually from all nodes, reducing the overhead and delay. ACOMS-C clusters the sensory field, and MS stops only at the collection points to obtain the data, which in turn reduces the delay. ACOMS-SC, delay is slightly greater than ACOMS-C, as the sleep-wake up schedule results in waiting for the collection of data. The proposed ACOMS-Q works with a mobile sink that visits only the active cluster heads with the efficient Q-learning incorporation on the ACO-TSP algorithm to find the optimized path with efficient scheduling, outperforming all the other methods. Figure 12 shows the efficiency of ACOMS-Q over other algorithms with respect to delay.

Network lifetime: The metric is closely related to energy consumption [34]. Distance is directly related to energy, and the LEACH protocol follows direct long transmission to the static sink (CH to sink), resulting in high energy consumption. In ACO-TSP, the mobile sink collects data, eliminating the long-distance transmission, but all nodes are active all the time. Incorporating the clustering mechanism, ACOMS-C, cluster heads are always active and aggregate data to send to MS on its arrival. Integrating a scheduling mechanism, ACOMS-SC, increases the network lifetime as the cluster heads are scheduled to wake up based on the arrival time of the mobile sink. ACOMS-Q outperforms by learning the network to activate only the nodes based on the priority of high data in the buffer and low energy, as evident in Figure 13. This results in optimized paths with high network longevity.

Tour length: The tour length parameter substantiates the capability of the proposed protocol to locate the best and most effective trajectory with the shortest path. The path traveled by the mobile sink towards sensor nodes for data gathering is called the tour length. It is evident from the results that tour length is minimal in ACOMS-Q, measuring 50.24m, as it considers only the active nodes for path finding. This finds to be a significant improvement in ACOMS-Q compared with other protocols, ACOMS-C and ACOMS-SC, that rely only on pheromone level and heuristic information to find the best tour. ACOMS-Q integrated the Q-learning algorithm that efficiently makes decisions from rewards by balancing exploration and exploitation, resulting in global optima.

Energy consumption: Energy is a crucial parameter for network longevity as sensor nodes are battery-powered [35]. Energy consumption in LEACH is highest due to long-distance transmission and multihop inefficiencies. In ACO-TSP, consumption is reduced with the integration of a mobile sink, but nodes are active, which results in energy drain. In clustered ACO, overall energy consumption is less, as only cluster heads are involved in aggregation. In ACOMS-SC, scheduling optimizes the energy consumption with the switching of sleep and wakeup modes. ACOMS-Q achieves higher efficiency through a machine learning approach and optimized path selection over the active nodes, which is simulated and shown in Figure 14. Energy per round for ACOMS-Q is found to be 0.106J, overperforming all the existing algorithms. Finding the active CHs and traversing only over those CHs improved the efficiency of the algorithm.

Packet Delivery Ratio (PDR): ACOMS-Q uses reinforcement learning and achieves superior reliability with dynamic selection of paths with the active cluster heads. With a static and stable infrastructure, LEACH also maintains a competitive PDR. ACOMS-SC outperforms the unscheduled approaches, such as ACO-TSP and ACOMS-C, as the scheduled approach provides better timing coordination, in turn reducing the occurrence of collisions and stabilizing the energy. The Q-learning enabled in ACOMS-Q results in updating the path decisions continuously from the previous actions and rewards. The intelligent scheduling in ACOMS-Q results in higher reliability than the approaches with static architecture and basic mobility, with a PDR of 98.1%, as evident from Figure 15.

ACOMS-Q, the proposed system, is compared with prior approaches to analyze the performance, and the obtained results are organized in Table 17. Also, comparison is evaluated on 95% Confidence Intervals (CI) and is tabulated in Table 18. It is evident that ACOMS-Q shows the overall best performance in all the metrices. The average delay of ACOMS-Q falls in the CI range of 126-150, showing a statistically significant improvement compared to other protocols. The energy consumption of ACOMS-Q falls in the CI range of 0.096-0.116, showing extensive improvement

with the LEACH protocol. Even though the confidence interval range of ACOMS-Q for the network lifetime parameter is high, the lower bound round is 8630, which is comparatively higher than the upper bound of all other approaches. This wider CI in ACOMS-Q is due to the exploration and continuous learning in RL. It is also observed that for all other metrices, ACOMS-Q provides a narrow CI range outperforming the other protocols.

