J-COSINE (Journal of Computer Science and Informatics Engineering)
Vol. 10, No. 1, June 2026 E-ISSN:2541-0806
Accredited Sinta-4 by RISTEKDIKTI Decree No. 79/E/KPT/2023 P-ISSN:2540-8895

Implementation of Static Routing with Path Redundancy on LoRa SX1276 and ESP32 Based on Graph Theory

L. Ahmad Syamsul Irfan Akbar*, Misbahuddin, Muhamad Syamsu Iqbal, A. Sjamsjiar Rachman, Giri Wahyu Wiriasto, DjulFikri Budiman, Made Sutha Yadnya
Electrical Engineering Department University of Mataram
Jl. Majapahit 62, Mataram, Lombok, NTB-INDONESIA
Email: [irfan, misbahuddin, msiqbal, asrachman, giriwahyuwiriasto, djulfikry, msyadnya]@unram.ac.id

Corresponding Author

***Abstract*—The rapid development of the Internet of Things (IoT) has increased the demand for energy-efficient wireless communication systems and others under various environmental conditions. LoRa (Long Range) technology has attracted considerable attention due to its ability to support long-distance transmission with low power consumption. However, most existing LoRa network implementations rely on single-hop or non-redundant static routing to the gateway, making them vulnerable to link degradation or node failure. This study enhances the static routing mechanism with path redundancy implemented on an ESP32 microcontroller and an SX1276 LoRa transceiver module to improve communication reliability in multi-hop LoRa networks. The network topology is modeled using graph theory, where each node is represented as a vertex and communication paths are represented as weighted edges based on the Received Signal Strength Indicator (RSSI). The primary and backup routes are derived from the two shortest simple paths obtained from the weighted graph. Experimental results demonstrate that the proposed approach achieves an average Packet Delivery Ratio (PDR) of 91.4% while maintaining communication continuity during primary path failures. The proposed approach demonstrates the feasibility of a reliable multi-hop LoRa network using static routing with graph-based path redundancy.**

Key Words: LoRa SX1276, ESP32, Static Routing, Path Redundancy, Graph Theory, Internet of Things.

I. INTRODUCTION

The advancement of the Internet of Things (IoT)[1] has created a high demand for wireless communication systems that are both energy efficient and capable of operating over long distances[2]. One of the prominent technologies addressing this need is LoRa[3][4], which belongs to the LPWAN (Low Power Wide Area Network) category[5][6]. LoRa enables data transmission across several kilometers with minimal power consumption, making it suitable for applications such as environmental monitoring[7][8][9], agriculture[10][11][12], smart cities[13][14][15], and industrial automation[16]. Despite its advantages, most existing LoRa network implementations rely on static routing or single hop communication to a gateway[17]. This static configuration, while simple and energy efficient, lacks adaptability and fails to provide redundancy when communication links or intermediate nodes become unavailable. As a result, the network is highly vulnerable to link failures, signal attenuation, or environmental interference, resulting in packet loss and communication delays[18].

Among various radio features available in LoRa networks, the Received Signal Strength Indicator (RSSI) has attracted significant attention because it can be obtained directly from LoRa transceivers with low deployment cost and without requiring additional hardware. Consequently, RSSI has been widely used for link-quality assessment and communication performance evaluation in LoRa-based systems[19][20][21]. Furthermore, several LoRa routing studies have employed RSSI measurements to estimate link quality and support routing decisions in multi-hop communication environments[22][23]. However, RSSI measurements may be affected by environmental conditions such as temperature, humidity, atmospheric pressure, and rainfall, which can introduce temporal variations in link quality and communication performance[24].

Most previous research has focused on the use of dynamic and adaptive routing mechanisms to improve the performance of multi-hop LoRa networks. One commonly used approach is RSSI-based on-demand routing, such as AODV variants, which dynamically establish communication paths as data is being sent. This approach has been shown to improve packet delivery success, but is accompanied by increased control message overhead due to route discovery and maintenance. However, some of these studies are still limited to simulation-based evaluations, so their performance does not fully represent real-world implementation conditions[25][26][27]. Other research attempts to reduce routing table processing overhead by modifying routing metrics or implementing clustering mechanisms in the data forwarding process. While effective in environments with rapidly changing topology, these approaches still rely on relatively frequent exchanges of control messages, thus limiting scalability in LoRa networks with low bandwidth and duty cycle constraints[28][29][30].

Unlike these approaches, this study targets semi-static LoRa deployments where network topology changes infrequently. Instead of continuously discovering and main-

taining routes during operation, the proposed method computes routing paths offline using graph theory and stores both primary and backup routes in each node before deployment. This design significantly reduces routing overhead because no route discovery packets, routing updates, or route maintenance procedures are required during normal network operation. The inclusion of precomputed backup paths further enables communication continuity when the primary route becomes unavailable, providing reliability improvements while preserving the simplicity of static routing.

To overcome the overhead problem and link failure, this paper proposes the implementation of static routing with path redundancy in LoRa-based networks using an ESP32 microcontroller[31] and a Semtech SX1276 LoRa transceiver module[32]. The proposed routing framework employs graph theory[33] to model the network topology and generate routing tables offline based on RSSI measurements. Each node stores a precomputed primary path and a backup path, enabling automatic failover without invoking dynamic routing algorithms during runtime. Therefore, the main contribution of this work is the integration of graph-based offline route computation and lightweight static routing with path redundancy, providing a low-overhead communication solution for semi-static LoRa multi-hop networks implemented and validated on real ESP32-SX1276 hardware. Unlike existing graph-based LoRa routing approaches that typically rely on dynamic route discovery, adaptive route maintenance, or simulation-based evaluations, the proposed method eliminates runtime routing overhead through fully precomputed routing tables while maintaining communication reliability through backup-path failover in a real-world deployment.

