Large-Scale Environmental Sensing of Remote Areas on a Budget

Dixin Wu, *Student Member, IEEE*
Alexandru S. Bogdan
Jörg Liebeherr *Fellow, IEEE*

***Abstract*—By enabling large-scale in-situ environmental monitoring of remote areas, the Internet-of-Things (IoT) can play a crucial role in quantifying and responding to climate change. Sensing of uninhabited and many rural regions creates a need for inexpensive battery-powered IoT systems that can be deployed across large areas. Today, such systems are woefully unavailable. This paper presents a scalable IoT architecture for low-cost and low-power in-situ environmental sensing. The architecture is anchored by self-organizing LoRa mesh networks that can be scaled to a hundred nodes, covering a hundred or more square kilometers, at a cost of less than US$15 per node. A low-power design enables nodes to operate for years on two AA batteries in many sensing applications. LoRa mesh networks connect to a cloud-based IoT backend via a battery-powered modular gateway, which supports Internet access over a WiFi network, a cellular network, and a low-earth orbit satellite system.**

I. Introduction

The Internet-of-Things (IoT) creates a paradigm shift for climate research and sustainable agriculture by enabling large-scale in-situ (on-site) sensing systems that collect and deliver environmental data in real-time [1]. However, environmental monitoring in remote areas and developing countries remains impeded by a lack of suitable IoT networking solutions. Practicable networking solutions for these regions must scale to large geographic areas, be cost-effective in terms of capital and operating costs, and be low-power so that deployments do not depend on access to an electric grid. General IoT networking solutions should also be deployable in a variety of environments, ranging from rural farming communities to the Arctic wilderness. Lastly, to serve diverse applications that include smart irrigation and monitoring of greenhouse gas emissions, the network subsystem should not be coupled to a specific sensing application.

While IoT network systems for environmental sensing abound, they are generally either low-cost or suitable for large areas, but not both. Low-cost IoT technologies, such as BLE and Zigbee, have a limited range [2, Table 2], while cellular radios, such as GSM, NB-IOT, and LTE-M, cover large distances, but incur a higher cost. Since recently, low-power wide-area networks (LPWANs) have enabled remote monitoring for applications where a long transmission range and long battery life are preferred over high throughput. A prime example is LoRa, a proprietary wireless radio technology that operates in unlicensed ISM bands and has communication ranges of several kilometers [3]. LoRa is complemented by an open-source protocol stack, called LoRaWAN, for building LPWANs. Public LoRaWAN infrastructures such as *The Things Network* [4] and *Helium* [5] provide low-cost wireless networking for applications with low throughput requirements. With LoRaWAN, sensor nodes send their data to a LoRaWAN gateway, thereby forming a star network topology. Thus, not only must each LoRa radio be within communication range of a gateway, the performance of communication also degrades when the number of connected radios grows large [6].

Recently, several efforts have been made to overcome the limitations of LoRaWAN, by organizing LoRa radios in a mesh network [7], [8]. A mesh network extends the communication range via multi-hop forwarding, whereby LoRa nodes relay data from other LoRa nodes towards a gateway. Since there is no requirement that all nodes are within range of a gateway, mesh networks can cover larger distances. Costs can be further reduced by replacing LoRaWAN gateways by simpler (and less costly) gateway devices that connect to the Internet using standard IoT protocols. Given the low cost of LoRa radios, LoRa mesh networks can significantly reduce the cost of large sensor deployments. However, until now there does not exist a LoRa mesh solution that scales to large networks, is low-cost, has low power requirements, and can operate autonomously without a need for global configuration.

This article describes a low-cost IoT architecture for large-scale environmental sensing that is built with LoRa mesh networks. The main building block of the architecture is a recently developed protocol for LoRa multi-hop mesh networking that can be scaled to a hundred nodes [9]. A unique feature of this protocol, compared to other LoRa mesh networks, is that the network is self-organizing, in the sense that the network forms without the need for central coordination or management [10]. With self-organization, the network can accommodate newly joining nodes and can adapt to node failures. The components of the IoT architecture are (1) one or more LoRa mesh networks, (2) a gateway with a modular design that provides Internet connectivity, and (3) a cloud-based back-engine for processing and storing collected data. The presented IoT architecture has a number of salient characteristics:

- *Scalable:* Simulation experiments in [9] show that a LoRa mesh network with 100 nodes distributed over an area of several hundred square kilometers can achieve a packet delivery ratio above 90 percent for all nodes, where packet delivery ratio expresses the percentage of packets

Dixin Wu is with Spero Analytics. Alexandru Bogdan is with Advanced Micro Devices Inc. Jörg Liebeherr is with the Department of Electrical and Computer Engineering, University of Toronto. The work is supported in part by the University of Toronto Centre for Global Engineering (CGEN).

