Long-Lived LoRa: Prolonging the Lifetime of a LoRa Network

Sezana Fahmida[†§], Venkata P Modekurthy[†§], Mahbubur Rahman[‡], Abusayeed Saifullah[§], Marco Brocanelli[§]

[§] Wayne State University and [‡] CUNY Queens College

Abstract—Prolonging the network lifetime is a major consideration in many Internet of Things applications. In this paper, we study maximizing the network lifetime of an energy-harvesting LoRa network. Such a network is characterized by heterogeneous recharging capabilities across the nodes that is not taken into account in existing work. We propose a link-layer protocol to achieve a long-lived LoRa network which dynamically enables the nodes with depleting batteries to exploit the superfluous energy of the neighboring nodes with affluent batteries by letting a depleting node offload its packets to an affluent node. By exploiting the LoRa's capability of adjusting multiple transmission parameters, we enable low-cost offloading by depleting nodes instead of high-cost direct forwarding. Such offloading requires synchronization of wake-up times as well as transmission parameters between the two nodes which also need to be selected dynamically. The proposed protocol addresses these challenges and prolongs the lifetime of a LoRa network through three novel techniques. (1) We propose a lightweight medium access control protocol for peer-to-peer communication to enable packet offloading which circumvents the synchronization overhead between the two nodes. (2) We propose an intuitive heuristic method for effective parameter selections for different modes (conventional vs. offloading). (3) We analyze the energy overhead of offloading and, based on it, the protocol dynamically selects affluent and depleting nodes while ensuring that an affluent node is not overwhelmed by the depleting ones. Simulations in NS-3 as well as real experiments show that our protocol can increase the network lifetime up to 4 times while maintaining the same throughput compared to traditional LoRa network.

I. INTRODUCTION

LoRa (Long Range) is a leading low-power wide-area network (LPWAN) technology that enables low-power (milliwatts) wireless devices to transmit at low data rates (kbps) over long distances (kms) using narrowband (kHz) [1]. It can be deployed for direct communication between numerous end devices (also called *sensor nodes*) and a gateway in many wide-area Internet of Things (IoT) applications including smart agriculture [2], smart city [3], and environmental monitoring [4]. While these end devices are usually powered by traditional single-use batteries, energy-harvesting technologies that exploit the light [5], the vibrations [6], and/or the heat [7] of the environment have emerged as an efficient alternative (e.g., in smart agriculture, environmental monitoring) for providing sustainable energy. The carbon footprint of such energy-harvesting technologies is many times lower than the other energy sources including single-use batteries [8],

[9]. However, relying on energy-harvesting for recharging the sensor nodes can lead to heterogeneous state of charge for the batteries in the network. Thus, it is crucial to regulate the energy consumption of the nodes dynamically to ensure maximum network availability.

Prolonging the network lifetime is a major consideration in most IoT applications. We consider *network lifetime* as the interval starting at any given point in time until the first node depletes its battery. In many IoT applications the lifetime directly implies the availability of the services as failure to collect sensor data from even one node may lead to unmet application requirements. In this paper, we propose the first method to maximize the network lifetime in a generic setup of an energy-harvesting LoRa network [10], [11].

Recently there has been some research focusing on adopting a general energy minimization strategy for all nodes to ensure the energy-efficiency of LoRa networks [12], [13]. However, following the same energy minimization strategy for all nodes may not always be beneficial for prolonging the lifetime in a network where nodes may have heterogeneous recharging capabilities. Insufficient energy generation in a recharge cycle may lead to an unexpected depletion of a node's battery even with the most energy-efficient transmission parameter allocation. Hence, in order to maximize the network lifetime it is required to have an approach that prevents such depletion. For example, a *depleting* node that has not harvested enough energy can drain its battery rapidly if proper action is not taken. However, there may be *affluent* nodes in the network that have superfluous residual energy in their batteries that will remain unused in the current recharge cycle. We propose a link-layer protocol to maximize the network lifetime by dynamically enabling the depleting nodes to exploit the superfluous energy of the neighboring affluent nodes.

In our approach, the superfluous energy of an affluent node is exploited by letting a depleting node *offload* its workload to the former. Namely, if the two nodes are within communication range of each other, a depleting node offloads its packets when needed to an affluent node which then transmits those to the gateway. In a LoRa network, transmission to a close-by node can be made at much less energy by adjusting *multiple* transmission parameters compared to directly forwarding to the gateway. We exploit this feature of the LoRa technology and enable low-cost (in energy) offloading by depleting nodes instead of high-cost direct forwarding. Note that, in this strategy, the overall energy consumption at the affluent node and, possibly, the total energy needed for delivering a packet

---

[†]Co-first-author

to the gateway may be higher than that needed for direct forwarding from the depleting node. However, as counter-intuitive as it may seem, it is very effective in maximizing the lifetime of the network because it prevents the depleting node from rapidly consuming its battery. This idea forms the central thesis of this paper to achieve a long-lived LoRa network.

Enabling packet offloading between nodes in a LoRa network raises three main challenges. First, the energy-efficient offloading of packets requires synchronization of wake-up times between the nodes, which typically introduces additional energy overheads. Second, a successful offloading requires that the nodes be operating on the same transmission parameters. Thus, we need to decide energy-efficient transmission parameters to enable offloading. Third, we need to dynamically select depleting and affluent nodes without overwhelming the latter, while also preventing the rapid battery depletion of the former.

We address the above challenges by developing the proposed link-layer protocol which enables a long-lived LoRa network. It prolongs the lifetime of a LoRa network through three novel techniques. (1) We propose a lightweight Medium Access Control (MAC) protocol for peer-to-peer communication to enable packet offloading which circumvents the synchronization overhead between the two nodes, thereby ensuring low power consumption and reliable packet delivery during offloading. (2) We propose an intuitive heuristic method for parameter selections for different modes (conventional vs. offloading) to ensure that only the required energy for successful transmission is spent in each mode. (3) We analyze the energy overhead for an affluent node when a packet is offloaded to it. Based on this analysis, the protocol dynamically selects affluent and depleting nodes during network operation as well as the duration in such modes so that an affluent node is not overwhelmed by the depleting ones. We have evaluated the proposed protocol through extensive simulations in NS-3 and physical experiments. The results show that it can increase the network lifetime up to 4 times while maintaining the same throughput compared to traditional LoRa network.

In the rest of the paper, Section II describes background and model. Section III describes related work. Section IV describes the protocol to prolong the network lifetime. Section V provides the evaluation results. Section VI concludes the paper.

II. Background and System Model

A. An Overview of LoRa and LoRaWAN

LoRa is a physical layer technology for LPWAN with a communication range of 3-10 miles [1] (depending on the environment). Modulation in LoRa is derived from the *chirp spread spectrum (CSS)*. In CSS, one bit of information or chirp is a signal continuously varying in frequency, often represented as multiple chips. Spreading the frequency of the signal enables the successful reception of a packet at long ranges and low/negative signal-to-noise ratios. The amount of frequency spreading applied to the signal in a chirp is called the *spreading factor* of transmission. LoRa adopts seven orthogonal spreading factors in the range $[6, 12]$ for simultaneous reception on the same channel. Each spreading factor is a binary log-scale representation of the number of chips present in one symbol, i.e., spreading factor 6 represents 64 chips are present in one symbol. In CSS, apart from spreading factors, nodes can be configured to use different bandwidth settings, channel, coding rates, transmission powers.

In North America, LoRa supports sixty-four channels with a maximum bandwidth of 125kHz and eight channels with a maximum bandwidth of 500kHz for uplink communication between a node and a gateway. Additionally, it supports eight downlink channels with a maximum bandwidth of 500kHz. A packet in LoRa communication supports four levels of forward error detection and correction mechanisms called *coding rate*. Coding rate ranges from $\frac{4}{5}$ to $\frac{4}{8}$. Lower coding rates enable better error detection and correction mechanism; however, they increase the packet size and energy consumption.
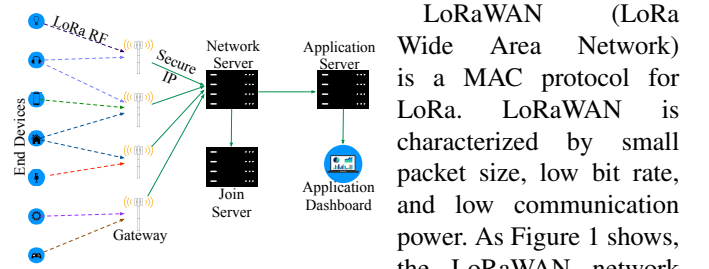

Fig. 1. LoRa Network Architecture

LoRaWAN (LoRa Wide Area Network) is a MAC protocol for LoRa. LoRaWAN is characterized by small packet size, low bit rate, and low communication power. As Figure 1 shows, the LoRaWAN network consists of nodes/end-devices, gateway, network server, application server, and a join server. A *gateway* acts as a relay between wireless communication (with the nodes) and wired communication (with the network server). The *network server* manages the network parameters, security, and application requirements. The *application server* interprets sensor and application data from the sensors. Nodes in the network initiate a join request upon deployment, which is handled by the *join server*.

