Burst-MAC: A MAC Protocol for Handling Burst Traffic in LoRa Network

Aakriti Jain
Department of Computer Science
Wayne State University
Detroit, MI, USA

Md Ashikul Haque
Department of Computer Science
Wayne State University
Detroit, MI, USA

Abusayeed Saifullah
Department of Computer Science
Wayne State University
Detroit, MI, USA

Haibo Zhang
Department of Computer Science
University of Otago
New Zealand

***Abstract*—While LoRa networks typically handle very infrequent communications, they can occasionally encounter sudden data bursts in response to unpredictable events such as a forest fire or volcanic eruption. LoRaWAN, the MAC protocol for LoRa, lacks a collision avoidance mechanism due to the severe energy constraints of the nodes. Burst traffic can result in substantial collisions, retransmissions, and packet loss when dealing with burst traffic, leading to rapid depletion of node batteries, increased network latencies, and significantly degraded throughput. In this paper, we propose Burst-MAC, a low-overhead MAC protocol for LoRa networks to efficiently handle burst traffic. When the network traffic is light and infrequent, Burst-MAC operates just like LoRaWAN. Upon detecting a data burst, it switches to burst mode that functions as follows: (i) nodes with bursty data are organized into virtual groups based on channel and spreading factor, and (ii) nodes within each group transmit in a semi-distributed TDMA fashion with each node determining its transmission time slots using a hash function to eliminate the need for centralized schedule distribution. Hash function collisions are resolved through piggybacking with acknowledgements from the gateway, ensuring bounded latencies for transmitted packets. We evaluate Burst-MAC through both physical experiments on LoRa devices and NS-3 simulations under various burst scenarios. Our results demonstrate that Burst-MAC significantly outperforms LoRaWAN and other baselines in several aspects, including a 4.5× improvement in packet reception rate, up to 6× reduction in energy consumption, and up to 2.5× decrease in latency.**

I. Introduction

The rise in demand for Internet-of-Things (IoT) applications has spurred the emergence of various Low-Power Wide-Area Network (LPWAN) technologies in recent years, with LoRa (Long Range) standing out as a prominent player [1], [2]. Operating within the unlicensed sub-GHz spectrum, LoRa facilitates long-range connectivity, low power consumption, and low data-rate transmissions. It enjoys global commercial availability, boasting over 600 documented use cases, spanning from smart sensing applications in civil infrastructures, such as health and well-being monitoring [3], to environmental monitoring such as forest fire monitoring [4], sailing monitoring [5], and agriculture monitoring [6], [7]. These applications typically require a network of many devices to be located far from central operations centers, in remote or hazardous locations with challenging terrain or offshore, where long-range and low-power communication is essential.

While LoRa applications typically involve very infrequent communications, they can occasionally experience sudden bursts of data. Burst traffic occurs when numerous devices simultaneously or rapidly send data packets within a short timeframe in response to an event, leading to a temporary surge in network activity. This scenario may occur, for instance, in smart cities during peak hours when numerous active devices generate a high volume of data. Another scenario arises during significant events or emergencies necessitating real-time data transmission from multiple devices. This often results from the activation of sensors deployed in affected areas, which promptly respond to events by generating burst traffic. These sensors, engineered to monitor and detect specific conditions, transmit real-time data to convey critical information during emergencies. For instance, in forest fire monitoring [8] or volcanic activity monitoring, numerous sensors are dispersed across large forest areas or near volcanoes to detect potential fires or eruptions, respectively. On normal days, these sensors transmit data packets infrequently. However, upon detecting a fire or eruption, sensors near the affected area begin generating burst traffic, likely indicating intense heat or smoke. This sudden uptick in data transmission plays a pivotal role in alerting authorities and facilitating timely responses such as evacuation in the event of a fire or eruption.

Currently, LoRa is not designed to handle burst traffic efficiently, since its *Medium Access Control (MAC)* protocol, known as LoRaWAN, adopts ALOHA, a straightforward protocol with no collision avoidance mechanism. Such a protocol makes the handling of burst traffic highly challenging as burst traffic will lead to massive collisions, retransmissions, and packet loss, which in turn will result in rapid battery depletion, excessive network latencies, and significantly degraded throughput. Simply applying CSMA (carrier sense

multiple access) or listen-before-talk (LBT) for LoRa [9], [10] approaches will also lead to massive collisions and huge energy consumption under packet burst. Although Time Division Multiple Access (TDMA) is effective in managing bursts or high traffic, blindly applying it to a large-scale LoRa network can severely compromise the overall network performance in latency, throughput, and energy efficiency. This is primarily because LoRa applications require the LoRa nodes to transmit quite infrequently, while burst traffic is a rare, unpredictable, and transient event. Additionally, the associated overhead related to time synchronization and schedule distribution would impose significant burdens on energy-constrained LoRa nodes.

In this paper, we propose ***Burst-MAC***, a low-overhead MAC protocol for LoRa networks to handle burst traffic. Burst-MAC overcomes the above mentioned challenges through the following novel mechanisms:

- Burst-MAC operates the same as LoRaWAN when the network traffic is light and swithes to burst mode once a burst is detected, thereby avoiding any time-synchronization requirement during normal periods. We propose methods to detect bursts at both the gateway and the LoRa nodes.
- Burst-MAC leverages the concept of virtually grouping bursty nodes based on channel and spreading factor to limit each node's collision domain within its group. Such a virtual grouping mechanism allows one node per each group to transmit concurrently by exploiting the parallel packet reception capability of the LoRa gateway using different channels or spreading factors.
- Burst-MAC employs a novel hash-based semi-distributed scheduling where nodes individually compute their transmission slots without centralized transmission scheduling. It employs a novel mechanism for resolving hash-function collisions through piggybacking in the downlink acknowledgements, thereby ensuring bounded latency.