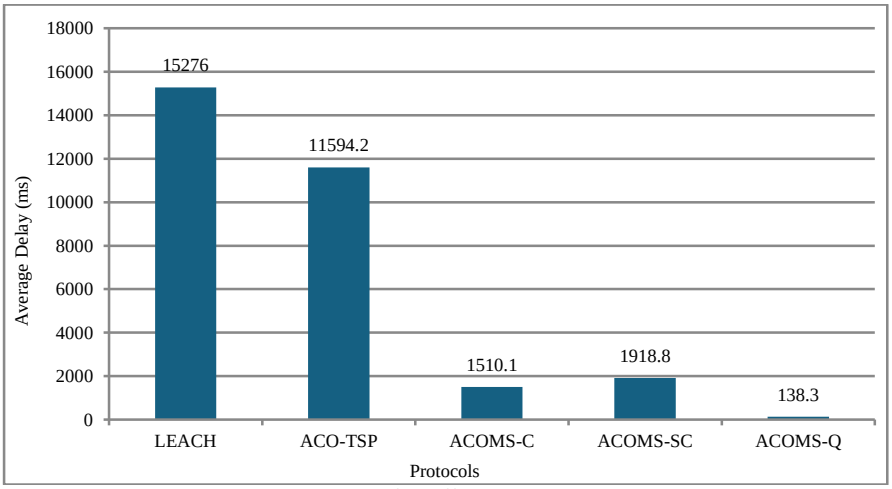

Fig. 12 Average delay comparison for different algorithms on 100-node scenarios

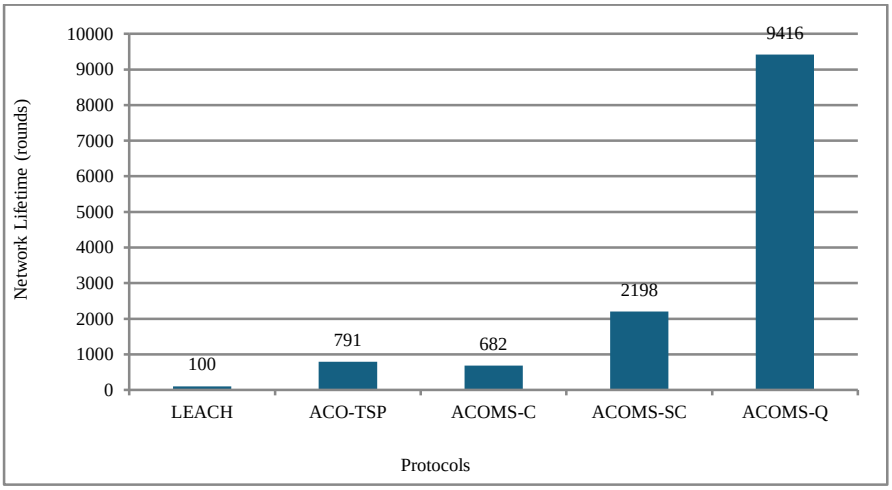

Fig. 13 Comparison of network lifetime for different algorithms on a 100-node scenario

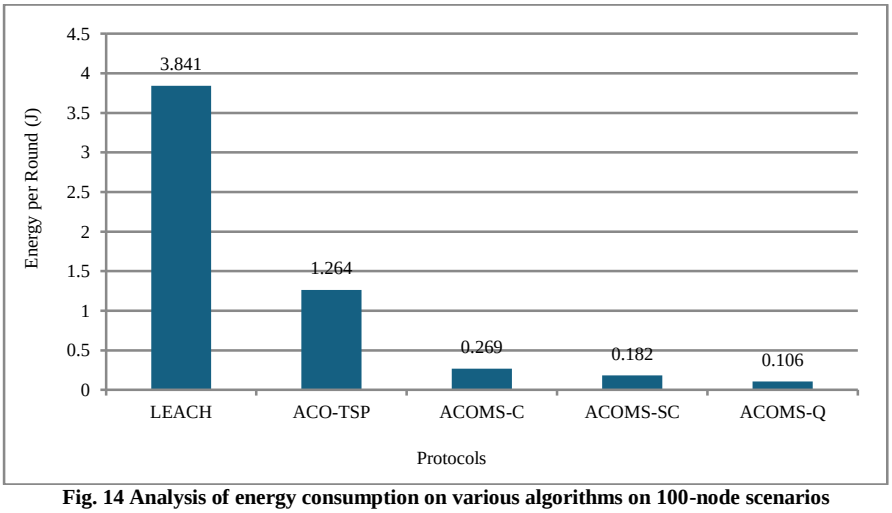

Fig. 14 Analysis of energy consumption on various algorithms on 100-node scenarios

