A Routing Protocol for LoRA Mesh Networks

Daniel Lundell* ada10dlu@student.lu.se
Anders Hedberg* anders.hedberg@sensefarm.com
Christian Nyberg‡ christian.nyberg@eit.lth.se
Emma Fitzgerald† emma.fitzgerald@eit.lth.se

*Sensefarm AB
Mobilvgen 10
SE-223 62 Lund
Sweden

†Department of Electrical and
Information Technology
Lund University
SE-221 00 Lund
Sweden

***Abstract*—A limitation of current LoRa networks is their single-hop nature. This causes difficulties in areas with poor Internet access, such as remote rural areas, or challenging radio environments, for example in metropolitan areas, as the LoRa gateway must be placed at a location with backhaul access to the network server, but must nonetheless be reachable by all end devices. To facilitate these applications, we present a new routing protocol to enable mesh networking with LoRa, allowing for multihop networking between gateways to extend coverage. Our protocol is tailored specifically to the requirements of LoRa networks. We have developed a proof-of-concept implementation of the protocol and have shown its effectiveness in both laboratory tests and a field trial in a real-world LoRa deployment.**

I. Introduction

LoRaWAN [1] is a Low Power Wide Area Network (LPWAN) specification designed to provide communication at low data rates, combined with long range and low energy usage [2]. LoRaWAN targets key requirements of the Internet of Things such as secure, bidirectional communication, mobility, and localization services, and is maintained by the LoRa Alliance [3]. LoRa technologies are divided into two sub-technologies that are tightly knitted together: LoRa and LoRaWAN, where LoRa is the radio technology used and LoRaWAN is the accompanying link layer protocol.

LoRa gateways act as transparent bridges from end devices to a network server. End devices use single-hop radio communication with gateways, which then connect to the network server via IP connections, for example Ethernet, 3G, or WiFi. In agricultural application scenarios in rural areas, communication between the gateways and network server cannot always be guaranteed, but may be required. It can also be difficult to provide coverage to large areas.

These are problems that Sensefarm, a company working in the area of IoT for agriculture, have often encountered in concrete use cases from their clients — farms and other agricultural enterprises. An example use case could be a farm where Internet connectivity is only available in a central office building, but sensors may be placed in fields distant from that building. In such cases, for example due to attenuation from passing through walls, the transmission range of the gateway may only be some hundreds of metres or perhaps up to a kilometre, not the tens of kilometres theoretically possible with LoRa. Another use case, outside of the agricultural context, is a metropolitan network, where it is often difficult to place gateways at the height required for good coverage, and they must instead be placed at street level, reducing the achievable range. This is a problem we have observed in the deployment of the Lund Open City Sensor Network [4]. In such cases, there is a need to connect repeaters to the network to extend the range. For this, an extensive mesh network over many hops is not required, only a few hops to extend coverage, and these relay nodes will typically be static, with only the end devices mobile. However, multihop relaying of data is not supported by existing LoRa technology.

978-1-5386-4725-7/18/$32.00

In this paper, we present a new protocol to provide communication and routing between LoRa gateways in order to provide coverage in remote areas. Our protocol is based on the Hybrid Wireless Mesh Protocol (HWMP) and Ad-hoc On-Demand Distance Vector Routing (AODV), adapted to the demands of LoRa networks and devices. We have developed a proof-of-concept implementation of our protocol and demonstrated its effectiveness in both laboratory tests as well as a field trial in an existing commercial LoRa deployment. This paper is based on [5].

The remainder of this paper is organized as follows. Section II will detail the related work in this area. In Section III, we explain our routing protocol, and in Section IV, we present the results of our performance evaluation of the protocol. Finally, Section V concludes this paper.

II. Related Work

There is a large body of existing work on routing for multihop wireless networks, in particular for networks with energy constraints and dynamic topology, as in our case. For many agricultural IoT applications, nodes are not mobile, so the topology does not change rapidly, however connections may be unreliable and nodes may leave the network due to a loss of power.