We have evaluated Burst-MAC through both physical experiments on LoRa devices and NS-3 simulations under various burst scenarios. The results demonstrate that Burst-MAC significantly outperforms LoRaWAN and other baselines in several aspects, including a 4.5× improvement in packet reception rate, up to 6× reduction in energy consumption, and up to 2.5× decrease in latency.

The remainder of this paper is organized as follows. Section II presents an overview of LoRa. Section III reviews the related works. Section IV describes the system model. Section V presents the design of Burst-MAC. Section VI and Section VII present the experiment results and simulation results, respectively. Section VIII concludes the paper.

II. An Overview of LoRa

LoRa is a pioneering LPWAN technology designed for long-distance, low-power applications. In its protocol stack, *LoRa* and *LoRaWAN* are referred to as the physical layer (PHY) and the link layer, respectively. As shown in Fig. 1, its typical network architecture consists of one or more gateways, many end-devices (LoRa nodes), and a network server. Nodes are

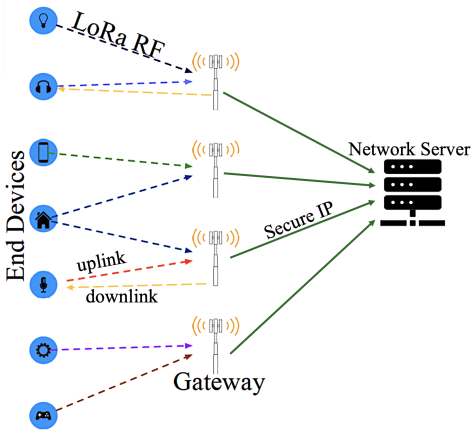


Fig. 1: The LoRa network architecture.

sensors/actuators that directly communicate with the gateway wirelessly over long-range. LoRaWAN divides the available spectrum into multiple *uplink* and *downlink* channels. Nodes send data to the gateway on the uplink channels and the gateway responds on the downlink channels.

1) LoRa Physical Layer : The LoRa PHY layer implements Chirp Spread Spectrum (CSS) modulation for communication. Spreading the signal over a wide bandwidth makes it less susceptible to noise and interference. A LoRa transceiver has five runtime-adjustable transmission parameters: bandwidth (BW), *spreading factor (SF)*, transmission (Tx) power, carrier frequency, and coding rate (CR). These parameters influence the transmission duration, energy consumption, reliability, and range. In the US, LoRa defines 64 uplink channels of 125 kHz bandwidth in 200 kHz increments in the 902.3-914.9 MHz band, and 8 downlink channels of 500 kHz bandwidth in 600 kHz increments in the 923.3–927.5 MHz band. SF determines the number of bits encoded in a symbol and ranges between 7 and 12. A higher SF increases the Signal to Noise Ratio (SNR) and, therefore, receiver sensitivity and the signal range, but lowers bitrate and lengthens communication time.

LoRa signal data rate is given by $\mathrm{SF} * \frac{4/(4+\mathrm{CR})}{2^{\mathrm{SF}}/\mathrm{BW}}$ whose value ranges between 0.3 kbps and 27 kbps. For the 915 MHz band in the US, the maximum Tx power is limited to 30 dBm (1 Watt) under the FCC regulations. In practice, LoRa nodes typically use lower Tx power (between 2 dBm and 20 dBm) to conserve battery. Under FCC regulations in the US, there is no duty-cycle requirement for spectrum utilization. However, all 125 kHz channels are subject to a maximum dwell time of 400 ms.

2) LoRaWAN MAC Layer: LoRaWAN supports three classes of operation: Class A, B, and C. *Class A* nodes are allowed for bidirectional communication where each node's uplink transmission is followed by two short downlink receive time-windows. Nodes locally make transmission decisions on a random time basis similar to the ALOHA protocol. Class A is the lowest power-consuming mode and intended for applications that require downlink communication from the gateway only after a node's uplink transmission. Downlink

communications from the gateway at any other time must wait until the next scheduled uplink. **Class B** nodes are also allowed for bidirectional communication, where they allow for more receive time slots for the nodes. Class B nodes open extra receive windows in addition to the Class A receive windows at pre-scheduled times. To resolve clock drifts, nodes synchronize their time with the gateway through periodic beacons. *Class C* nodes have continuously open receive windows that close only when transmitting. This mode consumes the highest energy but offers the lowest latency. Due to its middle-ground approach, Class B is the best choice for applications that require regular and timely server-initiated communication without the continuous power drain of Class C.

A message from a node to the gateway and vice versa may be *confirmed* or *unconfirmed*. For confirmed messaging, the sender requests an ACK from the receiver. When a node sends a confirmed message to the network server, it makes up to 8 Tx attempts until it gets an ACK. In unconfirmed messaging, a sender does not request an ACK from the receiver.

III. Related Work

Existing work on LoRa mostly focuses on improving performance [11]–[13], energy consumption and coverage [14]–[16], scalability [17]–[19], improving SNR [14], [20], lowering the required SNR to decode LoRa packets [21], and and packet collision resolution [22]–[29]. FTrack [23], mLoRa [25], and CoLoRa [26] use the temporal or spatial domain to resolve collisions. ScLoRa [29] utilizes cumulative spectral coefficient integrating both frequency and power information, to separate symbols in the overlapped signals of collided LoRa transmissions. Such collision recovery methods are able to decode a collision of only several packets and mostly rely on the PHY layer intervention. Since a burst scenario may lead to collisions of numerous packets, applying these collision recovery methods would be quite ineffective while consuming substantial time and energy for attempting to decode.

CSMA [30], LBT [9], [10], and random backoff mechanisms [31] have been explored in LoRa to reduce the probability of packet collisions. A collision-resolving MAC by imposing additional periodic beacons sent by LoRa gateways to reduce the occurrence of data packet collision has been studied in [32]. A CSMA-based mechanism has been introduced in [33] to alleviate severe uplink collisions. A traffic-aware channel and backoff window size allocation scheme has been studied in [34] to improve network capacity and latency by taking into account the fluctuation of different channel qualities and various traffic-buffer types for the optimum contention policy. The above protocols have not been designed to handle burst traffic. Specifically, CSMA, LBT, or backoff based approaches will lead to massive collisions, huge energy consumption, and unbounded latency under packet burst.

