

Received 21 June 2023, accepted 5 July 2023, date of publication 13 July 2023, date of current version 20 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3295119

## RESEARCH ARTICLE

# CDEIR: Intelligent Routing for Efficient Wireless Sensor Networks Using BUG Algorithm

P. SUMAN PRAKASH<sup>1</sup>, K. NAGARAJU<sup>2</sup>, A. VISHNUVARDHAN REDDY<sup>3</sup>, SHAIK FATHIMABI<sup>4</sup>, EBENEZER JANGAM<sup>4</sup>, SURBHI BHATIA KHAN<sup>5,6</sup>, (Senior Member, IEEE), AND AREEJ ABBAS MALIBARI<sup>7</sup>, (Member, IEEE)

<sup>1</sup>Department of Artificial Intelligence, G. Pullaiah College of Engineering and Technology, Kurnool, Andhra Pradesh 518002, India

<sup>2</sup>Department of Computer Science and Engineering, Indian Institute of Information Technology Design and Manufacturing Kurnool, Kurnool, Andhra Pradesh 518008, India

<sup>3</sup>Department of Computer Science and Engineering, G. Pulla Reddy Engineering College, Kurnool, Andhra Pradesh 518007, India

<sup>4</sup>Department of Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, Andhra Pradesh 520007, India

<sup>5</sup>Department of Data Science, School of Science, Engineering and Environment, University of Salford, Manchester, M5 4WT Salford, U.K.

<sup>6</sup>Department of Electrical and Computer Engineering, Lebanese American University, Byblos, Lebanon

<sup>7</sup>Department of Industrial and Systems Engineering, College of Engineering, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

Corresponding author: P. Suman Prakash (sumanprakashp@gmail.com)


Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R151), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**ABSTRACT** Wireless Sensor Networks (WSNs) are used to monitor specific areas of the environment by networking multiple sensors and collecting data for analysis. However, due to limited processing capabilities, the collected data needs to be transmitted to a Base Station (BS) that has high computational power and storage capacity. As SNs are not directly connected to the BS, multihop data transmissions through other SNs are required to reach the BS, which leads to congestion and additional energy consumption. To address these challenges, we propose a novel algorithm called Congestion, Delay, Energy-aware Intelligent Routing (CDEIR) using the BUG algorithm. The CDEIR approach identifies congested nodes using a Directed Spanning Tree and avoids traffic through them by paving an alternate path. This approach minimizes delay and optimizes energy consumption while avoiding congestion, all within a short computational time. We demonstrate the effectiveness of the CDEIR approach through theoretical analyses and simulations.

**INDEX TERMS** Bug algorithm, congestion avoidance, delay-aware, directed spanning tree, energy-efficiency, wireless sensor networks.

## I. INTRODUCTION

A responsive Wireless Sensor Network (WSN) is a collection of interconnected Sensor Nodes (SNs) that detect changes in their environment and relaying that information to a central hub or data processing unit in a timely and efficient manner [1]. These SNs can be deployed in a variety of domains, such as environmental monitoring, transportation [2], industrial automation, and smart cities [3]. One of the key features of a responsive WSN is its ability to adapt to changing conditions. This means that the network is able to detect changes in the environment and adjust its behavior

The associate editor coordinating the review of this manuscript and approving it for publication was Zhangbing Zhou .

accordingly. For example, in an environmental monitoring application, the network might adjust the frequency of data collection based on changes in weather conditions or air quality. It is also important for a responsive WSN to have the capability to operate efficiently and reliably. In order to achieve this, SNs must be carefully designed and optimized, as well as communication protocols and SNs that are energy-efficient [4].

In WSNs, routing is one of the most challenging issues, because data routing happens through multi-hop communication among SNs. But, nodes in WSNs are limited in terms of energy, processing power, and communication capabilities. Therefore, efficient routing algorithms are needed to ensure data is routed efficiently. To achieve this, routing algorithms

must be flexible enough to adapt to the rapidly evolving network environment. They must also consider the limited resources of the nodes. Additionally, the algorithms must be capable of handling the high traffic load and dynamically adjust directional paths according to the current conditions in the system [5], [6]. To do this, routing algorithms must efficiently process data, find shorter paths, and learn from past experiences. They must also be able to detect and avoid congestion, prioritize certain types of traffic, and minimize the number of hops to reduce latency [7]. Furthermore, they must be able to quickly respond to changes in the network topology and make decisions based on the current conditions of the network [8].

Many existing algorithms perform clustering and routing through various clustering approaches. They construct the shortest multihop route towards BS using a traveling salesperson approach. However, this approach has some disadvantages. For example, it can be computationally expensive, and it can be difficult to find the optimal solution. Additionally, this approach can be sensitive to changes in the network, and it can be difficult to scale up [9], [10]. As a result, we propose a novel approach to routing that aims to avoid congestion, minimize delays, and reduce energy consumption. The proposed algorithm is called Congestion, Delay and Energy-aware Intelligent Routing (CDEIR) algorithm. We construct a Directed Spanning Tree (DSP) and identify the weights of each node, which guides the load at each SN. In WSNs, this load helps identify the most efficient route while predicting congestion. As a result of the intelligent routing architecture, congestion can be minimized, delays are limited, and energy efficiency can be improved. In short, the contributions of this paper are summarized as follows:

- We construct a Directed Spanning Tree repeatedly with SNs and its available data packets, so that it helps to identify the weight/load of each node. By using these weights, we are able to predict congested network situations even more accurately.
- Once the DSP approach is determined, the load is used as an input for the Bug algorithm. As a result, the most efficient route is identified while avoiding the predicted congested nodes, and maintaining the shortest route at the same time.
- We simulate this proposed CDEIR approach using a Python simulator. We perform simulations considering different metrics related to congestion, delay and energy, to show the superiority of the proposed work over existing works.
- We perform theoretical analysis, and we confirm the superiority of the CDEIR algorithm. Furthermore, we estimate that the computational complexity of the proposed algorithm is the least among those already available.

The remaining sections of this paper are organized as follows. The literature study on recently published congestion, delay, and energy-aware algorithms and their limitations is

presented in Section II. The problem formulation for the proposed CDEIR is defined in Section III. The proposed CDEIR algorithm and its complete discussion through illustrative examples are presented in Section IV. The experimental results are evaluated in Section V, using both numerical and theoretical ways is presented in subsections V-B, and V-C, respectively. Finally, this paper is concluded with a future scope in Section VI.

## II. LITERATURE STUDY

In this section, we perform the literature study on recently published congestion, delay and energy-aware routing algorithms for WSNs.

A Reinforcement Learning (RL)-based optimal routing strategy is proposed by rabhu et al. [11] to achieve energy efficiency for WSNs. This scheme focuses on routing table compression, optimal data transmission paths to BS, and limiting energy consumption in BSs. Despite this, the degree of computational complexity of this work is high because it relies heavily on algorithmic learning. A balanced routing strategy for efficient data transmission, saving energy and efficient bandwidth utilization is described by Yu et al. [12]. A mathematical model embedded within each SN determines the transmission radius of each SN. Additionally, this helps maintain efficient data routing. It prolongs the network's lifespan by maintaining even power utilization between layers. A congestion control approach called RTTICC was proposed by Suman et al. in [13]. In RTTICC, SNs with congested traffic in the network are identified by applying a mathematical strategy. By avoiding congested nodes on a route, SNs construct dynamic routing paths. This model uses a RRT<sup>1</sup> algorithm used in robot path construction, but this approach requires more computations. This work further extended using meta-heuristic approaches for relay node selection in [14].

By using a multi-objective Deep Reinforcement Learning (DRL) approach, the scheme minimizes routing delays, minimizes energy consumption, and reduces packet loss by Agarwal et al. [15]. The DRL enables the scheme to quickly identify fault nodes and route data packets around them to maintain reliability. Additionally, the DRL can adjust the route dynamically to account for changes in the network such as traffic congestion or changes in node availability. This approach is further extended through mobile sink-based data routing in [16]. The Deep Graph Reinforcement Learning (DGRL) model incorporates a graph-based state representation and a graph-based reward function to learn an optimal control policy by Huang et al. in [17]. This enables the traffic control scheme to adapt to the given traffic conditions and optimize the energy consumption of the nodes in the network. Similarly, graph-aware deep learning is used for routing in WSNs by Yang et al. [18]. A feed forward back propagation is used to optimize data routing for air pollution applications by Dixit and Jindal [19]. In [20], Dorga et al. implement

<sup>1</sup>Rapidly-exploring Random Tree.

dynamic clustering and intelligent routing. In this, a metric is used to measure the load of a node based on the number of packets in the queue. In addition, it utilizes a clustering algorithm to group nodes with similar loads and route traffic away from congested nodes. This allows traffic to be routed more efficiently and reduces congestion risk.

A routing protocol based on  $A^*$  and the ant colony algorithm for efficient routing for data gathering in WSNs was proposed by Ibrahim and Ahmed [21]. The  $A^*$  algorithm considers the shortest path between two nodes, while the ant colony algorithm is used to identify cluster heads in systems. The combination of these two algorithms allows for faster and more efficient data gathering in WSNs. The grasshopper optimization algorithm (GOA) and golden eagle optimization (GEO) together improve routing efficiency for WSNs in [22]. Equilibrium optimization and genetic algorithms are employed for efficient clustering and routing for energy-efficient WSNs in [23]. Equilibrium optimization is used to determine the optimal number of clusters for a WSN by minimizing communication energy consumption. Genetic algorithms are used to generate the most efficient routing paths between the clusters, based on the WSN topology. From [24], the Krill herd and Cuckoo search algorithms are used together to achieve efficient clustering and routing algorithms to achieve long-term network lifetimes for WSNs. The use of Recurrent Neural Networks (RNN) is proposed for AI-based interactive routing for WSNs in [25]. This is because RNN can learn from experience and adapt to changing situations quickly. Thus, they can be used for routing in rapidly changing and unpredictable environments.

The Bayesian machine learning algorithms used in [26] are able to identify patterns in the data collected by the WSNs. This enables them to predict the most efficient data transmission routes. This in turn leads to more accurate routing and better overall performance approximately 86.67% compared with existing routing algorithms. Chakraborty et al. [27] introduced a Q-learning-based routing algorithm for balancing load among SNs. This algorithm works by using an iterative learning process to determine the most efficient route for the nodes to take, based on the current load in the network. It adapts to changes in the network by continuously learning and updating the most appropriate route for the nodes to take. Keum and Ko [28] proposed a method through Q-learning results focusing on trustworthiness, quality of services, and energy efficiency, this technology prioritizes mission-critical data trustworthiness. Similarly, Saleh et al. also proposed a similar method in [29] using mobile edge nodes for trust-aware routing. Tewari and Tripathi [30] used a Neuro-fuzzy approach for minimizing hotspot issues and constructing efficient routing.

As a whole, most existing algorithms use several meta-heuristics and learning algorithms, but routing algorithms are not responsive or intelligent. All the literature is compared using Table 1 for easy to refer. In this context, this paper introduces a routing algorithm that can be aware

of congestion, delay, and energy to construct an intelligent routing strategy for responsive WSNs.