II. LITERATURE REVIEW

Table I summarizes several representative LoRa routing studies and highlights the differences between the proposed approach and previous work. Unlike previous LoRa routing approaches that rely on dynamic routing updates, cluster formation mechanisms, or hybrid routing strategies during network operation, the proposed method computes both primary and backup routes offline using graph theory and RSSI measurements. The resulting routing information is converted into an embedded static route array and stored in each ESP32 node prior to deployment. During runtime, packet forwarding relies solely on HELLO-based failover between precomputed routes, eliminating route discovery and route maintenance overhead while preserving communication resilience. These characteristics distinguish the proposed approach from existing LoRa mesh routing solutions and represent the main state-of-the-art contribution of this work for semi-static LoRa multi-hop deployments.

TABLE I. COMPARISON WITH PREVIOUS LORA ROUTING STUDIES

Reference	Routing Type	Route Discovery	Main Limitation
Pham et al.[25]	AODV-based Mesh Routing	Dynamic	High end-to-end delay and packet loss affected by network size and payload length
Chong et al.[26]	AODV-inspired Multi-Route Routing	Dynamic	Repeated route re-establishment introduces additional routing overhead and latency in LoRa networks
Thaenthong et al.[27]	R-AODV-based Mesh Routing	Dynamic	Communication delay increased significantly during route recovery after node failures
Kebeng et al.[28]	Cluster-Based AODV (CLUBAA)	Dynamic	Improved scalability but experienced frequent route discovery failures in some scenarios
Huynh et al.[29]	HEAT-based Mesh Routing	Hybrid	Trade-off between routing overhead, latency, and scalability in mesh routing protocols
Gupta et al.[30]	Cluster-Based LoRa Routing	Dynamic	Additional mechanisms such as data aggregation are required to further improve performance
Proposed Work	Graph-Based Static Routing with Path Redundancy	Static	Designed for semi-static deployments without route discovery overhead

III. RESEARCH METHODOLOGY

A. System Design and Architecture

The proposed system is designed to implement multi-hop communication using a LoRa SX1276 transceiver and an ESP32 microcontroller. The system architecture consists of 6 ESP32 nodes configured to operate as data sources and relay nodes in a mesh topology. Each node is equipped with a LoRa SX1276 transceiver operating at 923 MHz, in accordance with the frequency allocation regulations for LoRa devices in Indonesia. The pin mapping between the ESP32 and LoRa SX1276 can be seen in tableII. In addition to using a custom-built SX1276-ESP32, we also used a TTGO T-Beam V1.1 LoRa-ESP32 board, as shown in the figure 1.

TABLE II. ESP32 AND LORA SX1276 PINS MAPPING

ESP32 Pin	LoRa SX1276 Pin
GND	GND
3.3 V	3.3 V
GPIO 2	DIO0
GPIO 14	RESET
GPIO 5	NSS
GPIO 18	SCK
GPIO 23	MOSI
GPIO 19	MISO

Each node is assigned a unique Node ID and hardware MAC address to distinguish it from other nodes in the network. At startup, each node initializes its transceiver and periodically broadcasts HELLO packets containing its Node ID. These packets are used solely to verify neighbor availability and support the failover mechanism between the primary and backup routes. No RSSI-based routing decisions are performed during runtime. The system does not rely on a centralized gateway; instead, routing and forwarding decisions are handled locally by each node based on the static routing table generated offline using graph analysis.

B. Static Routing with Path Redundancy

Each node in the LoRa network is configured with a static routing table that contains both a primary and a

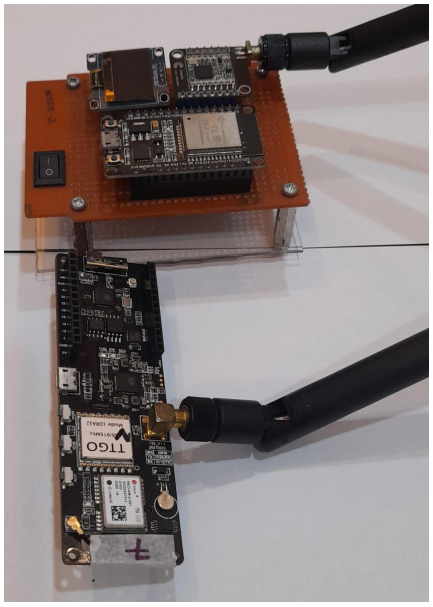

Fig. 1. Boards with LoRa radio used for the experimentation.Bottom board: TTGO T-Beam V1.1 LoRa-ESP32. Top board: custom LoRa-ESP32

backup route for each destination. Unlike dynamic routing, these routes are pre-computed offline using Python and NetworkX based on the measured RSSI values of all possible communication links. During runtime, the ESP32 node uses this static table to forward packets through the most reliable path, while periodically verifying the link health via HELLO messages.