LoRaWAN supports three classes of operation. For the uplink communication, LoRaWAN adopts pure ALOHA for transmission in all three classes, where a node transmits whenever it has a packet. In *class A*, every transmission precedes two receive time windows for successfully receiving an acknowledgment (ACK). In *class B*, the gateway periodically transmits a beacon for time synchronization. Between two beacons, a node periodically wakes up to receive any packets sent by the gateway. In *class C*, nodes are continuously listening to a packet transmission from the gateway.

B. System Model

The nodes in the network are powered by batteries recharged through energy harvesting devices (e.g., solar panels). The nodes collect and transmit sensor information to the gateway. They typically use class A mode for transmitting their packets to the gateway and sleep for the rest of the time to conserve energy. We denote the transmission parameters such as spreading factor, channel, coding rate, and bandwidth of a node v by SF_v, C_v, CR_v, and BW_v, respectively.

The *line-powered* gateways listen for packets, acknowledge them, and forward them to the network server and application server. In addition to providing security, the network server

stores the location and current battery level of each network node. Because node localization is out of scope for this paper, we assume it is performed at network deployment time through manual configuration or some existing localization techniques for low-power networks [14]. The battery level of each network node is embedded in the existing uplink packet to the network server for timely state of charge updates.

Typically, the nodes periodically sense the state of the system. However, in some cases, the sensed information may not change over time; hence, some nodes may locally determine if the sensed data needs reporting, which leads to aperiodic transmissions with a known minimum inter-arrival time between packets. When a node does not have any packets to transmit for long intervals, it communicates a heartbeat message with the gateway to maintain connectivity. The period of the heartbeat message is set by the network manager.

Transmission Energy Model. The energy consumed for packet transmissions from a node v can be modeled as follows:

$$E_{\text{tx}}(v) = P_{\text{tx}}(PL_v) \times N_{\text{symbols}}(v) \times T_{\text{symbol}}(v) \qquad (1)$$

where $P_{\text{tx}}(PL_v)$ represents the power consumed by the LoRa chip to transmit a packet at power level PL_v, $T_{\text{symbol}}(v)$ represents the time required to transmit one symbol with spreading factor SF_v and bandwidth BW_v, and $N_{\text{symbols}}(v)$ denotes the number of symbols in a packet. The number of symbols in a packet from v including a header and cyclic redundancy check (CRC) code is modeled as follows:

$$N_{\text{symbols}}(v) = preamble_v + 4.25 + 8 +$$
$$\max\left(\left\lceil \frac{8\,payload_v - 4\,SF_v + 24}{SF_v - 2\,DE_v} \right\rceil \frac{1}{CR_v}, 0\right)$$

where $preamble_v$ represents the length of the preamble and $payload_v$ represents the length of the payload. Equation (2) models $T_{\text{symbol}}(v)$ and Equation (3) describes the low data rate optimization mode DE_v:

$$T_{\text{symbol}}(v) = \frac{2^{SF_v}}{BW_v} \qquad (2)$$

$$DE_v = \begin{cases} 1, & \text{if low data rate optimization is enabled} \\ 0, & \text{otherwise} \end{cases} \qquad (3)$$

During a receive window of length $T_{\text{rx}}(v)$ seconds, node v consumes energy as follows:

$$E_{\text{rx}}(v) = P_{\text{rx}}(DE_v) \times T_{\text{rx}}(v) \qquad (4)$$

where $P_{\text{rx}}(DE_v)$ is the power consumed by the LoRa chip.

Recharge Energy Model. Each network node is equipped with an independent green energy source that recharges the node's local battery. Typically, the instantaneous power generated by any green source varies over time but repeats over a fixed time interval of length ζ called a *recharge cycle*. A recent study has shown that the average power generated for one hour interval by a solar panel repeats every 24 hours [15]. Thus, the network server can estimate the energy generated at node v during a recharge cycle as the integral of the estimated instantaneous power ($P_{\text{recharge}}(v)$) over the recharge cycle, as shown in Equation (5).

$$B_v^* = \int_0^\zeta P_{\text{recharge}}(v)dt \qquad (5)$$

When the network is deployed, the batteries are usually fully charged. Typically, the energy harvesting system is conservatively designed to be sized (1) proportionally to the battery size of each node, i.e., the energy stored in a fully-charged battery is no less than B_v^*, and (2) so that the average energy consumed during a recharge cycle is no greater than the average recharged energy during the same period. Thus, we define *battery budget* of a node v during ζ time units as B_v^*, an estimation of the energy generated during that interval. The network server synchronizes the start and end of the recharge cycle to ensure all nodes reset the battery budget at the same time and perpetuate network communication. In this conservative scenario, within an interval of length ζ, neighboring nodes can have different battery budgets due to environmental effects such as shadows that impact the energy generation. Similarly, nodes can have different energy consumption due to the heterogeneity of hardware and/or communication patterns.

Network Lifetime. We calculate the network lifetime as the time interval from the start of the recharge cycle until the first node consumes its battery budget. Although there are other definitions of network lifetime, we choose this definition of network lifetime as it hampers the data collection from at least one node in the present or the future intervals. Given the above considerations, we regulate the nodes' energy consumption in each interval according to the estimated B_v^* energy recharged within each interval ζ. This strategy ensures (1) there is no downtime in the current interval and (2) the batteries' energy level at the end of the current interval is higher than a certain level necessary to ensure continuous operations for the next.

III. RELATED WORK

Due to their ability to overcome the coverage and scalability limitations in traditional short-range wireless sensor network [16], LPWANs have received considerable attention recently [17]–[23]. While many LPWANs have been designed (e.g, LoRa [1], SigFox [24], IQRF [25], RPMA [26], DASH7 [27], Telensa [28], NB-IoT [29], SNOW [30], [31], and LTE Cat M1 [32], [33]), LoRa has gained popularity due to its wide availability. It has been the focus of much research in both academia and industry [34]. A comprehensive review of these works can be found in [35]. Most of these existing works have focused on enhancing throughput [36], [37], latency [38], [39], reliability [21], [40]–[43], or scalability and coverage [22], [44]–[47] of a LoRa network.

Energy efficiency of a LoRa network was studied in [12] using additional hardware and in [13], [48] by minimizing energy consumption at all nodes through transmission parameters selection [13], [48]. However, these approaches are not applicable to maximize the network lifetime of an energy harvesting LoRa network. Insufficient energy generation in a recharge cycle may lead to an unexpected depletion of a node's battery even with the most energy-efficient transmission

parameters. On the contrary, our proposed approach minimizes such depletion, thereby maximizing the network lifetime.

IV. PROPOSED PROTOCOL FOR LONG-LIVED LoRa

For nodes with battery recharging capabilities, minimizing energy consumption is not always beneficial for maximizing the lifetime of the network. For example, if a node is fully charged at time 0 and is expected to generate B^* units of energy by time ζ, then consuming any amount of energy in the range $[0, B^*]$ will still result in a full battery at time ζ. Thus, the network lifetime is not negatively affected as long as it is ensured that a node's energy consumption is no greater than its expected energy generation (or battery budget).

We rely on the above observation to maximize the network lifetime by enabling the nodes to exploit superfluous residual energy of neighboring nodes. This is done through a packet offloading mechanism where a node whose battery budget is insufficient to deliver all subsequent packets directly to the gateway dynamically forwards a subset of its future packets to a neighboring node with an affluent battery budget. The neighboring node forwards these packets to the gateway on behalf of the depleting node. Since we select neighboring nodes that are significantly closer to the node than the gateway, transmission to a neighboring node can be made at much less power by adjusting multiple transmission parameters than transmitting directly to the gateway. Note that, in this strategy, the overall energy consumption at an affluent node increases and the total energy needed for delivering a packet to the gateway may be higher than that required for direct transmission to the gateway from a depleting node. However, it conserves energy at the depleting node. Thus, it helps prolong the network lifetime as long as we select the affluent and depleting nodes dynamically and ensure that the latter are not overloaded.