### III. PROBLEM DEFINITION

We consider an area  $A$  with a set of SNs ( $\mathcal{S} = \{S_i | \forall 1 \leq i \leq n\}$ ) which are deployed randomly. SNs are connected with other based on a communication range ( $r_c$ ) to communicate control signals. Data packets can also be transmitted between  $S_i$  and  $S_j \forall i, j \in (1, n)$  over the transmission range ( $r_t$ ). A single BS ( $S_0$ ) collects data from all the nodes in the network. We assume connected SNs and base station are treated as an undirected graph  $G$ . From  $G$ , we assume edges are communication paths between SNs, and vertices are SNs. We consider  $\mathcal{S} \cup S_0$  are considered as non-stationary from deployment to until end of simulation. Routing process of  $\mathcal{G}$  uses the tree topology while transmitting data from  $\mathcal{S}$  to  $S_0$ .  $NN_i = \{S_j | d(S_i, S_j) \leq r_c \forall i \neq j \wedge S_j \in \mathcal{S}\}$  is considered as neighbour node set of  $S_i$ , where  $d(S_i, S_j)$  indicates the distance from node  $S_i$  to  $S_j$ . The initial  $G$  can be split into  $k$  number of trees during routing, whereas each tree  $\mathcal{T}_k$  contains a common root node i.e.,  $S_0$ , and  $\mathcal{T}_k \subseteq G$ . The nodes involved in each  $\mathcal{T}_k$  dynamically change during simulation while routing data.  $\Delta_k$  is considered as the depth of  $\mathcal{T}_k$ , which is also defined as the number of edges between  $S_i, \forall 1 \leq i \leq \Delta_k$  and  $S_0$ .  $\mathcal{T}_k$  is uni-directional from leaf node  $S_i, \forall 1 \leq i \leq \Delta_k$  to  $S_0$ .  $\vartheta$  is considered as packet service rate and it is unique  $\forall \mathcal{S}$  until end of  $T$ , where  $T$  indicates total simulation time. The energy model used is different for various networks, however in this paper we use according to Donta et al. [31].

While sending  $\Lambda$  data packets from  $S_i$  to  $S_j$  energy exhaust is calculated according to Eq. (1)

$$E_{ij}^t = (\Lambda \times a) + (\Lambda \times b \times d_{ij}^2) \quad (1)$$

where  $a$  and  $b$  are amplification, and processing energy for a single bit data, respectively.  $\Lambda$  is the amount of data packets transmitted between  $S_i$  and  $S_j$ . Similarly, a node  $j$  take to receive  $\Lambda$  data from its  $NN_i$  is considered as Eq (2).

$$E_i^{rx} = r \times \Lambda \quad (2)$$

where  $r$  means the energy exhaust for receiving a single bit from  $NN_i$ . Overall, a SN  $S_i$  's energy exhaust for both transmission and receiving of data is reepresented in Eq (3)

$$E_i = E_{ij}^{tx} + E_i^{rx} \quad (3)$$

After a particular amount of energy among available energy is exhausted, we calculate remaining energy using Eq. (4).

$$\xi_i = \begin{cases} E_0 - E_i & \text{First iteration} \\ \xi_i - E_i & \text{Otherwise} \end{cases} \quad (4)$$

The network lifespan of WSN in this paper is considered as the time until the first node exhausts its energy completely as per Donta et al. [9], and it is calculated mathematically as shown in Eq. (5).

$$\mathcal{N} = E_0 \times (\xi)^{-1} \quad (5)$$

TABLE 1. Summary of recently published literature study.

Reference	Method/s	Congestion	Delay	Energy-efficiency	Remarks
Prabhu et al. [11]	Reinforcement learning	✗	✓	✗	Routing table compression
Yu et al. [12]	Mathematical models	✗	✗	✓	Balance energy among the SNs
Suman et al. in [13]	RRT	✓	✗	✓	Efficiently identify congested nodes
Suman et al. [14]	ACO	✗	✓	✓	efficiently relay node selection
Agarwal et al.[15]	DRL	✗	✓	✓	Computationally heavy
Agarwal et al.[16]	Mobile sink-based approach	✗	✓	✓	Delay increased
Huang et al. [17]	DGRL	✓	✗	✓	Computational intensive
Yang et al. [18]	Graph-based deep learning	✗	✗	✓	Computational intensive
Dixit et al.[19]	Feed forward back propagation	✗	✓	✗	Air pollution control
Dogra et al.[20]	Dynamic clustering	✓	✗	✗	Delay increased
Ibrahim et al.[21]	A* algorithm	✗	✓	✗	Computational and memory intensive
Roberts et al. [22]	GOA and GEO	✗	✗	✓	Near-optimal solution, but not efficient
Heidari et al.[23]	Equilibrium optimization	✗	✗	✓	Efficient topology construction
Zachariah et al.[24]	Krill herd and Cuckoo search	✗	✗	✓	Quick alternate selection
Sumathi et al.[25]	RNN	✗	✗	✗	Adding intelligent to decide the route
Vanitha et al. [26]	Bayesian machine learning	✗	✓	✓	Efficient routing selection
Chakraborty et al.[27]	Q-learning	✓	✗	✗	Incremental learning for efficient routing
Keum et al. [28]	Q-learning	✗	✗	✓	Enable trust-worthiness at SNs
Saleh et al.[29]	Mobile edge node	✗	✗	✓	Enable trust-worthiness at SNs
Tewari et al.[30]	Neuro-fuzzy approach	✗	✗	✓	Efficient hotspot problem
<b>Proposed CDEIR</b>	Heuristic and Bug2 algorithm	✓	✓	✓	Controlling congestion, delay and energy efficiency

where  $E_0$  indicate the initial available energy of SNs, and  $\Upsilon$  indicates the maximum energy consumed at a time  $t$  in  $G$ , and it is calculated using Eq. (6).

$$\Upsilon = \max_{i \in n} (E_i) \tag{6}$$

where  $E_i$  is calculate using Eq. (3).

PDR refers to the ratio between packets received by the sink ( $\mathcal{R}$ ) and packets transmitted by the SNs ( $\Upsilon$ ) during the time  $T$ . This ratio gives us an indication of the network’s overall performance in terms of packet delivery rate. It tells us how many packets are being successfully delivered by the SNs and how many are being received by the sink. It is computed mathematically using Eq. (7).

$$PDR = \frac{1}{\Upsilon} \times \mathcal{R} \tag{7}$$

where  $\mathcal{R} \leq \Upsilon$ , and  $\Upsilon$  is calculated using the Eq. (8)

$$\Upsilon \cong \sum_{i=1}^n P(\mathcal{S}_i) \tag{8}$$

where the data packets available at  $\mathcal{S}_i$  in a particular time  $t$  is denoted using  $P(\mathcal{S}_i)$ .

A BS’s throughput ( $\sigma$ ) is measured by the number of packets it receives during a given unit of time  $T$  and mathematically denoted as shown in Eq. (9).

$$\sigma = \frac{1}{T} \times \mathcal{R} \tag{9}$$

Latency ( $L$ ) of a packet refers to the amount of time it takes to receive a packet ( $p$ ) from  $\mathcal{S}_i$  to  $\mathcal{S}_0$ . The goal of latency optimization is to minimize the total time spent sending a packet from its source to its destination. To achieve this, techniques such as packet queuing, packet scheduling, and packet routing are optimized. The  $\mathcal{L}$  includes the radio propagation delay ( $\mathcal{L}_r$ ), queuing delay ( $\mathcal{L}_q$ ), transmission

delay ( $\mathcal{L}_t$ ), and signal processing delay ( $\mathcal{L}_s$ ). From these,  $\mathcal{L}_r(k) \approx \mathcal{L}_s(k) \leq 1$ , so we neglect  $\mathcal{L}_r(k)$  and  $\mathcal{L}_s(k)$  because of no effect on outcome. It is computed using Eq. (10).

$$\mathcal{L}(p) = \begin{cases} \mathcal{L}_t(p) + \mathcal{L}_q(p) & \text{For Successful} \\ (\mathcal{L}_t(p) + \mathcal{L}_q(p)) \times \eta(p) & \text{For re-transmission} \end{cases} \tag{10}$$

where the  $\eta(p)$  denotes retransmission count of packet  $p$ . The average latency ( $\mathcal{L}_a$ ) of a WSN is measured using Eq. (11)

$$\mathcal{L}_a = \sum_{i=1}^n \left( \sum_{p=1}^{P(\mathcal{S}_i)} (\mathcal{L}(p)) \right) \times (\mathcal{R})^{-1} \tag{11}$$

As part of the CDEIR, the objective is to improve the PDR,  $\mathcal{N}$  and  $\sigma$  of the WSNs by minimizing their  $E_a$  (measured using Eq. (16)) and  $\mathcal{L}_a$ .

#### IV. PROPOSED CDEIR

There are two stages in the proposed CDEIR algorithm: DST construction and route construction. During the DST construction stage, a spanning tree is constructed according to data availability with each SN direction specifying transmission order. Then, during the route construction stage, the optimal route is determined for each SN direction by considering global minima. It is calculated by adding the number of hops and residual energy of each node in the path while avoiding congested nodes. This helps to ensure that the optimal intelligent route is chosen while conserving energy and reducing delay. The following subsections provide in-depth information on DST construction and intelligent route determination.

##### A. DST CONSTRUCTION

Each SN in the network is measured using DST in order to determine its weight and it further helps to identify whether



the node is congested or not, which is motivated from [32]. From the given deployment graph  $G$ , it constructs a spanning tree  $T$ . At a time  $t$ , each sensor node is assigned a weight ( $p$ ) based on how many packets it generates or relays. A DST is constructed in two phases as shown below:

**Step I:** If node  $S_i$  is within BS premises, then it will be connected to BS.

**Step II:** The SN  $S_i$  forwards its data to the SN  $S_j$  when  $p_i < \max(p_j) \forall j \in N(i)$  is met. Therefore,  $S_j$  becomes the parent of  $S_i$ . If two neighbor nodes have the same  $p$ , the neighbor with the smallest hop count becomes the parent node.

**Step III:** Otherwise ( $p_i \geq \max(p_j) \forall j \in N(i)$ ),  $S_i$  chooses its parent  $S_j$  based on its hop count to the base station instead of looking at the neighbour nodes. It chooses the node with the minimum weight when two or more nodes have the same hop count. Nevertheless, the tie is not broken, so the node with more children becomes the forwarding node for  $S_i$ .

The weight is denoted using  $\mathcal{W}$ , and it indicates the weight of the node and all its children's weight. The  $\mathcal{W}$  is calculated once the  $\mathcal{T}$  is constructed. The weight of  $S_i$  i.e.,  $\mathcal{W}_i$  is calculated using Eq. (12)

$$\mathcal{W}_i = \begin{cases} p_i & \text{iff } N(i) = \phi \\ \left( \sum_{j \in N(i)} p_j \right) + p_i & \text{Otherwise} \end{cases} \quad (12)$$

where  $N(i)$  is the children of the node  $S_i$ .

We provide an illustrative example for better understanding of the proposed DST construction. In this illustration we consider 17 SNs and each associated with a different amount of packets at a time  $t$ , as shown in Fig. 1(a). The initial weights are also shown in Table 2. SNs deployed randomly are connected via edges if the transmission ranges for both are within the range of each sensor node. According to the communication ranges of the SNs, initial connections are assumed. Initially, step I is validated with all SNs in WSNs. For this example, step I decides nodes  $\{S_4, S_5, S_7, S_{10}, S_{17}\}$  to be connected to BS i.e.,  $S_0$  because they are in proximity to it. The direction to be adjusted towards  $S_0$ .