An on-demand TDMA method that employs low-energy wake-up radios to enable unicast and broadcast modes for node triggering and time slot allocation has been studied in [35]. The work [36] gave a TDMA-based control for LoRa multi-channel transmissions, incorporating an urgent ALOHA channel and negative ACK system to achieve a one-hop, out-of-band control plane for wireless sensor networks. [37] combined the time slotted channel hopping with TDMA to improve network throughput and dependability. While the TDMA approach is recognized for its effectiveness in managing bursts or high traffic due to high utilization of the time slots, blindly applying it to a large-scale LoRa network can severely degrade overall network performance in terms of latency, throughput, and energy efficiency. This is primarily because the applications require the LoRa nodes to transmit packets quite infrequently, while burst traffic is a rare, unpredictable, and transient event. Additionally, the associated overhead related to global time synchronization and centralized schedule creation and distribution would impose significant burdens on energy-constrained LoRa nodes.

We have proposed Burst-MAC to handle burst traffic by confining collision domain of each node within a small group of nodes, allowing concurrent transmissions from the groups (one node per group). It employs a semi-distributed TDMA approach limited within a small group of nodes, where nodes determine their transmission time slots using a hash function, eliminating the need for centralized schedule distribution, thereby significantly minimizing overheads associated with traditional TDMA approaches.

IV. System Model

In this paper, we consider an LPWAN based on LoRa under FCC regulations in the US, for various wide-area monitoring applications such as smart cities, forest fire detection, and volcanic activity monitoring. We consider a dense deployment consisting of n number of nodes, m_{up} uplink channels, m_{down} downlink channels, and s spreading factors. While the applications typically involve very infrequent communications, they can occasionally encounter sudden bursts of data. Burst traffic occurs when numerous devices simultaneously or rapidly transmit data packets within a short time frame in response to a sudden and unpredictable event, such as a forest fire or volcanic eruption. Burst traffic may be experienced by all or a fraction of the nodes. Namely, we consider a mixed criticality model, where the nodes near the epicenter of the event may experience burst traffic, while the nodes far away from the point of interest may not.

Burst is unpredictable and can occur anytime and can last for any duration of time (typically for a very short time). To ensure the reliability in data transfer, we adopt the confirmed uplink messaging of LoRaWAN. The nodes retransmit the packet if an ACK is not received. The maximum number of retransmissions is configurable. Note that the gateway acknowledges only the first received Tx of a packet. We do not allow the gateway to send multiple ACKs for multiple Tx attempts of the same packet. In Burst-MAC, we adopt the class-B mode of LoRaWAN in the nodes because of its ability to open extra receive time-windows. These additional windows are necessary to accommodate frequent ACKs during burst traffic.

LoRa nodes support multiple channels and can transmit on any available channel as specified by the link-layer protocol. A gateway can concurrently receive on different channels. According to the LoRa specification by Semtech [38], LoRa uses OVSF (Orthogonal Variable Spreading Factor), a variant of CDMA known for its orthogonality. Spreading factors (SFs) are orthogonal to each other, meaning that concurrent transmissions on different SFs do not interfere. Thus, a gateway can receive signals from multiple nodes on the same channel as long as they use different SFs. However, some studies suggest that LoRa's SF are quasi-orthogonal.

To verify orthogonality in practice, we conducted experiments in our laboratory with the following setup: two USRP gateways and two LoRa nodes as transmitters, utilizing a 915MHz channel, 125kHz bandwidth, SFs of (7, 10) and (7, 8), and a transmission interval of 2s. A total of 100 packets were transmitted. Our experiment showed that, with different SFs for the two concurrent Tx's (e.g., 7, 10; 7, 8) on the same channel, the receiver receives all the packets for both SFs. We conclude that different SFs are nearly 100% orthogonal, if not completely.

Based on Semtech's specification and our experimental validation, we consider orthogonal SFs in this paper. We exploit the parallel packet reception capability of a LoRa gateway by pairing channels and SFs. Each unique pair of a channel and a SF is called a *virtual channels (VC)*. Since the SFs are orthogonal, VCs are also orthogonal, i.e., transmissions on distinct VCs do not interfere with each other.

V. Design of Burst-MAC

In this section, we present an overview of our low-overhead MAC protocol, **Burst-MAC**, for handling bursty traffic in LoRa networks. Fig. 2 shows the workflow of Burst-MAC. Once a burst is detected, the gateway and the nodes experiencing burst switch to the burst mode, in which the LoRa nodes will switch to Class B mode and the gateway will start broadcasting beacon packets periodically to synchronize the nodes with bursty traffic. To reduce the probability of a burst packet loss due to collision between transmissions from burst and non burst nodes, the burst nodes use a higher Tx power for uplink transmissions than nodes operating in normal LoRaWAN mode, enabling capture effect at the gateway.