Routing protocols for such networks consist of five core components — route discovery, route selection, route maintenance, data forwarding, and route representation and metric — along with multiple auxiliary components [6]. Mobile ad-hoc routing protocols are divided into three categories, based on the network topology information used for route discovery:

proactive, reactive or hybrid [7]. Within each of these three categories, there is a multitude of protocols [8], and as such, a comprehensive review cannot be provided here. We will instead examine a representative selection of protocols that are commonly used and referenced in the literature. In addition to the protocols we describe below, there are many derivatives with their roots in one of these protocols [6], [7], [9].

Optimized Link State Routing Protocol (OLSR) [10] is a proactive protocol that adapts link state routing for use in mobile ad-hoc networks (MANETs) by use of multipoint relays. Each nodes selects a set of its single-hop neighbours to act as relays, such that all two-hop neighbours are reachable through at least one relay. Relay nodes then maintain and share topology information, thus limiting the communications overhead as only nodes chosen as relays rebroadcast this information. OLSR is well-suited to large and dense networks with random and sporadic traffic. However, LoRa networks for agricultural IoT will typically be relatively small, with only a few gateways needed to cover even a large area. As such, the added overhead of choosing relays and updating topology information is unnecessary in our case.

Destination-Sequenced Distance Vector Routing (DSDV) [11] instead adapts distance vector routing to MANETs. Here, the key difference to standard distance vector routing as used in fixed link networks is that a sequence number field is added to the routing table. This allows nodes to selectively update their routing tables only when receiving a DSDV packet with a higher sequence number than the information the node already has. While DSDV thus has lower control overhead than OLSR [12], continual updates are nonetheless unnecessary for networks with static nodes, as in a typical LoRa rural deployment scenario.

Since reactive protocols require sharing of topology information only when routes fail or a new route needs to be established, they allow for a reduced control overhead, and thus energy cost, in comparison to proactive protocols [13]. Perhaps the best-known reactive routing protocol for MANETs is Ad-hoc On-Demand Distance Vector Routing [14]. In AODV, a node initiates route discovery by flooding the network with Route Request (RREQ) packets, and each node that receives the RREQ packet stores information on the source, destination, and the node it received the packet from. This information is then used to create a reverse path, using a Route Reply (RREP) packet sent from the destination back to the source. Upon route failure, for example if a node moves too far from its previous position, a Route Error (RERR) packet is used to notify all affected nodes, which may then prompt a new route discovery. AODV is designed for tens to thousands of nodes, and can handle a variety of mobility and traffic rates.

Dynamic Source Routing (DSR) [15] is another reactive protocol somewhat similar to AODV, in that it also uses RREQ flooding to perform route discovery. However, in DSR, a list of hops from source to destination is collected in the RREQ packet as it traverses the network. The source node may thus receive several RREP packets with different routes to the destination. It then chooses among the available routes based on the route metric, but also caches the unused routes, which can then be used in case of route failure, saving a further expensive route discovery process. DSR is designed for networks of up to 200 nodes, with potentially high mobility.

While these reactive protocols provide reduced overhead as compared with proactive protocols, this comes at the cost of significantly higher delays in sending data when a new route needs to be discovered [13]. This may be unacceptable in applications where alarm messages need to be received within a certain time. Another concern is that, according to the LoRa protocol, end devices (depending on device class) open receive windows at specified times, and can only receive downlink data during these windows. If the network is too large, route discovery may take too long, resulting in a response from the server coming too late to be received by the end device.

One potential solution for balancing overhead with route discovery delay is a hybrid routing approach. Zone Routing Protocol (ZRP) [16] divides the network into zones around each node, defined in terms of the number of hops from the node. Within each zone, nodes use neighbour discovery to find other connected nodes, and a proactive protocol, Intrazone Routing Protocol, is used. However, between zones, a reactive protocol, Interzone Routing Protocol, is used instead. Such an approach is primarily useful in large networks, however in the use case we consider, the use of zones to divide up the network is not necessarily and will likely simply add extra overhead for little or no benefit.