Now apply step II (Fig. 1(b)) to the remaining nodes. Now we consider each node and create a direction edge from lower weight nodes to higher weight nodes. The node  $S_1$  makes a direction to node  $S_5$ , because node  $S_5$  provides more data packets (i.e., 21) which is higher compared to node  $S_1$  (i.e., 18). Next, node  $S_3$  makes a direction to node  $S_{17}$ , because node  $S_{17}$  contains more data packets (i.e., 27) compared to node  $S_3$  (i.e., 18). Next, node  $S_6$  makes a direction to node  $S_5$ , because node  $S_5$  contains more data packets (i.e., 21) compared to node  $S_6$  (i.e., 12). Next, node  $S_8$  makes a direction to node  $S_7$ , because node  $S_7$  contains more data packets (i.e., 12) than node  $S_8$  (i.e., 11). After that, node  $S_9$  makes a direction to node  $S_{10}$ , because node  $S_{10}$  contains more data packets (i.e., 19) than node  $S_9$  (i.e., 16).

Similarly, node  $S_{11}$  make a direction to node  $S_{13}$ ,  $S_{12}$  makes a direction to node  $S_{14}$ ,  $S_{13}$  make a direction to node  $S_{15}$ ,  $S_{14}$  make a direction to node  $S_{15}$ ,  $S_{15}$  make a direction to node  $S_{17}$ , and  $S_{16}$  make a direction to node  $S_{15}$ . After Step II, all nodes except  $S_2$  are connected. Now, we move to Step III (Fig. 1(c)), and node  $S_2$  can change data transmission direction to node  $S_4$ , because both  $S_2$   $S_4$  contains 19 data packets, but node  $S_4$  is already connected with minimum hop count from  $S_0$ .

Next, we calculate the Forwarding Weight (FW) of each node using Eq. (12). In this case, the weights of leaf nodes are the same as the FWs. For example, The FW of node  $S_1$  is same as its weight i.e., 18. Similarly, FW of node  $S_{16}$  is same as its weight i.e., 11. Other than leaf nodes, node FW is calculated by combining their own weight and all their children's weights. For example, the FW of  $S_5$  is the weight of itself and its children i.e.,  $S_1$  and  $S_6$ . So, the FW of  $S_5$  is considered in this case as  $(21 + 18 + 12 = 51)$ . The resultant forwarding weight of each SN for the DST is shown in Fig. 1(d) and is summarized using Table. 2.

Congested nodes are identified according to the buffer size and forwarding weight of sensor nodes. We use Eq. (13) to identify whether a node is congested or not.

$$C(S_i) = \begin{cases} \frac{\rho \times \mathfrak{P}}{\mathfrak{G} \times B_i + \epsilon} & NN_i \neq 0 \\ 0 & \text{Otherwise} \end{cases} \quad (13)$$

where  $\rho$  indicates mean processing delay at source node  $S_i$ ,  $\mathfrak{P}$  indicates the amount of data packets  $p$  are competitive to occupy buffer  $B_i$ , which is computed as  $(\mathcal{W}_i - p_i)$ .  $\mathfrak{G}$  is denoted as the mean time gap between two data packets competing for  $B_i$ .  $\epsilon$  is a constant for avoiding divided by zero error.

From the running example shown in Fig.(1), and the computed  $\mathcal{W}$  shown in Table 2, we can find congested nodes. We identify that two nodes possibly cause congestion in  $G$ , and they are  $S_{15}$  and  $S_{17}$ . Now, it is necessary to balance the weights in these routes to mitigate congestion, and the Bug2 algorithm is used to balance and adjust routing in the next subsection i.e., Section IV-B.

### B. INTELLIGENT ROUTING

The intention of Intelligent Routing (IR) is to mitigate congestion by altering the data forwarding route towards  $S_0$ , by considering the number of relay nodes (which help to reduce energy consumption due to unnecessary relays), and also avoid the delay to reach data packets timely to the BS. In this context, we use the Bug2 algorithm [33], which is used to identify an optimal obstacle-aware path for a robot. The Bug2 algorithm is capable of detecting and avoiding obstacles while simultaneously minimizing the cost of routing. This is achieved by using two basic behaviors: bug-to-goal and bug-to-obstacle. The Bug2 algorithm is suitable for IR, as it is designed to optimize paths in both static and dynamic environments. In IR approach, Bug 2 consider an obstacle as a congested node. The IR through Bug2 is based on an

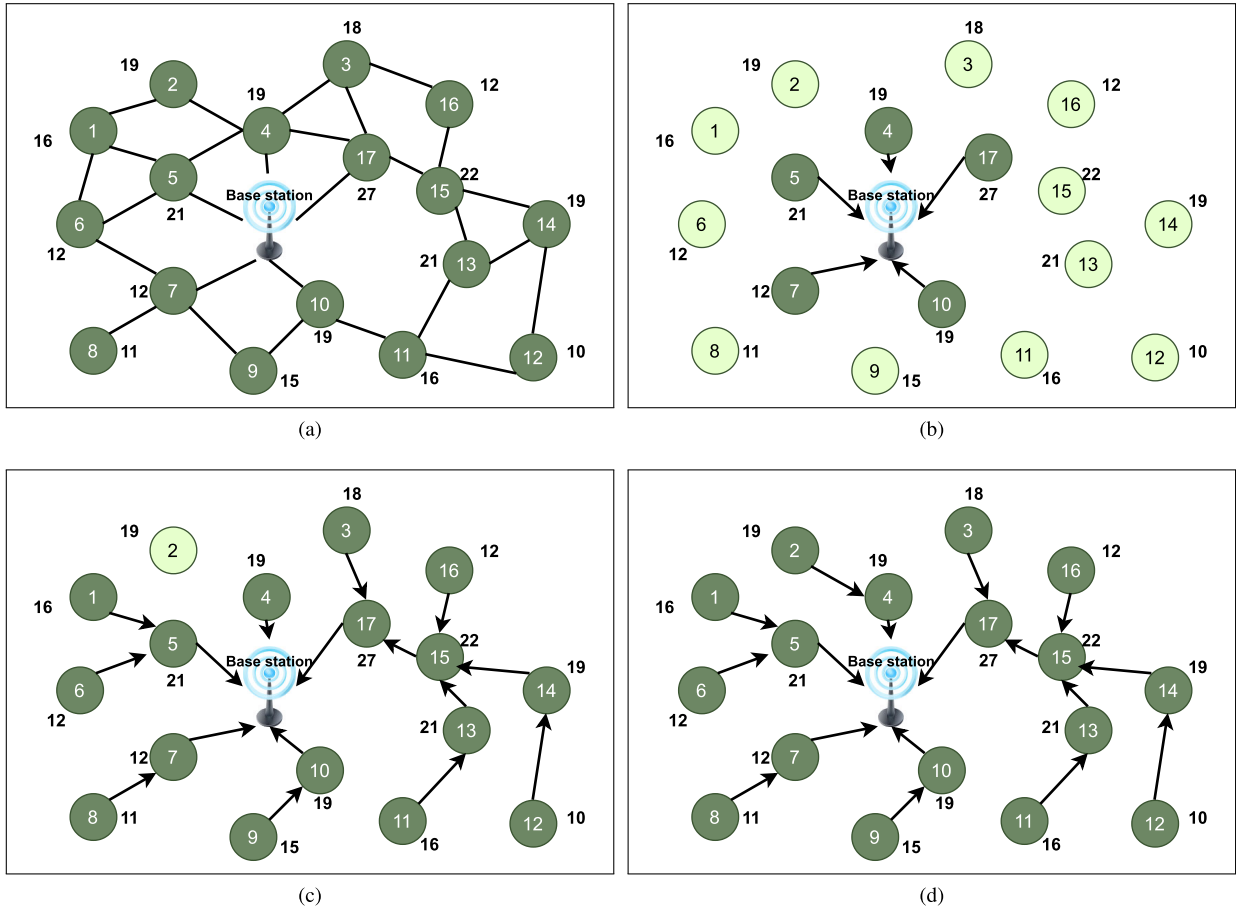


FIGURE 1. Example (a) WSN with its IDs and weights (b) Step I (c) Step II (d) Step III - Constructed DST.

TABLE 2. Weights and FW's for each sensor node in Illustrative example.

$S_i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
$p_i$	18	19	18	19	21	12	12	11	15	19	16	10	21	19	22	12	27
$W_i$	18	19	18	38	51	12	23	11	15	34	16	10	37	29	100	12	127

iterative process, wherein the solution is improved by taking into account local neighborhoods of the current solution. This helps to reduce the computational cost of the process, while still achieving local maxima. Also, it checks the alternate minimum distance between the source node  $S_i$  and the  $S_0$  simultaneously. Fig. 2 shows a flow diagram of the Bug algorithm to construct IR between a node and BS.

As soon as a congestion node (any  $S_j$ ) is recognized in a routing path, the Bug2 algorithm begins alternate possibilities around its neighbour nodes  $NN_j$ . In order to avoid  $S_j$ , it computes an alternative route from  $S_i$ 's current location  $(lo_i, la_i)$ , and the next visiting point  $(lo_j, la_j)$  is considered a non-congested route and the slope and y-intercept are determined using Equation, following which the next node in the route is considered as a straight line Eq. (14) and Eq. (15).

$$slope = \tanh\left(\frac{la_j - la_i}{lo_j - lo_i}\right) \tag{14}$$

$$intcpt = la_j - (slope \times lo_j) \tag{15}$$

After identifying the congestion node, the Bug2 algorithm determines the  $NN$  of the congested nodes and traverses them that way. Further, there are multiple routes to traverse around the congested nodes, but it determines which is the shortest path to traverse towards BS. The algorithm also calculates the number of hops from the node to BS and stores it in the node as shown in Algorithm 1. Finally, it sends the data back to BS for further processing.

From Algorithm 1, Initially, the routing starts from the leaf node. Then, it moves towards the closest neighbor which has less hop count from BS. Additionally, the nodes between them will be checked for congestion. If it find any congested node/s, it eliminates the route through congested node and builds an alternate path between the SN and BS. Finally, the packet is routed to the BS by the shortest alternate path. If a congested node is not found, the shortest path between SN and BS is selected. The packet is then routed to the BS along this path. The data packet is then transmitted from one node to the next until it reaches the BS. The BS then processes the