The gateway exploits LoRa's parallel packet reception capability on distinct VCs to divide the LoRa nodes into groups so that all nodes in the same group share the same VC. Within each group, the nodes experiencing burst traffic follow a semi-distributed TDMA fashion for packet transmissions, thereby obviating the need of any centralized schedule distribution, and minimizing energy consumption. Once synchronized with the gateway, each burst node determines its packet transmission time slot based on a pre-defined **hash function**. In Burst-MAC, the pre-defined hash function is shared by all nodes and the gateway. Hence, the gateway can pre-compute the schedules for all burst nodes and identifies those that might end up mapping to the same transmission time slots, thereby enabling the gateway to identify the colliding nodes. By monitoring

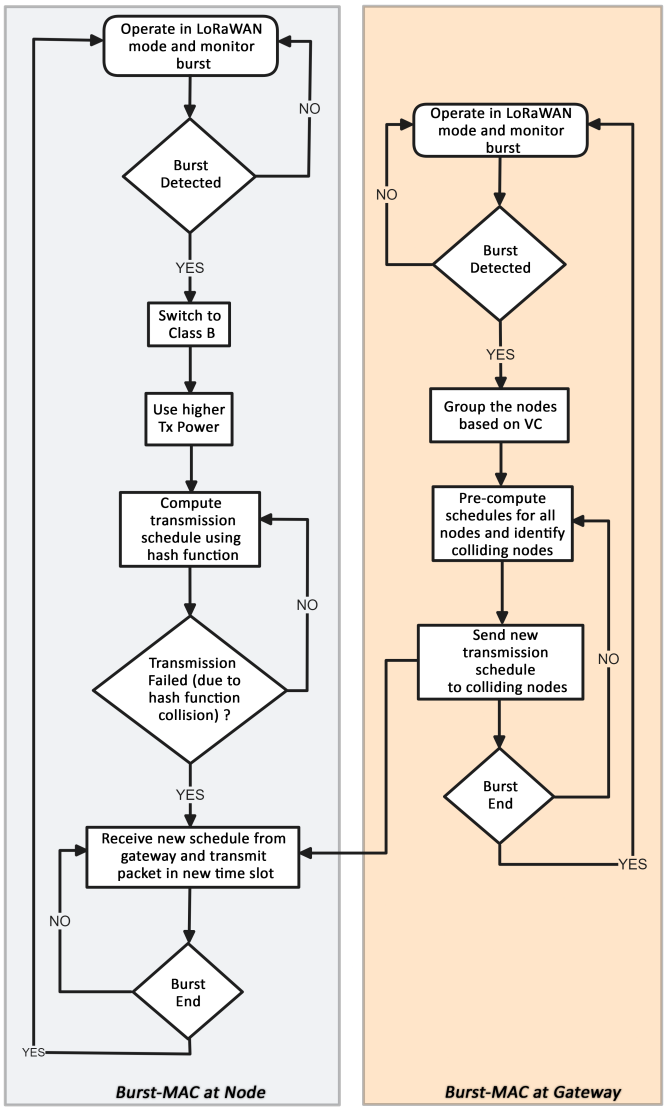


Fig. 2: Workflow of Burst-MAC

the time slots which are not being used by any nodes, the gateway assigns the unused transmission slots to the colliding nodes. When the burst is over, nodes and the gateway switch back to normal LoRaWAN mode. Burst-MAC also handles link failures, network and workload dynamics. In the following sections, we explain the steps involved in Burst-MAC in detail.

A. Determining the Start of Burst Traffic

In Burst-MAC, both the gateway and the nodes must detect the onset of burst since they need to switch from the normal LoRaWAN mode to Burst-MAC. The LoRa gateway identifies the onset of burst by monitoring the number of VCs experiencing collisions. During burst traffic, the probability of packet collisions increases significantly as many nodes begin transmitting packets in quick succession. The gateway determines that the network is experiencing burst if it observes that the number of VCs experiencing collisions exceeds a predefined threshold. Once the gateway switches to Burst-MAC,

it will piggyback a notification in the downlink communication to notify the burst nodes to switch to Class B mode. However, simply monitoring collisions on the VCs is insufficient for detecting burst, as there may be instances where only one node is transmitting on each VC, resulting in no collisions, even though the node is still experiencing burst. Therefore, to accurately determine the onset of burst, along with monitoring VC collisions, we utilize an explicit feedback from the nodes to the gateway, indicating about their burst status.

Each node detects the onset of burst by monitoring its packet generation rate. If its packet generating rate exceeds a predefined threshold, the node identifies itself as being in burst mode and switch to Burst-MAC. The burst nodes also inform the gateway about the burst status by sending explicit feedback through an uplink packet. Specifically, one bit in the packet's payload is reserved to indicate the burst status. Nodes set this bit to '0' if not experiencing a burst and to '1' if they are. When a node sends an uplink packet to the gateway, if the bit is set to '1', the gateway concludes that the node is experiencing burst traffic.

This packet transmission may not be entirely reliable since the uplink packet from the node may be lost due to collisions in a burst scenario. However, in emergency events like earthquakes or volcanic eruptions, many nodes undergo burst and transmit packets to the gateway. The gateway will receive at least some packets even if many packets end up colliding. It concludes that the network is in burst mode if it receives at least one packet in which the burst bit is set. At this point, the gateway may only know the presence of a burst in the network and not exactly which nodes are in burst. It can determine which nodes are in burst after a few rounds of communication.

Upon ascertaining the presence of burst, the nodes and the gateway switch to Class-B mode. After switching, the gateway broadcasts a time-synchronized beacon periodically, which is received by all the nodes in burst mode. These beacons provide a timing reference for the nodes, allowing them to synchronize their clocks with the gateway. Thereafter, they use Burst-MAC for subsequent transmissions. Note that, it is not required for the gateway and the burst nodes to switch to Burst-MAC at the same time. A burst node can switch to Burst-MAC once it detects burst or receives the notification from the gateway.

B. Grouping Nodes based on Virtual Channels

LoRaWAN allows parallel packet reception on multiple m_{up} channels. When combined with s SFs, a total of $s * m_{\text{up}}$ orthogonal virtual uplink channels are allowed for concurrent packet transmissions. Upon determining the onset of burst, the grouping kicks off and the gateway groups the nodes based on their VCs. Each VC forms a unique group and the nodes having the same VC are assembled into one group. Such a grouping allows one node from each group to transmit uplink packets concurrently, as nodes in different groups do not interfere with each other. Thereby each node's collision domain is limited within its group.

The number of VCs in a network is configurable. Typically in US, upto 64 m_{up} channels and six SFs (7-12) are permissible, thereby allowing a maximum of 384 VCs. Such a large number of VCs makes grouping efficient to limit collisions, as each group will have only a small number of nodes.