On the other hand, a hybrid protocol that is more applicable to agricultural LoRa networks can be found in Hybrid Wireless Mesh Protocol [17], used in IEEE 802.11s mesh networks [18]. HWMP is based on a combination of AODV and tree-based routing [19], and can be used in either on-demand or proactive mode. Proactive mode requires that a root node be configured within the network, which is suited to cases where a single LoRa gateway has a backbone Internet connection, while other gateways are used as relays to extend coverage to a larger area. In this mode, the root node periodically sends routing and metric information down the network tree, allowing each node to learn a route to the root node. Because this information sharing is restricted to following the tree structure, it requires significantly less overhead than other proactive protocols that share topology information homogeneously throughout the network. In this work, we use HWMP as the basis for our routing protocol, although substantial changes were required to adapt the standard HWMP to LoRa-based networks.

Aside from control overhead and route discovery delay, performance when sending data traffic should also be considered when designing a routing protocol. In [13], the throughout and average delay for DSR, AODV and ZRP were compared using simulations in NS-2. AODV achieved the highest throughput, regardless of the number of nodes in the network. However, AODV and ZRP were shown to have higher average delay than DSR, due to the route caching used in DSR. Similar results were obtained in [20], although here DSDV was shown to outperform AODV for smaller networks. AODV and HWMP were compared in [21], using simulation studies in NS-3, and

HWMP was shown to perform better than AODV in terms of packet delivery, throughput, and end-to-end delay.

In [22] the authors propose a multihop protocol based on LoRa and concurrent transmission (CT), a recently proposed multi-hop protocol. The concept is tested by several proof-of-concept experiments, that show that this concept show good performance indoors, especially where the nodes are close to each other. How it would work outdoors with large distances is not clear.

III. Routing Protocol for LoRa Networks

There are a number of important considerations to take into account when designing a routing protocol for use with LoRa. First and foremost, the protocol must not interfere with current LoRa implementations, in order to maintain compatibility with the large userbase of existing LoRa devices. Further to this, there are some key differences between LoRa and other radio access technologies that affect the design and implementation of routing protocols.

LoRa networks utilize a star-of-stars topology, with end devices connected to gateways, which are in turn connected to a network server. While end devices' transmissions may be received by multiple gateways, downlink transmissions will reach the end device via a single gateway. In a typical MANET, the topology is more homogeneous, though some routing algorithms for mesh networks include various types of hierarchical network structures [9]. When considering multihop routing for LoRa networks, each gateway may be either fitted with a backbone Internet connection, or may be a relay node with only LoRa connections to other gateways and to end devices. This gives two possible routing configurations: a single root gateway with a stable connection to the Internet, or discovery of a route to the nearest (for a given metric) gateway with an Internet connection. The total path length will typically be quite short, as LoRa is a long-range technology and thus only a few hops are required to cover even large areas.

In Europe, LoRa operates on the 868 MHz ISM band, which requires a maximum 1% duty cycle per device. Any routing protocol must thus impose minimal extra traffic. Moreover, each LoRaWAN packet has a header of at least 13 bytes, while the payload can then vary between 59 and 230 bytes, depending on regional regulations. This places a severe restriction on the amount of extra data that can be included in headers to facilitate routing, while still leaving space for payload data. There is also a strict time constraint on the receive windows. After sending an uplink packet, an end device opens receive windows at 1 and 2 second delays for downlink transmissions. This puts an upper limit on the maximum round trip time, including possible route discovery and establishment, in order to provide bidirectional communication between end devices and the server. Each end-device is given a unique DevEUI and DevAddr from the server, and the server is reached through an IP-address. Gateways, however, are not designated any higher-layer addresses, and as such only have MAC addresses. Because of the packet size constraints, it is not feasible to implement IP on top of LoRaWAN, so instead our routing protocol must rely only on the data available in LoRaWAN.