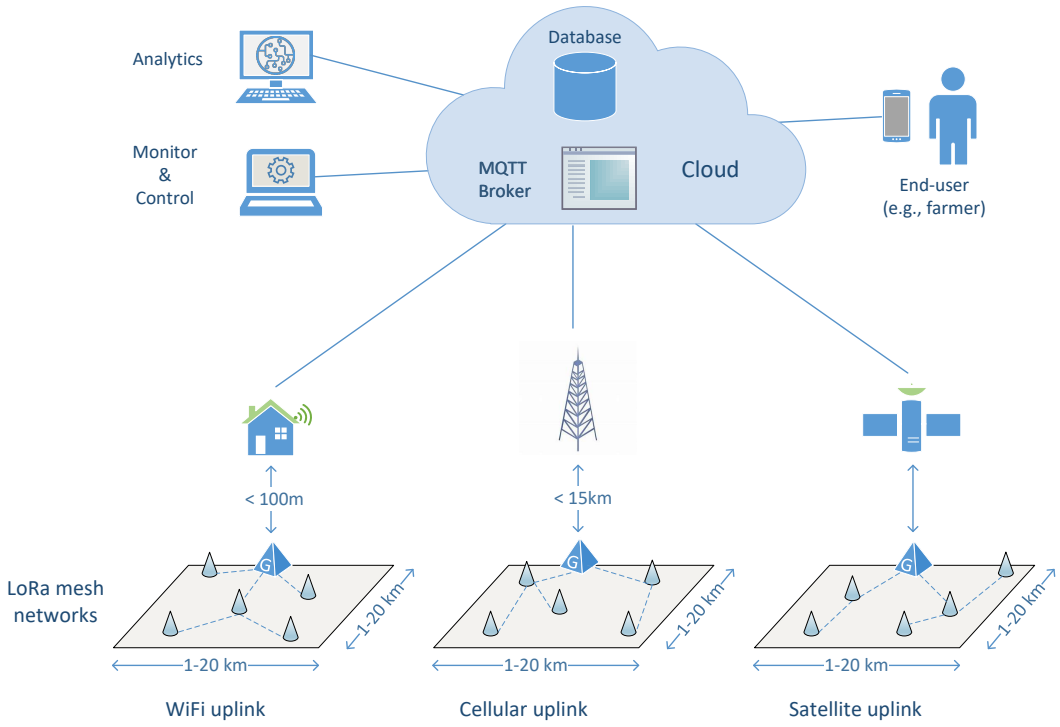


Fig. 1. CottonCandy IoT Architecture.

that are successfully forwarded from a source node to the gateway. Larger areas are covered by deploying multiple mesh networks.

- *Versatile:* By designing a gateway that can connect to a WiFi network, a cellular network, and a low-earth orbit satellite system, network deployments can take advantage of available terrestrial communication infrastructures, but can also operate in the absence of such infrastructures.
- *Cost effective:* Our prototype mesh node has a unit cost below US$15. The gateway node presented in this paper can be assembled for less than US$100 without a satellite uplink and around US$400 with a satellite modem.
- *Low power:* The LoRa mesh network and the gateway are active only during certain periods, so-called *duty cycles*. Between duty cycles all nodes and the gateway hibernate in a lower-power state. An energy-aware design ensures thousands of duty cycles on two AA batteries for mesh nodes and on four AA batteries for gateway nodes.
- *Application-independent:* The network system of the IoT architecture is only concerned with the formation of mesh networks and the delivery of data, but does not assume or require application-specific knowledge. In particular, mesh nodes and gateways process and format sensor data, but do not interpret their content. Access to sensors at a mesh node is provided through customizable callback functions. Interpretation of sensor data occurs only in the cloud.

To the best of our knowledge, there currently does not exist an IoT system for periodic measurements that can cover a similar area, on a similar scope, at a comparable cost. Existing LoRa mesh network solutions are still inadequate for large-scale environmental sensing. For instance, Lundell et al. [11] implemented multi-hop relaying only among LoRaWAN gateways. The mesh networks presented in [12] and [13] do not support battery-powered applications. Jiang et al. [14] reduced the energy consumption by adopting duty cycles. However, their centralized TDMA scheduler cannot scale due to the significant overheads of flooding TDMA schedules.

We believe that LoRa is a game changing technology for low-bandwidth applications due to low cost, (relatively) long range, and low power requirements. However, the potential of LoRa technology has so far not been fully exploited. The architecture presented shows that LoRa mesh networks provide an affordable option for deploying agricultural IoT monitoring systems.

II. CottonCandy IoT Architecture

Figure 1 provides an overview of the IoT architecture. The bottom of the figure depicts three LoRa mesh networks, each with a different method for accessing the Internet (WiFi, cellular, and satellite). Mesh nodes, represented as cones in the figure, self-organize in a rooted spanning tree topology. The root of each spanning tree is a gateway node, indicated by a pyramid and labeled as ‘G’, that provides access to the

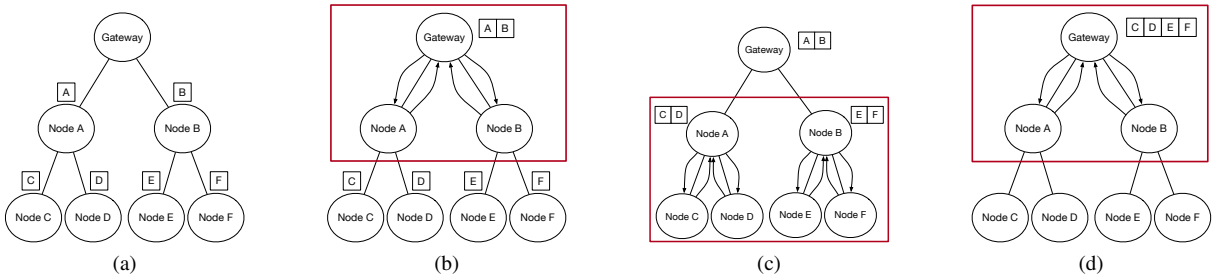


Fig. 2. Recursive data collection. (a) Each node (other than the gateway) has a data item indicated by a square. (b) The gateway requests data from nodes A and B. (c) Nodes A and B request data from nodes C–F. (d) In the next round of requests, the root collects the data items of nodes C–F from nodes A and B.