C. Hash-based Scheduling for Uplink Transmissions

After the grouping of nodes, the burst nodes within each group follow a TDMA schedule to send the burst messages to the gateway. An intuitive method for sending schedules to all nodes is to leverage a centralized scheduling approach. However, this approach presents several challenges, including (a) centralized scheduling at the gateway requires the dissemination of the constructed schedule to the bursty nodes in each group. In densely deployed networks, broadcasting the schedule to all nodes becomes cumbersome. The downlink packet would need to contain a schedule for each node in the network, resulting in a significantly large packet. Sending such a large downlink packet during burst is challenging primarily due to bandwidth limitations and energy constraints. Large packets also require more airtime, leading to substantial transmission delays. (b) In LoRa networks, the downlink capacity is typically more constrained than the uplink capacity. Transmitting a large packet to all nodes simultaneously may result in excessive energy consumption. If the packet is missed by a node, it needs to be retransmitted, further consuming excessive energy. This situation can negatively impact the overall performance of the network, leading to delays, packet loss, and reduced reliability. We address these challenges by proposing a hash-based semi-distributed scheduling policy where nodes individually compute their transmission slots.

In Burst-MAC, nodes operating in burst mode within a group use a hash function to determine their packet transmission schedules. Upon joining the network, all nodes are assigned the same hash function, which is also known to the gateway. The selection of the hash function for Burst-MAC is flexible and can be chosen by the network designer based on application requirements. The simplest hash function is $h(K) = k \ mod \ M$, where k is the node ID and M is the number of nodes in a group. The hash value is the time slot allocated to node k for uplink transmission. We define a *superframe* for each group where the total number of time slots equals the number of nodes in the group. Note that while the hash function is the same for all nodes, the number of nodes M can be different for different groups. For example, consider a group of 10 nodes with node IDs {1231, 1232, 1243, 1244, 1235, 1245, 1266, 1287, 1299, 1270}. Using the hash function, $h(K) = k \ mod \ 10$, the 10 nodes are mapped to the following time slots: 1,2,3,4,5,5,6,7,9,0 as shown in Fig. 3. Hence, the superframe contains 10 time slots. Each node uses its computed schedule for subsequent transmissions.

Using a hash function to compute transmission schedule minimizes overhead and complexity in Burst-MAC as the nodes autonomously compute their schedules, thus obviating the need of a centralized schedule dispersion policy. Hash functions also offer flexibility for accommodating changes or additions to the network. Nodes can easily adapt to new hash

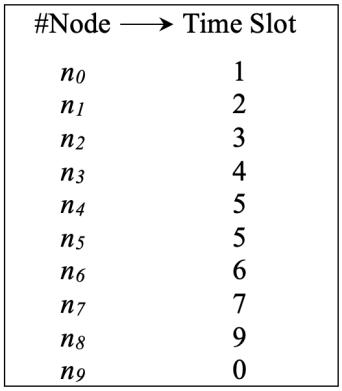
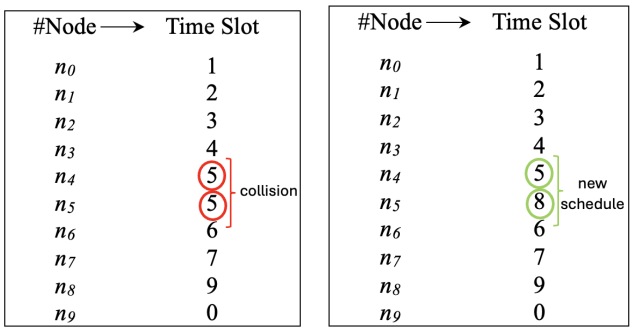


Fig. 3: Computing transmission schedule using Hash Function



(a) Nodes with colliding time slots (b) Dispersing new time slots

Fig. 4: Hash function time slot collision resolution

functions and recompute their schedules as needed, making Burst-MAC a dynamic and adaptable solution.

The time slot length for each node depends on the SF they operate on. Nodes are grouped based on the VC, ensuring that all nodes within a single group operate on the same SF and use time slots of the same length. The length of time slots varies across different groups as each group uses a unique SF. A single time slot may not always be sufficient for packet transmission, especially for nodes operating on higher SFs that may require longer slots. Therefore, we calculate time slot length for nodes based on their SF. Let L represent the length of the smallest time slot. Nodes with the smallest SF are assigned the smallest time slot; for instance, SF 7 corresponds to each time slot of length L. As the SF increases, the length of time slots also increases in a harmonic progression. For example, SF 8 is assigned a time slot of length $2L$, SF 9 is assigned a time slot of length $3L$, and so forth.

D. Ensuring Collision Free Transmissions

Hash functions have an underlying limitation of collisions, which occur when two different inputs produce the same hash value. In Burst-MAC, all the nodes experiencing burst in a group compute their transmission schedule using the pre-defined hash function. However, collision may occur if two or more nodes are hashed to the same time slot. For instance, in the above mentioned example of a group of 10 nodes, both n_4 and n_5 map to time slot 5, resulting in a packet collision if both nodes transmit in that slot. Such a situation is undesirable and needs collision resolution.

An intuitive approach involves using more complex hash functions to reduce the likelihood of mapping to the same time slot. Although these hash functions can decrease the probability of collisions, no hash function can ensure entirely collision-free time slot computation. To tackle this problem, we propose a collision resolution method in which the gateway identifies the colliding nodes and computes new schedules for the colliding nodes. The gateway holds information such as node IDs, the numbe of nodes in each group, and the hash function. By utilizing this information, the gateway can pre-compute the packet transmission schedules for all nodes experiencing burst. This allows the gateway to predict which nodes might end up sharing the same time slot, enabling proactive collision detection and resolution. Upon identification of colliding nodes i.e, the nodes mapping to the same time slot, the gateway sends a new transmission schedule to these nodes.