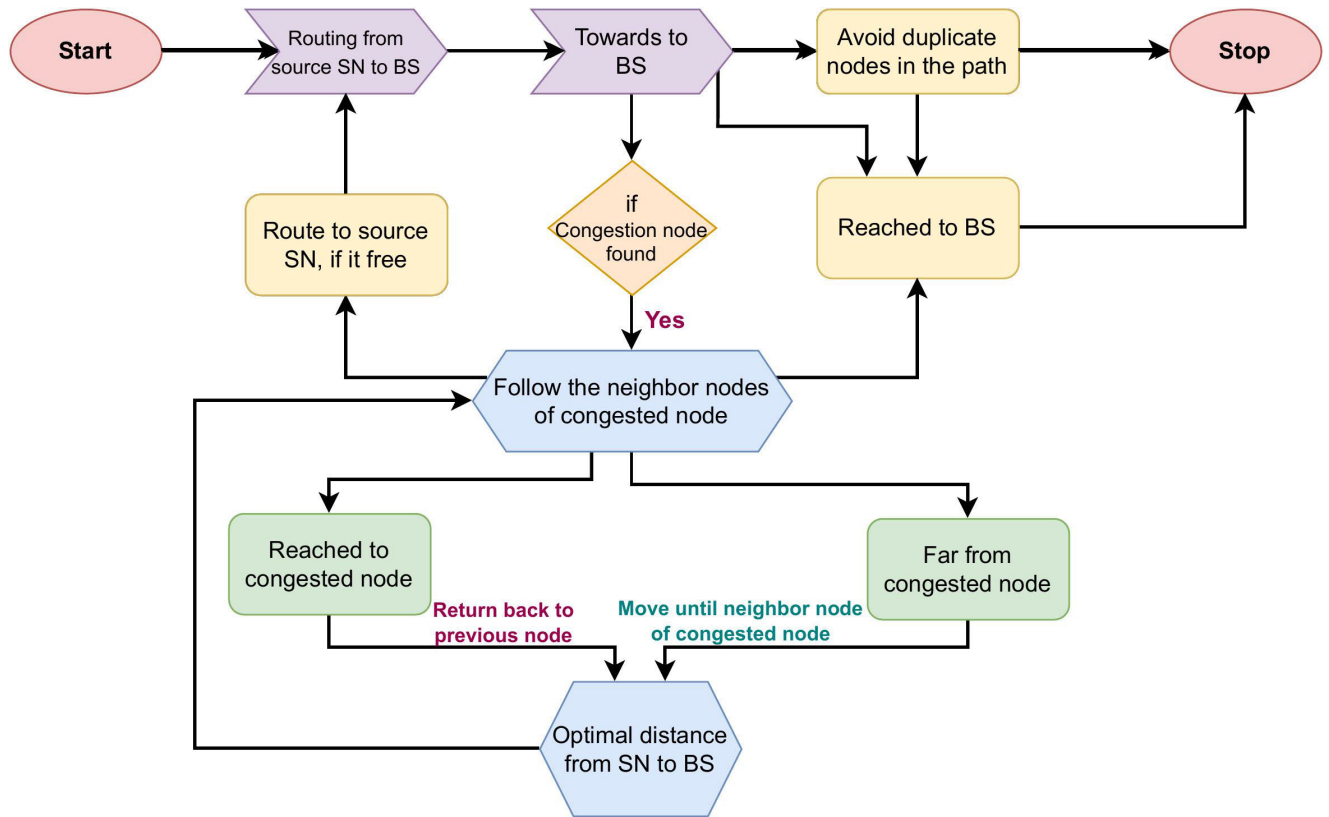


FIGURE 2. The general working procedure of the BUG algorithm during path construction.

**Algorithm 1** Intelligent Route Construction Using Bug Algorithm

```

1: while TRUE do
2:   repeat
3:     Move MS from current location to BS.
4:   until  $O_i = TRUE$ 
5:   if status(S) == VISITED then
6:     STOP
7:   end if
8:   repeat
9:     Travel near  $C_i$ 's neighbor
10:  until status(S) == VISITED
11:  Calculate least hop-count
12:  Move to the initial SN  $S_i$ 
13:  if Move to next SNs then
14:    GOAL not reached
15:    EXIT
16:  end if
17: end while
    
```

packet and responds back to the SN. The response follows the same path in reverse. In this way, all the congested nodes are avoided in the routes, while the congested nodes become normal by emptying their buffers and by avoiding the relay data from its neighbour nodes. Thus, delays and energy are

optimized, and congestion is reduced. The data packet is then transmitted back to the SN, and the communication is complete.

We use the running example shown in Fig. 1(b), and optimize the route by avoiding congested nodes. From Table 2, and Eq. (13), we identify the node  $S_{15}$  is congested (which is highlighted in Fig. 3(a)). From this figure (Fig. 3(a)), the nodes  $S_{11}$  (16 packets),  $S_{12}$  (10 packets),  $S_{13}$  (21 packets),  $S_{14}$  (19), and  $S_{16}$  (12 packets) are traveling through node  $S_{15}$  followed by  $S_{17}$  and  $S_0$ . In this way, the node  $S_{15}$  relays 78 packets along with 22 packets. It appears that the buffer  $B_{15}$  is almost full in this case which causes congestion at this time  $t$ . Choose alternate routes for each node towards BS once the congested node has been identified. First, the leaf node  $S_{11}$  with its 16 packets are transmitted to its neighbours either  $S_{10}$  or  $S_{12}$ , but the Bug2 algorithm chooses  $S_{10}$ . After diverting  $S_{11}$  packets to  $S_{10}$ , still there is no congestion possibility at node  $S_{10}$ . Further, the node  $S_{13}$  diverts its packets to node BS, through  $S_{11}$  followed by node  $S_{10}$ . However, this decision does not cause any congestion at  $S_{11}$  or  $S_{10}$ . Similarly, the data packets from node  $S_{16}$  are also routed to node  $S_3$ , and node 15 is free from congestion, and there are no further congested nodes. The updated routes which are generated through Bug2 while avoiding congestion are shown in Fig. 3(b). Using this process, alternate routes will be generated dynamically in every iteration, while improving

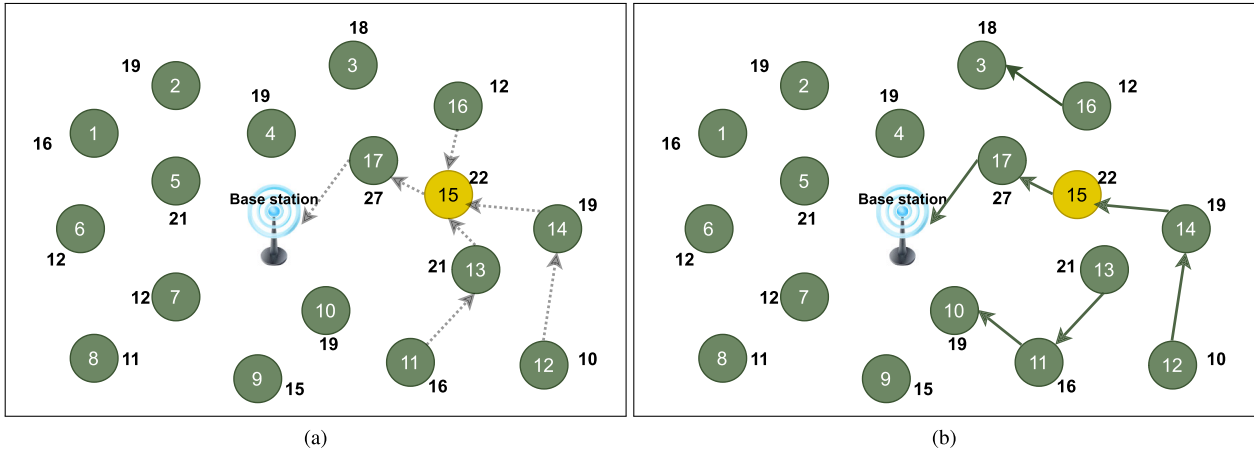


FIGURE 3. Illustrative example of Route construction (a) Congested node identified (b) Alternate path using Bug2 is determined.

throughput, decreasing delay, and reducing energy consumption, as well as avoiding congestion. As a result, the network can adapt to changing conditions in real-time, leading to improved performance for the overall system. This can be seen as a major advantage compared to traditional static routing methods.

V. RESULT AND COMPARISON

In this section, we present simulation setup and simulators used to evaluate numerical results. Further, we plot numerical evaluations, and mention the reasons for the improvements in proposed work. Finally, we provide a theoretical analysis on the proposed CDEIR algorithm.

A. SIMULATION SETUP

The proposed CDEIR and existing approaches are implemented and tested using a Python simulator (Python 3.11). There are 500-1000 SNs deployed within an area of 600 square meters, and these deployments are generated according Sah et al. [34]. All deployed SNs are equipped with a 194.4 KJ battery capacity, and all batteries are fully charged at the beginning of the deployment. For both the existing and proposed models, the simulation time is considered 120 hours. We summarize the remaining metrics used during simulation in table 3. Furthermore, these metrics were used to compare the performance of proposed work and existing approaches at various levels, including the lifespan of the WSN, energy-efficiency, buffer usage, throughput, latency, and connectivity efficiency. A time-driven network and an event-driven network are used to estimate these metrics. A WSN with event-driven acquisition acquires packets only when an event occurs in the network, while a WSN with time-driven acquisition acquires packets continuously regardless of external events occurring in the network. The properties of event-driven acquisition and time-driven acquisition are similar to Kafi et al. [35].

B. NUMERICAL EVALUATION

In this section, different metrics (lifetime, energy efficiency, latency, etc.) are evaluated with numerical values

TABLE 3. Simulation Parameters.

Parameter	Value
Simulator	Python 3.11
$n$	500 – 1000 SNs
$A$	600 sq.m
Node type	Zolertia Z1 mote
$E^{max}$	96 kB
$S_0$	(300,300)
$a$	0.061 J
$b$	0.019 J
$c$	0.021J
$r_t$	14 m
$r_c$	19 m
Protocol	MAC: TDMA Application: MQTT-SN
Data transfer rate	45 Kbps
Packet generation	4 pkt/sec
Packet size	128 bits
$E^{max}$	192.4 KJ
Channel bandwidth	$10^{+3} Hz$
Noise spectral power density	$10^{-9} W/Hz$
Length of a Frame	7s
Topology	Tree topology

to demonstrate the superiority of the proposed algorithm for CDEIR. This helps to objectively compare the proposed algorithm to existing algorithms in terms of how well it performs in specific tasks. It also helps to identify areas for improvement and assess trade-offs between different metrics.

1) NETWORK LIFETIME

One of the most important metrics used to measure the performance of a WSN is its lifetime ( $\mathcal{N}$ ).  $\mathcal{N}$  is calculated by taking into account the initial battery charge ( $\mathcal{E}_i$ ) and the power consumed by the nodes for a given period of time ( $\mathcal{T}$ ). The WSN’s lifetime is then determined by how long it takes for the battery charge to drop to zero, which is signified by the Eq. (5). Fig. 4(a) shows event-driven scenario and 4(b) shows time driven scenarios of  $\mathcal{N}$  for the given metrics shown in Table 3.

From Fig. 4(a), we notice the change in  $\mathcal{N}$  such as improvement of proposed CDEIR by  $\approx 1.223112-21.21121222\%$  compared with NFEER,  $\approx 1.9654-22.89618\%$  compared



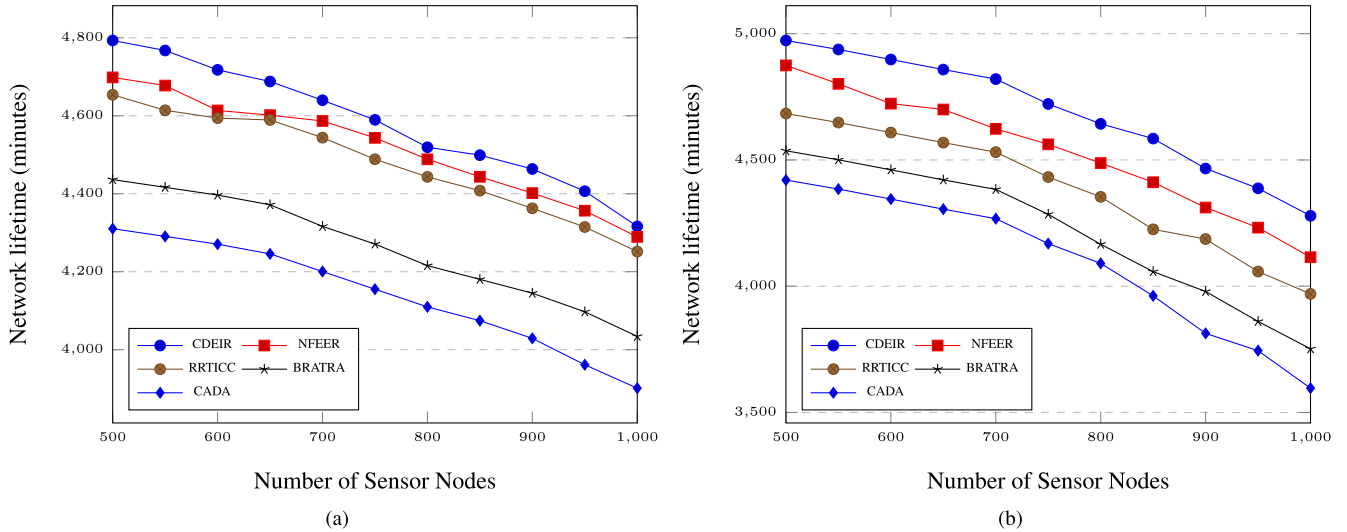


FIGURE 4. Network lifespan of s (a) Event-driven WSNs (b) Time-driven WSNs.

with RRTICC,  $\approx 7.57245\text{-}72.77457\%$  over BRATRA, and  $\approx 9.9954754\text{-}12.547156\%$  compared with CADA for event-driven WSNs. Similarly, in the scenario#2 (Fig. 4(b)), the proposed CDEIR performed better than the NFEER, RRTICC, BRATRA, and CADA in terms of  $\mathcal{N}$ . NFEER resulting maximum of 4698 minutes, RRTICC achieve maximum of 4683 minutes, BRATRA achieve maximum of 4536 minutes, and CADA 4420 minutes and overall  $\mathcal{N}$  of CDEIR sustain minimum of 5.62-7.21% longer compared to existing NFEER, RRTICC, BRATRA, and CADA. So, CDEIR in event-driven acquisition can sustain a longer lifetime since it only acquires packets when an event occurs, while time-driven acquisition requires more energy to acquire packets continuously. Moreover, event-driven acquisition has higher accuracy in  $\mathcal{N}$  since the SN maintains its energy during events, while time-driven acquisition continuously exhausts its energy.