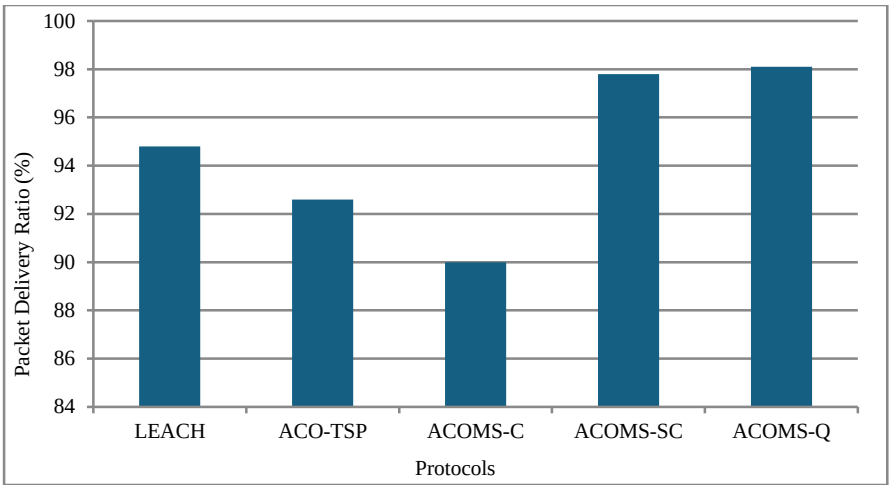

Fig. 15 Comparative analysis of PDR on different algorithms on 100-node scenarios

Table 17. Performance comparison of various protocols with metrics

Metric	LEACH	ACO-TSP	ACOMS-C	ACOMS-SC	ACOMS-Q
Tour Length of MS(m)	Static sink	883.53	125.38	125.38	50.24
Average Delay(ms)	15276.0	11594.2	1510.1	1918.8	138.3
Maximum Delay(ms)	33607.11	24068.17	3071.99	4573.71	228.01
Energy Consumption(J)	3.841	1.264	0.269	0.182	0.106
Network lifetime (rounds)	100	791	682	2198	9416
PDR (%)	94.8	92.6	90.0	97.8	98.1

Table 18. Performance comparison of various protocols with 95% confidence intervals

Metric	LEACH	ACO-TSP	ACOMS-C	ACOMS-SC	ACOMS-Q
Average Delay(ms)	15276.0±642	11594.2±369.22	1510.1±156.57	1918.8±240.79	138.3±12.37
Energy Consumption(J)	3.841±0.17	1.264±0.02	0.269±0.01	0.182±0.01	0.106±0.01
Network lifetime (rounds)	100±0	791±15	682±51	2198±139	9416±786
PDR (%)	94.8±1.34	92.6±0.32	90.0±0.00	97.8±0.51	98.1±0.03

6. Conclusion

The research proposes mathematical analysis and simulation of a hybrid optimization framework incorporating a reinforcement Q-learning algorithm into the ACO bio-inspired algorithm. This integration provides efficient data collection and enhances the intelligence in decision-making using a mobile agent.

ACO-TSP is widely used to locate the shortest routing tour resulting in low energy consumption and delay. The intelligent decision-making-based proposed approach, ACOMS-Q, uses ACO-TSP and an RL-based Q-learning algorithm for the shortest path selection, taking Q value along with heuristic information and pheromone level, thereby increasing the path-finding accuracy and overall network lifetime.

Mathematical formulations were developed to model the network with the integration of learning-based node prioritization with the heuristic-driven ACO algorithm. The algorithm performs efficient clustering and dynamically schedules node wakeup times based on Q values obtained from buffer occupancy and residual energy, ensuring that high-priority nodes are visited earlier in the sink's route. The suggested approach dramatically enhances important performance measures like delay, energy usage, network lifetime, tour length, and delivery ratio according to simulation results produced with MATLAB. ACOMS-Q is compared with LEACH, ACO-TSP, ACOMS-C, and ACOMS-SC on various metrices and it is found that the proposed methodology outperforms all the other protocols with the highest network lifetime, energy efficiency, and reduced delay.