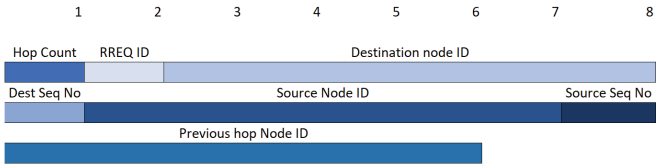

Fig. 1. RREQ packet format

A. Protocol Details

We take HWMP (see Section II) as a starting point for our LoRa routing protocol. HWMP provides low end-to-end delays for data traffic by the use of proactive routing, while at the same time not requiring network-wide flooding of topology information. However, HWMP has a number of features that are superfluous in a LoRa networking scenario, resulting in large protocol headers. AODV, on the other hand, has much more streamlined header information. We will thus combine the overall protocol design of HWMP with some aspects borrowed from AODV.

Our protocol works on a tunneling principle. When a gateway receives a packet from an end device, if it does not itself have an Internet connection, it will route the packet in one of two possible ways. If a root node has been configured in the network, we apply HWMP's proactive mode and the gateway forwards the packet to the route node along the already-established route. If there is no root node, the gateway will look for a route to an Internet-connected gateway. Routing occurs only between gateways and is transparent to both the end devices and the LoRa network server.

The routing protocol messages from AODV and HWMP need to be modified to work with LoRaWAN. Gateways recognize routing packets using the LoRaWAN MAC header, where we set the MType field to 111, reserved in the LoRaWAN specification for non-standard message formats. We use the RFU field, which is reserved for future use, to 001 to identify our routing protocol. The original header is kept so that the packet can be recognized as a LoRaWAN packet on the network. After the LoRaWAN header, an identifier is added to indicate which type of packet follows: RREQ, RREP, RERR, or a data packet. Gateways within the network use their 8-byte MAC addresses, also called node ID in the following, to identify themselves. Since end devices do not participate in routing, we reduce the needed overhead by using the 4-byte Device Address to identify end devices.

1) Packet Formats: The RREQ packet format is shown in Figure 1. It contains the following fields: hop count, RREQ ID, destination ID, destination sequence number, source node ID, source sequence number, and metric. All other fields present in AODV have been removed to reduce the header size. The previous hop field contains the node ID of the node the packet was received from. It is not present in AODV, but has been added to compensate for the lack of an IP layer. For

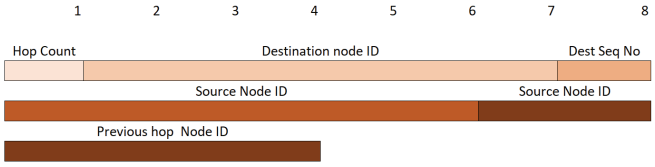

Fig. 2. RREP packet format

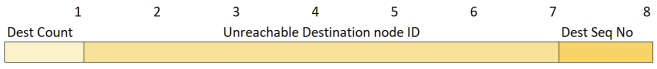

Fig. 3. RRER packet format

RREQ, the destination node ID is set to all ones to indicate a broadcast, and any gateway can then answer if it has an Internet connection.

The RREP packet format is shown in Figure 2. Again, a number of fields have been removed from AODV to reduce overhead, and the previous hop field has been added. The rest of the fields are similar to those for RREQ packets. The RRER packet format, shown in Figure 3, is largely unchanged compared to AODV; although some unneeded fields have been removed. For data packets (Figure 4), additional header fields for source and destination node IDs have been added.

2) Routing and Device Tables: Each gateway maintains a routing table containing five fields: destination, next hop, destination sequence number, and hop count, which we use as the routing metric. The routing table is updated when information about a route with a higher destination sequence number or lower hop count is received.

Each time a gateway receives an uplink message from an end device it stores the device address and destination node ID in a device table. Later, when the gateway receives a downlink message, it checks in the device table to see if the destination device is reachable via a direct LoRaWAN connection. If not, the downlink message is forwarded to the next gateway. Downlink messages in LoRaWAN always use (one of) the same gateway(s) that transported the corresponding uplink message. Thus, the next hop for downlink messages can be recorded in the device table as uplink messages are processed. For each uplink message received from another gateway, the device address is written to the device table, with the sending gateway's node ID as the corresponding destination node ID.