### 2) AVERAGE ENERGY DRAIN

The AED helps to determine how much energy is being consumed by each node, as well as the overall performance of the network. By understanding the AED, network administrators can better manage the energy consumption of their nodes and optimize their WSNs. Furthermore, the AED can be used to identify the most energy-efficient nodes in the network, allowing network administrators to determine which nodes should be used more often. This helps to ensure that the network is running at its optimal efficiency, while also reducing the overall energy consumption. We measure AED using Eq. 16.

$$E_a = \frac{\sum_{i=1}^n E_i}{n} \tag{16}$$

In this paper, we analyze the AED of the CDEIR and existing works based on event-driven (as shown in Fig. 5(a))

and time-driven WSNs(as depicted in Fig. 5(b)). In both scenarios, we noticed improvements in the proposed CDEIR algorithm. In the first scenario, the improvements plotted in Fig. 5(a) shows  $\approx 8.2147887\text{-}19.77844667\%$  lower efficient compared with NFEER, RRTICC resulting in 9.87-20.47% of lower efficiency than CDEIR. Similarly, BRATRA results in 19.9982-29.34 percent, and CADA approaches 21.43-37.51 percent lower energy efficiency compared with CDEIR approach. On the other side, Fig. 5(b) illustrates performance variations in Time-driven WSNs applications between the CDEIR algorithm and others. The performance improvement of AED over NFEER is approximately  $1.67 - 9.8898\times$ , RRTICC and CDEIR is  $\approx 2.38 - 6.995\times$ ,  $4.712 - 8.1211\times$ , and  $7.323 - 12.79112\times$  in CADA. This is due to its ability to find optimal routes using a combination of data from the environment, energy efficiency, and delay cost. This also demonstrates that CDEIR achieves better energy efficiency in both scenarios. This provides a benefit to the user and WSNs in terms of energy savings, cost savings, and improved performance.

### 3) FAIRNESS INDEX OF ENERGY DRAIN

In order to identify the energy drain of a specific region of WSN and an inability to identify the bottleneck through the AED metric, the fairness index ( $\mathcal{F}$ ) is estimated. In WSNs, congestion causes a particular node or region to be congested, so the  $\mathcal{F}$  metric is useful. The range of  $\mathcal{F}$  exists between 0 and 100, where a higher value indicates a better result, and vice versa. We estimate  $\mathcal{F}$  according to Donta et al. [36] as shown in Eq. (17).

$$\mathcal{F} = \left(1 - \frac{\zeta \times 2}{\max\{E_a^t\} - \min\{E_a^t\} + \epsilon}\right) \times 100 \tag{17}$$

where the standard deviation of energy drain denoted using  $\zeta$ ,  $\max\{E_a^t\}$  indicates the maximum AED at time  $t$  and  $\min\{E_a^t\}$

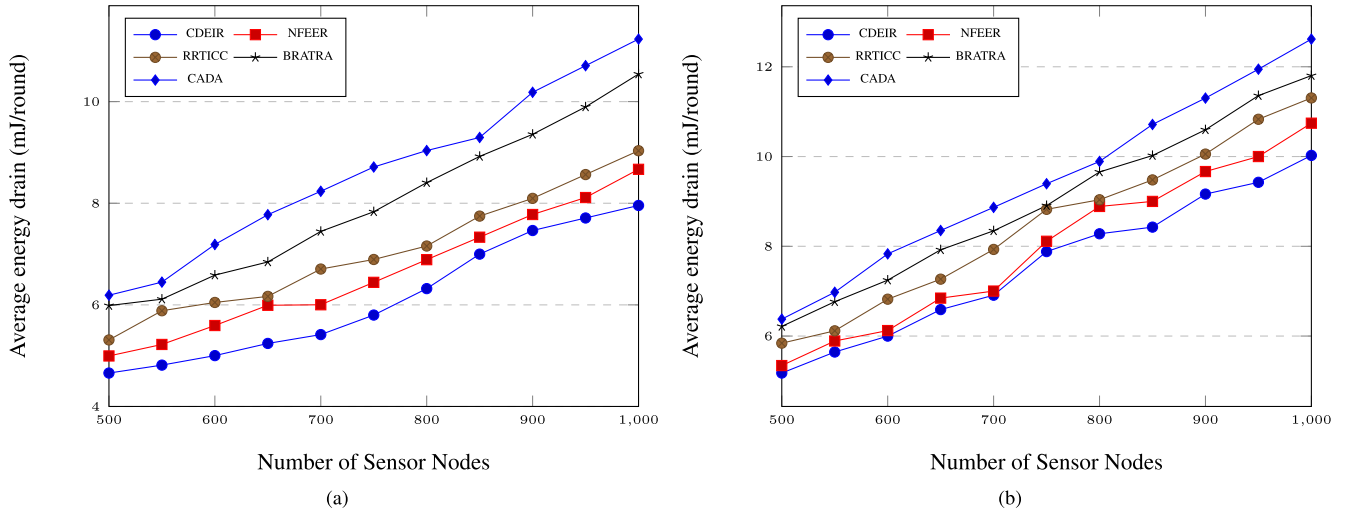


FIGURE 5. Average Energy Drain of (a) Event-driven WSNs (b) Time-driven WSNs.

is the minimum AED at a time  $t$ . In the event of similar max and min energies,  $\epsilon$  prevents the [divided by zero] error. In order to determine the  $\zeta$  value, we apply Eq. (18).

$$\zeta = \frac{1}{\sqrt{n}} \sqrt{\sum_{i=1}^n (E_i^c - E_a^t)^2} \quad (18)$$

A comparison is made between the  $\mathcal{F}$  of the proposed CDEIR and the  $\mathcal{F}$  of existing works using two scenarios. In these two scenarios, the number of  $n$  is varied between 500 and 1000, and plot using Fig. 6. From Fig. 6(a), we notice performance of the CDEIR algorithm is significantly better than that of the RRTICC algorithm, ranging from 1.21142 to 1.944574%. Both BRATRA and CADA have similar relationships, with cost or time near 2.711426-3.11028 $\times$  and 2.91142-7.1211452 $\times$ , respectively, lower than CDEIR. In Fig. 6(b), time-drive  $\mathcal{F}$  are plotted. Compared to RRTICC, BRATRA, and CADA, CDEIR has shown significant improvements in  $\mathcal{F}$  over the existing approaches that range from 1.2214 to 4.2214 $\times$ , 2.2214-5.1124 $\times$ , and 4.77584-6.774774 $\times$ . This demonstrates the effectiveness of CDEIR over existing approaches in terms of fairness index of energy. The use of Bug2 algorithm results in efficient alternate routing with high throughput and minimizes the cause of congestion in the data routes. Therefore, the improved results of CDEIR demonstrate its potential for improving the performance of responsive WSNs.

#### 4) BUFFER UTILIZATION

Efficient usage of buffers (BU) during data routing determines the efficiency of the routing process. The efficiency of the data collection process can be defined with the help of this metric, as well as the efficiency of resource use. BU ultimately determines how much data is collected, how much data is lost in the network, and what data collection method

is most effective. When BU is used efficiently, data can travel through the network more quickly and with less latency. This increases the speed of data collection and decreases the amount of data lost. Additionally, efficient usage of BU can help to identify the most efficient data collection methods, as certain methods can cause more data to be lost than others. Consequently, WSN congestion can be mitigated. We can use the Eq. (19) to estimate the BU.

$$BU = \sum_{i=1}^n \mathcal{B}_i \times (\mathcal{B}_{max} \times n)^{-1} \quad (19)$$

Using 500 to 1000 SNs as a range, we estimate the BU of the CDEIR algorithm for event-based and time-dependent WSNs. We notice that CDEIR outperforms existing applications, but it is also true that event-driven applications perform better. This is due to the fact that CDEIR is designed to process data from event-driven applications more efficiently, as it is able to quickly detect and respond to changes in the environment. Additionally, CDEIR reduces the need for frequent updates from time-dependent WSNs, which further contributes to its improved performance. From Fig. 7(a), existing RRTICC performs poorly i.e., approximately 1.112 - 2.28211% lower than the performance of CDEIR. Also, CDEIR outperforms BRATRA by 1.65589% and CADA by 3.665896%. NFEER also resulted low performance approximately 0.889-1.998%. BU performances are improved by at least 1.53% compared to existing and published approaches by CDEIR, as shown in Fig. 7(b) for time-driven scenario by approximately 1.6677%, 2.00121%-2.611427%, 2.99587%-4.33655662%, and 4.2211451%-6.88454% from NFEER, RRTICC, BRATRA, and CADA approaches, respectively. The proposed CDEIR algorithm reduces the probability of packet dropping and maintains quality of service. Furthermore, it also improves latency and reduces overall network cost.

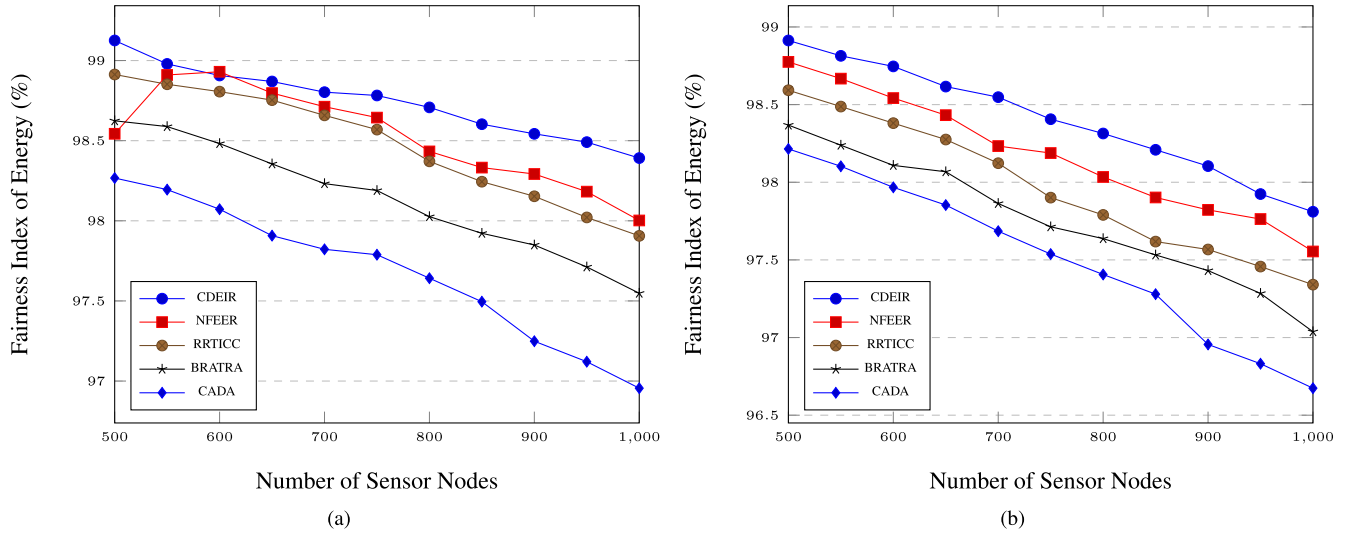


FIGURE 6. Fairness Index of Energy Drain of WSNs in (a) Event-driven (b) Time-driven.