The energy and buffer-sensitive mobile data gathering with the ACOMS-Q learning hybrid model delivers a scalable and flexible solution for many applications in WSN. The limitation of the ACOMS-Q is that even though the algorithm results in high network lifetime and reduced delay, meeting the objectives, the computational Complexity and training time with the updating of the Q table and learning policies when deployed in the real world will be high. Scalability is another limitation as the number of nodes increases; computation complexity increases with increased memory

requirement, which represents an important deployment barrier.As the system requires learning, slow convergence is a drawback when implemented on large-scale dynamic environments. The limitations related to mobility of the mobile sink in terms of path planning, obstacles, and battery usage need to be considered for practical field deployment. The future scope should be on improvements in reinforcement learning that result in less computational Complexity with fast convergence, yet maintaining good operational longevity and resource efficiency.

In the future, the system can also be expanded to real-time, multi-sink scenarios and can be incorporated with a deep neural network [36, 37]. Validation on the testbed to check the response of the algorithm can be taken as future work.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Dionisis Kandris et al., "Applications of Wireless Sensor Networks: An Up-to-Date Survey," *Applied System Innovation*, vol. 3, no. 1, pp. 1-24, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[2] Marco Dorigo, Mauro Birattari, and Thomas Stutzle, "Ant Colony Optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[3] Gilbert Laporte, "The Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms," *European Journal of Operational Research*, vol. 59, no. 2, pp. 231-247, 1992. [CrossRef] [Google Scholar] [Publisher Link]

[4] Oday Al-Jerew, Nizar Al Bassam, and Abeer Alsadoon, "Reinforcement Learning for Delay Tolerance and Energy Saving in Mobile Wireless Sensor Networks," *IEEE Access*, vol. 11, pp. 19819-19835, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[5] Islam Mosavvar, and Ali Ghaffari, "Data Aggregation in Wireless Sensor Networks using Firefly Algorithm," *Wireless Personal Communications*, vol. 104, no. 1, pp. 307-324, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[6] Vishal Kumar Arora, Vishal Sharma, and Monika Sachdeva, "ACO Optimized Self-Organized Tree-based Energy Balance Algorithm for Wireless Sensor Network: AOSTEB," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 12, pp. 4963-4975, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[7] Adam Kozłowski, and Janusz Sosnowski, "Energy Efficiency Trade-Off Between Duty-Cycling and Wake-Up Radio Techniques in IoT Networks," *Wireless Personal Communications*, vol. 107, no. 4, pp. 1951-1971, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[8] Praveen Kumar Donta, Tarachand Amgoth, and Chandra Sekhara Rao Annavarapu, "An Extended ACO-based Mobile Sink Path Determination in Wireless Sensor Networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 10, pp. 8991-9006, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[9] Vishal Kumar Arora, Vishal Sharma, and Monika Sachdeva, "A Multiple Pheromone Ant Colony Optimization Scheme for Energy-Efficient Wireless Sensor Networks," *Soft Computing*, vol. 24, no. 1, pp. 543-553, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Palvinder Singh Mann, and Satvir Singh, "Optimal Node Clustering and Scheduling in Wireless Sensor Networks," *Wireless Personal Communications*, vol. 100, no. 3, pp. 683-708, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[11] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA, vol. 2, pp. 1-10, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[12] O. Younis, and S. Fahmy, "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366-379, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[13] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29-41, 1996. [CrossRef] [Google Scholar] [Publisher Link]

[14] Jing-hui Zhong, and Jun Zhang, "Ant Colony Optimization Algorithm for Lifetime Maximization in Wireless Sensor Network with Mobile Sink," *GECCO'12: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation*, Association for Computing Machinery, New York, NY, United States, pp.1199-1204, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[15] Weimin Wen et al., "EAPC: Energy-Aware Path Construction for Data Collection using Mobile Sink in Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 18, no. 2, pp. 890-901, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[16] Jin Wang et al., "An Improved Ant Colony Optimization-based Approach with Mobile Sink for Wireless Sensor Networks," *The Journal of Supercomputing*, vol. 74, no. 12, pp. 6633-6645, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[17] D. Praveen Kumar, Amgoth Tarachand, and Annavarapu Chandra Sekhara Rao, "ACO-based Mobile Sink Path Determination for Wireless Sensor Networks Under Non-Uniform Data Constraints," *Applied Soft Computing*, vol. 69, pp. 528-540, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[18] Minal Shahakar, S.A. Mahajan, and Lalit Patil, "Enhancing Resource Utilization and Load Distribution with ACO and Reinforcement Learning in Dynamic Computing Infrastructure," *Panamerican Mathematical Journal*, vol. 34, no. 1, pp. 14-24, 2024. [CrossRef] [Publisher Link]