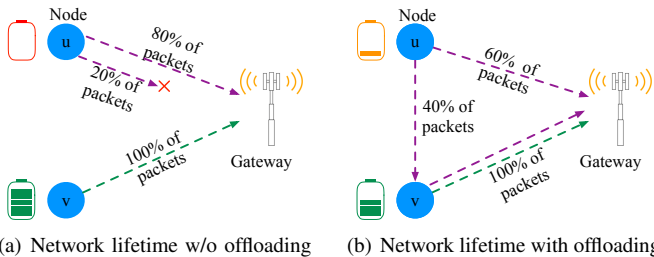(a) Network lifetime w/o offloading   (b) Network lifetime with offloading

Fig. 2. An example of maximizing network lifetime through packet offloading

Fig. 2 shows an example of the proposed offloading mechanism. In this example, two nodes, u and v, transmit packets to the base station. Packet transmission from node v consumes 10% of its battery, while that from node u consumes 100% of its battery for transmitting only 80% of the packets. In the traditional setting, u runs out of charge before the next charging cycle. Assuming u needs 50% energy to transmit a packet to v compared to the base station, offloading 40% of u's transmission to v can ensure u's (and thereby the network) lifetime is longer than the charging window.

Definition 1 (Affluent Node): A node v with an energy budget B_v^* and an estimated energy consumption $B_v(t, \zeta)$

during the interval $[t, \zeta]$, where t is the current time and ζ is the start of next interval, is said to be an affluent node at time t if and only if

$$B_v(t, \zeta) < B_v^*. \tag{6}$$

Definition 2 (Depleting Node): A node v with an energy budget B_v^*, residual energy β_v that needs to be saved for future, and estimated energy consumption $B_v(t, \zeta)$ during the interval $[t, \zeta]$, where t is the current time and ζ is the start of next interval, is a depleting node at time t if and only if

$$B_v(t, \zeta) > B_v^* - \beta_v \tag{7}$$

The value of β_v depends on the estimated energy generation in the next recharge cycle of node v and can be estimated by the network manager during network deployment. At time t, the estimate of energy consumption, battery budget for that interval, and Equations (6) and (7) are used to designate a node as affluent or depleting.

Packet offloading from a depleting node u to an affluent node v in our approach means transferring the responsibilities of transmitting a set of packets to the gateway from u to v. To enable packet offloading, we consider three modes of operation for a node: *offloading*, *lading*, and *conventional*. In the conventional mode, a node transmits packets directly to the gateway. In the lading mode, a node receives packets from a depleting node and forward them to the gateway. The offloading mode enables packet offloading to an affluent node.

Packet offloading increases the energy consumption of an affluent node but does not affect its lifetime when its total energy consumption is no greater than its battery budget. To ensure that the total energy consumption of an affluent node remains within its battery budget, we propose to find (i) the pairing between affluent nodes and depleting nodes that communicate with each other and (ii) an estimate of the operation time in lading and offloading mode for the affluent and depleting node, respectively. Since the solution to an optimal pairing of affluent-depleting nodes and operation time in lading and offloading mode is unknown, we propose a heuristic solution.

The proposed heuristic for finding the affluent-depleting node pair that communicate with each other depends on the node's energy consumption in the lading mode. Energy consumption in the lading mode depends on the link-layer protocol that enables packet offloading. In the following sections, we describe the proposed link-layer protocol that enables packet offloading between the depleting and affluent nodes, transmission parameter selection, energy overhead in the lading mode, and selection of affluent and depleting node pairs that communicate with each other.

A. Enabling Peer-to-Peer Communication for Offloading

Enabling communication between an affluent node and a depleting node requires synchronization between their sleep and wake-up times. Such a synchronization can incur huge energy overheads for both affluent and depleting nodes. To overcome this limitation, we adopt the low-power listening technique used in low-power network. In low-power listening, a node performs clear channel assessment (CCA) for a fixed duration
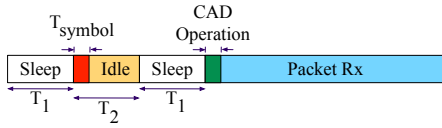
4

Fig. 3. Timing diagram in Lading Mode.

to determine the channel activity. If the node detects channel activity, it wakes up and listens to the packet; otherwise, it sleeps. We propose to adopt a similar approach for the lading mode of LoRa. However, adopting low-power listening for LoRa has the following challenges: (1) using CCA to monitor the start of a packet can lead to several false-negatives since nodes can decode packets below the noise level, and (2) due to long communication range, a node can be in communication range with many other nodes, which can increase the number of false wake-ups in lading mode. We address these limitations by adopting channel activity detection (CAD) for low-power listening. The proposed peer-to-peer communication between an affluent and a depleting node is described in the following sections. First, we describe the communication in the lading mode and then in the offloading mode.

1) Lading Mode for LoRa: In LoRa's lading mode, the affluent node periodically wakes up and performs CAD. Note that most of the existing and commercially available off-the-shelf LoRa communication chips such as SX1276, SX1277, SX1278, and SX1279 implement and support CAD [49]. During CAD, the affluent node receives signals for two symbols and probes for correlation between the received signal and a known preamble to identify any ongoing packet transmissions. If the affluent node detects a packet transmission, it continues listening to the packet transmission; otherwise, it sleeps. Because a node probes for correlation with a known symbol, CAD overcomes the false-negative detection of packets. After receiving a packet, the affluent node forwards the packet to the gateway. Since there is a minimum inter-arrival time between two packets at a node, the affluent node can transmit to the gateway in the interval between two offloading packets without interfering the depleting node's communications. The affluent node can also use this interval to transmit its own packets.

Fig. 3 shows the timing diagram of a node operating in lading mode. An affluent node wakes up once every $T_{\text{CAD}} = T_1 + T_2$ seconds, where T_1 and T_2 are timer interrupts. The node sleeps for T_1 seconds. Within the T_2 seconds, the node performs CAD for T_{symbol} seconds and remains in low-power idle mode for the rest if a packet is not detected (for example, after the first CAD operation in Fig. 3). If the node detects a packet during the CAD, for example, after the second CAD operation in Fig. 3, it wakes up and listens to the packet. On the other hand, in LoRa communication the preamble for all packets is the same. Thus, during CAD an affluent node may not distinguish between transmissions to the affluent node and gateway, which would result in a significantly high number of false wake-ups. To solve this problem, we use reverse I-Q signals for packet transmissions to lading node and regular I-Q signals for packet transmissions to the gateway. During CAD, an affluent node can only listen to regular I-Q or reverse I-Q

but not both, thus minimizing the number of false wake-ups.

2) Offloading Mode for LoRa: The objective of the offloading mode is to minimize the energy consumption at a node, thereby increasing the network lifetime. To meet this requirement, we propose to adopt a lightweight MAC protocol similar to LoRaWAN class A to send a packet. A node transmits a packet whenever it has data to send. If the affluent node acknowledges the packet reception in the next two time windows, it goes back to sleep until it has data to send. Otherwise, it retransmits the packet after a random back-off.

For a depleting node's packet to be successfully received by an affluent node through CAD, the duration of the preamble should be a minimum of $T_1 + 2T_2$. Since each symbol takes $\frac{2^{SF}}{BW}$, the length of the preamble is at least $\frac{(T_1+2T_2)BW}{2^{SF}}$. When $T_1 = T_2 = 4.1ms$ (the smallest feasible value possible for T_1 and T_2), the preamble should be at least 13 symbols long for a spreading factor of 7 and bandwidth of 125kHz.

B. Transmission Parameter Selection

Packet offloading between nodes requires both the affluent node and the depleting node to operate on the same channel, spreading factor, bandwidth, and coding rate. Due to the complex nature of this problem, we leave the optimal problem formulation and analyzing its complexity class as an open problem. In our approach, we develop a heuristic for transmission parameter selection in the conventional mode and offloading-lading mode to enable communication from node to gateway and from depleting node to affluent node, respectively.
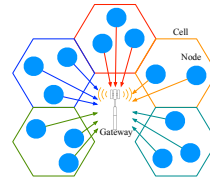

Fig. 4. A channel assignment example (each color represents a different transmission channel).

Channel Assignment. The objective of channel allocation is to ensure potential neighboring affluent and depleting nodes are assigned the same channel. The network server segregates the networks into cells and assigns the same channel to all the nodes within a cell. Furthermore, adjacent cells are assigned different channels to minimize the interference from depleting nodes in neighboring cells. Since neighboring nodes can have different communication patterns and battery budgets, the nodes within the same cell can have both depleting and affluent nodes. Fig. 4 shows an example of our channel allocation. To generate cells, the network server uses location-based segregation of nodes. Initially, the network server creates cells equal to the number of channels. At the end of each recharge cycle, it can further divide one cell into smaller cells if the number of affluent and depleting nodes within one cell is above a certain threshold. The network manager can set this threshold based on required performance improvement in the network lifetime. The same channel assignment is used in all operation modes.

