Opportunistic Routing towards Mobile Sink Nodes in Bluetooth Mesh Networks

Marcelo Paulon J. V.[a], Bruno José Olivieri de Souza[b] and Markus Endler[c]
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Brazil

Keywords: Wireless Mesh Networks, Bluetooth Mesh, Mobile Sinks, Data Collection, Controlled Flooding, Flooding Routing.

Abstract: This work evaluates sporadic data collection on a Bluetooth Mesh network, using the OMNET++ INET simulator. The data collector is a roaming sink node, which could be a smartphone or other portable device, carried by a pedestrian, a biker, an animal, or a drone. The sink node could connect to a mesh network in hard-to-reach areas that do not have internet access and collect sensor data. After implementing Bluetooth Mesh relay extensions, Low Power, and Friend features in OMNET++, we were able to propose and evaluate algorithms for mobility-aware, adaptive, routing of sensor data towards the sink node. While the long-term goal for this research is to implement the proposed algorithms on ESP32-based SoCs to monitor tree health, so far, the preliminary simulated results already reveal some interesting findings. One variation of a proposed routing algorithm achieved an 82.00% increase in unique data delivered to the sink node compared to Bluetooth Mesh's default routing algorithm. In that case, there was a 5.45% decrease in energy consumption for the same scenario. Also, the delivery rate increased by 58.22%.

1 INTRODUCTION

This work considers a scenario where several nodes equipped with sensors are spread in one area of difficult access, where each node monitors the health of a tree with sensors attached to its trunk and foliage, as well as some environmental variables such as temperature and humidity in the proximity of the tree. We further consider that sensor data accumulated at each mesh node can be retrieved by a mobile sink node (i.e., Mobile-Hub) when the sink gets sufficiently close to the mesh node during its continuous movement within the monitored arboretum mesh (i.e., MAM) region.

In any case, the goal is that the Mobile-Hub(s) should be able to collect as much sensor data from the whole network while on the move. But this requires an agile routing of the sensor data in the WSN towards the direction of the place where the Mobile-Hub is currently "having a rendezvous" with a mesh node.

As can be seen, this use case faces not only the challenges of intermittent connectivity (since the Mobile-Hub is only sporadically connected to the mesh network along its trajectory) but also of the energy constraints of the mesh network.

Bluetooth is a wireless technology that can be used for WSNs and may be an interesting option since most commercially available smartphones, as well as many microcontroller devices and system-on-chip devices (SoCs), support Bluetooth and its more recent Low Energy features (Baert et al., 2018). One example of an SoC that supports Bluetooth is the ESP32. (Giacomini et al., 2020) describes the implementation of a Bluetooth routing approach for IoT environments using ESP32 SoCs.

One option for organizing Bluetooth networks is by forming a mesh network with the Bluetooth Mesh standard (Baert et al., 2018) (which will be referenced as BTMesh in this paper). BTMesh's latest version (5.1) was officially released in 2019 (Bluetooth SIG, 2019), and it tries to achieve more efficient energy draw when compared to other technologies such as Wi-Fi and ZigBee. BTMesh routes packets across the network by adopting a relay strategy that consists of controlled flooding.

This work aims to propose two alternatives to BTMesh's default relay algorithm (MAM_0 and MAM_Δ) that may achieve higher packet delivery rates

[a] https://orcid.org/0000-0002-8382-0957
[b] https://orcid.org/0000-0002-1707-7755
[c] https://orcid.org/0000-0002-8007-9817

Paulon J. V., M., Olivieri de Souza, B. and Endler, M.
Opportunistic Routing towards Mobile Sink Nodes in Bluetooth Mesh Networks.
DOI: 10.5220/0010557000670075
In *Proceedings of the 18th International Conference on Wireless Networks and Mobile Systems (WINSYS 2021)*, pages 67-75
ISBN: 978-989-758-529-6

and lower energy draw when routing data towards a Mobile-Hub. Those alternatives were evaluated in a simulated data collection context, considering BTMesh's default relay algorithm (which this work will call BTM-R from here on) as a benchmark.

The results indicate that one of the proposed algorithms (MAM_{Δ}) achieves a higher packet delivery rate to the Mobile-Hub when compared to BTM-R. This delivery rate considers the number of unique data packets received by the Mobile-Hub versus how many packets were generated and sent by all other nodes. This work also evaluated the global energy draw, the number of packets received on the Mobile-Hub, and the end-to-end delay (from each BTMesh sensor to the Mobile-Hub). The MAM_{Δ} algorithm presented lower end-to-end delay and received more unique data packets than BTM-R. However, in some configurations, it performed worse in terms of energy draw.