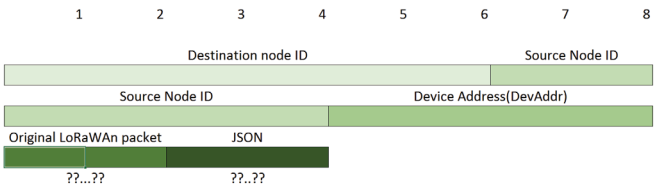

Fig. 4. Data packet format, "??…??" indicates variable field size

B. Protocol Operation

The protocol operates according to the following steps. First, an end-device wakes up and transmits a message. It then awaits the opening of its first receive window. The gateway is constantly listening and receives the message from the end-device. If root mode is used for routing, the gateway then checks if it has a valid path to the root node, and if so, it forwards the packet. If no such path exists or root mode is not used, the gateway performs route discovery. To do this, it sends out a RREQ message with the destination address set to broadcast and awaits a RREP message. When this is received, the gateway stores the route in its routing table, and the device address of the end device in its device table. It then encapsulates the data packet and forwards it.

The next gateway in the network receives the packet and checks if it is the destination. If not, it checks its routing table for a valid route. If it has a valid route, it updates the packet's source node ID field and forwards it, otherwise route discovery is performed as above. If, on the other hand, this gateway is the destination, it instead does the following. First, it records the device address and destination node ID in its device table. It then decapsulates the data packet to recover the original LoRaWAN message, sends this to the server, and awaits a response. If a response is received from the server the gateway performs a lookup in the device table to find the corresponding destination address. The device table entry is erased after the lookup, or if no response is received within a time limit. Next, the gateway encapsulates the data packet along with the JSON data from the original packet, and checks the routing table for a valid route to the destination. If none is found, route discovery is performed. Finally, the gateway forwards the packet towards the destination.

Each subsequent gateways along the downlink path checks if it is the destination, and if not, forwards the packet after updating the previous hop field. If it is the destination, it erases the device entry in the device table, and decapsulates the data packet into a LoRaWAN-compliant packet before transmitting it to the end-device.

Given the restrictions on duty cycle and packet length in LoRa systems, it is important to analyse the overhead of the routing protocol. In terms of control packets sent, in addition to the data packet initially sent from the end device, each gateway the packet must traverse adds two packets, RREQ and RREP. These add a total of 44 extra bytes that must be transmitted per gateway. For comparison, the LoRaWAN header adds 13 bytes to each data packet. Routing thus requires a significant increase in the control overhead per hop, however this is mitigated by the fact that LoRa networks will typically have a low network diameter in hops, due to the long range achievable for each hop.

IV. Protocol Evaluation

We tested our routing protocol in laboratory experiments with a linear topology, with a varying number of hops, as well as in a field experiment at the premises of one of Sensefarm's customers. The purpose of the tests was to measure some of

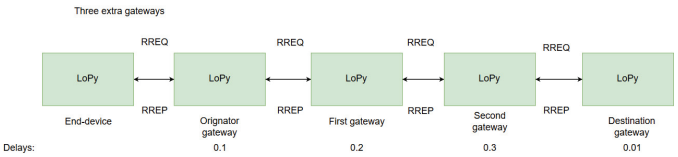

Fig. 5. Network topology for the laboratory experiments.

TABLE I
ROUTE CONSTRUCTION TIMES, WITH AND WITHOUT ADDED DELAYS.