Transmission power. In conventional mode, the nodes use transmission power based on ADR (adaptive data rate) adopted in LoRaWAN. In ADR, the nodes initially use the maximum transmission power to transmit a packet [50]. After every 20 packets received by the gateway, the nodes decrease

5

the transmission power by 3dBm to compute the minimum transmission power that results in a successful reception. We use a similar approach in the lading-offloading mode, where an affluent node starts with a transmission power of $PL_i = P_s + L_0 + 10n\log\frac{d}{d_0}$, where P_s is the receiver sensitivity, $L_0 = 7.7$dBm, the path loss at reference distance $d_0 = 1$m, and $n = 3.76$ is the path loss exponent. An affluent node decreases the transmission power if the signal to noise ratio (SNR) is significantly higher than the receiver sensitivity.

Coding rate. LoRaWAN specifies a coding rate of 4/5 to be used in the US915 band to limit the maximum dwell time in each channel [51]. We use a coding rate of 4/5 in both the conventional and offloading-lading mode.

Spreading factor. In the conventional mode, the nodes use spreading factors in the range [9,10] for the US band and [9, 12] for the EU band to enable long-distance communication between nodes and base stations. The ADR algorithm gives a specific spreading factor for each node based on the signal to noise ratio observed at the gateway. In the offloading mode, depleting nodes use lower spreading factors such as 7 and 8 to communicate with the affluent node. Lower spreading factors typically consume less power and are known to have shorter communication range, and hence, are better suited for packet offloading. Low spreading factors are used only for packet offloading to reduce the number of collisions with other transmissions, thereby conserving energy at the depleting node. In the lading mode, a node receives packets on low spreading factors but uses high spreading factor to transmit packets to the gateway like conventional mode.

C. Energy Overhead during Lading Mode

Typically, the energy consumed by the communication module in an embedded low-power device is higher than that of the computation platform [52]. Thus, in this work, we focus on estimating the energy overhead required by the communication module of an affluent node in the lading mode and on ensuring that the overhead does not decrease the lifetime of the affluent node. Note that the energy overhead derived in this section is based on the energy model for communication given in the datasheet of LoRa communication model [49]. In the lading mode, there are three main sources of energy overheads: periodic CAD, packet reception and forwarding, and acknowledgement transmission and reception.

Periodic CAD. Affluent node v in the lading mode performs CAD once every $T_{\text{CAD}}(v)$ seconds, where $T_{\text{CAD}}(v) = T_1 + T_2$ (as shown in Fig. 3). Within $T_{\text{CAD}}(v)$ interval, v consumes P_{rx} power for a duration of $\frac{2^{1+SF_{\text{offload}}}}{BW_{\text{offload}}}$ to probe for correlation. The affluent node consumes $P_{\text{RC-Oscillator}}$ for a duration of $T_2 - \frac{2^{1+SF_{\text{offload}}}}{BW_{\text{offload}}}$ to remain in low-power idle mode with only RC-Oscillator on. For the rest T_1 time units, the affluent node sleeps and consumes close to 0 energy. Thus, the total energy consumed during an interval of $T_{\text{CAD}}(v)$ seconds is

$$E_{\text{CAD}}(v) = T_2 P_{\text{RC-Oscillator}} + \frac{2^{(1+SF_{\text{offload}})}}{BW_{\text{offload}}}\left(P_{\text{rx}} - P_{\text{RC-Oscillator}}\right)$$

The actual power consumption values for P_{rx} and $P_{\text{RC-Oscillator}}$ can be obtained from the datasheet of LoRa

communication module, and typically power consumed in receive mode is twice that of low-power idle mode [49]. Because an affluent node v remains in lading mode for a total duration of T_{LM} time units, the total energy consumed during CAD is given by $\frac{T_{\text{LM}}(v)}{T_{\text{CAD}}}E_{\text{CAD}}(v)$.

Packet reception and forwarding. The total energy overhead to receive a packet in the lading mode is a product of the power drawn by the chip and time on air of the packet, which is given by the product of number of symbols in the packets and time to transmit one symbol. In the lading mode, a node receives packets transmitted to it using the offloading-lading transmission parameters. The number of symbols in u's packet when transmitted by u to v using the offloading-lading transmission parameters, is given as follows:

$$\begin{aligned}N_{\text{symbols}}(u \xrightarrow{u} v) = \text{preamble}_u + 4.25 + 8 \\ + \max\left(\left\lceil\frac{8\,\text{payload}_u - 4\,SF_{\text{offload}} + 24}{SF_{\text{offload}} - 2\,\text{DE}_{\text{offload}}}\right\rceil\frac{1}{CR_{\text{offload}}}, 0\right)\end{aligned} \quad (8)$$

Similarly, the time to transmit one symbol of node u's packet from u with offloading-lading transmission parameters is:

$$T_{\text{symbol}}(u \xrightarrow{u} v) = \frac{2^{SF_{\text{offload}}}}{BW_{\text{offload}}} \quad (9)$$

To receive a packet from u, the affluent node v consumes P_{rx} power for a duration of $N_{\text{symbols}}(u \xrightarrow{u} v)T_{\text{symbol}}(u \xrightarrow{u} v)$.

In the lading mode, the affluent node forwards offloaded packet using its conventional mode parameters. Thus, the number of symbols in u's packet when transmitted by v using v's conventional mode transmission parameters is given by

$$\begin{aligned}N_{\text{symbols}}(v \xrightarrow{u} \text{Gateway}) = \text{preamble}_u + 4.25 + 8 \\ + \max\left(\left\lceil\frac{8\,\text{payload}_u - 4\,SF_v + 24}{SF_v - 2\,\text{DE}_v}\right\rceil\frac{1}{CR_v}, 0\right)\end{aligned} \quad (10)$$

Similarly, the time to transmit on symbol of node u's packet from node v to Gateway with v's conventional mode transmission parameters is given by Equation (2). To forward an offloaded packet, an affluent node v consumes $P_{\text{tx}}(PL_v)$ power for $N_{\text{symbols}}(v \xrightarrow{u} \text{Gateway})T_{\text{symbol}}(v)$ duration. Thus, the total energy consumed by an affluent node v for receiving and forwarding one packet from u is given as follows:

$$\begin{aligned}E_{\text{FW}}(u, v) = P_{\text{rx}}\,N_{\text{symbols}}(u \xrightarrow{u} v)\,T_{\text{symbol}}(u \xrightarrow{u} v) \\ + P_{\text{tx}}(PL_v)\,N_{\text{symbols}}(v \xrightarrow{u} \text{Gateway})\,T_{\text{symbol}}(v)\end{aligned} \quad (11)$$

Note that the number of symbols and time to transmit a packet using the offload parameters, as given by Equation (8) and (9) are significantly lower than that using the conventional parameters, as given by Equation (10) and (2). Furthermore, the power level used for transmission in offloading-lading mode is lower than that of conventional mode. Thus the total energy consumed during a packet's offloading is significantly lower than that needed for its transmission to the gateway.

Acknowledgment transmission and reception. Similar to packet transmission, energy overhead during transmitting and receiving acknowledgments is a product of power drawn, number of symbols, and time to transmit one symbol. In the lading mode, a node transmits an ACK to a depleting node using the offloading-lading transmission parameters. Thus, the number of symbols in node v's ACK to node u is given by

Equation (12), assuming that the packet transmitted contains only the preamble and header (i.e., no payload and CRC).

$$N_{\text{symbols}}(v \xrightarrow{\text{ack}} u) = \text{preamble}_u + 4.25 + 8 +$$
$$\max\left(\left\lceil \frac{8 - 4\,SF_{\text{offload}}}{SF_{\text{offload}} - 2\,DE_{\text{offload}}} \right\rceil \frac{1}{CR_{\text{offload}}}, 0\right) \quad (12)$$