In the next section, we present related work in BTMesh networking and data collection in wireless sensor networks. Section 3 covers BTMesh and its characteristics. Section 4 describes the proposed data collection solution, explaining BTM-R, MAM_0, and MAM_{Δ} relay algorithms. Section 5 describes the simulation model and evaluation metrics, as well as the simulations. Section 6 presents and discusses the simulation results. Section 7 concludes by presenting future this works.

2 RELATED WORK

After the BTMesh specification was released, some studies and simulations for the BTMesh technology have been explored, such as (Baert et al., 2018) (Leon and Nabi, 2020) (Hansen et al., 2018). (Leon and Nabi, 2020) analyses BTMesh in a real-world environment and reports limitations for message delivery as a result.

(Hansen et al., 2018) evaluate three relay selection mechanisms with the intent of reducing the number of relay nodes in the BTMesh network to reduce costs while preserving a certain level of redundancy. Their work is orthogonal to the present work, as it focuses on the BTMesh network formation (in which the network topology is defined), whereas the present work focuses on analyzing routing for data collection without altering the BTMesh network topology.

The authors could not find extensions of BTMesh relay algorithms that could be directly compared to the algorithms this work describes in Chapter 4 - this is - simple extensions to BTMesh routing that can be implemented on top of BLE. For instance, such extensions can be implemented on a microcontroller with BLE support without needing to alter BLE functionality.

BTMesh adopts a flooding routing approach and there is an extensive amount of published work on this topic, with optimizations through concurrent-transmission based flooding (Mager and Zimmerling, 2016) (Ma et al., 2018) (Cheng et al., 2018) (Ma et al., 2020). The *Harmony* algorithm (Ma et al., 2020) was tested through experimental evaluation and, compared to the state-of-the-art at the time of publication, presented 50% higher delivery rates and shorter end-to-end latencies in the presence of harsh Wi-Fi interference. However, such routing algorithms optimizations differ significantly from the algorithms proposed by this work, as they are not designed considering Bluetooth compatibility. The advantage of preserving Bluetooth compatibility is to more easily implement and deploy applications, as many commercially available devices such as smartphones and microcontrollers support Bluetooth (Baert et al., 2018).

(Djedouboum et al., 2018) studied the current state of data collection in Wireless Mesh Sensor Networks and analyzed its challenges in the context of Big Data. They also discussed the challenges of data collection when mobility is involved, like contact detection with mobile data collectors, quality of service (QoS), and location detection. The MAM algorithms do not cover quality of service and location detection, which are out of the scope of the present work. However, contact detection is an important part of the routing process and the MAM algorithms would be defined in the context of their research as feature-based routing protocols that rely on route discovery.

3 BLUETOOTH MESH

BTMesh is a mesh standard based on Bluetooth Low Energy (BLE) that supports many-to-many communication using the wireless Bluetooth protocol.

The standard uses BLE-specific advertising and scanning as underlying mechanisms to achieve flooding-like communication (Bluetooth SIG, 2016). BTMesh flooding ensures that some nodes in the network, called Relay Nodes, repeat incoming messages so that they are relayed further until their destination is reached. Compared to conventional BLE advertising, BTMesh nodes do not send packets according to advertising intervals but send their packets directly after a random generated back-off time per channel. To scan the advertisement channels for incoming packets, the mesh nodes use a 100% duty cycle, meaning that they are permanently scanning unless they are sending a packet.

In order to prevent the obvious problems caused by uncontrolled message flooding, BTMesh introduces relay cache features. Only nodes that have the relay feature enabled (i.e., RNs) will forward received messages to neighbor nodes. There is an LRU (least recently used) cache on each relay node that stores packet signatures and ensures a relay node only relays a specific message once. Also, each message has a Time-To-Live (TTL) field that represents the number of hops. Messages are only relayed if they are not in the cache, and the number of hops is less than 127 (the number 127 is defined by the Bluetooth specification and corresponds to a 1-octet opcode).

The full-time duty cycle to scan the different BLE advertisement channels directly impacts the node's energy consumption. The BTMesh standard tries to solve this by introducing Friendship and Low Power node features. A BTMesh Friend Node (i.e., FN) has mainly two responsibilities: storing incoming messages for nearby Low Power Nodes (i.e., LPNs) and sending those messages to the LPNs (LPNs periodically query their FNs for new messages).

With the Friendship feature, Low Power nodes don't need to stay permanently scanning the network and can save battery power by keeping their radio stack disabled most of the time. Thus, according to the Bluetooth Mesh specification, FNs can function as intermediate storage and opportunistic relay nodes for the other "energy-restricted" mesh nodes that will awake typically only for communication with some FN during short periods to save energy.

Despite these advantages, BTMesh has two main problems considering this work's data collection scenario (described in section 1):

- Network size limitation - message routing is ad hoc, but performed through a controlled flooding approach that limits the number of hops to 127 (which could be insufficient for a vast area network application).
- Power consumption - although the flooding approach results in some flexibility in terms of handling nodes' neighbors change as well as low latency (packets always get sent through the shortest path), duplicate messages are sent through the network and this impacts power consumption.