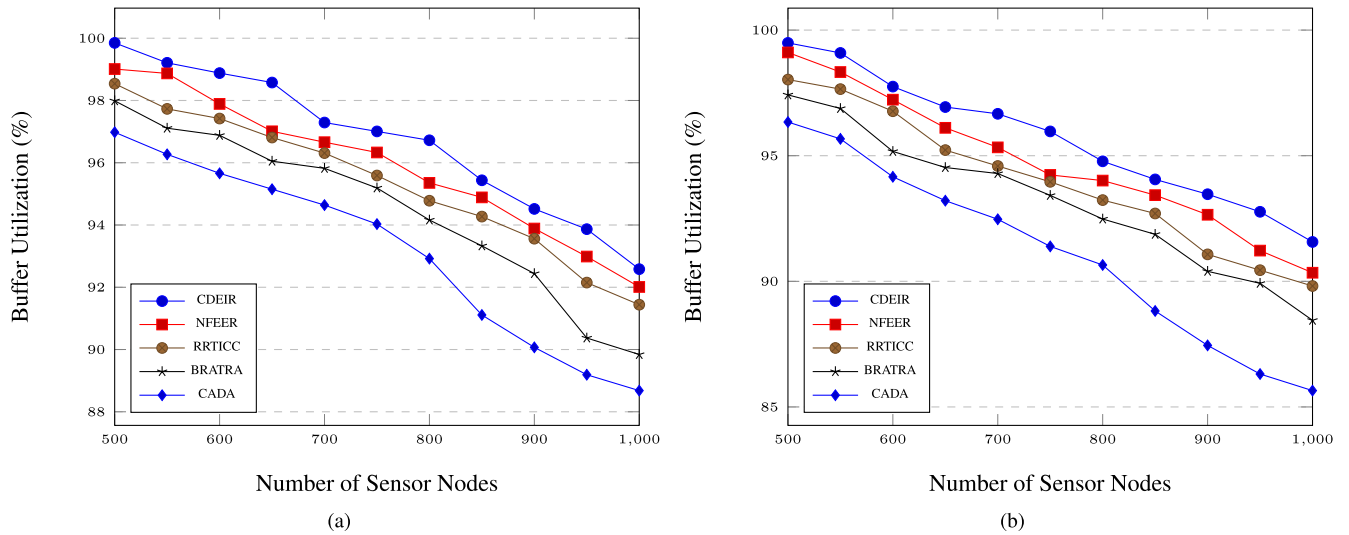


FIGURE 7. Buffer Utilization of (a) Event-driven WSNs (b) Time-driven WSNs.

5) NETWORK THROUGHPUT

Network throughput ( $\Gamma$ ) is the amount of data that can be transmitted between two points in a given amount of time. By dividing the number of packets ( $|P_i|$ ) acquired during simulation time ( $T$ ) by the number of packets acquired, we can determine the rate of data transmission. This is how we can determine the throughput of the network. This is represented mathematically by Eq. (20).

$$\Gamma = \frac{1}{T} \sum_{i=1}^n |P_i| \tag{20}$$

The  $\Gamma$  value is then estimated through variation of the  $n$  value between 100-1000 in Fig. 8(a). In comparison to NFEER, RRTICC, BRATRA, and CADA algorithms, CDEIR has improved  $\Gamma$  by 0.666667%, 0.799548%, 3.447554%,

and 5.01145221%, respectively. In order to comply with the CDEIR rules in time-driven algorithms, NFEER, RRTICC, BRATRA, and CADA have  $\Gamma$  greater than 0.4114114%, 0.4411254%, 0.5569985%, and 0.88999874%, respectively as shown in Fig. 8(b). On the basis of these numerical analyses, CDEIR and existing strategies achieve higher throughputs. It has been observed that the improved  $\Gamma$  of CDEIR is a result of the Bug approach used to determine the near-optimal congestion, delay, and energy aware route in the WSNs. This means that the Bug approach is able to reduce the overall energy consumption of the network by finding routes that have the shortest delays and lowest congestion levels. This results in a decrease in the overall energy consumption of the network, improving the overall throughput of the network.

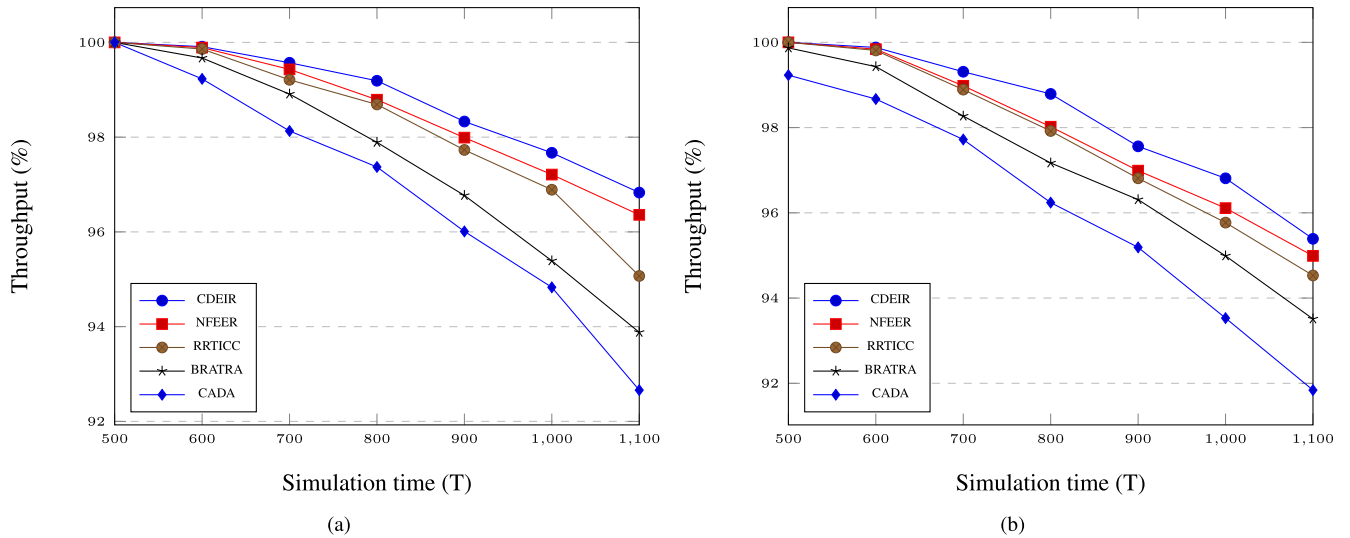


FIGURE 8. Throughput of (a) Event-driven WSNs (b) Time-driven WSNs.

### 6) END-TO-END DELAY

End-to-End (E2E) delays are applied in this study ( $\Delta$ ) to data packets returned by the BS ( $S_0$ ). Performance is improved when algorithms have a lower E2E. By applying E2E delays, we can reduce the amount of time it takes for data packets to be returned from SNs to the BS. This can help improve performance by reducing the amount of time spent waiting at SNs. Additionally, by using the formula in Eq. (21) we can accurately calculate E2E delays and ensure that performance is increased.

$$\Delta = \sum_{i=1}^n \sum_{j=1}^{P_i} (\varphi_{ij}^w + \varphi_{ij}^m) \times (n \times P_i)^{-1} \quad (21)$$

where  $\varphi_{ij}^m$  is the time awaiting of  $j^{th}$  packet of node  $i$  at MS and,  $\varphi_{ij}^w$  is the waiting time of  $j^{th}$  packet of node  $i$  at SN, before it is routed to the BS.  $P_i$  indicates the total packets generated by node  $i$ . In this paper, a number of existing approaches are evaluated, including the  $\Delta$  based CDEIR, as well as time-driven WSNs.

Based on the plots shown in Fig. 9(a), we understand how the E2E delay varies among the different algorithms when they are applied to event-driven WSN applications, and we also see that CDEIR has a lower delay than the other algorithms. Performance improvements of  $1.8876\text{-}5.47557\times$ ,  $2.3651\text{-}6.2368\times$ ,  $1.88475\text{-}7.69856\times$ , and  $5.055214\text{-}8.9954552\times$  are achieved by the proposed CDEIR over the existing NFEER, RRTICC, BRATRA, and CADA. In time-driven applications, we estimate the E2E delay of WSNs with 100 SNs in Fig. 9(b). Although the CDEIR algorithm performs better than the existing NFEER, RRTICC, BRATRA, and CADA strategies, which are respectively  $1.1128\text{-}3.44323\times$ ,  $1.451\text{-}3.99548\times$ ,

$1.99856\text{-}4.84457\times$ , and  $5.0120014\text{-}6.847754\times$ . The CDEIR algorithm provides a more efficient solution of reducing the end-to-end delay since it takes into account the size of the data packet and the load of the nodes on the network. This allows for the optimal routing of the packet and the reduction of transmission time. The CDEIR algorithm works by considering the size of the data packet, the load on the nodes and the available bandwidth. This allows it to select the most efficient route for the packet, reducing transmission time and increasing the chances of the packet being delivered on time.