1) Graph-Based Network Representation: The LoRa mesh network can be mathematically modeled as a directed weighted graph:

$$G = (V, E) \tag{1}$$

where V represents the set of nodes and E represents the set of wireless communication links between nodes. Each edge (i, j) has an associated weight $w(i, j)$ that represents the link cost, derived from the measured RSSI value as :

$$w(i, j) = -RSSI(i, j). \tag{2}$$

The path with the minimum cumulative weight is selected as the primary route, while the second-shortest path, ideally disjoint from the primary, is selected as the backup route. This ensures data delivery continuity even if the primary link fails due to interference or node unavailability

2) Implementation on ESP32: In firmware, static routing is implemented in the `LoRaRouting.cpp` module. The `staticroute.csv` file generated during the offline graph computation phase serves as an intermediate routing artifact. Before deployment, its routing entries are converted into an embedded static route array and compiled into the ESP32 firmware. Upon initialization, each node loads this pre-generated routing array, which defines the primary and backup next-hop IDs used for packet forwarding toward the destination node.

Routing decisions are not computed during runtime. All routes are determined offline using the graph-based routing algorithm, and no RSSI measurements are evaluated after deployment. During operation, HELLO packets are transmitted periodically only to verify the availability of the next-hop node on the primary route. If HELLO packets from the primary next-hop are not received within a predefined timeout interval, the primary route is considered unavailable and packet forwarding is automatically redirected to the backup route stored in the routing table. Three main background tasks are executed during runtime:

1) HELLO Sender: periodically broadcasts heartbeat packets containing the node ID.
2) RX Listener: receives incoming HELLO and DATA packets and processes packet forwarding.
3) Expiration Checker: monitors HELLO timeout events and triggers route failover when the primary next-hop becomes unavailable.

Each node continuously exchanges HELLO messages with its neighboring nodes to monitor the availability of the primary next-hop. Unlike dynamic routing protocols, no RSSI measurements are evaluated during runtime and no route computation is performed after deployment. If no HELLO packet is received from the primary next-hop within the predefined timeout interval, the primary route is considered unavailable and the system automatically switches to the backup route specified in the embedded static routing table. This mechanism provides route redundancy and failover capability without requiring route discovery or route recalculation. Algorithm 1 illustrates the complete runtime operation of the proposed static routing system on ESP32. After loading the embedded static routing table, the system continuously monitors HELLO messages, detects timeout events, and performs automatic failover between the primary and backup routes when necessary.

Algoritma1 presents the pseudocode of the static routing mechanism implemented in LoRaRouting.cpp. The algorithm defines the sequence of initialization, HELLO message transmission, packet forwarding, and failover logic. The algorithm runs continuously as a background process within the ESP32 firmware. By maintaining a simple yet reliable decision logic, it ensures that packet transmission automatically switches between primary and backup routes without requiring centralized coordination.

C. Graph Representation and Network Modeling

The reliability of communication between a source node S and a destination node D is analyzed using graph-theoretic principles. Each link (i, j) in the network is represented as a weighted edge, where the weight is inversely proportional to the measured RSSI value. Based on this representation, the routing structure of the LoRa network can be mathematically modeled as a directed weighted graph $G = (V, E)$.

To determine optimal routing paths for statically distributed LoRa nodes, an algorithmic procedure was

Algorithm 1 Static Routing with Path Redundancy on LoRa SX1276/ESP32

procedure INIT
  Load the embedded static route array generated from `staticroute.csv`; for each entry store (d, NH_p, NH_b) into SRT
  Set timeout parameter T_{stale}
  Spawn tasks: HELLOTX, RXHANDLER, AGING
end procedure
procedure HELLOTX
  loop
    Broadcast HELLO $\langle id \rangle$; sleep($T_{HELLO} + jitter$)
  end loop
end procedure
procedure RXHANDLER(pkt)
  Read payload $(type, s, d)$
  UpdateNeighbor(s, t_{now})
  if $type = HELLO$ **then**
    return
  end if
  if $d = NODE\_ID$ **then**
    DeliverToApp(pkt)
  else
    FORWARD(pkt, d)
  end if
end procedure
procedure FORWARD(pkt, d)
  Lookup $SRT[d] \rightarrow (NH_p, NH_b)$; if missing then drop/log and return
  $nh \leftarrow NH_p$ if HEALTHY(NH_p) else NH_b if HEALTHY(NH_b) else $\varnothing$
  if $nh = \varnothing$ **then**
    Drop/log "no healthy link"; **return**
  end if
  TransmitPacket(pkt, nh)
end procedure
function HEALTHY(n)
  if $n = \varnothing$ **then**
    return False
  end if
  $t_u \leftarrow NeighborTable[n].lastHello$
  $\Delta t \leftarrow t_{now} - t_u$
  return $(\Delta t \leq T_{stale})$
end function
procedure AGING
  loop
    For each neighbor n: if $(t_{now} - n.lastUpdated > T_{stale})$ then mark stale
    Sleep(T_{aging})
  end loop
end procedure