This work focuses on improving power consumption for data collection scenarios using Bluetooth Mesh, and does not try to overcome the Bluetooth Mesh 127 hop limitation.

The network topology significantly influences how the network will behave, as defining which nodes are relay nodes, friend nodes, or low power nodes is not done dynamically and, if not done correctly, can make the network inefficient and even disconnected (by exceeding the hop limit from one node to another, or by lack of relay nodes that can forward messages between them). This work's configured simulation topologies, described in section 5, are connected networks with relay nodes, friend nodes, and low power nodes.

4 PROPOSAL

This section describes the application of BTMesh for collecting data using a Mobile-Hub as described in section 1. The first approach used the standard BTMesh relay implementation (i.e., BTM-R) to direct messages towards the Mobile-Hub. Also, this work proposes two alternative relay algorithms for the specific scenario that is being discussed.

To collect data from the network, the Mobile-Hub sends a discovery packet periodically (every 1 second) while moving around the area. The discovery packet is used to notify the nodes that there is a data sink available to receive data, and those packets eventually reach every network node if they are received by one of the network relay nodes, provided the relay algorithm restrictions (e.g., BTM-R enforces a maximum hop limit). Once any node receives a discovery packet, it should send its sensor data as well as any stored sensor data towards the Mobile-Hub.

The BTM-R determines that the relay nodes only consider the number of hops made so far and whether they have already relayed the message, when evaluating if the message should be relayed.

The proposed alternatives to BTM-R only consider the scenario that was described in section 1 (routing data towards a single Mobile-Hub). This is an important difference between this work and alternative routing technologies for sensor networks that cover routing from any node to another in the network.

When describing alternative relay algorithms in this section, this work will consider packets that are not Discovery packets to be "data packets", which are packets sent by the Mesh network nodes containing sensor data that should be relayed to the Mobile-Hub.

Each subsection contains a pseudo-code with a possible implementation of the described relay algorithms. The code would be run on every relay node for each packet they receive, and would receive as input: the sender's address (`senderAddress`), the number of packet hops (`messageHops`), and the packet's content (`messageBody`). Global variables stored in the nodes, available across local executions, are initialized in the pseudo-code's *Init* session.

4.1 BTMesh Relay (BTM-R): Flooding

BTMesh's original relay algorithm (BTM-R) consists of a controlled flooding approach (Bluetooth SIG, 2018). The algorithm combines two strategies to manage the network flooding:

1. limit the number of packet hops to 127 (corresponds to a 1-octet opcode, as defined by the specification);
2. avoid the same node relaying a packet multiple times.

The pseudo-code *Algorithm 1* illustrates BTM-R's implementation logic. Firstly it computes the packet hash and checks if it is present on an LRU cache, storing this information on a variable named recentlyRelayed (lines 1 and 2).

If it was recently relayed (recentlyRelayed == true) or if the number of messages hops is greater than 126, it stops executing, and the packet gets discarded (lines 3-5). Otherwise, the number of message hops is incremented and the packet is relayed through a broadcast (lines 6 and 7).

Two noticeable problems with this approach are: I) due to BTM-R routing approach, the messages may get relayed excessively and delivered multiple times since they are sent through every route possible. If much data is coming from sensors, there may be competition on multiple routes to deliver it. II) the 127 hop limit could be a problem depending on the nodes' topology layout/distribution, possibly making certain nodes unreachable to others.

Algorithm 1: BTMesh Relay (BTM-R).

```
Input: senderAddress, messageHops, messageBody
1: byte hash ← hashMessage(messageBody)
2: bool recentlyRelayed ← isInLRUCache(hash)
3: if (recentlyRelayed == true or messageHops > 126) then
4:     return
5: end if
6: hops ← messageHops + 1
7: broadcastMessage(messageBody, hops)
```

4.2 MAM_0: Last Known Route

The MAM_0 algorithm is the first alternative routing algorithm this work designed and evaluated as a viable alternative to BTM-R for Mobile-Hub data routing. MAM_0 is based on a reactive routing strategy that only uses BTM-R's controlled flooding approach for Discovery packet propagation.

With the intent of maintaining a route to the Mobile-Hub, each node sets the last known directly connected node to have access to a Mobile-Hub. This information is updated on every node upon each received Discovery packet. With this approach, data packets are then no longer broadcast but are sent to a single node in each step.

The pseudo-code *Algorithm 2* illustrates MAM_0 implementation. On every relay node, a global variable `bestNodeAddress` is initialized as NULL. In line 1, the algorithm checks if the packet is a Discovery packet by looking at its body. If it is a Discovery packet, the global variable `bestNodeAddress` gets set to the packet sender address (line 2), the packet is relayed using BTM-R's mechanism, and the execution stops (lines 3-4).