### 7) EFFICIENCY OF CONNECTIVITY

In this study, the Efficiency of Connectivity (EoC) is estimated by counting the number of SNs that become isolated when their energy has evaporated completely. A detailed analysis of the data transmission efficiency between SNs is also provided. Eq. (22) shows the estimated EoC for the proposed CDEIR, which was obtained from Donta et al. [10].

$$EoC = \frac{n - \xi'}{n} \times 100 \quad (22)$$

where  $\xi'$  during time  $T$  is estimated using Eq. (23).

$$\xi' = \sum_{i=1}^n \mathcal{C}_i \quad (23)$$

where  $\mathcal{C}_i$  can be treated as Eq.(24).

$$\mathcal{C}_i = \begin{cases} 1 & \xi_i \equiv 0 \\ 0 & \text{Otherwise} \end{cases} \quad (24)$$

By applying CDEIR routing to WSNs and the effect on complete energy drain, we estimate the EoC of these SNs in



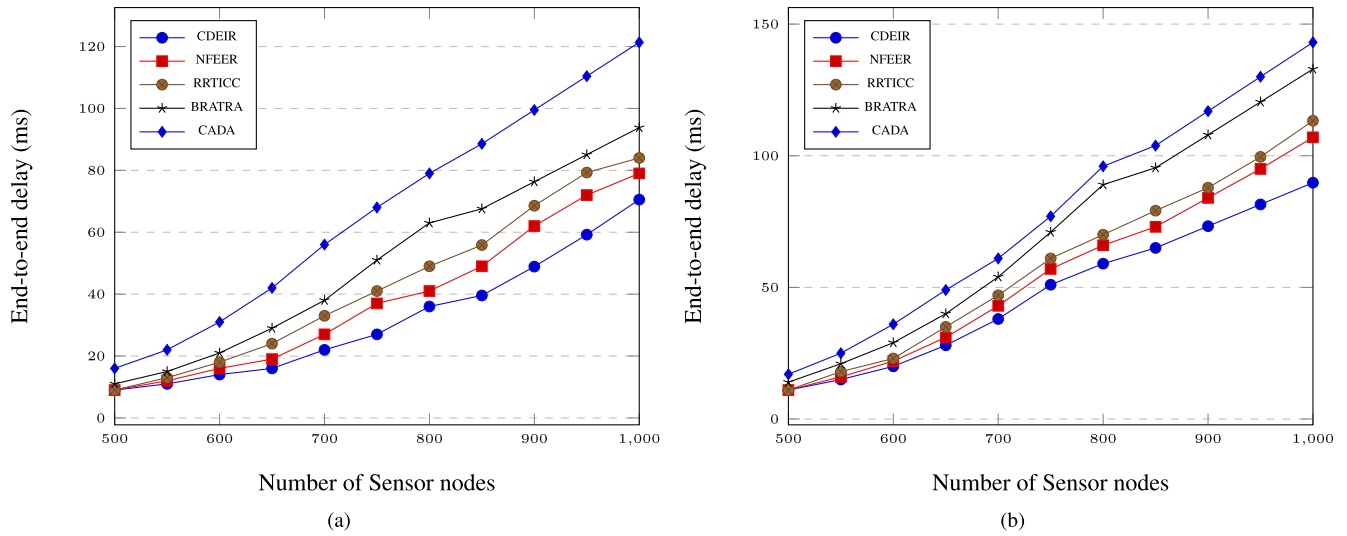


FIGURE 9. End-to-end delay of (a) Event-driven WSNs (b) Time-driven WSNs.

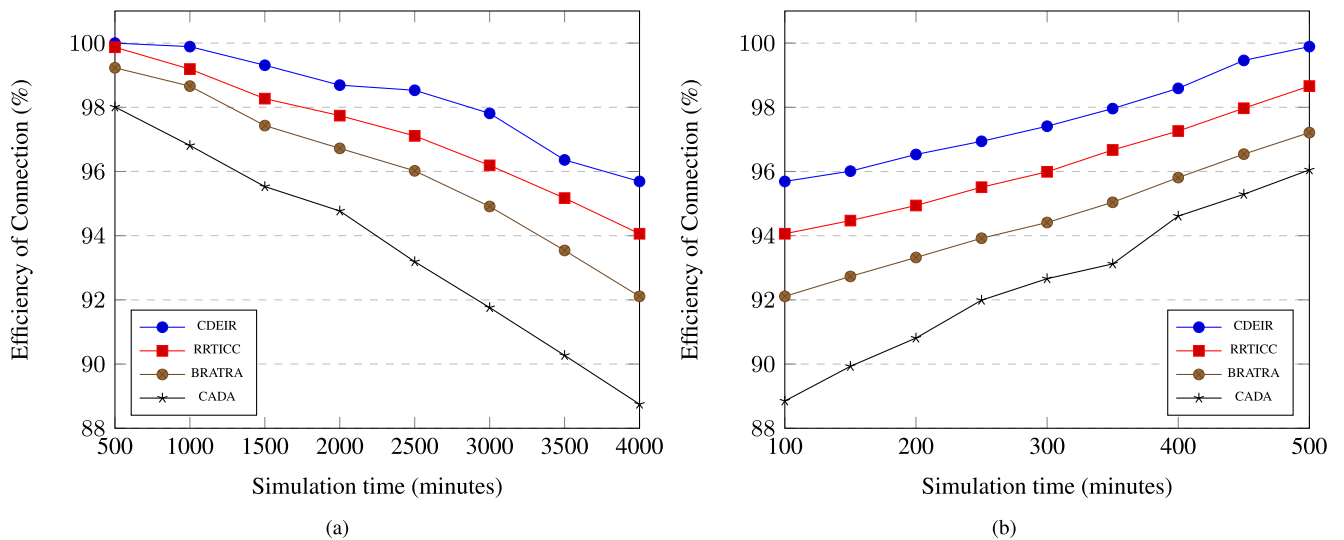


FIGURE 10. Efficiency of Connectivity Vs. (a) Simulation time (b) # Sensor nodes.

the WSNs. Two scenarios were considered: the first, increasing simulation time and plotting in Fig. 10(a) and second changing the number of SNs as shown in Fig. 10(b). The results showed that CDEIR routing was able to drastically reduce the energy consumption of WSNs, as well as lengthen the lifetime of the nodes. On the other hand, a value of 100% indicates that all SNs are fully connected (no isolated nodes in WSNs). A value of less than 100% indicates that there might be isolated nodes in the network. From Fig. 10(a), we enhance the simulation and observe the EoC of the proposed and existing approaches. According to the proposed CDEIR, SNs are well connected and there is no isolated node until the 564th time unit. Increasing the simulation time changes the alternate data packet transmission,

resulting in a decrease in EOC. As we vary the number of SN (100-600), we also observe an increase in EoC in WSNs, as shown in Fig. 10(b). The proposed CDEIR always results in higher EoC than the existing algorithms. This demonstrates the effectiveness of the proposed CDEIR in improving the energy efficiency of WSNs. This can be attributed to the fact that the proposed CDEIR algorithm balances the energy consumption of the nodes by evenly distributing the energy among them. This ensures that the nodes have sufficient energy to communicate with each other, leading to better connectivity and fewer isolated nodes. Additionally, the proposed algorithm also reduces the overall energy consumption of the network, leading to higher energy efficiency. Furthermore, CDEIT also provides better connectivity

and fewer isolated nodes compared to other existing algorithms.

### C. THEORETICAL EVALUATIONS

Theoretical analysis of the proposed CDEIR approach follows according to Banoth et al. [37], where they used to estimate the connection and coverage efficiency, which is also suitable for congestion avoidance.

*Theorem 1: The worst and average case time complexity of the CDE Intelligent Routing is  $O(n^3)$  and  $O(n \log n)$ , respectively.*

*Proof 1:* In order to determine the time complexity of the proposed CDEIR, we follow these steps. It is a combination of DST construction, congestion node identification and intelligent route construction. The DST of the proposed CDEIR algorithm will take approximately  $O(n \log n)$ , where  $n$  is the number of deployed SNs. Determining congestion among  $n$  nodes along with computing their  $\mathcal{W}$  and  $C(\mathcal{S}_i)$  needed  $O(n)$  time complexity. Finally, Algorithm 1 is  $O(n\delta^2)$ , where  $\delta$  is the maximum depth of the responsive WSN  $G$  from the  $\mathcal{S}_0$ . There is very often deployment of WSNs with  $\delta = n$ , in such case Algorithm 1 require  $O(n^3)$ , and overall Asymptotic time complexity of CDEIR is  $O(n^3)$  in the worst case. According to the average case, it takes  $O(n \log n)$ .

*Theorem 2: The responsive WSN (i.e.,  $G$ ) is well connected and any  $\mathcal{S}_i, \forall 1 \leq i \leq n$  can able to send its data either directly or through relay to  $\mathcal{S}_0$*

*Proof 2:* There is a strong connection between two nodes  $\mathcal{S}_i$  and  $\mathcal{S}_j$  if at a time  $t$  there are connections between them and if there is a path from  $\mathcal{S}_0$  to every node in  $G$ . A pair of undirected edges exists between any two arbitrary nodes in  $\mathcal{S} = \mathcal{S}_i, \forall 1 \leq i \leq n$ , either between  $(\mathcal{S}_i, \mathcal{S}_{i+1})$  ( $\forall (i+1) < n$ ) or between  $(\mathcal{S}_{i+1}, \mathcal{S}_i)$  and tested for all nodes from  $1 \leq i \leq n$ . A depth-first search tree is also used to determine if the  $\mathcal{S} = \mathcal{S}_i, \forall 0 \leq i \leq n$  is connected to the  $\mathcal{S}_0$ . The CDEIR algorithm proves that  $\mathcal{S}$  and  $\mathcal{S}_0$  are well connected if DFS( $G, \mathcal{S}_0$ ) and local connectivity of the nodes are satisfied.

### VI. CONCLUSION

For responsive WSNs, we present an intelligent routing approach considering congestion, delay and energy constraints and we named this as CDEIR algorithm. We construct a directed spanning tree within which based on the SNs relay and its own data, we determine the possible congested nodes in WSNs. Once the congested nodes are identified, we use a Bug algorithm which constructs an intelligent route while avoiding the congested nodes with a minimal hop count. The proposed CDEIR algorithm performs the above-mentioned operations with limited computational complexity which is better when compared with existing algorithms. Additionally, we confirm that CDEIR results in the shortest and most efficient routes, which further minimizes delay and energy usage. Also, it outperforms other approaches in terms of

several performance metrics such as lifespan, energy efficiency, throughput and delay. By means of simulations and theoretical proofs, we confirm that our approach performs better than existing approaches. Our work can be further extended by considering dynamic network conditions and testing real-world applications.

### CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

### REFERENCES

- [1] J. Kurose, E. Lyons, D. McLaughlin, D. Pepyne, B. Philips, D. Westbrook, and M. Zink, "An end-user-responsive sensor network architecture for hazardous weather detection, prediction and response," in *Proc. Asian Internet Eng. Conf.* Cham, Switzerland: Springer, 2006, pp. 1–15.
- [2] J. Song, H. Liu, L. Nie, Z. Ning, M. S. Obaidat, and B. Sadoun, "Network traffic prediction for intelligent transportation systems: A reinforcement learning approach," in *Proc. IEEE Global Commun. Conf.*, Dec. 2022, pp. 245–250.
- [3] A. Paszkiewicz and J. Węgrzyn, "Responsiveness of the sensor network to alarm events based on the Potts model," *Sensors*, vol. 20, no. 23, p. 6979, Dec. 2020.
- [4] D. P. Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Inf. Fusion*, vol. 49, pp. 1–25, Sep. 2019.
- [5] W. Osamy, A. M. Khedr, A. Salim, A. I. A. Ali, and A. A. El-Sawy, "A review on recent studies utilizing artificial intelligence methods for solving routing challenges in wireless sensor networks," *PeerJ Comput. Sci.*, vol. 8, p. e1089, Oct. 2022.
- [6] T. M. Behera, U. C. Samal, S. K. Mohapatra, M. S. Khan, B. Appasani, N. Bizon, and P. Thounthong, "Energy-efficient routing protocols for wireless sensor networks: Architectures, strategies, and performance," *Electronics*, vol. 11, no. 15, p. 2282, Jul. 2022.
- [7] V. Rajagopal, B. Velusamy, and S. Rathinasamy, "Double Q-learning-based adaptive trajectory selection for energy-efficient data collection in wireless sensor networks," *Int. J. Commun. Syst.*, vol. 36, no. 7, p. e5452, May 2023.
- [8] X. Wang, J. Li, Z. Ning, Q. Song, L. Guo, S. Guo, and M. S. Obaidat, "Wireless powered mobile edge computing networks: A survey," *ACM Comput. Surv.*, vol. 2023, pp. 1–12, Jan. 2023.
- [9] P. K. Donta, T. Amgoth, and C. S. R. Annavarapu, "Congestion-aware data acquisition with Q-learning for wireless sensor networks," in *Proc. IEEE Int. IoT, Electron. Mechatronics Conf. (IEMTRONICS)*, Sep. 2020, pp. 1–6.
- [10] P. K. Donta, T. Amgoth, and C. S. R. Annavarapu, "Delay-aware data fusion in duty-cycled wireless sensor networks: A Q-learning approach," *Sustain. Comput., Informat. Syst.*, vol. 33, Jan. 2022, Art. no. 100642.
- [11] D. Prabhu, R. Alageswaran, and S. M. J. Amali, "Multiple agent based reinforcement learning for energy efficient routing in WSN," *Wireless Netw.*, vol. 2023, pp. 1–11, Jan. 2023.
- [12] C. Yu, M. Ku, L. Wang, F. Huang, and W. Jia, "BRATRA: Balanced routing algorithm with transmission range adjustment for energy efficiency and utilization balance in WSNs," *IEEE Internet Things J.*, vol. 10, no. 2, pp. 1096–1111, Jan. 2023.
- [13] P. S. Prakash, D. Kavitha, and P. C. Reddy, "A rapidly-exploring random tree-based intelligent congestion control through an alternate routing for WSNs," *Int. J. Commun. Netw. Distrib. Syst.*, vol. 29, no. 1, pp. 71–94, 2023.
- [14] P. S. Prakash, D. Kavitha, and P. C. Reddy, "Delay-aware relay node selection for cluster-based wireless sensor networks," *Meas., Sensors*, vol. 24, Dec. 2022, Art. no. 100403.
- [15] V. Agarwal, S. Tapaswi, and P. Chanak, "Intelligent fault-tolerance data routing scheme for IoT-enabled WSNs," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16332–16342, Sep. 2022.
- [16] V. Agarwal, S. Tapaswi, and P. Chanak, "Energy-efficient mobile sink-based intelligent data routing scheme for wireless sensor networks," *IEEE Sensors J.*, vol. 22, no. 10, pp. 9881–9891, May 2022.