Number of hops	Total artificial delay (s)	Average time, with delays (s)	Average time, no delays (s)
1	0.11	0.27	0.16
2	0.51	0.82	0.31
3	1.11	1.58	0.47

the requirements imposed by the LoRaWAN specification. In order to meet the time constrains of the receive windows, time tests were done with and without multihop routing. The protocol implementation used a Pycom LoPy 1.0 [23]. The LoPy is a microcontroller that supports LoRa, WiFi and Bluetooth and can also act as a "nano-gateway", meaning that it does not have a full implementation of the LoRaWAN specification. The specification requires a gateway to be able to listen on at least three different channels simultaneously, whereas the LoPy uses a single channel [1]. However, it has been certified as an end-device by the LoRa-Alliance [24]. The LoPy runs MicroPython [25], a C implementation of the Python 3 programming language for microcontrollers.

For the laboratory tests five LoPy units were used, connecting in a daisy chain topology as shown in Figure 5. One LoPy acted as an end-device, and another acted as the final gateway connected to the server. The three intervening LoPy devices acted as intermediate gateways performing routing. The Open Source LoRa server [26] was used as the receiving server. Because of the LoPy's single channel restriction, an artificial delay of 0.1s was added at each gateway to prevent collisions. In an implementation with full LoRa gateways, it would be possible to remove these delays provided appropriate channel assignments were given to neighbouring gateways.

Route construction was performed with one, two, and three intermediate gateways, and 200 test runs were performed for each case. Gateways began each test with empty routing tables. In Table I numerical results for the recorded route construction times are shown, along with the times after the subtraction of the artificial delays. The 95 % confidence interval of the results are in all cases less than 0.1 % of the mean. The route construction time shows a linear increase with the number of intermediate gateways added.

A key concern is the feasibility of downlink transmission from the server to the end device, which must occur in time for at least one of the end device's receive windows. As can be seen in Table I, without our added delays, the total time added for route discovery comfortably allows for data transmission to occur and a downlink reply to be received in time for the receive windows at one and two seconds after the end device

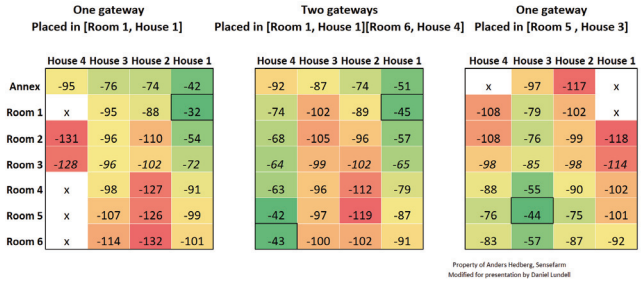

Fig. 6. Field test location with RSSI shown in dB for each section. Gateway locations are outlined in black, and sections with no coverage are shown in white and marked with an x.

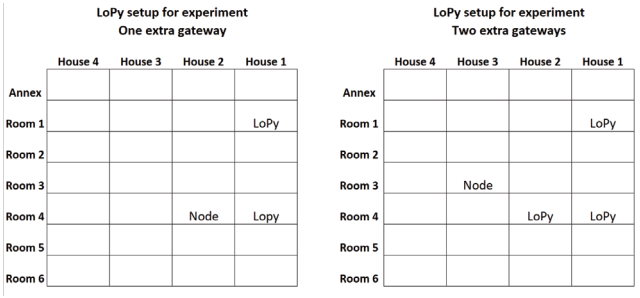

Fig. 7. Gateway placement during field tests.

first sends its data packet. Even with the extra delay added, it is nonetheless feasible to use the second receive window. However, it is clear that the delay induced by route discovery places a limit on the maximum number of hops a route may have and still allow downlink transmissions. If no downlink transmission is required, more hops could be added without disruption to the functioning of the network.

We also conducted a field test of a multihop LoRa network using our routing protocol at a Sensefarm customer location. This was a warehouse where tests conducted by Sensefarm had revealed that the existing gateways could not provide full coverage. For these tests, one or two Kerlink Gateways [27] were used. Figure 6 shows the RSSI (in dB) recorded for each section of the warehouse. Sections with no reception are shown in white and marked with an x, and sections where gateways were located are outlined in black.