If the packet is not a Discovery packet, the algorithm assumes it is a Data packet that should be directed towards a Mobile-Hub. Discovery packets are only generated by the passing Mobile-Hub and may be relayed by relay nodes further into the network.

The algorithm continues to execute if the incoming packet is not a Discovery packet. It checks if the global variable `bestNodeAddress` is set (line 6), and if it is, the message gets sent to this address with an incremented number of hops (lines 7-8).

This logic implies that if the variable `bestNodeAddress` is not set, data packets will not be relayed. Also, it implies that if it relays a data packet, it will only be relayed to a single node. It was designed with those characteristics in consideration, with the intent of reducing the number of messages propagated through the network and thus the overall energy consumption.

Algorithm 2: MAM_0 - Last known route.

```
Init: bestNodeAddress ← NULL
Input: senderAddress, messageHops, messageBody
1: if (isDiscoveryMessage(messageBody) == true) then
2:     bestNodeAddress ← senderAddress
3:     bluetoothMeshRelay(senderAddress, messageHops, messageBody)
4:     return
5: end if
6: if (bestNodeAddress != NULL) then
7:     hops ← messageHops + 1
8:     sendMessage(bestNodeAddress, messageBody, hops)
9: end if
```

4.3 MAM_{Δ}: Reactive Least-hop Route

The MAM_{Δ} algorithm also consists of a reactive routing approach, but it sets data packet routes to Mobile-Hubs based on its distance (in hops) to the Mobile-Hub. This distance is essentially the number of hops it takes from each node until the Mobile-Hub.

This approach requires a way of knowing in advance the number of hops from each node to the Mobile-Hub (and updating it often as this information changes as the Mobile-Hub moves). This is achieved through the discovery message packet hop information.

Upon receiving a Discovery packet, the relay node evaluates if the sender would be the best destination for sending data to the Mobile-Hub. This evaluation considers the number of hops, as well as an expiration time.

The expiration time is called expiry and, when expired, makes the next Discovery packet have its sender set as the best node regardless of the number of hops. This way, the best routes get preserved for some time, but the logic accounts for them eventually becoming old/invalid. A Δ (milliseconds) parameter is used to control this expiration's length.

If the route is not expired and the number of hops of an incoming Discovery packet is less than the best one, then the algorithm considers it to be the new best destination, and sets its state accordingly (storing the new best sender address and number of hops and resetting the expiry/timeout).

The pseudo-code *Algorithm 3* describes MAM_{Δ}'s implementation.

The variable `bestNodeAddress` is the address used to forward data packets, as the relay node considers it to be the best node to reach the Mobile-Hub.

The variable `bestNodeHops` is the number of hops from the current node to the Mobile-Hub. It is stored to decide whether or not the best node should be updated.

The variable `expiry` is the expiration time that was previously described. Upon receiving a Discovery packet, the algorithm will set the best node to the packet sender if the current time is greater than expiry.

In line 1, it checks if the packet is not a Discovery packet. If it's not a discovery packet, then it will be relayed if the best node is set (lines 2-5) and then execution will stop regardless of the best node being set or not (line 6).

If execution proceeds, it means the algorithm is handling a discovery packet. In line 8, it checks whether or not the current time is greater than expiry, and sets this boolean value to a variable called expired.

In line 9, it checks if the value of expired is true or the number of hops is less than the value held in the global variable bestNodeHops. If this check is valid, the bestNodeAddress global variable is set to the sender's address (line 10), the bestNodeHops global variable is set to the number of packet hops (line 11), and the expiry global variable is set to the current time plus the algorithm parameter Δ (line 12).

Line 14 is the final execution step run for all Discovery packets, to relay them using the BTM-R's logic. Similarly to MAM_0, Discovery packets are still always relayed and Data packets are no longer broadcast (but get sent to a single node).

The advantage of MAM_{Δ} when compared to MAM_0 is that MAM_{Δ} temporarily preserves routes considering the distance to the Mobile-Hub. It was designed with the goal of maintaining shorter routes to the Mobile-Hub, which would imply a smaller energy consumption.

Algorithm 3: MAM_{Δ} - Reactive least-hop route.