For each group, the gateway continuously monitors the usage of time slots within the superframe, tracking those that have not been assigned to any nodes. In a mixed criticality network scenario, a group may contain both burst and non-burst nodes. While burst nodes operate over Burst-MAC with scheduled transmissions, non-burst nodes function in normal LoRaWAN mode without following a schedule. The size of the superframe is based on the total number of nodes in a group, regardless of whether they are burst or non-burst nodes. Consequently, some superframes may have extra time slots that are not mapped to any burst nodes and remain unused. However, during sudden events like earthquakes or volcanic eruptions, most nodes enter burst mode, leaving each group with few or no non-burst nodes, and therefore very few unused slots. If certain time slots remain unused for an extended period, the gateway concludes that these slots are not mapped to any nodes and reassigns them to the colliding nodes as their new schedule. For instance, in the previous example, the superframe length is 10, meaning there are 10 available time slots for mapping. Nodes n_4 and n_5 are colliding because they both map to time slot 5. However, time slot 8 is not assigned to any node. The gateway sends a new transmission schedule to nodes n_4 and n_5, where node n_4 is reassigned time slot 5 while node n_5 is assigned new time slot 8. Now, the transmission is collision-free as each node transmits in a unique time slot as shown in Fig. 4.

In the first round of transmission, after the burst nodes transmit an uplink packet, the gateway sends ACK to the nodes. It sends a normal ACK to each of the burst nodes except the colliding ones. For the colliding nodes, it piggybacks the new time slot information along with the ACK. Burst nodes operating in Class-B switch to receive mode after uplink transmission to receive an ACK from the gateway. Nodes can receive ACKs in either of the two receive windows, RX1 and RX2, or in the ping slots. As illustrated in Fig. 5, Class B

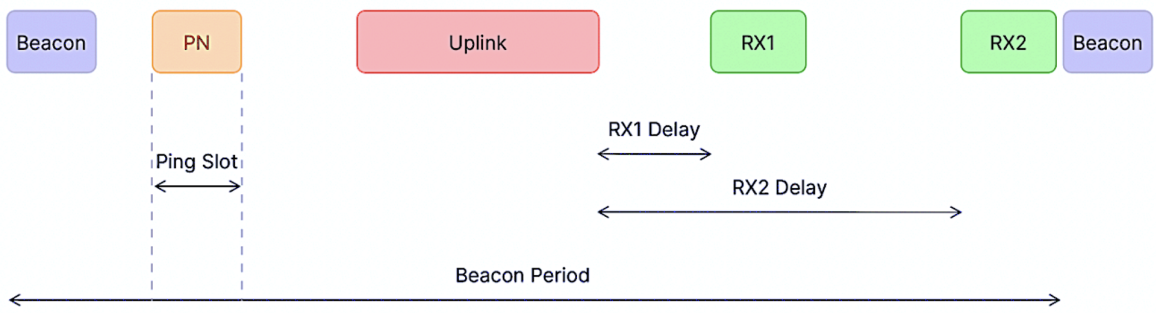


Fig. 5: Class-B operation mode in LoRa

mode enables a node to open additional receive windows at predetermined times, referred to as *ping slots*. Each ping slot is a designated time window during which the node listens for potential downlink messages from the gateway. By utilizing additional ping slots in Burst-MAC, nodes can frequently receive downlink messages without having to exclusively wait for RX1 and RX2, thereby increasing the likelihood of successfully receiving a downlink message during the first few rounds of collision resolution.

For the colliding nodes, the gateway does not need to receive uplink transmissions from these nodes to send downlink ACKs since it pre-computes the schedules for all nodes. Knowing the IDs of the colliding nodes, it transmits the new schedule piggybacked ACKs only to those nodes. These nodes then retransmit the collided packets using the new time slots. If any colliding nodes fail to receive the ACK and do not get their updated schedule, they will continue retransmitting the packet using the old/shared slot until the gateway assigns a new time slot. The maximum number of retransmissions may be determined by the network designer. By dynamically allocating the unused slots, Burst-MAC maximizes overall utilization, thereby enhancing the network's performance.

Furthermore, in a mixed criticality scenario where both burst and non-burst nodes coexist in a single group, transmissions from non-burst nodes could potentially interfere with packets from burst nodes operating on Burst-MAC. However, normal LoRaWAN communication is very infrequent, therefore the probability of such interference is low. Occasionally if interference occur, we resolve it by leveraging the *capture effect* that allows the stronger signal to be successfully received at the gateway even in the presence of interference. Specifically, in Burst-MAC, nodes in burst mode transmit packets at a higher Tx power than non-burst nodes. If a burst node and a non-burst node transmit packets simultaneously, the gateway prioritizes and processes the packets from burst nodes (sent at higher Tx power), treating the packets from non-burst nodes (sent at lower Tx power) as interference. This prioritization is crucial because transmissions from burst nodes are urgent messages indicating emergencies, which require prompt delivery over routine transmissions from non-burst nodes.

E. Downlink Communication in Burst-MAC

In a burst scenario, multiple nodes can transmit uplink packets to the gateway simultaneously. The ACKs for all these packets need be sent to the nodes on the downlink channel almost instantaneously. However, downlink communications for multiple nodes occur in sequel. Enabling ACK in such a scenario will significantly increase latency. Therefore, we propose to enable ACK only for the first few rounds of communication and disable it thereafter. It is necessary to enable ACK for the first few rounds to resolve any hash function collisions. It is reasonable to disable ACK after collision resolution since each burst node has been assigned a unique time slot in the superframe for uplink transmission. Hence, disabling ACK in a burst scenario does not significantly impact network performance. Moreover, in bursty scenarios, a new packet is imminent right after the previous transmission. Any previously undelivered or lost packet becomes irrelevant once a new packet arrives, as the new packet may hold the updated and most relevant information. Disabling ACK thus results in also disabling retransmissions, thereby reducing overall computation and energy overhead. Note that ACK is only disabled for burst scenarios; nodes operating in normal LoRaWAN mode still require ACK.