We performed the same tests as in our laboratory experiments, however due to time constraints, only one and two intermediate gateways could be tested, and only three test runs could be carried out for each case. The implementation used here was an older version than that used to obtain the results

TABLE II
ROUTE CONSTRUCTION TIMES FOR THE FIELD TEST. TIMES SHOWN IN SECONDS.

Number of intermediate gateways	1	2
Test 1	0.337943	0.889042
Test 2	0.338961	0.887943
Test 3	0.337911	0.880154
Average	0.338271	0.889046

TABLE III
CONSTRUCTION TIME COMPARISONS IN EXPERIMENT

Number of hops	Total artificial delay (s)	Average time, with delays (s)	Average time, no delays (s)
1	0.11	0.33	0.22
2	0.51	0.89	0.38

for the laboratory tests, but with largely the same protocol functionality. Three LoPy devices were placed as shown in Figure 7, and route construction was performed. The time taken for route construction in each test is shown in Table II, and the times without our added delays are shown in Table III. While we were only able to perform a limited number of tests, we were nonetheless able to demonstrate correct operation of our routing protocol in a real world scenario, and the route construction times were similar to those measured in our laboratory experiments.

V. CONCLUSION

In this paper, we have presented a new routing protocol, based on AODV and HWMP, to provide multihop transmission in LoRa networks. There remains, however, work to be done to provide a fully-featured protocol. A number of features present in AODV and HWMP to deal with route failure, such as route errors, route timers, and acknowledgements, have not yet been implemented in our protocol. Moreover, our experiments presented here provide only an initial proof of concept of the feasibility of multihop routing in LoRa, and more testing is required to comprehensively evaluate the performance of our protocol. Downlink transmission has also not yet been tested, although it is provided for in our protocol design. Security is a further concern that has not yet been addressed. Although our protocol preserves the end-to-end encryption of data provided by LoRa, at present it is vulnerable to some types of attacks, for instance a malicious gateway advertising a false route. These issues need to be addressed in future protocol versions.

Nonetheless, we have demonstrated the effectiveness of multihop routing for LoRa systems, including in a real world scenario where we were able to extend the network to areas that previously had poor or no coverage. Moreover, our routing protocol is fully compatible with the LoRaWAN standard, and is transparent to both end devices and the network server. This is critical for deployment to existing systems, since LoRa systems may have different operators for different network elements, but only the gateways need to be modified to support our protocol. Route construction delays were short enough to allow downlink transmission within the end device receive window times. This work opens up possibilities for use of LoRa technology in a wider range of use cases, particularly in remote or inaccessible areas.

REFERENCES

[1] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent, "LoRa alliance LoRaWAN specification," *LoRaWAN Specification, Release v1.0*, 2015.

[2] LinkLabs, "A comprehensive look at low power, wide area networks for 'internet of things' engineers and decision makers," https://cdn2.hubspot.net/hubfs/427771/LPWAN-Brochure-Interactive.pdf, (Accessed on 2017-04-25).

[3] "Lora alliance," https://www.lora-alliance.org/, (Accessed on 2017-07-05).

[4] "Lund Open City Sensor Network," http://www.futurebylund.se/lund-open-city-sensor-network, (Accessed on 2018-03-05).

[5] D. Lundell, "Ad-hoc network possibilities inside lorawan," Master's thesis, Lund University, 2017.

[6] M. Lee, J. Zheng, X. Hu, H. Juan, C. Zhu, Y. Liu, J. Yoon, and T. Saadawi, "A new taxonomy of routing algorithms for wireless mobile ad hoc networks: The component approach." *IEEE Communications Magazine*, vol. 44, no. 11, pp. 116–123, 2006.

[7] G. Walikar and R. Biradar, "A survey on hybrid routing mechanisms in mobile ad hoc networks." *Journal of Network and Computer Applications*, vol. 77, pp. 48–63, 2017.

[8] I. Ahmad, U. Ashraf, and A. Ghafoor, "A comparative QoS survey of mobile ad hoc network routing protocols." *Journal of the Chinese Institute of Engineers*, vol. 39, no. 5, pp. 585–592, 2016.