Here, $v \xrightarrow{\text{ack}} u$ denotes the transmission of ACK from node v to node u after a successful packet reception. The duration of each symbol in node v's ACK to node u ($T_{\text{symbol}}(v \xrightarrow{\text{ack}} u)$) is the same as $T_{\text{symbol}}(u \xrightarrow{u} v)$ since the transmission parameters are the same. Equation (9) gives the expression for $T_{\text{symbol}}(u \xrightarrow{u} v)$. The affluent node transmits an acknowledgment twice and each acknowledgment takes $N_{\text{symbols}}(v \xrightarrow{ack} u) T_{\text{symbol}}(u \xrightarrow{u} v)$ time to transmit and consumes $P_{\text{tx}}(PL_{\text{offload}})$ power. In the lading node, node v receives an ACK for node u's packet from the gateway using its transmission parameters. To receive an ACK, node v opens two receive window slots with a total length of $T_{\text{rx}}(v)$. For the duration of T_{rx}, it consumes P_{rx} power. Thus, the total energy consumed to transmit and receive an ACK is:

$$E_{\text{ack}}(u,v) = 2P_{\text{rx}}\,T_{\text{rx}}(v)$$
$$+ 2P_{\text{tx}}(PL_{\text{offload}})\,N_{\text{symbols}}(v \xrightarrow{\text{ack}} u)\,T_{\text{symbol}}(u \xrightarrow{u} v) \quad (13)$$

A depleting node u generates $\frac{T_{\text{LM}}(v)}{\tau_u}$ packets in the lading mode of v, where τ_u is its minimum inter-packet generation time. If each packet of node u needs an average of γ retransmissions to successfully reach v, the total energy consumed to offload $\frac{T_{\text{LM}}(v)}{\tau_u}$ packets including ACKs is given by

$$\frac{\gamma T_{\text{LM}}(v)}{\tau_u}\big(E_{\text{FW}}(u,v) + E_{\text{ack}}(u,v)\big)$$

A node does not perform CAD during the reception and transmission of a packet or an ACK. Thus, the energy correction is the product of $E_{\text{CAD}}(v)$ and the number of occurrences of CAD during the reception and transmission of both the offloaded packet and its ACK. The energy correction at node v per packet offloaded by u is given by Equation (14).

$$E_{\text{CR}}(u,v) = \frac{E_{\text{CAD}}(v)}{T_{\text{symbol}}}\bigg(2T_{rx}$$
$$+ 2N_{\text{symbols}}(v \xrightarrow{\text{ack}} u)\,T_{\text{symbol}}(u \xrightarrow{u} v)$$
$$+ N_{\text{symbols}}(u \xrightarrow{u} v)\,T_{\text{symbol}}(u \xrightarrow{u} v) \quad (14)$$
$$+ N_{\text{symbols}}(v \xrightarrow{u} \text{Gateway})\,T_{\text{symbol}}(v)\bigg)$$

Overall overhead. Considering packet offloading from multiple depleting nodes and three sources of energy overhead, the total energy overhead in the lading mode of a node v is given by Equation (15), where DN represents the set of depleting nodes, and AF_DN represents the set of affluent and depleting node pairs that communicate with each other.

$$E_{\text{LM}}(v) = \frac{T_{\text{LM}}(v)}{T_{\text{CAD}}}E_{\text{CAD}}(v) +$$
$$\sum_{\substack{\{u,v\}\in\text{AF\_DN}\\ u\in\text{DN}}} \frac{\gamma T_{\text{LM}}(v)}{\tau_u}\big(E_{\text{FW}}(u,v) + E_{\text{ack}}(u,v) - E_{\text{CR}}(u,v)\big)$$
$$(15)$$

Note that Equation (15) gives an upper bound on the energy overhead on the lading mode, assuming that nodes are always performing CAD during the lading mode. In practice, the nodes can be sleeping for short periods, i.e., between inter-arrival times of packets. To simplify the computation of affluent and depleting node pairs that communicate with each other, we use this upper bound on the lading mode's energy consumption. As we explain in the following section, this energy overhead for an affluent node is used to find the affluent and depleting node pairs that communicate with each other.

D. Selecting Different Modes of Operation

1) Workload Prediction: The network server distinguishes affluent nodes from depleting nodes based on the total energy consumption of the node in the current recharge cycle. Unfortunately, the unreliability in wireless networks makes it difficult to predict the number of transmission attempts each node will make before the packet has been successfully received at the gateway. In case of packet reception failure at the gateway, a node can retransmit it up to 8 times. However, the actual number of retransmissions can be less than that.

To estimate the number of retransmissions from the nodes, we use a sliding window approach. Each node embeds the retransmission attempt information in the packet it transmits. The network server keeps the retransmission count from the last w packets in a sliding window. The size of the sliding window can be decided by the network manager. The average retransmission count of the window is used as the retransmission overhead, γ, for the network. There are existing works that use various machine learning techniques such as Auto-Regressive Moving Integrated Average Technique (ARIMA), Seasonal ARIMA [53], Support Vector Machines, Multi-Layer Perception, [54] Gaussian Process and Long Short Term Memory (LSTM) architecture [55] to predict the wireless network traffic pattern. However, such approaches increase the complexity of the system. Thus, we leave the integration of such prediction algorithms in our approach as a future work.

2) Selection of Affluent and Depleting Nodes: The network server selects the affluent and depleting nodes based on their expected energy consumption in the conventional mode. For each node in v in the network, it estimates the total number of transmission attempts $N_{\text{tx}}(v)$ in the current recharge cycle as $N_{\text{tx}}(v) = \gamma \times \frac{T_\zeta}{\tau_v}$, where T_ζ denotes the time interval from current time t to the start of the next recharge cycle. Using this information, the network server estimates the energy consumption in conventional mode for each node v as $E_{\text{CM}}(v) = N_{\text{tx}} \times E_{\text{tx}}(v)$, where, $E_{\text{CM}}(v)$ is the energy consumption in conventional mode and $E_{\text{tx}}(v)$ is the energy consumption for packet transmission given by Equation (1). The network server can estimate the battery budget B_v^* for each node v in the network. Thus, if $B_v^* - \beta_v > E_{\text{CM}}(v)$, it assigns node v to the set of affluent nodes AN. Otherwise, we assign node v to the set of depleting nodes DN. In this way, the network server divides all nodes into two disjoint sets AN and DN as presented in Algorithm 1. The server runs this Algorithm after every packet reception.

Algorithm 1 Selecting Affluent and Depleting Nodes

Input: Set of all nodes V
Output: Disjoint sets AN, DN
$AN \leftarrow \emptyset$
$DN \leftarrow \emptyset$
for $v \in V$ **do**
    $E_{\text{CM}}(v) \leftarrow N_{\text{tx}} \times E_{\text{tx}}(v)$
    if $E^{\text{CM}}(v) < B_v^* - \beta_v$ **then**
        $AN \leftarrow AN \cup \{v\}$
    else
        $DN \leftarrow DN \cup \{v\}$
    end if
end for

Algorithm 2 Selecting Lading and Offloading Mode

Input: Disjoint sets AF, DN
Output: Set AF_DN
for $v \in AN$ **do**
    $AF\_DN \leftarrow \{v\} \times DN$
    for $(u, v) \in AF\_DN$ **do**
        sort AF_DN in ascending order of $E_{\text{OM}}(u, v)$
        calculate $T_{LM}(v)$ for AF_DN from Equation (15)
        while Equation 16 is not satisfied **do**
            Remove last pair (u, v) from AF_DN
            calculate $T_{\text{LM}}(v)$ for AF_DN from Equation (15)
        end while
    end for
end for

E. Selecting Lading and Offloading Mode

After executing Algorithm 1, if the network server identifies any depleting node $u \in DN$, it immediately starts to look for a suitable affluent node $v \in AN$ in the same cell which can support the lading mode of operation. To this end, the network server creates the set AF_DN of affluent and depleting nodes that communicate with each other. Each member in the set AF_DN is a pair (u, v) where u is the depleting node and v is the affluent node. For every pair $(u, v) \in AF\_DN$, the lading time $T_{\text{LM}}(v)$ for affluent node v should be at least as large as τ_u, one sampling period of u, to successfully enable offloading of at least one packet from u to v. Thus, every pair $(u, v) \in AF\_DN$ should satisfy the constraint:

$$T_{\text{LM}}(v) \geq \tau_u \;\; \forall (u, v) \in AF\_DN. \tag{16}$$

Furthermore, in order to prevent depletion of budget at the affluent node, the time for lading mode should be carefully selected so that energy consumption in lading mode does not exceed the superfluous residual energy E_r for v, calculated as the difference between node v's budget and $E_{\text{CM}}(v)$. Thus, for an affluent node v, the energy consumption in lading mode E_{LM} is bounded as

$$E_{\text{LM}}(v) \leq E_r(v). \tag{17}$$

Since optimal pairs of affluent and depleting nodes are unknown, we rely on a greedy heuristic which enables an affluent node to assist as many depleting nodes as possible without hampering its own lifetime. To this end, the network server constructs the set AF_DN by pairing an affluent node v with every depleting node u in the same cell. Then, the network server calculates the time in lading mode $T_{\text{LM}}(v)$ such that Equation (17) is satisfied. Such an approach prevents an affluent node from becoming a depleting node. Note that, if node v is paired with a large number of depleting nodes, $T_{\text{LM}}(v)$ is shortened to keep $E_{\text{LM}}(v)$ bounded. Thus, the estimated value of T_{LM} might not satisfy Equation (16). In other words, the affluent node may not be able to support packet offloading from these many depleting nodes $u \in AF\_DN$ with its superfluous residual energy. Thus, the network server needs to remove some pairs from AF_DN to increase $T_{\text{LM}}(v)$ and satisfy Equation (16). Since our goal is to prolong the lifetime of depleting nodes, we select to remove the pairs with the highest *offloading overhead* $E_{\text{OM}}(u, v)$ given by the following equation which represents the energy consumption for node u to offload one packet to node v:

$$E_{\text{OM}}(u, v) = P_{\text{tx}}(PL_{\text{OFFLOAD}}) \times N_{\text{symbols}}(u \xrightarrow{u} v) \, T_{\text{symbol}}(u \xrightarrow{u} v)$$