In regions such as Europe, with symmetric uplink and downlink channels i.e., $m_{\text{up}} = m_{\text{down}}$, each group of nodes receive downlink ACKs on the same VC as that of the uplink transmission. For example, group G_1 operating on VC 1 (physical channel 1 and SF 7), uses VC 1 for both uplink and downlink transmissions. However, in regions such as the US, with asymmetric uplink and downlink channels i.e., $m_{\text{up}} > m_{\text{down}}$, multiple groups of nodes use the same downlink channel to receive ACK from the gateway. In LoRa, the downlink channel bandwidth is much larger than the uplink channel, allowing the downlink channel to accommodate ACK for multiple nodes. The gateway encodes ACK for multiple nodes in one downlink packet and transmit it on downlink channel which is chosen per LoRa specification [39]. In Class-A and Class-B, the channel used in RX1 is calculated as uplink channel number % 8, and the SF used in RX1 is the same for uplink transmission. RX2 uses a fixed data rate and frequency, and the SF is configurable (default is SF 12 at 923.3MHz). A LoRa nodes does not open RX2 if it receives an ACK in RX1.

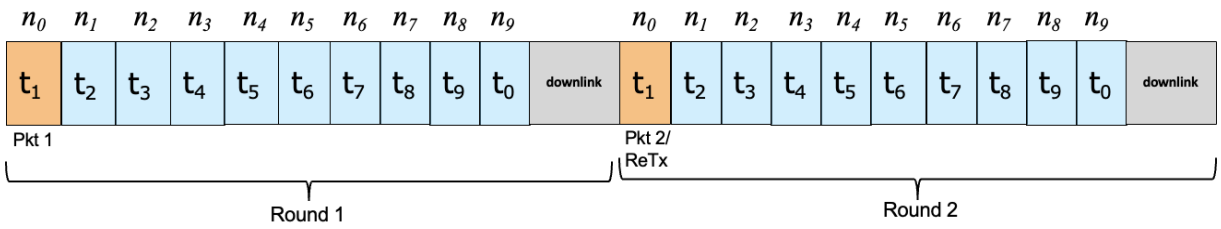


Fig. 6: Timeslots in Burst-MAC

In burst scenario, if at most one node per group (i.e. nodes using the same VC) transmits, the gateway will follow the above procedure to send the ACK in the two RXs. If the node fails to receive the downlink in RX1, RX2 is configured to the SF same as the uplink to send the aggregated ACK. If more than one node in a group transmits simultaneously, a collision occurs at the gateway. The gateway either fails to demodulate all these packets or demodulate only the packet transmitted with the highest TX owing to capture effect. In such a case, the gateway only needs to send at most one ACK for this group and still follows the above procedure to send the ACK.

F. Handling Network and Workload Dynamics

Once a schedule is computed for a burst node and a time slot is assigned, the node uses this slot to transmit its packet and then sleeps until the next packet arrives. If a node has multiple packets to transmit, it uses the same time slot in subsequent rounds. For example, node n_0 is assigned time slot t_1, it transmits its first packet during this slot and sleeps for the remainder of the superframe length (t_2-t_0). The node wakes up again at time slot t_1 in round 2 superframe cycle to transmit the second packet, if available (Fig. 6). If no new packet is available, the node may either sleep or attempt to retransmit a previously failed packet, but only if the failure occurred within the first few rounds before ACK is disabled. If a new packet is available, the node drops the old packet and transmits the new one in the assigned time slot.

A node in the network can switch from non-burst to burst mode at any time. Within a group, both burst and non-burst nodes coexist. Since the superframe length is based on the total number of nodes in the group, it remains fixed even when a node's status changes. However, the used and unused time slots within a superframe can vary with this change. A time slot previously mapped to an idle non-burst node will now be utilized by a node that has switched to burst mode. Burst-MAC minimizes unused slots by dynamically reassigning them to burst nodes, which is particularly beneficial when there are very few burst nodes in a group. For example, if only 1 out of 10 nodes in a group is in burst mode, the superframe length is still 10 time slots. Initially, 9 of these slots are unused. After a few rounds, the gateway assigns these unused slots to the burst node. As a result, the burst node does not have to wait for the next superframe cycle and can transmit multiple packets within a single superframe. If a new node joins the network at any time, it is assigned a VC, and the grouping is redone. Consequently, the superframe length and time slot assignments are adjusted accordingly.

Furthermore, if a node switches to burst mode while some nodes in its group are already in burst, it can join the burst pool if its burst appears before collisions have been resolved, since ACK is still enabled. However, if a new node switches to burst mode after collisions have been resolved, it may not receive a time slot before the current burst ends because ACK is disabled. Such a scenario is rare in practice since typically nodes in an area experience bursts nearly simultaneously (e.g., during events like earthquakes). Burst-MAC can be made capable of handling this scenario by periodically enabling ACK.

G. Handling the end of Burst Traffic

As traffic decreases and the burst scenario ends, the nodes and the gateway switch back to regular LoRaWAN from Burst-MAC. The nodes monitor the reduction in packet generation. Once it falls below a certain threshold, the nodes transition from burst to non-burst mode. Specifically, the nodes switch back to Class-A from Class-B mode of operation and no longer rely on scheduled packet transmission. Nodes also send an explicit feedback by setting the payload bit to '0' to inform the gateway of this transition. The gateway observes an increase in unused schedules as nodes switch to non-burst mode. Using this information and the explicit feedback from the nodes, the gateway concludes that the nodes have exited burst mode. Consequently, the gateway also reverts to normal LoRaWAN operation and stops sending schedules. In case the gateway exit Burst-MAC and some burst nodes are still operating in Class-B mode, these nodes will exit Burst-MAC if they didn't receive any beacons from the gateway.