Algorithm 2 Graph-Based Static Routing with Path Redundancy

Require: $Link.csv$
Each entry $l \in L$ contains $\langle src, dst, RSSI, bidirectional \rangle$
Initialize directed graph $G \leftarrow (V, E)$
for all $l \in L$ **do**
  $u \leftarrow l.src$
  $v \leftarrow l.dst$
  $w \leftarrow -l.RSSI$ ▷ RSSI converted to positive cost
  Add directed edge $(u \rightarrow v)$ with weight w to G
  if $l.bidirectional = true$ **then**
    Add directed edge $(v \rightarrow u)$ with weight w to G
  end if
end for
$R \leftarrow \emptyset$
for all $s \in V$ **do**
  for all $d \in V$ **do**
    if $s = d$ **then**
      continue
    end if
    if exists path from s to d in G **then**
      Compute $P \leftarrow k$ shortest simple paths from s to d
      $P_1 \leftarrow$ first path in P ▷ Primary path
      $P_2 \leftarrow$ second path in P ▷ Backup path (if exists)
      $NH_p \leftarrow$ first hop after s in P_1
      if P_2 exists **then**
        $NH_b \leftarrow$ first hop after s in P_2
      else
        $NH_b \leftarrow \varnothing$
      end if
      Append $\langle s, d, NH_p, NH_b \rangle$ to R
    end if
  end for
end for
Export R to `staticroute.csv`

developed using the Python NetworkX library to construct a graph that computes primary and backup paths and estimates overall network reliability. The steps of this process are summarized in pseudocode presented in Algorithm2. The algorithm operates offline using collected RSSI data to generate a static routing table, which is then applied to each ESP32 node. The algorithm shows the main steps involved in constructing the graph, calculating the routing paths, and exporting the static routing table to the ESP32 device. The process begins by loading the measured RSSI data for each node path recorded in the file $(links.csv)$, constructing the network graph G(V,E), and determining the shortest and second-shortest paths using graph theory. The resulting calculations are then exported to the file *staticroute.csv*. The file *staticroute.csv* is used as an intermediate output of the offline graph computation. Its routing entries are then converted into a static array and embedded into the ESP-IDF firmware before deployment.

During runtime, each ESP32 node loads the embedded static routing array upon boot and forwards packets based on predefined primary and backup next-hop entries. The runtime forwarding mechanism uses periodic HELLO messages to monitor the availability of the primary next-hop. If no HELLO packet is received within the prede-

fined timeout interval, the primary route is considered unavailable and packet forwarding is automatically redirected to the backup route. This design provides path redundancy while avoiding route discovery, route maintenance, and dynamic routing table recalculation during network operation.

D. Overall Process Integration

The proposed system integrates a graph-theoretic path computation stage using Python with a static routing execution stage running on ESP32 nodes. The overall workflow is summarized in Figure2. First, the quality of the pairwise links between nodes is empirically measured by periodically exchanging HELLO packets and recording the received signal strength indicator (RSSI). These measurements are collected into a link database and stored as $link.csv$, which contains tuples of the format (src, dst, RSSI, bidirectional). In the offline stage, this data is processed using Python and NetworkX. The LoRa network is modeled as a directed weighted graph $G = (V, E)$, where each node in the network is mapped to a vertex, and each wireless link is mapped to a directed edge with weight $w(i, j) = -RSSI(i, j)$. Using this model, the shortest path is selected as the primary route, and the second-shortest path is selected as the backup route. The resulting routing table is exported as $staticroute.csv$.

At runtime, the ESP32 nodes use the embedded static routing table generated during the offline stage. Packet forwarding is normally performed through the primary route specified in the routing table. To monitor route availability, nodes periodically exchange HELLO messages. If the source node does not receive HELLO packets from the primary next-hop within the predefined timeout interval, the primary route is considered unavailable and packet forwarding is automatically redirected to the backup route. Since all routing decisions are computed offline, no RSSI-based route selection, route discovery, or route recalculation is required during network operation.

IV. RESULTS AND DISCUSSION

A. Formation of Static Routing Table using Python and NetworkX

The static routing table used by the ESP32-based LoRa network was generated offline using a Python program implemented in Google Colab and based on the NetworkX library. The program (algortm2) processed the RSSI measurements stored in *links.csv*, as illustrated in figure 3, which contained the measured RSSI values between all pairs of nodes. Each entry in the file represented a communication link between two nodes, including its directionality and measured signal strength.

In this study, NODE_0 was designated as the source node and NODE_5 as the destination node, while NODE_1, NODE_2, NODE_3, and NODE_4 served as intermediate forwarding nodes. Based on the graph constructed from the RSSI measurements, the program computed the two lowest-cost paths between NODE_0 and NODE_5. The path with the lowest accumulated cost was selected as the primary route, while the second-best path was designated as the backup route. The next-hop information for both routes was then extracted and stored in the *staticroute.csv* file, in the format: $(src, dst, primary\_next, backup\_next)$, table III explains the roles of the two files used.

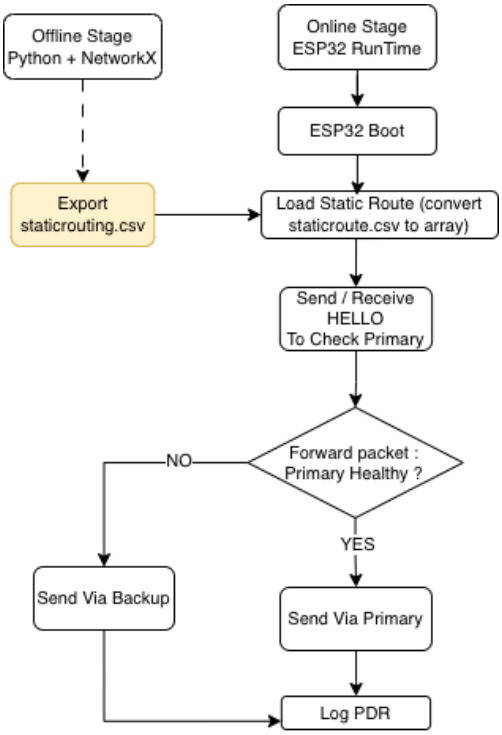

Fig. 2. Offline and online process integration

TABLE III. INPUT AND OUTPUT FILES FOR OFFLINE GRAPH COMPUTATION

File	Description
links.csv	Measured RSSI values between node pairs used to construct the weighted graph.
staticroute.csv	Generated routing table containing the primary and backup next-hop information for communication from NODE_0 to NODE_5.