Internet and initiates collection of data from mesh nodes in a duty cycle. Using the CottonCandy LoRa mesh protocol from [9], networks with as many as 100 nodes can be built at a density as low as 0.25 nodes/km^2, yielding a coverage of up to 400 km^2 for a single mesh network.

The mesh networks operate in a time-driven fashion, where nodes are active only during a duty cycle and remain in a low-power state between duty cycles. If sensor readings are required outside of a duty cycle or if sensors operate asynchronously, e.g., a motion sensor, a mesh node may need to be activated between duty cycles. The elapsed time between duty cycles is application-dependent and is expected to range from fractions of an hour to a week. The frequency of duty cycles is a configurable parameter.

In a data cycle, the amount of sensor data collected from each mesh node in a duty cycle is small, ranging from 4–64 bytes. Data is delivered from a mesh node to the gateway via multi-hop forwarding. When a node forwards messages, it can concatenate data items from multiple sources in a single message, up to a maximum message size.

A gateway delivers data from the mesh network to a cloud system via the Internet. The gateway has three options for accessing the Internet. If the gateway is within range of a wireless access point (typically < 100 m) using WiFi incurs the least cost. A cellular radio can be used if the gateway is within range of a base station (typically < 15 km). A satellite uplink is the option of last resort for remote locations without a terrestrial communication infrastructure. By accessing a low-earth orbit satellite system, such as Iridium, a gateway can connect to the Internet mostly anywhere, however, at a high cost.

Gateways upload data items to a cloud system with the MQTT protocol [15]. The data is then stored in a cloud-based database. Application-specific data analytics tools access the data base and present results to users, either as visualizations or as part of a decision-support system (e.g., whether to irrigate a certain field).

In addition to supporting data collection and storage, the cloud system also facilitates monitoring and control of a mesh network. Monitoring functions can be used to query information on the network topology, geographical location (assuming nodes are equipped with GPS), battery status, or application-specific information about attached sensors. An example of a control function is the adjustment of the frequency of duty cycles.

III. CottonCandy Mesh Network Protocol

We recently developed a network protocol, called *CottonCandy*, that enables LoRa nodes to self-organize in a mesh network with a spanning tree topology. The protocol builds the basis of the proposed IoT architecture. We next give a summary of CottonCandy. We refer to [9] for a detailed description and an extended performance evaluation.

CottonCandy is intended for time-driven sensor applications, where nodes report data to a gateway in coordinated duty cycles. To support long-term measurements of environmental data on a large scale with low power requirements, CottonCandy emphasizes conservation of power and mitigation of packet collisions.

CottonCandy nodes form a rooted spanning tree network topology, where the root is a gateway device with access to the Internet. The hierarchical structure of a spanning tree enables simple, recursive protocols for both uplink and downlink communications. With respect to a reference node, we use the terms ‘upstream’ and ‘downstream,’ respectively, to describe nodes that are closer to the gateway and further away from the gateway than the reference node. In the spanning tree, each node other than the gateway has an upstream neighbor, which is the node closer to the gateway to whom it forwards data. Downstream neighbors of a node are those nodes for which the node is the upstream neighbor.

All CottonCandy nodes start their duty cycles at about the same time. The timespan until the next duty cycle is disseminated by the gateway in the preceding duty cycle.

CottonCandy mesh protocol: CottonCandy takes advantage of the fact that long-term measurements of environmental factors from multiple sensors generally do not require a reliable data delivery service and have some tolerance for packet losses. Rather than recovering packet losses through retransmissions, CottonCandy reduces the occurrences of packet losses by adapting to network conditions. Packet losses are reduced by avoiding packet collisions, that is, transmissions with overlapping frequencies and air times, and by performing flow control. Packet collisions are mitigated by distributed methods for channel selection, media access control, proximity-based neighbor discovery, and a request-driven method for data collection.

Packet collisions are trivially avoided by having nodes transmit on different frequencies. LoRa operates in unlicensed ISM bands (902–928 MHz in North America, 867–869 MHz in Europe, 470–510 MHz in China), from which LoRa radios

select a channel with a configurable bandwidth of 125, 250, or 500 kHz. In CottonCandy, each node selects a *private channel*, which is used for communication with the node's downstream neighbors. Information about private channels is exchanged over a *public channel* that is used by all nodes. By restricting the use of the public channel and by carefully coordinating transmissions, collisions on the public channel can be mostly avoided. Collisions are further reduced by imposing a limit on the maximum number of downstream neighbors of a node. CottonCandy uses a random access protocol with an adaptive random backoff delay before each transmission. Each node adapts the length of the backoff interval to the number of downstream neighbors to meet a target collision rate of 5 percent.