Init: bestNodeAddress $\leftarrow$ NULL, bestNodeHops $\leftarrow$ 0, expiry $\leftarrow$ 0
Input: senderAddress, messageHops, messageBody
1: **if** (isDiscoveryMessage(messageBody) == false) **then**
2: **if** (bestNodeAddress != NULL) **then**
3: hops $\leftarrow$ messageHops + 1
4: sendMessage(bestNodeAddress, messageBody, hops)
5: **end if**
6: **return**
7: **end if**
8: bool expired $\leftarrow$ NOW() $>$ expiry
9: **if** (*expired* == true **or** $messageHops < bestNodeHops$) **then**
10: bestNodeAddress $\leftarrow$ senderAddress
11: bestNodeHops $\leftarrow$ messageHops
12: $expiry \leftarrow NOW() + \Delta$
13: **end if**
14: bluetoothMeshRelay(senderAddress, messageHops, messageBody)

5 SIMULATION

The simulations rely on OMNET++ INET framework v5.6.1, an open-source model suite for simulating wired, wireless and mobile networks. For the best of our knowledge, the INET framework lacks any official implementation of the BTMesh standard and BLE. (Kajdocsi et al., 2019) describe a BTMesh partial implementation using the OMNET++ framework,

however, the authors state that their simulation model made it impossible to evaluate a network with more than 30 nodes, and also it did not contain one of BTMesh's core features, Friend Nodes.

The goal of this work's simulations is to evaluate the feasibility and the performance of the proposed variants of relay algorithms in a data collection scenario with one Mobile-Hub - initially with 50 fixed nodes and then a configurable number of nodes.

This work involved creating a model that supported those initial requirements (50 nodes and BTMesh Friendship), and this was achieved by using INET's IEEE 802.15.4 model as a base, to be used across all simulations. This standard was chosen as it defines low-rate wireless personal area networks (LR-WPANs) like ZigBee, which has similar features to BLE. This work's simulation model has the following characteristics that make it similar to BTMesh:

(a) maximum transmission range = 100 meters

(b) transmission rate = 1 Mbps (BTMesh's transmission rate)

(c) nodes can be configured as Low Power or Relay+Friend

The following metrics were collected and used to compare the different Relay Algorithms described in section 4:

- End-to-end delay (ms): the elapsed time in milliseconds from the moment a packet is sent by the source (sensor node) until the packet arrives at the Mobile-Hub.
- Delivery rate (%): the delivery rate of all of the generated data packets to the Mobile-Hub, this means the number of successfully delivered packets to a Mobile-Hub divided by the total number of sent data packets, multiplied by 100.
- Mobile-Hub received packets (bytes): the amount in bytes of received packets. The charts distinguish between unique and repeated data.
- Energy Draw (Joules): the amount of energy that was drawn, by all of the network nodes.

BLE advertising packets, the type of packet used by BTMesh, only implement a simple collision avoidance mechanism. It changes the advertising channels sequentially and also has a random delay between 0 and 10ms for consecutive sends on the same channel, according to Bluetooth Core v5.0 specification (Bluetooth SIG, 2016). On Bluetooth Core v5.1 specification (Bluetooth SIG, 2019), collision avoidance is slightly improved by allowing advertising channels to be chosen at random instead of sequentially.

This work's simulations account for the possibility of packets being lost, with a radio interference model

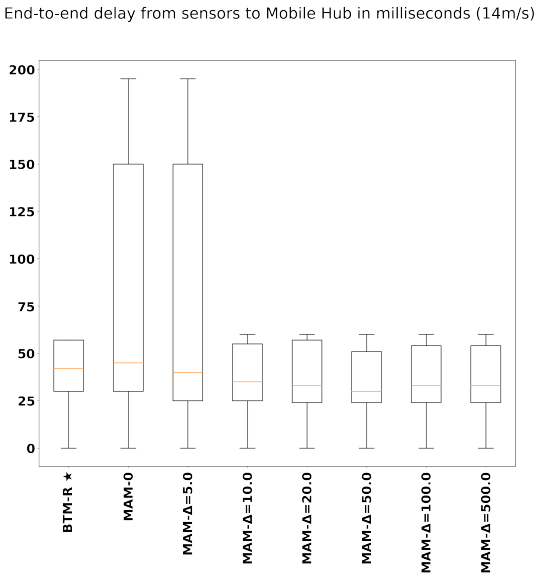

Figure 1: Average delay comparison between BTM-R and MAM relay with least-hop route.

and CSMA/CA simulation as well as a Mobile-Hub movement model. The radio layer was imported from INET's 802.15.4 model, which uses CSMA/CA not BTMesh's simpler collision avoidance approach. The movement model was imported from INET, called *CircleMobility* (Varga, 2020), in which the node simply moves around a circle at a fixed speed. For this work's 50 node-simulation, a fixed radius of 400 meters and a constant speed of 14 m/s were used.

For the MAM_{Δ} algorithm, the simulations varied the algorithm's Δ parameter.

Varying execution time was not very significant in the context of this work, since it is comparing relay algorithms. The execution time only needed to be big enough for the Mobile-Hub to visit some of the nodes and for some of the routes be overridden according to the Δ parameter.