The resulting routing table is converted into a static route array and embedded into the firmware of all ESP32-SX1276 nodes prior to deployment. During network operation, packets are forwarded through the primary next hop under normal conditions. If the primary route becomes unavailable due to loss of connectivity with the corresponding next hop, packets are automatically routed through the backup route. Since all routing paths are pre-computed offline using Python and NetworkX, no route discovery is required during runtime, reducing the computational burden and enabling efficient packet forwarding on resource-constrained embedded devices.

The integration of offline graph computation and real-time static routing has proven crucial for improving network reliability. By pre-computing primary and backup routes on the host computer, as illustrated in Figure 2, the ESP32-SX1276 node only needs to perform lightweight operations during runtime, including periodic HELLO message transmission, link health monitoring, and packet

forwarding. Packets are typically transmitted over the primary route. If the source node does not receive a HELLO message from the next-hop node associated with the primary route within a predetermined timeout period, the primary route is deemed unavailable, and traffic is immediately redirected to the pre-computed backup route. Because both routes are determined offline, failover can be performed quickly without additional path computation or route discovery, making this approach suitable for resource-constrained LoRa devices.

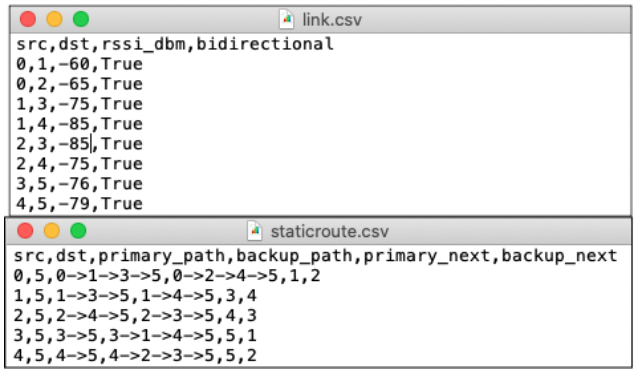

Fig. 3. Input link.csv and Output staticroute.csv on Python NetworkX lybrary

B. *Packet Delivery Ratio (PDR) Evaluation*

The Packet delivery ratio (PDR) was evaluated to measure the consistency of the proposed static routing system with path redundancy. This experiment was conducted between a classroom and a laboratory at the Department of Electrical Engineering, University of Mataram.

The experiment involved six LoRa nodes (NODE_0–NODE_5). NODE_0 served as the source node, while NODE_5 acted as the destination node. The remaining nodes (NODE_1–NODE_4) functioned as intermediate forwarding relays based on a static routing table (staticroute.csv) generated from RSSI values measured in $link.csv$.

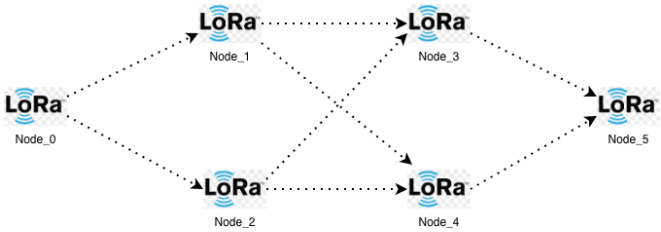

Fig. 4. Topology of LoRa nodes

Each link's RSSI ranged between –60 dBm and –85 dBm, forming the partially meshed topology: $0 \leftrightarrow 1$, $0 \leftrightarrow 2$, $1 \leftrightarrow 3$, $2 \leftrightarrow 3$, $1 \leftrightarrow 4$, $2 \leftrightarrow 4$, $3 \leftrightarrow 5$, and $4 \leftrightarrow 5$. The routing table computed using Python + NetworkX defined for each source–destination pair both a primary and backup path. For example, communication from Node 0 to Node 5 used:

Primary: $0 \rightarrow 1 \rightarrow 3 \rightarrow 5$, Backup: $0 \rightarrow 2 \rightarrow 4 \rightarrow 5$.

During the test, NODE_0 transmitted 20 packets sequentially to NODE_5. Each relay node forwarded packets according to the precomputed routing table. Under normal conditions, packets were transmitted through the primary route. When the source node failed to receive HELLO messages from the next-hop node on the primary route within the predefined timeout interval, the primary route was considered unavailable and packet forwarding was automatically redirected to the backup route. Table IV summarizes the experimental outcomes. The PDR was calculated as:

$$\text{PDR} = \frac{N_{\text{received}}}{N_{\text{sent}}} \times 100\%. \tag{3}$$

TABLE IV. PDR RESULTS FOR DIFFERENT TRANSMISSION DELAYS

Delay (s)	TX	RX-Average	Loss Packet-Average	PDR (%)
10	20	16.0	4.0	80.0
12	20	17.0	3.0	85.0
15	20	19.0	1.0	95.0
20	20	19.0	1.0	95.0
30	20	19.3	0.7	96.6
40	20	19.3	0.7	96.6
AVERAGE	**20**	**18.3**	**1.7**	**91.4**

Tabel IV summarizes the Packet Delivery Ratio (PDR) obtained. Each experiment was repeated three times for every transmission delay. The results indicate that the transmission delay has a significant impact on communication reliability. When the delay was set to 10 seconds, the average number of successfully received packets was 16 out of 20 packets, resulting in a PDR of 80%. Increasing the delay to 12 seconds improved the average PDR to 85%, while delays of 15 and 20 seconds further increased the PDR to 95%.