When a large number of nodes send data toward the gateway, data may accumulate at nodes close to the gateway, which increases the risk of congestion and packet collisions. This can be prevented by having upstream nodes prompt their downstream neighbors to upload data. Data collection can then proceed in a recursive fashion, where, starting at the gateway, nodes request data items from their downstream neighbors. When a node replies to the request, it subsequently requests data from its own downstream neighbors. This results in a recursive data collection process, which is illustrated in Fig. 2. The transmission of requests continues as long as there is data to be collected. The explicit requests for data act as a flow control mechanism, which prevents nodes close to the gateway from becoming overwhelmed with data items from downstream nodes.

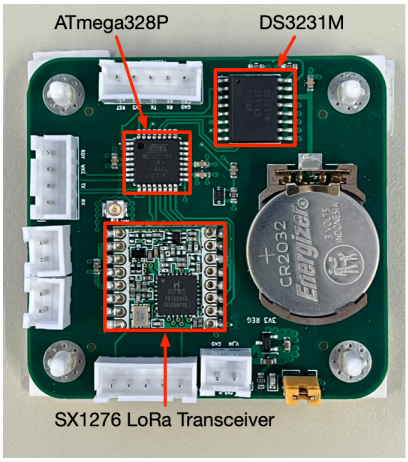
The CottonCandy protocol runs a distributed rooted spanning tree algorithm with the gateway node as the root of the tree topology. Here, a newly joining node only needs to identify an existing node that becomes its upstream neighbor in the tree topology. Joining the network does not require global coordination or configuration. Since nodes in a CottonCandy network are in hibernation most of the time, new nodes can only be added during a duty cycle. Thus, joining nodes must know when duty cycles commence. Additionally, nodes must learn the private channel of the node that becomes its upstream neighbor. Both types of information are sent by nodes that are currently part of the spanning tree at the start of a duty cycle (before data collection) on the public channel. A new node keeps listening to these messages. Once one of these messages is received, the new node hibernates until the start of the next duty cycle, at which point it knows how to contact a node on its private channel. When the new node contacts a CottonCandy node, it uses a low transmission power. If this is not successful, it incrementally increases the transmission power and retries in the next duty cycle. Since low transmission power translates into a small transmission range, this process reduces interference between nodes, thereby further reducing the likelihood of packet collisions.

The impact of single-node failures in the spanning tree network is minimized by the ability to self-organize. Failures or departures of nodes are handled using timeouts. If a node does not receive messages from its upstream neighbor for some time, it rejoins the network as a new node. If a node does not receive messages from one of its downstream neighbors for some time, it removes the neighbor from its neighbor table. For the given example in Fig. 2, when nodes A and B fail, nodes C–F no longer have an upstream neighbor that provides a path to the root, and they will rejoin the network as new nodes. This results in the formation of a new topology. In the particular example, one of the nodes must be within radio range of the gateway, such that it can rejoin the network as a downstream neighbor of the gateway. Once the node has restored connectivity to the gateway, other impacted nodes can rejoin the network as its downstream neighbors. If no node is within range of the gateway, there does not exist a path to the gateway, and the topology cannot be repaired.

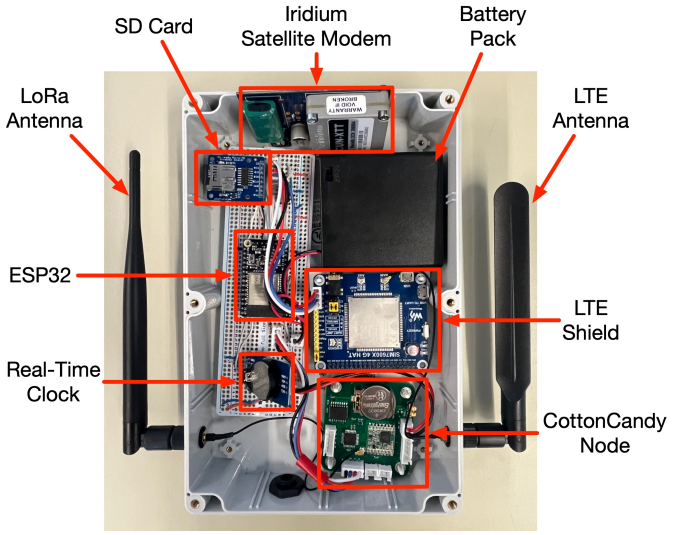
Regulations: Transmissions on unlicensed ISM bands have regional regulations on transmission power and channel usage. The maximum transmission power of CottonCandy nodes can be configured to comply with constraints on the so-called effective radiated power, which is specified in some regulations. In North America, the Federal Communications Commission (FCC) additionally imposes a 400 ms maximum dwell time on each channel. In Europe, the European Radiocommunications Committee (ERC), restricts the maximum utilization on most frequency sub-bands in an hour to 1 percent, meaning that a node can transmit for a total of 36 seconds every hour. The CottonCandy protocol is flexible enough to comply with such regional regulations. For example, dwell time limits can be satisfied by tuning physical-layer parameters and the maximum packet size. Multiple data collections in an hour can be avoided by setting the time between duty cycles sufficiently large. Further, the use of private channels splits transmissions of an individual node between its own private channel and the private channel of its upstream neighbor.