- [17] R. Huang, W. Guan, G. Zhai, J. He, and X. Chu, "Deep graph reinforcement learning based intelligent traffic routing control for software-defined wireless sensor networks," *Appl. Sci.*, vol. 12, no. 4, p. 1951, Feb. 2022.
- [18] S. Yang, C. Tan, D. Ø. Madsen, H. Xiang, Y. Li, I. Khan, and B. J. Choi, "Comparative analysis of routing schemes based on machine learning," *Mobile Inf. Syst.*, vol. 2022, pp. 1–18, Jun. 2022.
- [19] E. Dixit and V. Jindal, "IEESEP: An intelligent energy efficient stable election routing protocol in air pollution monitoring WSNs," *Neural Comput. Appl.*, vol. 34, no. 13, pp. 10989–11013, Jul. 2022.
- [20] R. Dogra, S. Rani, H. Babbar, S. Verma, K. Verma, and J. J. P. C. Rodrigues, "DCGCR: Dynamic clustering green communication routing for intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16197–16205, Sep. 2022.
- [21] M. E. Ibrahim and A. E. Ahmed, "Energy-aware intelligent hybrid routing protocol for wireless sensor networks," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 3, p. e6601, 2022.
- [22] M. K. Roberts and P. Ramasamy, "Optimized hybrid routing protocol for energy-aware cluster head selection in wireless sensor networks," *Digit. Signal Process.*, vol. 130, Oct. 2022, Art. no. 103737.
- [23] E. Heidari, A. Movaghar, H. Motameni, and B. Barzegar, "A novel approach for clustering and routing in WSN using genetic algorithm and equilibrium optimizer," *Int. J. Commun. Syst.*, vol. 35, no. 10, p. e5148, Jul. 2022.
- [24] U. E. Zachariah and L. Kuppusamy, "A hybrid approach to energy efficient clustering and routing in wireless sensor networks," *Evol. Intell.*, vol. 2022, pp. 1–13, Jan. 2022.
- [25] A. C. Sumathi, A. Javadpour, P. Pinto, A. K. Sangaiah, W. Zhang, and S. M. Khaniabadi, "NEWTR: A multipath routing for next hop destination in Internet of Things with artificial recurrent neural network (RNN)," *Int. J. Mach. Learn. Cybern.*, vol. 13, no. 10, pp. 2869–2889, Oct. 2022.
- [26] C. N. Vanitha, S. Malathy, R. K. Dhanaraj, and A. Nayyar, "Optimized Pollard route deviation and route selection using Bayesian machine learning techniques in wireless sensor networks," *Comput. Netw.*, vol. 216, Oct. 2022, Art. no. 109228.
- [27] I. Chakraborty, P. Das, and B. Pradhan, "An intelligent routing for Internet of Things mesh networks," *Trans. Emerg. Telecommun. Technol.*, vol. 2022, p. e4628, Aug. 2022.
- [28] D. Keum and Y.-B. Ko, "Trust-based intelligent routing protocol with Q-learning for mission-critical wireless sensor networks," *Sensors*, vol. 22, no. 11, p. 3975, May 2022.
- [29] A. Saleh, P. Joshi, R. S. Rathore, and S. S. Sengar, "Trust-aware routing mechanism through an edge node for IoT-enabled sensor networks," *Sensors*, vol. 22, no. 20, p. 7820, Oct. 2022.
- [30] P. Tewari and S. Tripathi, "An energy efficient routing scheme in Internet of Things enabled WSN: Neuro-fuzzy approach," *J. Supercomput.*, vol. 2023, pp. 1–25, Jan. 2023.
- [31] P. K. Donta, S. N. Srirama, T. Amgoth, and C. R. Annavarapu, "iCoCoA: Intelligent congestion control algorithm for CoAP using deep reinforcement learning," *J. Ambient Intell. Humanized Comput.*, vol. 2023, pp. 1–16, Jan. 2023.
- [32] T. Amgoth and C. S. R. Annavarapu, "ACO-based mobile sink path determination for wireless sensor networks under non-uniform data constraints," *Appl. Soft Comput.*, vol. 69, pp. 528–540, Aug. 2018.
- [33] K. N. McGuire, G. C. H. E. de Croon, and K. Tuyls, "A comparative study of bug algorithms for robot navigation," *Robot. Auto. Syst.*, vol. 121, Nov. 2019, Art. no. 103261.
- [34] D. K. Sah, K. Cengiz, P. K. Donta, V. N. Inukollu, and T. Amgoth, "EDGF: Empirical dataset generation framework for wireless sensor networks," *Comput. Commun.*, vol. 180, pp. 48–56, Dec. 2021.
- [35] M. A. Kafi, D. Djenouri, J. Ben-Othman, and N. Badache, "Congestion control protocols in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1369–1390, 3rd Quart., 2014.
- [36] P. K. Donta, B. S. P. Rao, T. Amgoth, C. S. R. Annavarapu, and S. Swain, "Data collection and path determination strategies for mobile sink in 3D WSNs," *IEEE Sensors J.*, vol. 20, no. 4, pp. 2224–2233, Feb. 2020.
- [37] S. P. R. Banoth, P. K. Donta, and T. Amgoth, "Target-aware distributed coverage and connectivity algorithm for wireless sensor networks," *Wireless Netw.*, vol. 2023, pp. 1–16, Jan. 2023.



**P. SUMAN PRAKASH** received the B.Tech. degree from Jawaharlal Nehru Technological University Hyderabad, India, the M.Tech. degree from Karunya University, and the Ph.D. degree from Jawaharlal Nehru Technological University Anantapur, India. He is currently an Associate Professor with the Department of Artificial Intelligence, G. Pullaiah College of Engineering and Technology (Autonomous), Kurnool, Andhra Pradesh, India. He serves different academic positions, such as an Examination Section In-Charge, and the Head of the Department for Internet of Things, G. Pullaiah College of Engineering and Technology. His research interests include wireless sensor networks, artificial intelligence, machine learning, the Internet of Things, and distributed systems. He is on the editorial boards of several journals.



He is on the editorial boards of several journals.

**K. NAGARAJU** received the Ph.D. degree from the National Institute of Technology Calicut. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Indian Institute of Information Technology, Design and Manufacturing, Kurnool. His research interests include wireless sensor networks, artificial intelligence, machine learning, deep learning, natural language processing, theory of computer science, and real-world applications in these areas.



He is on the editorial boards of several journals.

**A. VISHNUVARDHAN REDDY** received the B.Tech. degree in computer science and engineering from Sri Venkateswara University, Tirupati, Andhra Pradesh, India, in 2006, and the M.E. degree in software engineering from Jadavpur University, Kolkata, West Bengal, India. He is currently an Associate Professor with the Department of Computer Science and Engineering, G. Pulla Reddy Engineering College (Autonomous), Kurnool, Andhra Pradesh. He is having 13 years of teaching experience. His research interests include mobile ad-hoc networks, cross-layer design, and wireless mesh networks.



She is on the editorial boards of several journals.

**SHAIK FATHIMABI** received the M.Tech. degree from the University College of Engineering, Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India, and the Ph.D. degree from the Department of Computer Science and Engineering, National Institute of Technology, Warangal, India. She is currently a Senior Assistant Professor with the Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, Andhra Pradesh. She has published a number of



**EBENEZER JANGAM** received the B.Tech. degree in computer science and engineering from Jawaharlal Nehru Technological University Hyderabad, in 2005, the M.Tech. degree in computer science and engineering from the National Institute of Technology Karnataka, Surathkal, in 2009, and the Ph.D. degree in computer science and engineering from the Indian Institute of Technology, Dhanbad, in 2021. He was an Assistant Teacher with the National Institute of Technology Warangal and an Assistant Professor with the Vignan Foundation for Science, Technology and Research, Guntur. He has more than 15 years of experience in teaching and research. He is currently a Senior Assistant Professor with the Department of Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada. He has published 20 papers in refereed journals and international conferences. He has received the ISEA Fellowship for three years.



**SURBHI BHATIA KHAN** (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering (machine learning and social media analytics). She also received a Project Management Professional Certification from reputed Project Management Institute, USA. She is currently with the Department of Data Science, School of Science, Engineering and Environment, University of Salford, Manchester, U.K. She has more than 11 years of academic and teaching experience in different universities. She was a recipient of the Research Excellence Award from King Faisal University, Saudi Arabia, in 2021. Her research interests include information systems, machine learning, sentiment analysis, and data science.



**AREEJ ABBAS MALIBARI** (Member, IEEE) received the Ph.D. degree in computer science-application artificial intelligence in supply chain management from the University of Essex, U.K., in 2010. From 2011 to 2013, she was a Supervisor with the Department of Information Systems, King Abdulaziz University. At the same time, she was an Honorary Lecturer with the School of Computer Science and Electronic Engineering, University of Essex, U.K. She was a Coordinator of academic accreditation (ABET) implementation with the College of Computing and IT, King Abdulaziz University, in 2012 and 2018. From 2017 to 2019, she was the Vice Director of the Computer Skills Unit. She was also a Supervisor of Female Campus with Saudi Electronic University, Jeddah. Since 2019, she has been the founding Dean of the College of Engineering, Princess Nourah bint Abdulrahman University (PNU), Riyadh. She leads the hydrogen production related projects and the autonomous mobility projects PNU. She has more than 40 publications in computer science and engineering. Her research interests include artificial intelligence applications in energy efficiency, supply chain management, e-commerce, e-business, medical application image processing, and education. She is a member of ACM. During her service, she received several professional training certifications, some of which in the fields: leadership by INSEAD (1) and LMI (4), several in supply chain management by ISCEA (3), academic accreditation by ABET (3), and many other in the fields of computer science and engineering. Through her carrier, she serves the community in many aspects weather its related to her profession or not.

• • •