The highest performance is achieved at transmission delays of 30 and 40 seconds, where the average number of received packets reaches 19.3 out of 20 and the corresponding PDR reaches 96.6%. At these delays, the average packet loss is reduced to only 0.7 packets. The overall average PDR across all experiments is 91.4%, indicating that the proposed static routing mechanism with precomputed primary and backup routes provides reasonably reliable end-to-end communication in the tested multi-hop LoRa network. Figure 5 presents an example of the serial log generated during the experiment, showing the reception of data packets at the destination node and the corresponding Packet Delivery Ratio (PDR) calculation.

The observed PDR improvement with slower transmission rates can be attributed to the reduced probability of packet collisions and channel contention among LoRa nodes. A longer interval between packet transmissions provides additional time for intermediate nodes to complete packet forwarding and exchange HELLO messages before the next packet is sent and received. Consequently, the network experiences less packet loss and achieves more stable communication performance. These results indicate that the transmission interval configuration plays a significant role in determining the reliability of static multi-hop LoRa communication.

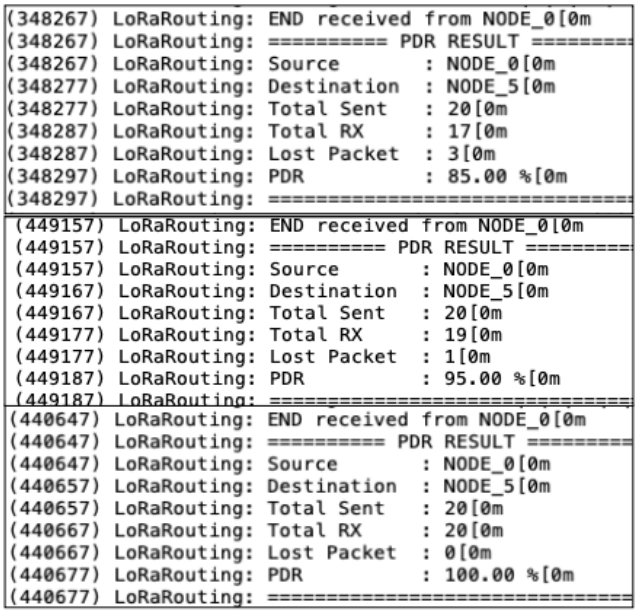

Fig. 5. Serial monitor output showing packet reception and PDR calculation at the destination node

This PDR value is driven by the RSSI-based route selection process performed during the offline graph generation phase. RSSI measurements collected from all communication links are used to generate the routing table using the Python and NetworkX frameworks. Links with stronger RSSI values were assigned lower routing costs and were therefore more likely to be selected as part of the primary route. Consequently, packet forwarding occurred through communication paths that provided better link quality and a lower probability of packet loss.

The availability of precomputed backup paths also significantly improves communication reliability. For the source-destination pair, the routing table stores a primary route and an alternative backup route. The backup route was generated offline together with the primary route using the graph-based routing algorithm. Therefore, when a failover event occurred, no additional route computation or route discovery process was required. The source node simply switched to the alternative next-hop specified in the precomputed routing table. The routing table was not generated dynamically from RSSI measurements collected during data transmission but was precomputed offline, converted into a static route array, and embedded into each ESP32 node before deployment. This redundancy increased communication resilience because alternative forwarding paths were immediately available whenever the primary route became unavailable. No route discovery or route maintenance is required during runtime, reducing communication and packet delivery overhead on the ESP32 node.

It should be noted that RSSI measurements are not continuously monitored during the PDR evaluation phase. In contrast, RSSI values are collected in advance and used exclusively for offline route calculation. The resulting routing information is first stored in *staticroute.csv*, then converted into an embedded static route array in the ESP-IDF firmware prior to deployment. Therefore, forwarding decisions during runtime are based entirely on the precomputed routing table, not on dynamic RSSI measurements or route retransmissions.

While RSSI is a practical indicator of link quality, its value can vary due to environmental conditions, wireless interference, and propagation effects. Nevertheless, the average PDR of 91.4% indicates that the combination of RSSI-based offline route selection and path redundancy is effective in maintaining reliable end-to-end communication under the tested deployment conditions.

One limitation of the present study is the absence of a direct baseline comparison against static routing without path redundancy or alternative routing approaches under identical deployment conditions. The primary objective of this work was to demonstrate the feasibility and practical implementation of graph-based static routing with precomputed backup paths on real LoRa hardware. Consequently, the reported PDR values reflect the performance of the proposed approach only. Future studies should include comparative experiments to quantify the extent of performance improvement introduced by the path redundancy mechanism.