Hardware: We have designed a prototype of a CottonCandy mesh node with low-end commodity components. The node has an ATmega328P microcontroller (MCU) with only 32 kB flash memory and 2 kB RAM, which is connected to a Semtech SX1276 LoRa transceiver with default settings for LoRa PHY parameters ($\mathrm{SF} = 7, \mathrm{BW} = 125$ kHz, $\mathrm{CR} = 4/5$). A DS3231M real-time clock (RTC) provides timekeeping, which enables the MCU and LoRa transceiver to wake up from a deep sleep mode within and between duty cycles. At the start of a duty cycle, the RTC wakes up the MCU, which in turn switches on the LoRa transceiver. The hardware platform operates on 2 AA batteries with 3V input voltage. Figure 3a presents our custom-built circuit board of a CottonCandy node, which has a unit cost below US$15.

Energy Consumption: The CottonCandy mesh nodes are optimized for long-term operations on limited battery power. A node hibernates between duty cycles with a constant current draw at only 17 μA. Within each duty cycle, the energy consumption of a node largely depends on the number of its descendants, that is, the set of nodes that are downstream in the spanning tree topology. This is a consequence of the need to relay data items from descendants. We refer to [9] for detailed power profile and investigation of the relationship between energy consumption and location in a balanced tree topology. Table I summarizes the energy consumption in a 100-node



(a) CottonCandy mesh node.



(b) CottonCandy gateway.

Fig. 3. Devices of a CottonCandy network.

TABLE I
ENERGY CONSUMPTION OF COTTONCANDY NODES IN A DUTY CYCLE.

Number of descendants	Min. (mAh)	Avg. (mAh)	Max (mAh)
0	0.07	0.47	1.14
5	0.38	0.85	1.70
10	0.53	1.01	1.84
40	1.11	1.34	1.81
80	1.70	1.78	1.83

network for nodes with different number of descendants. Even for a node with 80 descendants, our simulations yield an energy consumption of less than 2 mAh in a duty cycle.

This allows CottonCandy nodes to operate for several thousand data cycles on two AA batteries. For an application with a few sensor readings per day, the battery lifetime hence exceeds one year.

Limitations: Several factors impact the performance of the CottonCandy protocol. The multi-channel communication enables orthogonal transmissions at different partitions of a spanning tree. Its effectiveness is essential to the packet delivery performance of the CottonCandy protocol, especially for large networks. Specifically, the risk of collisions is associated with the number of private channels with respect to the network density. A dense node placement with a small number of channels can incur a high collision rate. In principle, the CottonCandy protocol can support hundreds of nodes. However, increasing the number of nodes also increases the depth of the spanning tree. Since each additional hop prolongs the time to fetch data from the furthest nodes, deeper trees incur a higher energy consumption. We also note that the duration of data collection must not exceed the configured time interval between duty cycles. Given that every node transmits an 8-byte payload in each duty cycle, our evaluations of the protocol [9] show that a data collection for 20 descendants takes around 7 minutes to complete. For practical applications, which may take measurements only once every few hours, a limit of the network size to 100 nodes ensures that the data collection can be completed within 15 minutes which results in an energy consumption of $<$2 mAh in each duty cycle. The latter is crucial to support thousands of duty cycles on 2 AA batteries.

IV. MODULAR GATEWAY

The gateway in the CottonCandy IoT architecture has the dual roles of (1) collecting data items from nodes in the mesh network, and (2) transmitting these data items via the Internet to a cloud system. These roles are realized by a modular design with two components: a downlink module that collects sensor data from the mesh network, and an uplink module that forwards the collected data to a cloud system. Both modules run on separate MCUs.

The downlink module is a CottonCandy node that serves as the root of the spanning tree mesh topology. The ability to run as root is encoded in the address of the node. Whenever the downlink module receives a data item, it passes it on to the uplink module.

Operation of uplink module: The uplink module is responsible for establishing connectivity to the Internet, for formatting sensor data that arrives from the downlink module, and for uploading data items to the cloud. For Internet access, the uplink module may use WiFi, a cellular network, or a low earth orbit satellite system. The module is also equipped with an SD card for backup storage.

With WiFi and cellular, the gateway structures the received data in JSON format and publishes it via the MQTT protocol to a cloud-based MQTT broker. The published data is interpreted by a cloud function and posted to a database. The JSON data contains identifiers of the gateway and the node from where the data originated, a timestamp, and the sensor data sent by the node.

Due to the high cost per bit of satellite communications, the uplink module uploads raw data to the satellite network provider, without putting it in JSON format. From the satellite network provider, the data is forwarded to a cloud function via HTTP.

We emphasize that the gateway does not interpret data that it receives from the mesh network. The advantage of this is that the gateway does not need to be aware of specifics of the sensing application and sensor equipment.

Monitor and Control: In addition to uplink communication of sensor data from the mesh network to the cloud, there is a data path from the cloud to the gateway for performing monitor and control functions. With an MQTT broker, a system administrator can publish a request to query sensor nodes for battery status, location, as well as the status of sensors. Since the gateway is active only during the duty cycles of the CottonCandy network, commands to the gateway can be issued only during a duty cycle. To facilitate coordination with the gateway, the gateway indicates the start and end of a duty cycle to the cloud using a separate MQTT topic.