In the above equation, PL_{OFFLOAD} is the transmission power for node u in offloading mode. Thus, the network server greedily removes the pair (u, v) with the highest offloading overhead from the set AF_DN until node v is able to support packet offloading for every pair (u, v) in the set as described in Algorithm 2. After the set AF_DN is constructed, for each pair $(u, v) \in AF\_DN$, the network server enables offloading and lading mode in u and v, respectively, for the duration of $T_{\text{LM}}(v)$. Note that, the network server only provides an upper bound of time in lading and offloading mode.

1) Transitioning between Conventional and Lading/Offloading Modes: Upon deciding the set AF_DN, the network server broadcasts the information via acknowledgments. One challenge in transitioning from conventional mode to lading/offloading mode is estimating the start of the lading/offloading mode. Since nodes transmit a heartbeat message periodically, the network server can predict the arrival of the next heartbeat message from each node that is transitioning from conventional mode to lading/offloading mode. This information can be used to estimate the next time instant before which all nodes will have a minimum of one communication (either heartbeat or data) with the gateway. The network server chooses this instant as the start of the lading and offloading mode of operation. This ensures that all nodes that are transitioning to offloading/lading modes receive information about the transition.

The network server chooses the affluent node to start its lading mode a few seconds before the offloading mode starts. Such a proactive approach circumvents any time synchronization errors between the devices. When its reserve energy is exhausted, an affluent node can use piggy-backed acknowledgments to inform the depleting nodes to transition from offloading mode to conventional mode. Once all depleting nodes are notified to switch to the conventional mode, an affluent node can transition to conventional mode.

V. EVALUATION

We have evaluated our results in simulation and experiment.

A. Simulation

1) Simulation Setup: We evaluate the efficiency of our approach by conducting simulations using up to 1200 nodes and a single gateway placed in a disc of radius 3500m. Our implementation is based on the LoRaWAN module for NS-3 [56]. The nodes and the gateway use 8 channels between

902.3 and 903.7 MHz with a channel bandwidth of 125 kHz. To conform with the US regulations, we do not use SF11 and SF12. Unless stated otherwise, the parameters were set using our transmission parameter selection approach.

We configure a percentage of nodes to transmit at a higher rate of 20 to 30 packets per hour. All other nodes transmit at a rate of 2 to 4 packets per hour. The packet interarrival rate remained unchanged for the duration of the simulation. For each node, a rechargeable battery was simulated by assigning battery budgets randomly in the range [6,25] joules for the current recharge cycle. The heterogeneity of battery budgets and traffic patterns of the nodes introduce depleting nodes in the network. Affluent and depleting nodes, if any, were selected based on Algorithm 1 after every packet reception at the gateway. For simplicity, we use a fixed retransmission overhead, $\gamma = 2$. Energy consumption for packet transmission (including retransmissions), packet reception and CAD are subtracted from this budget. Packet transmission from a node was stopped upon depletion of its budget. For each setup, we reported the network lifetime and throughput for a single run. Network throughput was measured by the number of bytes received per unit of time at the gateway.

Baseline. To our knowledge, no existing work studied lifetime maximization of energy-harvesting LoRa networks. Thus we compare the performance of our approach (shown as 'LLL' i.e., Long-Lived LoRa in figures) with standard LoRaWAN.

2) Results under Varying Number of Nodes: We first evaluate our approach under varying number of nodes, where approximately 3% of the nodes are depleting. Figure 5 demonstrates the performance in terms of lifetime and throughput. In Figure 5(a), the network lifetime remains fairly constant for our approach despite the increase in number of nodes whereas for LoRaWAN the lifetime decreases as the number of nodes increase (at 200 nodes the lifetime is 10.42 Hours for LoRaWAN which decreases to 7.53 Hours at 1200 nodes). The small variations in network lifetime across experiments in Figure 5(a) are mainly due to the randomly assigned battery budgets. As the figure shows, our approach (LLL) is consistently better than the legacy LoRa (LoRaWAN). Our approach has a lifetime of 23.8 Hours on average compared to 7.53 Hours in LoRa. When the number of nodes is 800, we achieve more than 4 times better performance in terms of lifetime. As shown in Figure 5(b), LLL maintains throughput no less than the standard LoRaWAN under the same set up.

It is worth noting that, in our approach, offloading may increase the latency of some (offloaded) packets. However, the applications deployed in a LoRa network such as data collection in smart agriculture [2], smart metering [57], smart cities [3] and wild-life monitoring [58] are usually not time-sensitive. Rather, lifetime maximization is of greater importance as it ensures seamless data collection from all sensors. Thus, by increasing the latency of some packets, we are significantly prolonging the lifetime of the network while keeping the throughput no less than the standard LoRaWAN.

3) Results under Varying Number of Depleting Nodes: We now evaluate our approach under varying percentage of
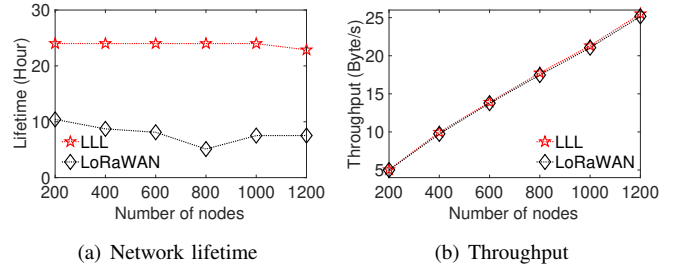

(a) Network lifetime   (b) Throughput

Fig. 5. Network lifetime and throughput under varying number of nodes

depleting nodes. We fix the number of nodes at 1000 and vary the percentage of depleting nodes in the network from 2 to 10. As shown in Figure 6(a), the lifetime of our approach is significantly better than the traditional LoRa, as the percentage of the depleting nodes increases. On average, the network lifetime for our approach is 22.58 Hours, whereas for LoRaWAN, the lifetime is 6.03 Hours. We observe the lifetime for our approach decreases as the percentage of depleting node increase as the set of affluent nodes are not able to support a large number of depleting nodes (at 2% depleting nodes, the lifetime is 24 Hours which decreases to 22 Hours at 10% depleting nodes). Figure 5(b) shows that our approach has network throughput of 23 Byte/s where LoRaWAN has 22.39 Byte/s. As we introduce more depleting nodes in the network by increasing the packet rate at some nodes, the throughput of both approaches increase. However, our approach prevents early depletion of batteries and thus achieves higher throughput than LoRaWAN under many depleting nodes.
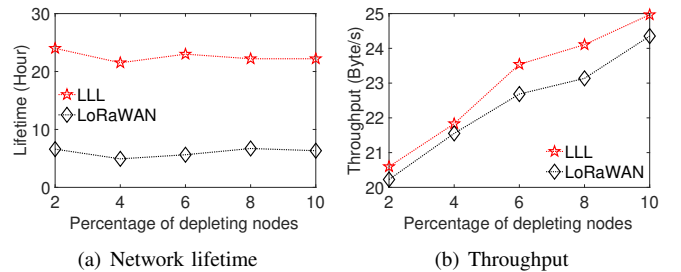