V. CONCLUSION AND SUGGESTIONS

This study demonstrated the successful implementation of a static routing system with path redundancy on LoRa SX1276 and ESP32 devices using a graph-theoretic approach. By integrating offline route computation through Python and NetworkX with an embedded static routing table, the system achieved reliable multi-hop communication without dynamic route recomputation. Experimental results obtained from a six-node network showed an average Packet Delivery Ratio (PDR) of 91.4%, indicating that the precomputed primary and backup routes were effective in maintaining communication continuity when the primary forwarding path became unavailable. These findings suggest that the proposed approach provides a reliable and low-overhead routing framework suitable for IoT applications such as environmental monitoring, smart agriculture, and infrastructure monitoring systems.

Despite these promising results, several limitations remain. The proposed routing mechanism is intended for networks with semi-static topologies, where routing tables are generated offline based on RSSI measurements. Consequently, significant environmental changes or node mobility may reduce route optimality if the routing table is not periodically updated. In addition, the experimental evaluation was conducted on a small-scale six-node testbed in a campus environment, which may not fully represent the performance characteristics of larger LoRa mesh deployments. Furthermore, energy consumption was not directly measured and was only indirectly inferred from the reduction of routing overhead and lightweight runtime processing.

Future work will focus on extending the proposed approach through the development of a hybrid static-

dynamic routing architecture capable of adapting to topology changes while maintaining low communication overhead. Additional studies will also evaluate network scalability using a larger number of nodes and investigate the incorporation of complementary routing metrics, such as battery power status and packet loss rate, to further enhance communication reliability and routing resilience under more challenging operating conditions.

ACKNOWLEDGMENTS

Thank you to the University of Mataram's leadership for financing this study under the DIPA BLU capacity building scheme (contract number 1775/UN18.L1/PP/2024).

REFERENCES

[1] A. Celik, I. Romdhane, G. Kaddoum, and A. M. Eltawil, "A top-down survey on optical wireless communications for the internet of things," *IEEE Communications Surveys Tutorials*, vol. 25, no. 1, pp. 1–45, 2023.

[2] H. H. R. Sherazi, L. A. Grieco, M. A. Imran, and G. Boggia, "Energy-efficient lorawan for industry 4.0 applications," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 891–902, 2021.

[3] Y. Mao, Y. Ge, Y. Fan, W. Xu, Y. Mi, Z. Hu, and Y. Gao, "A survey on lora of large language models," *Frontiers of Computer Science*, vol. 19, no. 7, p. 197605, 2024. [Online]. Available: https://doi.org/10.1007/s11704-024-40663-9

[4] Z. J. R. Kamoona and M. Ilyas, "Investigating the performance of lora communication for nominal lora and interleaved chirp spreading lora," in *2022 International Conference on Artificial Intelligence of Things (ICAIoT)*, 2022, pp. 1–7.

[5] H. Rajab, H. Al-Amaireh, T. Bouguera, and T. Cinkler, "Evaluation of energy consumption of lpwan technologies," *EURASIP Journal on Wireless Communications and Networking*, vol. 2023, no. 1, p. 118, 2023. [Online]. Available: https://doi.org/10.1186/s13638-023-02322-8

[6] S. Y. Mane, "Lpwan's – overview, market scenario and performance analysis of lora, sigfox using nb-fi range calculator," in *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, 2021, pp. 1–4.

[7] N. Okafor, R. Ingle, U. Okwudili Matthew, M. Saunders, and D. T. Delaney, "Assessing and improving iot sensor data quality in environmental monitoring networks: A focus on peatlands," *IEEE Internet of Things Journal*, vol. 11, no. 24, pp. 40 727–40 742, 2024.

[8] D. Abhiram, K. D. Reddy, P. K. Reddy, and M. Belwal, "Iot-based environmental monitoring system," in *2025 3rd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, 2025, pp. 764–769.

[9] J. P. L. Sánchez, A. S. Paipilla Arenas, and H. Paz Penagos, "Environmental monitoring system based on lora communication," in *Applied Computer Sciences in Engineering*, J. C. Figueroa-García, J. A. López-Sotelo, J. F. Moreno-Trujillo, and E. E. Gaona-García, Eds. Cham: Springer Nature Switzerland, 2026, pp. 202–210.

[10] A. Pagano, D. Croce, I. Tinnirello, and G. Vitale, "A survey on lora for smart agriculture: Current trends and future perspectives," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3664–3679, 2023.

[11] Y.-T. Ting and K.-Y. Chan, "Optimising performances of lora based iot enabled wireless sensor network for smart agriculture," *Journal of Agriculture and Food Research*, vol. 16, p. 101093, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666154324001303

[12] V.-D. Bui, T.-T.-H. Nguyen, and G.-T. Bui, "Application of lora technology for monitoring and controlling irrigation systems in agriculture," in *Proceedings of the 4th Annual International Conference on Material, Machines, and Methods for Sustainable Development (MMMS2024)*, B. T. Long, H. X. Nang, P. T. Huy, Y.-H. Kim, K. Ishizaki, K. Hyungsun, D.-T. Nguyen, V. V. Truong, N. T. Hong Minh, and P. Duc An, Eds. Cham: Springer Nature Switzerland, 2025, pp. 429–436.

[13] L. Kane, V. Liu, M. McKague, and G. R. Walker, "Network architecture and authentication scheme for lora 2.4 ghz smart homes," *IEEE Access*, vol. 10, pp. 93 212–93 230, 2022.