Interactions in a duty cycle: The downlink module controls the activity of the uplink module in a duty cycle. Between duty cycles, both modules hibernate in a low-power state. Within a duty cycle, whenever data arrives from the mesh network, the downlink module wakes up the uplink module and forwards the received data. The uplink module stores the data internally and then returns to a low-power state. When the downlink module has completed its data collection, it informs the uplink module and starts hibernation. Only then does the uplink module establish access to the Internet and transmit all data that was collected in the duty cycle. The batch transmission of data yields significant power savings, compared to uploading data to the Internet as it arrives from the downlink module. After transmitting the stored data on the uplink, the uplink module enters a low power state.

Hardware: Figure 3b depicts a prototype of the CottonCandy gateway with two MCUs: (1) an ATmega328P for the downlink module, and (2) an ESP32 for the uplink module. The ATmega328P runs the same code as any other CottonCandy node. The only difference to other CottonCandy nodes is that it acts as the root of the spanning tree mesh topology. The ESP32 has 520 kB RAM and 4 MB flash memory with an onboard WiFi radio. It is attached to a cellular shield and a RockBLOCK 9602 Iridium SatComm modem with an internal antenna. The two modules communicate with each other over a serial port. Additionally, the downlink module connects to two pins on the uplink module, which are used to indicate the arrival of new data items and the completion of data collection. The gateway also has an SD card for backup storage. The system is powered by 4 AA batteries.

The cloud system is hosted in the Google Cloud ecosystem. The MQTT broker is provided by Google Cloud IoT, processing is performed by Google cloud functions, and the database is realized with Google Firebase.

Power profile: Figure 4 provides power profiles of a gateway in a duty cycle in a small mesh network for WiFi, cellular, and satellite uplinks. Each duty cycle shows three distinct phases, which are separated in the graphs by vertical dashed lines. During data collection, which starts shortly after $t = 15$ s, power consumption is dominated by LoRa transmissions by the downlink module. Each LoRa transmission creates a spike of up to around 120 mA. When the data collection is completed, at around $t = 150$ s, the uplink module transmits all collected data to the Internet. Here, the consumed energy and the duration of the upload depends on the type of uplink. With WiFi (Fig. 4a), the upload is completed within 2 s, where the current draw is between 150–900 mA. With a cellular uplink (Fig. 4b), connecting to the Internet requires an initialization which extends the upload phase to more than 20 s, with current levels in the range 200–900 mA. The satellite module also requires an initialization. The initial spike in the upload phase observed in Fig. 4c is due to charging a supercapacitor on the satellite modem. Once the supercapacitor is charged, the current draw in the remainder of the upload phase is below 200 mA. For all uplink types, once data has been uploaded, both modules of the gateway hibernate in a low-power state with a current draw of 0.07 mA. For the satellite uplink, due to parasitic drain from the RockBlock satellite modem, the current is initially 0.19 mA, but then decreases to 0.07 mA.

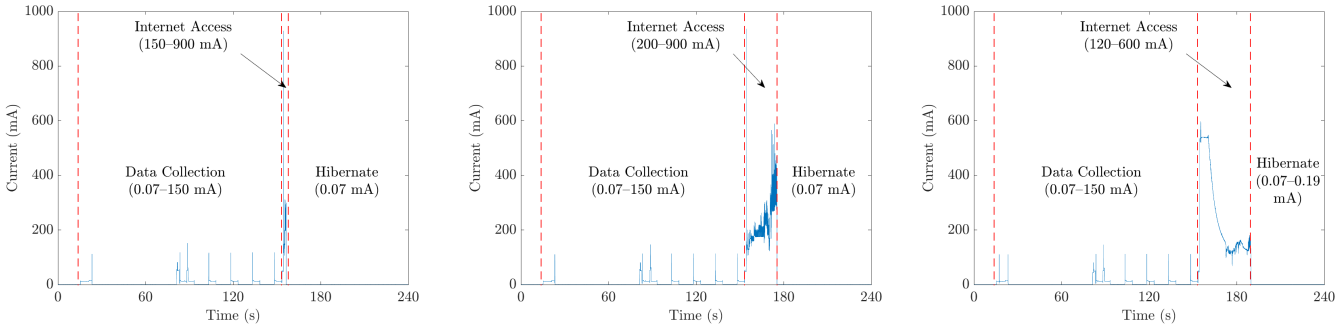
With the power profiles, we can estimate the energy consumption in a duty cycle for a mesh network with 100 nodes. Initialization of the uplink in the upload phase consumes about 0.03 mAh for WiFi, 1 mAh for cellular, and 2 mAh for satellite uplinks. Publishing an MQTT message with one data item over WiFi and cellular uplinks, requires around 0.06 mAh and 0.14 mAh, respectively. With this, we can infer that for WiFi and cellular, the gateway of a 100-node mesh network consumes at most 3–5 mAh in each duty cycle.

For a satellite uplink, each message transmission consumes 0.5–2 mAh. If one message is sent for each data item, the energy consumption in each duty cycle of a 100-node mesh network can exceed 100 mAh. This can be significantly reduced by aggregating data items. For example, with a maximum payload size of 340 bytes, we can fit 40 8-byte data items in a single transmission, thereby reducing the consumed energy to below 10 mAh per duty cycle.