[9] E. Alotaibi and B. Mukherjee, "Survey paper: A survey on routing algorithms for wireless ad-hoc and mesh networks." *Computer Networks*, vol. 56, pp. 940–965, 2012.

[10] T. Clausen and P. Jacquet, "Rfc 3626: Optimized link state routing protocol (OLSR)," https://www.rfc-editor.org/rfc/rfc3626.txt, 2003, (Accessed on 2017-04-25).

[11] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *ACM SIGCOMM computer communication review*, vol. 24, no. 4. ACM, 1994, pp. 234–244.

[12] S. Mohapatra and P. Kanungo, "Performance analysis of AODV, DSR, OLSR and DSDV routing protocols using NS2 simulator," *Procedia Engineering*, vol. 30, pp. 69–76, 2012.

[13] P. Kavita and S. Abhishek, "A comprehensive performance analysis of proactive, reactive and hybrid MANETs routing protocols." *International Journal of Computer Science Issues, Vol 8, Iss 6, Pp 432-441 (2011)*, no. 6, p. 432, 2011.

[14] C. Perkins, E. Belding-Royer, and S. Das, "Rfc 3561 - ad hoc on-demand distance vector (AODV) routing," https://tools.ietf.org/html/rfc3561, 2003, (Accessed on 04/25/2017).

[15] D. Johnson, Y.-c. Hu, and D. Maltz, "Rfc 4728 - the dynamic source routing protocol (DSR) for mobile ad-hoc networks for IPv4," https://tools.ietf.org/html/rfc4728, 2007, (Accessed on 2017-04-25).

[16] Z. J. Haas, M. R. Pearlman, and P. Samar, "The zone routing protocol (ZRP) for ad hoc networks," https://tools.ietf.org/html/draft-ietf-manet-zone-zrp-04, 2002, (Accessed on 05/05/2017).

[17] A. Joshi, M. Bahr *et al.*, "Hwmp specification," *IEEE P802*, vol. 11, pp. 802–11, 2006.

[18] G. R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11s: the WLAN mesh standard," *IEEE Wireless Communications*, vol. 17, no. 1, 2010.

[19] L. Borsani, S. Guglielmi, A. Redondi, and M. Cesana, "Tree-based routing protocol for mobile wireless sensor networks," *Eighth International Conference on Wireless On-Demand Network Systems and Services*, p. 164, 2011.

[20] P. Manickam, T. G. Baskar, M. Girija, and D. D. Manimegalai, "Performance comparisons of routing protocols in mobile ad hoc networks," *arXiv preprint arXiv:1103.0658*, 2011.

[21] I. Ibrahim, N. A. Latiff, S. S. Yusof, N. N. A. Malik, and S. S. Ariffin, "Performance comparison of AODV and HWMP routing protocols in wireless mesh networks," *2013 IEEE International RF and Microwave Conference (RFM)*, p. 116, 2013.

[22] Z. G. K. D. S. M. Liao, C.H. and H. Morikawa, "Multi-hop lora networks enabled by concurrent transmission," *IEEE Access*, vol. 5, pp. 21 430–21 446, 2017.

[23] "PyCom LoPy," https://www.pycom.io/wp-content/uploads/2017/06/lopySpecsheetJune.pdf, (Accessed on 2017-06-28).

[24] "LoRa certification test report," https://www.lora-alliance.org/portals/0/CertifiedProducts/pycom_lopy/TERD-WTS-TR-27%20-%20LoRa_Certification_Test_Report_201701.006.pdf, (Accessed on 04/25/2017).

[25] "Micropython — python for microcontrollers," https://micropython.org/, (Accessed on 2017-05-05).

[26] "LoRa app server — open-source LoRaWAN application-server," https://docs.loraserver.io/lora-app-server/, (Accessed on 2017-07-05).

[27] Kerlink, "Wirnet station 868 mhz," http://www.kerlink.fr/en/products/lora-iot-station-2/wirnet-station-868, (Accessed on 2017-03-05).