[14] K. B. Shah, S. Visalakshi, D. P. Guragain, and R. Panigrahi, "Advancing smart city sustainability with internet of things and artificial intelligence aided low-cost digital twin systems for waste management," *Microsystem Technologies*, vol. 31, no. 10, pp. 2783–2796, 2025. [Online]. Available: https://doi.org/10.1007/s00542-024-05827-4

[15] A. Osorio, M. Calle, J. Soto, and J. E. Candelo-Becerra, "Routing in lora for smart cities: A gossip study," *Future Generation Computer Systems*, vol. 136, pp. 84–92, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22001996

[16] B. S. Kumar, S. Ramalingam, V. Divya, S. Amruthavarshini, and S. Dhivyashree, "Lora - iot based industrial automation motor speed control monitoring system," in *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, 2023, pp. 11–15.

[17] J. R. Cotrim and J. H. Kleinschmidt, "Lorawan mesh networks: A review and classification of multihop communication," *Sensors*, vol. 20, no. 15, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/15/4273

[18] H. Alhomyani, M. Fadel, N. Dimitriou, H. Bakhsh, and G. Aldabbagh, "Modeling the performance of a multi-hop lorawan linear sensor network for energy-efficient pipeline monitoring systems," *Applied Sciences*, vol. 14, no. 20, 2024. [Online]. Available: https://www.mdpi.com/2076-3417/14/20/9391

[19] A. Vazquez-Rodas, F. Astudillo-Salinas, C. Sanchez, B. Arpi, and L. I. Minchala, "Experimental evaluation of rssi-based positioning system with low-cost lora devices," *Ad Hoc Networks*, vol. 105, p. 102168, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870520301736

[20] K. Chara, F. Srairi, S. Ghedjati, and F. Radjah, "Efficient indoor rssi analysis for iot-based weather stations using lora protocol in agricultural applications," in *2024 12th International Conference on Systems and Control (ICSC)*, 2024, pp. 104–108.

[21] V. S. V. G. Sarma Lanka and D. Kothandaraman, "Mobility-aware lora protocol for reliable iot communications," *IEEE Access*, vol. 14, pp. 5869–5890, 2026.

[22] L. Prade, J. Moraes, E. de Albuquerque, D. Rosário, and C. B. Both, "Multi-radio and multi-hop lora communication architecture for large scale iot deployment," *Computers and Electrical Engineering*, vol. 102, p. 108242, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790622004773

[23] Y. Lalle, M. Fourati, L. C. Fourati, and J. P. Barraca, "Routing strategies for lorawan multi-hop networks: A survey and an sdn-based solution for smart water grid," *IEEE Access*, vol. 9, pp. 168 624–168 647, 2021.

[24] L. Aarif, M. Tabaa, and H. Hachimi, "RSSI prediction and optimization of transmission power for improved LoRa communications performance," *Annals of Telecommunications*, vol. 80, no. 5, pp. 501–520, Jun. 2025. [Online]. Available: https://doi.org/10.1007/s12243-024-01059-9

[25] V. D. Pham, D. T. Le, R. Kirichek, and A. Shestakov, "Research on using the aodv protocol for a LoRa mesh network," in *Distributed Computer and Communication Networks*, V. M. Vishnevskiy, K. E. Samouylov, and D. V. Kozyrev, Eds. Cham: Springer International Publishing, 2020, pp. 149–160.

[26] M. J. Ying Chong, S. Kit Lim, K. K. Tan, and J. Huey Khor, "Aodv-inspired routing with next-hop caching and path selection

for lora mesh networks," in *2025 IEEE Industrial Electronics and Applications Conference (IEACon)*, 2025, pp. 75–80.

[27] J. Thaenthong, T. Pimsen, and K. Chidchuea, "R-aodv: Reverse ad-hoc on-demand distance vector routing algorithm with lora mesh based for emergency alert system," in *2023 15th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2023, pp. 143–148.

[28] T. O. Kebeng, S. M. Sheikh, and M. Kgwadi, "A cluster based directional forwarding routing protocol for bandwidth constrained networks," *International Journal of Engineering Trends and Technology*, vol. 70, no. 9, pp. 155–166, 2022. [Online]. Available: https://doi.org/10.14445/22315381/IJETT-V70I9P216

[29] H.-K. Huynh and H.-A. Pham, "Heat routing algorithm for multi-hop communication in iot-enabled lora-based wireless mesh networks," in *2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, 2022, pp. 756–760.

[30] S. Gupta and I. Snigdh, "Clustering in LoRa networks, an energy-conserving perspective," *Wireless Personal Communications*, vol. 122, no. 1, pp. 197–210, 2022. [Online]. Available: https://doi.org/10.1007/s11277-021-08894-2

[31] D. Hercog, T. Lerher, M. Truntič, and O. Težak, "Design and implementation of esp32-based iot devices," *Sensors*, vol. 23, no. 15, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/15/6739

[32] J. L. Soler-Fernández, O. Romera, A. Diéguez, J. D. Prades, and O. Alonso, "Lora power model for energy optimization in iot applications," *Sensors*, vol. 26, no. 1, p. 301, 2026. [Online]. Available: https://doi.org/10.3390/s26010301

[33] N. Alharbi, L. Mackenzie, and D. Pezaros, "Enhancing graph routing algorithm of industrial wireless sensor networks using the covariance-matrix adaptation evolution strategy," *Sensors*, vol. 22, no. 19, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/19/7462