Each simulation was run for 200 seconds, which is the default for OMNET++'s simulations. The network was composed of 13 LPNs and 37 FNs (with all FNs being Relay Nodes), and a single Mobile-Hub that circled around the mesh nodes. This map was manually generated by the authors, and is a connected network. The following variations values were tested: Delta (in milliseconds) 5, 10, 20, 50, 100, 500

The Mobile-Hub's circular trajectory was across a 400 meters radius, at a constant speed of 14 m/s (equivalent to a quadcopter), and covered only a subset of the network nodes. The Mobile-Hub connected to 10 Relay Nodes (20% of all network nodes).

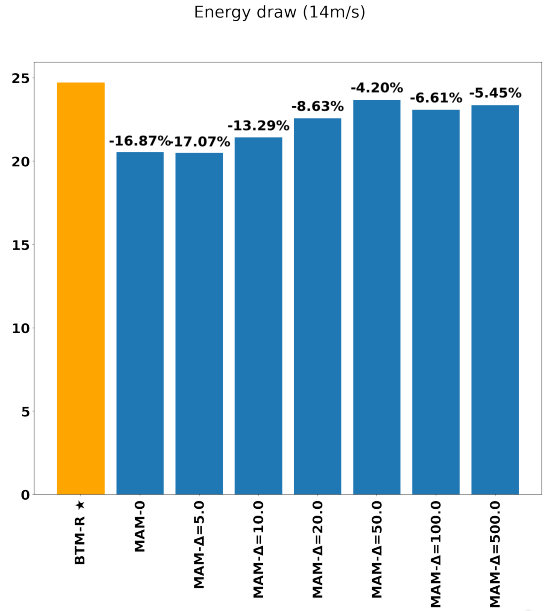

Figure 2: Energy Draw (Joules).

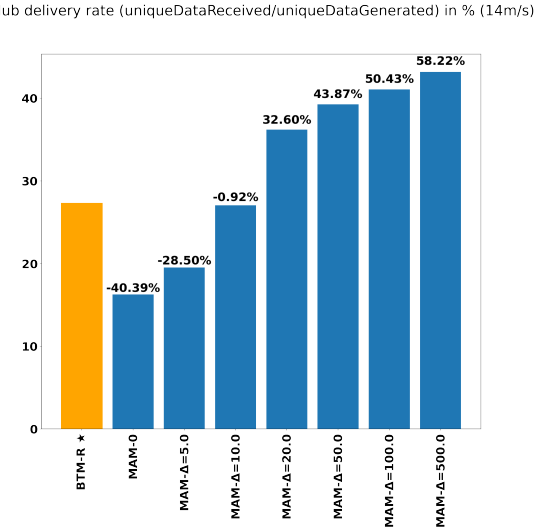

Figure 3: Delivery Rate (%).

6 RESULTS

This section presents the four evaluated metrics in the next subsections.

6.1 End-to-End (Sensors to Sink) Delay

Figure 1 shows the end-to-end delay (from the moment data packets are sent by the source node until they reach the Mobile-Hub) in milliseconds. The chart shows the results, presented as box plots, for BTM-R, MAM_0, and MAM_Δ with varied Δ values. Each box plot representing the data includes the 25% quartile (Q1), median (marked in red), and the 75% quartile (Q3). Outliers have been omitted to facilitate visualization. Higher values indicate that the delay was greater, which means that messages took more time to be delivered to the Mobile-Hub.

BTM-R's median was of 42ms while MAM_0's median was of 45ms. However, all of MAM_Δ's medians were lower than BTM-R's, with the best one being 30ms. Those results indicate that MAM_0 performs worse in terms of end-to-end delay when compared to BTM-R, and that the MAM_Δ algorithm performed better than BTM-R and MAM_0.

6.2 Energy Draw

Figure 2 presents the energy draw of all Mesh nodes in Joules. The values were aggregated as a sum, in which the Mobile-Hub energy draw was neglected, and are displayed on a bar chart. The horizontal axis presents each relay algorithm that was used, and the vertical axis contains the energy draw values in Joules. Higher values indicate that more energy was consumed, however, this is not necessarily an indicator of worse overall performance since more messages could have been sent or the simulation presented a higher packet delivery rate. On each alternate algorithm bar, there is a percentage indicating the percentage comparison between each value and BTM-R's.

The results (Figure 2) show that MAM_0 energy draw was as low as 16.87% less than BTM-R's. The chart indicates that the lowest energy draw for the tested scenarios was with the MAM_Δ algorithm with Δ = 5, 17.07% less than BTM-R. For MAM_Δ with Δ=50, most energy was consumed among the alternative algorithms, 4.2% less than BTM-R. Those results indicate that all the proposed alternatives consumed less energy than BTM-R; however, the energy consumption varied according to the algorithm's Δ parameter.