(a) Network lifetime   (b) Throughput

Fig. 6. Lifetime and throughput under varying % of offloading nodes

4) Results under Duty Cycle Enabled EU Band: We have evaluated our approach in the EU band using 8 channels ranging from 868 MHz to 869.55 MHz under varying number of nodes. Duty-cycle for each channel was set according to the regulatory rules. As such, the same channel cannot be reused for multiple transmissions immediately. To comply with this, we change our channel assignment in offloading and lading mode such that after each transmission, an affluent node asks the depleting nodes through ACKs to change to the same channel for the next transmission. However, in case of more than one depleting nodes with different transmission patterns, this introduces an overhead as ACKs cannot be transmitted to all of them immediately. Thus, in Figure 7 we see a slight decrease in lifetime and throughput for our approach. While throughput of both approaches increases with the number of nodes in Figure 7(b), the throughput for our approach is slightly lower than LoRa. However, as shown in Figure 7(a), we observe that our approach is significantly better

than LoRaWAN in terms of lifetime. On average our approach has lifetime of 20.2 Hours compared to 7.35 Hours in LoRa.
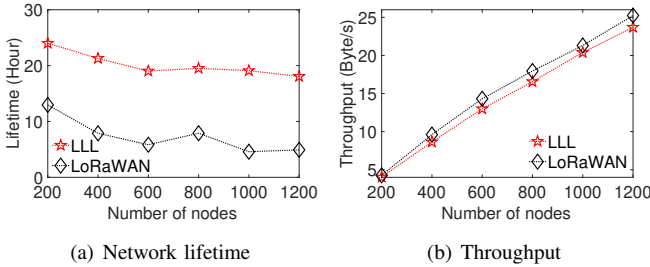

(a) Network lifetime   (b) Throughput

Fig. 7. Lifetime and throughput under varying number of nodes in EU band

B. Experiments

In this section, we conduct a proof-of-concept experiment of our design with the commercially available LoRa nodes, gateway, and network server to show that the simulation results are consistent also in our real testbed, i.e., the measured network lifetime increases while maintaining a similar throughput.

1) Implementation: In our implementation, we use the Dragino SX1276 LoRa transceiver HAT [59] on Raspberry Pi 3 [60] as the LoRa nodes. On the other hand, we use the RAK2245 HAT [61] on Raspberry pi 3 to build a gateway. Specifically, each LoRa node runs a custom built version (for our protocol) of the LMIC 1.6 LoRa/LoRaWAN library [62] and the Raspberry Pi hosting the gateway runs the open-source ChirpStack LoRaWAN network server [63] locally.

2) Experimental Setup: We use 15 LoRa nodes and one gateway in our experiments. Figure 8 shows the locations of the nodes in our indoor deployment of an area of (30x20) ft^2, where the dark circle represents the gateway and the rest represent the nodes (labeled from N1 to N15). Note that
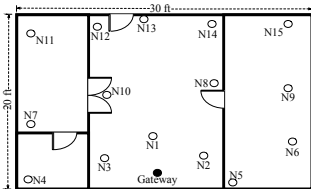

Fig. 8. Gateway and node's positions in indoor deployment.

we were unable to perform wide-area outdoor experiments due to the COVID-19 pandemic [64]. The gateway has two half-duplex transceiver radios, and each node has one half-duplex transceiver radio. Our LoRa network operates in the 915 MHz frequency band and adopts no duty cycle requirements (as per the LoRa regional regulations [1]). Each node in our experiments maintains a variable (i.e., storage area in the node's memory) to emulate its energy budget. At the beginning of the experiments, each node is assigned a different energy budget that ranges between 6 to 20 joules. Based on the radio states (e.g., transmit, receive, CAD, sleep, etc.), each node deducts appropriate energy from its budget according to the energy model presented in Section IV-C which maintains the accuracy of the energy model given in the SX1276 datasheet [49] and that of the simulation setup in Section V-A1.

In conventional LoRa mode (Class-A), each node adopts the ADR. In offloading mode, a node uses a channel bandwidth of 125 kHz, SF of 7, transmit power of 2 dBm (minimum for the SX1276 LoRa module in the LMIC library), and coding

rate of $\frac{4}{5}$. Using these settings, we enable 3 to 15 nodes to transmit data in the experiments. Each node transmits its CPU temperature (2 bytes) to the gateway with the ACK bit set. In each run of the experiment (e.g., with different number of nodes), two-thirds of the nodes adopt an inter-packet interval of 2 to 4 minutes. On the other hand, one-third of the nodes send packets with an interval of 40 to 60 seconds. It may thus introduce depleting nodes in the network. Unless stated otherwise, these are our default experimental setups.

Note that the Raspberry pi 3 used as LoRa nodes in our setup has a higher static power consumption than light-weight sensors used in real deployments. We did not attach a battery to the nodes as the raspberry pi's static power usage would overwhelm the power consumed in transmissions and rapidly deplete the battery, instead the nodes were line-powered. In this proof-of-concept setup, the supply current values from the SX1276 datasheet [49] were used to calculate the total energy consumption in our proposed Algorithms.

3) Results: With the above settings, we evaluate the network lifetime and throughput of our design and compare with LoRaWAN. Figure 9(a) shows the network lifetime for both of the approaches with varying number of nodes. On average our approach has lifetime of 30.24 minutes, compared to 17.84 minutes in LoRa. As the nodes use a higher packet rate than simulation, the lifetime is much lower than observed in simulation. This figure also shows that as the number of nodes increases, the lifetime in LoRaWAN decreases sharply compared to our approach. Figure 9(b) shows the overall network throughput for both of the approaches with varying number of nodes. As shown in this figure, on average, the throughput in LoRaWAN and our approach remain competitive (18.72 vs 19 Bytes/s). The results in simulations and experiments exhibit similar trends in network lifetime and throughput.
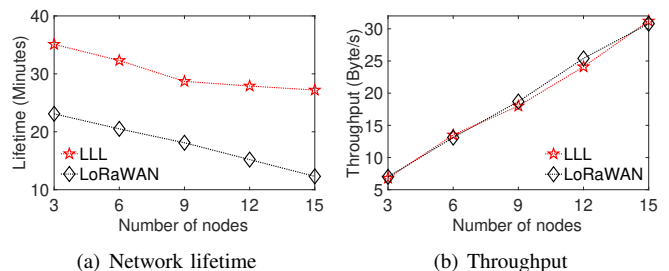

(a) Network lifetime   (b) Throughput

Fig. 9. Network lifetime and throughput under varying number of nodes

VI. Conclusion

As LoRa networks are widely being adopted for various IoT applications, prolonging their lifetime is a very important research problem. In this paper, we have proposed a link-layer protocol to prolong the lifetime of an energy-harvested LoRa network. Simulations and experiments show that our protocol can increase the network lifetime up to 4 times compared to traditional LoRa network. In our future work we intent to study how to preserve the privacy of packets while offloading.

REFERENCES

[1] "Lora alliance," https://lora-alliance.org/.

[2] Business Insider, "Smart farming in 2020: How IoT sensors are creating a more efficient precision agriculture industry," https://www.businessinsider.com/smart-farming-iot-agriculture, 2020.

[3] GSMA, "Air Quality Monitoring with IoT and Big Data," https://www.gsma.com/iot/iot-big-data/smart-london-air-quality-monitoring-big-data/, 2020.

[4] IBM, "How AI, IoT and weather tech can help better detect deadly wildfires," https://www.ibm.com/blogs/think/2019/08/ai-detect-wildfires/, 2020.

[5] PowerFilm Solar, "Solar power for industrial IoT," https://www.powerfilmsolar.com/about-us/the-horizon-blog/2018/07/06/solar-power-for-industrial-iot, 2020.

[6] Revibe Energy, "Powering the industrial IoT through vibrations," 2020. [Online]. Available: https://revibeenergy.com

[7] PHYS.ORG, "Body heat through flexible fabric could power IoT devices for health monitoring for people, pets, machinery," https://phys.org/news/2020-01-body-flexible-fabric-power-iot.html, 2020.

[8] M. Pehl, A. Arvesen, F. Humpenoder, A. Popp, E. G. Hertwich, and G. Luderer, "Understanding future emissions from low-carbon power systems by integration of life-cycle assessment and integrated energy modelling," *Nature Energy*, no. 2, pp. 939—-945, 2017.

[9] D. Roberts, "Batteries have a dirty secret," 2018, https://www.vox.com/energy-and-environment/2018/4/27/17283830/batteries-energy-storage-carbon-emissions.

[10] F. Wu, J.-M. Redouté, and M. R. Yuce, "We-safe: A self-powered wearable iot sensor network for safety applications based on lora," *IEEE Access*, vol. 6, pp. 40 846–40 853, 2018.

[11] H. H. R. Sherazi, G. Piro, L. A. Grieco, and G. Boggia, "When renewable energy meets lora: A feasibility analysis on cable-less deployments," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5097–5108, 2018.