V. Prototype Deployment

We have deployed a CottonCandy IoT network at the Koffler Scientific Reserve in King Township, Ontario, a biological field station that occupies 348 hectares of fields, wetlands, grasslands, and forest. The deployed network has 14 CottonCandy nodes (Fig. 5a) and one gateway (Fig. 5b) which are placed in open terrain of 70 hectares, mostly without unobstructed line of sight of each other. All devices are encased in waterproof IP65 enclosures with externally mounted antennas and mounted approximately one meter above ground at a distance of 100–300 meters to each other. Figures 5c–5f show the placement of nodes and the spanning tree topologies of CottonCandy that result from multiple restarts of the network. Each restart of the network, which is forced by powering down the gateway for some time and then powering back on, generally results in different spanning tree topology. The topologies show that some nodes select a geographically remote node as upstream neighbor. Here, the signal strength between nodes is less determined by geographical proximity than by terrain, vegetation, and line-of-sight.

The deployed network has been in operation from May until October 2022. To increase the amount of collected data, the frequency of duty cycles has been set to 10 minutes, and batteries of nodes were replaced after 7,500 duty cycles. In each duty cycle, each mesh node reported readings from a temperature sensor using messages with a 7-byte node header



(a) WiFi uplink. (b) Cellular uplink. (c) Satellite uplink.

Fig. 4. Power profiles of uplink modalities.

and an 8-byte payload. Node failures that have occurred occasionally have been caused by wildlife or mechanical issues (loose batteries). A detailed analysis of a measurements over a 10-day period with 1,500 duty cycles, showed a packet delivery ratio (PDR) between 89.5–98.5 percent [9].

Fault recovery of the network has been tested in experiments, where the gateway is turned off for several consecutive duty cycles, causing all nodes to become de-synchronized and attempt rejoining the network. After the gateway is restarted the nodes rebuild the network. In three performed trials of this experiment, all 14 (non-gateway) nodes rejoined the spanning tree within 5–7 duty cycles.

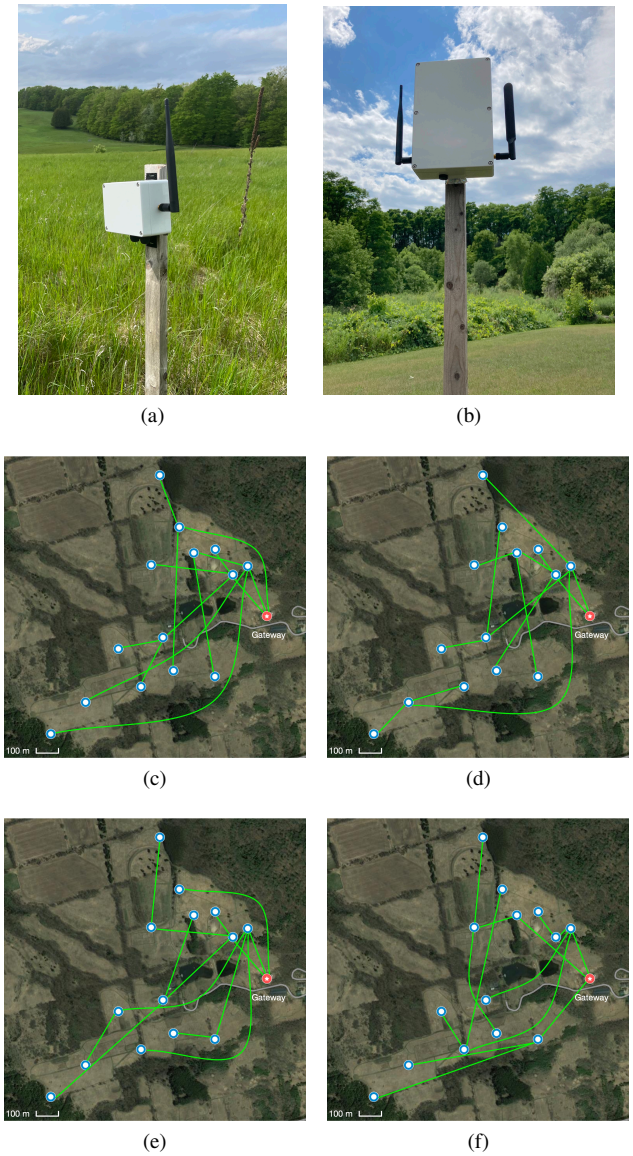
Benchmark: To demonstrate the scalability and coverage of the presented IoT architecture, we refer to the results of our large-scale simulation in [9]. The simulation deploys 100 nodes across 380 km^2 with a single gateway at the center. The distance between the gateway and the furthest node is over 12 km. In each duty cycle, each node uploads an 8-byte payload to the gateway. The results show that the mesh network can achieve an average PDR of over 90 percent, regardless of node location. To the best of our knowledge, so far there does not exist a LoRa mesh network solution that achieves similar coverage at this packet delivery performance.

VI. Challenges and Future Directions

The long range and low cost of LoRa has an enabling quality for low-cost large-scale IoT systems. We believe that the potential of LoRa technology, and, specifically, of LoRa mesh networks remains largely unexplored.

Network Scalability: Much future work is needed to understand the scalability limits of LoRa mesh networks. For example, a large number of hops increases the likelihood of packet collisions. This could be compensated by introducing a (hop-by-hop) ACK/NACK scheme with retransmissions. Presently, the CottonCandy mesh protocol does not attempt retransmissions due to their energy overheads. The main obstacle here is the access to a sufficiently large area and personnel to set up a network with more than 100 nodes.