6.3 Delivery Rate

Figure 3 presents the delivery rate (to the Mobile-Hub) of all Mesh nodes data packets, in percentage. The values consider the amount of unique data packets received divided by the amount of unique data generated by each sensor node. The horizontal axis presents each relay algorithm that was used, and the vertical axis contains the delivery rate percentage values. Higher values indicate that more unique messages were delivered successfully to the Mobile-Hub. On each alternate algorithm bar, there is a percentage indicating the percentage comparison between each value and BTM-R's.

The results (Figure 3) show that MAM_0 delivery rate was 16.26%, which is 40.39% lower than BTM-R's 27.29% rate, and it was the lowest delivery rate among all others tested scenarios. For MAM_Δ with Δ=500, the delivery rate was 43.18%, the highest among the tested scenarios, 58.22% greater than BTM-R. Those results indicate that, for the tested scenarios, the proposed alternatives can also achieve higher and lower delivery rates when compared to BTM-R, depending on how the alternative algorithms are parameterized.

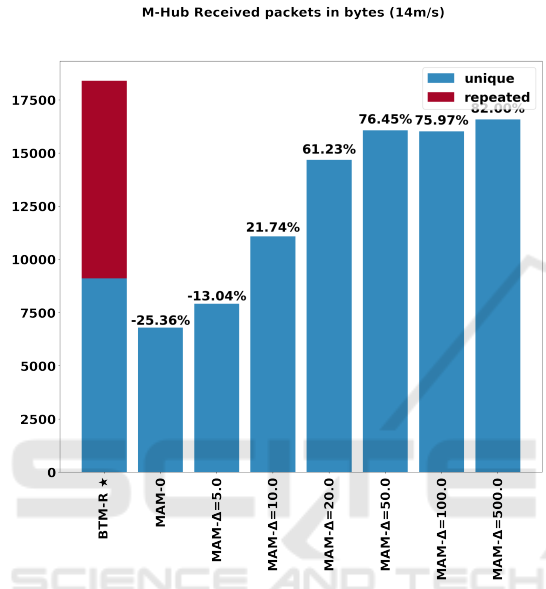

Figure 4: Packets received comparison between BTM-R and MAM_Δ with least-hop route.

6.4 Received Packets

Figure 4 presents the amount of data packets received by the Mobile-Hub, in bytes. The values consider the amount of unique data packets received and display them on a bar chart indicating how many of them were duplicates (if any). The horizontal axis presents each relay algorithm that was used, and the vertical axis contains the amount of data packets in bytes. Higher values indicate that more messages were received by the Mobile-Hub; however, unique values are painted in blue and repeated values in red. On each alternate algorithm bar, there is a percentage indicating the percentage comparison between each unique value portion and BTM-R's unique value portion.

The results (Figure 4) show that on the BTM-R simulation, the Mobile-Hub collected 9.1k unique bytes and a total of 18.4k bytes of data. Thus, in this simulation, 50.5% of the collected data were duplicate packets. MAM_0 received 6.79k bytes of unique data packets, which is 25.36% lower than BTM-R's, and it was the algorithm with the lowest amount of unique data received among all other tested scenarios. For MAM_Δ, with Δ=5, the Mobile-Hub collected 7.92k bytes of unique data, 13.04% less than BTM-R. For all other tested Δ values, results presented a higher amount of unique data packets collected when compared to BTM-R. With Δ=500, the amount of unique data received was 16.57k bytes, the highest among the simulations, 82% greater than BTM-R. Those results indicate that the proposed alternatives can achieve higher and lower amounts of unique data packets received by the Mobile-Hub when compared to BTM-R, depending on how the alternative algorithms are configured. Also, it indicated that MAM_0 and all tested MAM_Δ algorithms did not result in the delivery of duplicated data packets to the Mobile-Hub.

6.5 Analysis and Tradeoffs

Compared to BTM-R, MAM_0 consumed less energy in the tested scenarios (-16.87%) and lower end-to-end delay in most cases. However, delivery rate (Figure 3) was lower compared to BTM-R (-40.39%). Also, the received packets in bytes values were lower than BTM-R's (-25.36%). This indicates that, overall, BTM-R outperforms MAM_0.

The MAM_Δ algorithm was tested with different Δ values. For Δ=5, the delivery rate was lower than BTM-R's in the tested scenarios (-28.50%). The received packets in bytes were 13.04% lower than BTM-R. The energy draw was lower than BTM-R's in the tested scenarios (-17.07%). This indicates that, in most cases, BTM-R outperforms MAM_Δ with Δ=5.

For Δ=10, the delivery rate was lower than BTM-R's (-0.92%). The received packets in bytes were higher than BTM-R in the tested scenarios (+21.74%). The energy draw was lower than BTM-R's in the tested scenarios (-13.29%). This indicates that MAM_Δ with Δ=10 outperforms BTM-R in terms of amount of data collected and energy draw, however, performs worse regarding delivery rate.