[12] R. Piyare, A. L. Murphy, M. Magno, and L. Benini, "On-demand lora: Asynchronous tdma for energy efficient and low latency communication in iot," *Sensors*, vol. 18, no. 11, p. 3718, 2018.

[13] W. Gao, W. Du, Z. Zhao, G. Min, and M. Singhal, "Towards energy-fairness in lora networks," in *IEEE ICDCS*, 2019.

[14] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[15] A. Jahid, K. H. Monju, S. Hossain, and F. Hossain, "Hybrid power supply solutions for off-grid green wireless networks," *International journal of green energy*, vol. 16, no. 1, pp. 12–33, 2019.

[16] A. Saifullah, S. Sankar, J. Liu, C. Lu, R. Chandra, and B. Priyantha, "CapNet: A real-time wireless management network for data center power capping," in *2014 IEEE Real-Time Systems Symposium*, 2014, pp. 334–345.

[17] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 855–873, 2017.

[18] A. Saifullah, M. Rahman, D. Ismail, C. Lu, J. Liu, and R. Chandra, "Enabling reliable, asynchronous, and bidirectional communication in sensor networks over white spaces," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '17. Association for Computing Machinery, 2017.

[19] D. Ismail, M. Rahman, and A. Saifullah, "Low-power wide-area networks: Opportunities, challenges, and directions," in *Proceedings of the Workshop Program of the 19th International Conference on Distributed Computing and Networking*, ser. Workshops ICDCN '18. Association for Computing Machinery, 2018.

[20] M. Rahman and A. Saifullah, "Integrating low-power wide-area networks in white spaces," in *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2018, pp. 255–260.

[21] A. Dongare, R. Narayanan, A. Gadre, A. Luong, A. Balanuta, S. Kumar, B. Iannucci, and A. Rowe, "Charm: exploiting geographical diversity through coherent combining in low-power wide-area networks," in *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 2018, pp. 60–71.

[22] R. Eletreby, D. Zhang, S. Kumar, and O. Yağan, "Empowering low-power wide area networks in urban settings," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '17, 2017, p. 309–321.

[23] M. Rahman, D. Ismail, V. P. Modekurthy, and A. Saifullah, "Implementation of lpwan over white spaces for practical deployment," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, ser. IoTDI '19. Association for Computing Machinery, p. 178–189.

[24] "Sigfox," https://www.sigfox.com/en.

[25] http://www.iqrf.org/technology.

[26] https://www.ingenu.com/technology/rpma.

[27] http://www.dash7-alliance.org.

[28] "Telensa," https://www.telensa.com.

[29] 3gpp.org/news-events/3gpp-news/1785-nb_iot_complete.

[30] A. Saifullah, M. Rahman, D. Ismail, C. Lu, J. Liu, and R. Chandra, "Low-power wide-area network over white spaces," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1893–1906, 2018.

[31] A. Saifullah, M. Rahman, D. Ismail, C. Lu, R. Chandra, and J. Liu, "SNOW: Sensor network over white spaces," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '16. Association for Computing Machinery, 2016, p. 272–285.

[32] https://www.u-blox.com/en/lte-cat-m1.

[33] "LTE Advanced Pro," 2017, https://www.qualcomm.com/invention/technologies/lte/advanced-pro.

[34] M. Bor, J. Vidler, and U. Roedig, "Lora for the internet of things," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, ser. EWSN '16, 2016, pp. 361–366.

[35] R. A. and P. A., "Survey on lora technology: Solution for internet of things," *Intelligent Systems, Technologies and Applications*, vol. 1148, pp. 259–271, 2020.

[36] X. Wang, L. Kong, L. He, and G. Chen, "mLoRa: A multi-packet reception protocol in LoRa networks," in *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, 2019, pp. 1–11.

[37] X. Xia, Y. Zheng, and T. Gu, "Ftrack: Parallel decoding for lora transmissions," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems*, 2019.

[38] W. Wu, Y. Li, Y. Zhang, B. Wang, and W. Wang, "Distributed queueing based random access protocol for lora networks," *IEEE Internet of Things Journal*, 2019.

[39] L. Leonardi, F. Battaglia, and L. L. Bello, "Rt-lora: A medium access strategy to support real-time flows over lora-based networks for industrial iot applications," *IEEE Internet of Things Journal*, 2019.

[40] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, and S. Pollin, "Improving reliability and scalability of lorawans through lightweight scheduling," *IEEE Internet of Things Journal*, 2018.

[41] P. J. Marcelis, V. S. Rao, and R. V. Prasad, "Dare: Data recovery through application layer coding for lorawan," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2017, pp. 97–108.

[42] C. Pham, A. Bounceur, L. Clavier, U. Noreen, and M. Ehsan, "Investigating and experimenting interference mitigation by capture effect in lora networks," in *Proceedings of the 3rd International Conference on Future Networks and Distributed Systems*. ACM, 2019, p. 31.

[43] C. Orfanidis, L. M. Feeney, M. Jacobsson, and P. Gunningberg, "Investigating interference between lora and ieee 802.15. 4g networks," in *Wireless and Mobile Computing, Networking and Communications (WiMob),*. IEEE, 2017, pp. 1–8.

[44] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do lora low-power wide-area networks scale?" in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2016, pp. 59–67.

[45] M. Slabicki, G. Premsankar, and M. Di Francesco, "Adaptive configuration of lora networks for dense iot deployments," in *16th IEEE/IFIP Network Operations and Management Symposium (NOMS 2018)*, 2018, pp. 1–9.

[46] A. Mahmood, E. G. Sisinni, L. Guntupalli, R. Rondon, S. A. Hassan, and M. Gidlund, "Scalability analysis of a lora network under imperfect orthogonality," *IEEE Transactions on Industrial Informatics*, 2018.

[47] F. Van den Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Scalability analysis of large-scale lorawan networks in ns-3," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2186–2198, 2017.

[48] M. N. Ochoa, A. Guizar, M. Maman, and A. Duda, "Evaluating lora energy efficiency for adaptive networks: From star to mesh topologies," in *Wireless and Mobile Computing, Networking and Communications (WiMob),*. IEEE, 2017, pp. 1–8.

[49] Semtech, "SX1276 datasheet," 2019. [Online]. Available: https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R0000001OKx/JUYM3TvBMenQzU4LS8ZJcM58BIjCcoZcUpHV0gnZ.y0

[50] L. alliance, "LoRaWAN ADR," https://lora-alliance.org/resource-hub/lorawanr-specification-v11, 2017.

[51] ——, "LoRaWAN ADR," https://lora-alliance.org/resource-hub/rp2-101-lorawanr-regional-parameters, 2020.

[52] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, vol. 3. IEEE, 2003, pp. 1713–1723.

[53] Yantai Shu, Minfang Yu, Jiakun Liu, and O. W. W. Yang, "Wireless traffic modeling and prediction using seasonal arima models," in *IEEE International Conference on Communications, 2003. ICC '03.*, vol. 3, May 2003, pp. 1675–1679 vol.3.

[54] A. Y. Nikravesh, S. A. Ajila, C. Lung, and W. Ding, "Mobile network traffic prediction using mlp, mlpwd, and svm," in *2016 IEEE International Congress on Big Data (BigData Congress)*, June 2016, pp. 402–409.

[55] S. Cao and W. Liu, "Lstm network based traffic flow prediction for cellular networks," in *Simulation Tools and Techniques*, H. Song and D. Jiang, Eds. Cham: Springer International Publishing, 2019, pp. 643–653.

[56] D. Magrin, M. Capuzzo, and A. Zanella, "A thorough study of lorawan performance under different parameter settings," *IEEE Internet of Things Journal*, 2019.

[57] "Smart meter," https://www.semtech.com/lora/lora-applications/smart-electricity-metering.

[58] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, "An analysis of a large scale habitat monitoring application," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*. ACM, 2004, pp. 214–226.

[59] "Dragino gps/lora hat," 2020. [Online]. Available: https://www.dragino.com/products/lora/item/106-lora-gps-hat.html

[60] "Raspberry pi 3 b+," 2020. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/

[61] "Rak2245 hat/pilot gateway," 2020. [Online]. Available: https://cdn-shop.adafruit.com/product-files/4284/4284_Get_Start_with_RAK2245_Pi_HAT_V2.4R.pdf

[62] "Lmic 1.6 for rpi gps/lora hat," 2020. [Online]. Available: https://github.com/wklenk/lmic-rpi-lora-gps-hat

[63] "Chirpstack lorawan network server," 2020. [Online]. Available: https://www.chirpstack.io

[64] "World Health Organization," 2020. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019