[19] Zhou Wu, and Gang Wan, "An Enhanced ACO-based Mobile Sink Path Determination for Data Gathering in Wireless Sensor Networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[20] Muralitharan Krishnan, and Yongdo Lim, "Reinforcement Learning-based Dynamic Routing using Mobile Sink for Data Collection in WSNs and IoT Applications," *Journal of Network and Computer Applications*, vol. 194, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[21] Madana Srinivas, and Tarachand Amgoth, "Data Acquisition in Large-Scale Wireless Sensor Networks using Multiple Mobile Sinks: A Hierarchical Clustering Approach," *Wireless Networks*, vol. 28, no. 2, pp. 603-619, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[22] Praveen Kumar Donta et al., "Data Collection and Path Determination Strategies for Mobile Sink in 3D WSNs," *IEEE Sensors Journal*, vol. 20, no. 4, pp. 2224-2233, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[23] Vikram Rajpoot et al., "Analysis of Machine Learning based LEACH Robust Routing in the Edge Computing Systems," *Computers and Electrical Engineering*, vol. 96, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[24] Peng Xiong, Dan He, and Tiankun Lu, "A Q-Learning based Target Coverage Algorithm for Wireless Sensor Networks," *Mathematics*, vol. 13, no. 3, pp. 1-14, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[25] Yiyang Liu et al., "Improved Dyna-Q: A Reinforcement Learning Method Focused via Heuristic Graph for AGV Path Planning in Dynamic Environments," *Drones*, vol. 6, no. 11, pp. 1-17, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[26] Benjamin Freed et al., "Unifying Model-based and Model-Free Reinforcement Learning with Equivalent Policy Sets," *Reinforcement Learning Conference*, pp. 1-19, 2024. [Google Scholar] [Publisher Link]

[27] Xu Wang et al., "Deep Reinforcement Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 4, pp. 5064-5078, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[28] Lieping Zhang et al., "A Self-Adaptive Reinforcement-Exploration Q-Learning Algorithm," *Symmetry*, vol. 13, no. 6, pp. 1-16, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[29] Bandar Al-Ghamdi, Marwane Ayaida, and Hacène Fouchal, "Scheduling Approaches for Wireless Sensor Networks," *2015 15th International Conference on Innovations for Community Services (I4CS)*, Nuremberg, Germany, pp. 1-6, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[30] Zhenchun Wei et al., "A Q-Learning Algorithm for Task Scheduling based on Improved SVM in Wireless Sensor Networks," *Computer Networks*, vol. 161, pp. 138-149, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[31] Yunsick Sung, Eunyoung Ahn, and Kyungeun Cho, "Q-Learning Reward Propagation Method for Reducing the Transmission Power of Sensor Nodes in Wireless Sensor Networks," *Wireless Personal Communications*, vol. 73, no. 2, pp. 257-273, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[32] Hui Li et al., "Energy Efficient Mobile Sink Driven Data Collection in Wireless Sensor Network with Nonuniform Data," *Scientific Reports*, vol. 14, no. 1, pp. 1-19, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[33] Shashank Singh, Veena Anand, and Pallav Kumar Bera, "A Delay-Tolerant Low-Duty Cycle Scheme in Wireless Sensor Networks for IoT Applications," *International Journal of Cognitive Computing in Engineering*, vol. 4, pp. 194-204, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[34] Ouadoudi Zytoune, Youssef Fakhri, and Driss Aboutajdine, "Lifetime Optimization for Wireless Sensor Networks," *2009 IEEE/ACS International Conference on Computer Systems and Applications*, Rabat, Morocco, pp. 816-820, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[35] Arif Ullah et al., "A Hybrid Approach for Energy Consumption and Improvement in Sensor Network Lifespan in Wireless Sensor Networks," *Sensors*, vol. 24, no. 5, pp. 1-18, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[36] G. Pushpa et al., "Optimizing Coverage in Wireless Sensor Networks using Deep Reinforcement Learning with Graph Neural Networks," *Scientific Reports*, vol. 15, no. 1, pp. 1-21, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[37] M. Senthamilselvi, and C. Ranjeeth Kumar, "Multi-Agent based DRL with Federated Learning for Data Transmission in Mobile Sensor Networks," *Automatika*, vol. 66, no. 3, pp. 475-490, 2025. [CrossRef] [Google Scholar] [Publisher Link]