For higher Δ values ($>= 20.0$), MAM_Δ presented delivery rates that were higher than BTM-R in the tested scenarios. The highest Δ value that was simulated (Δ=500) presented the highest delivery rates: 43.18% (a 58.22% increase compared to BTM-R's rate). In all tested cases, the energy draw was lower compared to BTM-R. The scenario that consumed most energy (Δ=50) represents only a 4.20% energy draw decrease compared to BTM-R, with a 43.87% delivery rate increase, and a 76.45% received packets in bytes increase. However, in some cases in which the energy draw was lower, the delivery rate

and amount of unique received packets in bytes was also lower (MAM_0 and $MAM_{\Delta=5}$).

The results indicate that, with the correct tuning (Δ parameter), MAM_Δ may achieve a significantly better performance compared to MAM_0 and BTM-R in terms of unique received packets and delivery rate, as well as energy efficiency (when we consider the amount of energy drawn proportionally to the higher delivery rates and higher unique data packets received).

7 CONCLUSION AND FUTURE WORK

This work proposed two alternative approaches to relaying messages in BTMesh networks to a mobile sink node named Mobile-Hub. The proposed approaches were implemented and evaluated with the BTMesh standard model on a simulator (OMNET++ INET framework) by executing multiple simulations to collect the desired metrics: energy draw, Mobile-Hub data delivery rate, Mobile-Hub amount of data received, and end-to-end delay (time elapsed from the sensor node data being sent to the network until it reaches the Mobile-Hub).

The preliminary results show that one of the proposed relay algorithms, MAM_Δ, achieved betters results in all of the evaluated metrics when compared to BTMesh's default relay algorithm (BTM-R).

Extending the MAM_Δ algorithm to handle multiple Mobile-Hubs and heterogeneous data collection by type (e.g., multiple Mobile-Hubs, that subscribe to different data types) should be an exciting path to explore as a ramification of this work.

Experimenting with different mobility types, as well as running simulations with different network topologies varying the number of nodes as well as FN/LPN densities should also bring interesting results and discussions. Field tests using microcontrollers with sensors and a quadcopter as Mobile-Hub are also in the roadmap for this research.

ACKNOWLEDGEMENTS

This study was financed in part by AFOSR grant FA9550-20-1-0285.

REFERENCES

Baert, M., Rossey, J., Shahid, A., and Hoebeke, J. (2018). The Bluetooth Mesh Standard: An Overview and Experimental Evaluation. *Sensors*, 18(8):2409.

Bluetooth SIG, I. (2016). Bluetooth core specification version 5.0.

Bluetooth SIG, I. (2018). An intro to bluetooth mesh part 2. URL: *https://www.bluetooth.com/blog/an-intro-to-bluetooth-mesh-part2/*.

Bluetooth SIG, I. (2019). Bluetooth core specification version 5.1 feature overview.

Cheng, L., Niu, J., Luo, C., Shu, L., Kong, L., Zhao, Z., and Gu, Y. (2018). Towards minimum-delay and energy-efficient flooding in low-duty-cycle wireless sensor networks. *Computer Networks*, 134:66–77.

Djedouboum, A., ARI, A., Gueroui, A., Mohamadou, A., and Aliouat, Z. (2018). Big data collection in large-scale wireless sensor networks. *Sensors*, 18.

Giacomini, E., D'Alterio, F., Lacava, A., and Cuomo, F. (2020). Blues: A self-organizing ble mesh-network paradigm for iot environments. In *2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, pages 409–414.

Hansen, E. A. J., Nielsen, M. H., Serup, D. E., Williams, R. J., Madsen, T. K., and Abildgren, R. (2018). On relay selection approaches in bluetooth mesh networks. In *10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 1–5.

Kajdocsi, L., Dörömbözi, A., and Kovács, J. (2019). Development of bluetooth mesh core stack using omnet++. In *2019 IEEE 17th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 23–28.

Leon, E. D. and Nabi, M. (2020). An experimental performance evaluation of bluetooth mesh technology for monitoring applications. In *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6.

Ma, X., Zhang, P., Li, X., Tang, W., Wei, J., and Theel, O. (2018). DeCoT: A dependable concurrent transmission-based protocol for wireless sensor networks. *IEEE Access*, 6:73130–73146.

Ma, X., Zhang, P., Liu, Y., Boano, C. A., Kim, H. S., Wei, J., and Huang, J. (2020). Harmony: Saving Concurrent Transmissions from Harsh RF Interference. *Proceedings - IEEE INFOCOM*, 2020-July:1024–1033.

Mager, F. and Zimmerling, M. (2016). Mixer. *PCI-Paint and Coatings Industry*, 2016(AUGUST).

Varga, A. (2020). Circlemobility - inet framework. URL: *https://doc.omnetpp.org/inet/api-current/ neddoc/inet.mobility.single.CircleMobility.html*.