Real-Time Data Reporting: Synchronized duty cycles are beneficial for energy efficiency, but they impose challenges for event-driven applications. Network activities are paused when nodes enter a deep sleep state between duty cycles. The



(a) (b) (c) (d) (e) (f)

Fig. 5. CottonCandy deployment at the Koffler Scientific Reserve: a) CottonCandy WSN node; b) gateway device; c–f) snapshots of various self-organizing topologies observed in our experiments on fault recovery of the network.

presented IoT solution currently does not support real-time reporting of data items, which is crucial to early-warning systems, e.g., forest fire monitoring. An ongoing effort is to adapt the mesh network for these event-driven environmental sensing projects by enabling on-demand data collection between duty cycles. One possible solution is to replace deep sleep with a low-power receiving mode that sniffs packets periodically at a low current draw.

Over-the-Air Firmware Update: The complexity of mesh networks introduces challenges to over-the-air firmware update, by which contents of the latest firmware are disseminated to a designated group of nodes. In a mesh network, matching firmware versions are critical for nodes on the same relaying path. Due to the limited bandwidth of LoRa, large file transfer can take minutes to complete even at a single hop. A firmware in a multi-hop mesh can incur significant energy overheads. At present, the ability of over-the-air firmware updates in LoRa mesh networks, e.g., with a reliable multicast protocol, remains largely unexplored.

VII. Conclusions

Precision agriculture and climate change research are two major application areas that can benefit from low-cost large-scale IoT systems. Many developing countries are vulnerable to a threat of droughts and a general lack of fresh water. IoT systems that monitor soil quality can enable a careful management of water resources, however, such systems must be affordable and scalable.

This paper has made a case for using multi-hop LoRa mesh networks to develop low-cost IoT solutions for in-situ environmental sensing of large geographical areas. The IoT system presented in this paper takes advantage of a LoRa mesh protocol that can form networks with one hundred nodes that cover an area of more than one hundred square kilometers and deliver more than 90 percent of packets, at a cost of less than US$15 per node. The paper also addresses several open challenges in the development of LoRa mesh networks and explores their potential solutions.

References

[1] L. Burton, K. Jayachandran, and S. Bhansali, "The 'real-time' revolution for in situ soil nutrient sensing," *Journal of The Electrochemical Society*, vol. 167, no. 3, 2020.

[2] F. Montori *et al.*, "Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues," *Pervasive Mob. Comput.*, vol. 50, pp. 56–81, 2018.

[3] K. Mekki *et al.*, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, 2019.

[4] The Things Network, "The Things network," 2019. [Online]. Available: https://www.thethingsnetwork.org/

[5] A. Haleem *et al.*, "Helium: A decentralized wireless network," Helium Systems Inc., Release 0.4.2, 2018. [Online]. Available: http://whitepaper.helium.com/

[6] F. Adelantado *et al.*, "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, 2017.

[7] Z. Sun, H. Yang, K. Liu, Z. Yin, Z. Li, and W. Xu, "Recent advances in LoRa: A comprehensive survey," *ACM Trans. Sen. Netw.*, vol. 18, no. 4, pp. 67:1–67:44, 2022.

[8] J. R. Cotrim and J. H. Kleinschmidt, "LoRaWAN mesh networks: A review and classification of multihop communication," *Sensors*, vol. 20, no. 15, p. 4273, 2020.

[9] D. Wu and J. Liebeherr, "A low-cost low-power LoRa mesh network for large-scale environmental sensing," TechRxiv, July 2022. [Online]. Available: 10.36227/techrxiv.20365050.v1

[10] J. Liebeherr, M. Valipour, and T. Y. Zhao, "Elements of application-layer internetworking for adaptive self-organizing networks," *Proc. IEEE*, vol. 107, no. 4, pp. 797–818, 2019.

[11] D. Lundell *et al.*, "A routing protocol for LoRa mesh networks," in *Proc. 19th IEEE WoWMoM*, 2018, pp. 14–19.

[12] C. Liao *et al.*, "Multi-hop LoRa networks enabled by concurrent transmission," *IEEE Access*, vol. 5, pp. 21 430–21 446, 2017.

[13] H. Lee and K. Ke, "Monitoring of large-area IoT sensors using a LoRa wireless mesh network system: Design and evaluation," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 9, pp. 2177–2187, 2018.

[14] X. Jiang *et al.*, "Hybrid low-power wide-area mesh network for IoT applications," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 901–915, 2021.

[15] A. Al-Fuqaha *et al.*, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surv. Tuts.*, vol. 17, no. 4, pp. 2347–2376, 2015.

Dixin Wu (S'17) received the MASc degree in Electrical and Computer Engineering at the University of Toronto in 2023. He is currently with Spero Analytics where he commercializes sensor network technologies for large-scale environmental sensing.

Alexandru S. Bogdan received the BASc degree in Engineering Science at the University of Toronto in 2022. He is currently with Advanced Micro Devices Inc. in Toronto, Canada.

Jörg Liebeherr (S'88, M'92, SM'03, F'08) received the Ph.D. degree in Computer Science from the Georgia Institute of Technology in 1991. He was on the faculty of the Department of Computer Science at the University of Virginia from 1992–2005. Since Fall 2005, he is with the Department of Electrical and Computer Engineering at the University of Toronto. His research interests are self-organizing networks and network calculus.