H. Performance Analysis of Burst-MAC

As illustrated in Fig. 5, a burst node will synchronize with the gateway once it receives a beacon packet, and then it will compute its transmission slot using the pre-defined hash function. Let N_i be the number of nodes in group G_i using spreading factor i and n_i be a burst node in G_i. Since the length of the superframe for G_i is N_i, in the worst case, n_i is assigned slot N_i for its uplink transmission. Suppose the uplink transmission from n_i fails because multiple burst nodes are assigned to transmit in the same time slot. The gateway will find unused slots and assign them to the colliding nodes. In the worst case, the other nodes that collided with n_i are assigned new slots before N_i and n_i remains to use slot N_i.

Since the length of the superframe is equal to the number of nodes in a group and not all nodes in a group can be in burst mode in the same superframe, each burst node in G_i can be assigned a distinct slot once collisions are detected at the gateway. Assuming the downlink communication for rescheduling doesn't introduce extra delay and uplink packets will not be lost due to other reasons except collision, in the worst case the first packet from n_i can be received by the gateway with in $2N_i$ times slots after receiving a beacon packet. Once collisions are resolved, the maximum latency for delivering each of the subsequent packet is N_i time slots.

VI. Proof-of-Concept Experiments

In this section, we evaluate the performance of Burst-MAC through small-scale physical experiments to demonstrate the viability of our approach.

A. Setup

We conduct an experiment to evaluate our approach in an indoor setup as depicted in Fig. 7. We employ five Dragino LoRa shields [40] with Arduino Uno R3 [41] as the LoRa nodes and a USRP B200 [42] as the LoRa gateway. We use a single 915 MHz LoRa channel to mimic a network with a large number of nodes. Furthermore, we transmit one packet every 10 seconds from each node to occupy the channel almost all the time. Regarding other parameters, we use a spreading factor of 7, a bandwidth of 125 kHz, and a coding rate of 4/5 in our experiment. We use LoRaWAN as a baseline to compare our approach.

B. Metric

For evaluation, we employ the following metrics.

1) Packet Reception Rate: We compare the Packet Reception Rate (PRR) for Burst-MAC against LoRaWAN. PRR is defined as the ratio of total number of packets received at the gateway to the total number of packets transmitted by the LoRa nodes.

$$\text{Packet Reception Rate} = \frac{\text{Total Received Packets}}{\text{Total Transmitted Packets}}$$

2) Energy: We define energy consumption as the total amount of energy dissipated by a node to successfully transmit an uplink packet to the gateway. We measure the average energy consumption per packet per node for both our approach and regular LoRa as,

$$\text{Energy} = \frac{\text{Total energy for transmission and retransmission}}{\text{Number of successful packet transmission}}$$

3) Latency: We measure latency for uplink packets which are successfully received by the gateway. We evaluate the latency per node for both Burst-MAC and LoRaWAN.

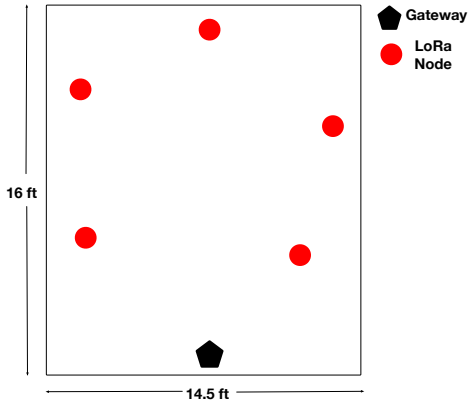


Fig. 7: Setup.

C. Results under Varying Percentage of Burst Nodes

We vary the percentage of nodes under burst conditions and measure the packet reception rate, energy consumption per packet, and end-to-end latency. We present the results in Fig. 8. It can be seen that, when the percentage of burst nodes increases, the packet reception rate achieved by Burst-MAC drops slightly but always stays above 80%, whereas the packet reception rate achieved by LoRaWAN drops significantly, achieving less than 10% when all five nodes are in burst mode. Compared to LoRaWAN, Burst-MAC demonstrates a very high packet reception rate for all percentages of burst nodes.

Fig. 8(b) shows the energy efficiency of Burst-MAC in comparison with LoRaWAN. It can bee seen that the average energy consumption per received packet in Burst-MAC only slightly increases when the percentage of burst nodes increases. However, average energy consumption per received packet in LoRaWAN increases exponentially. This is attributed to the lower packet loss and having little to no overhead for the usage of time slots in Burst-MAC.

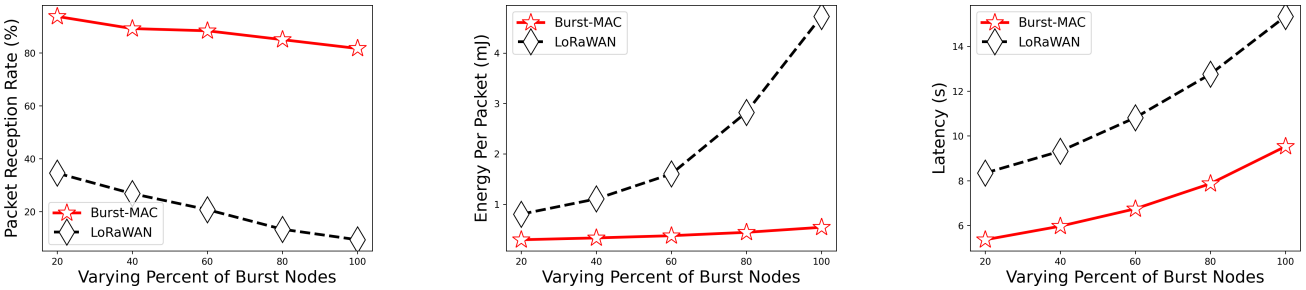
Fig. 8(c) shows that Burst-MAC outperforms LoRaWAN regarding end-to-end latency owing the high packet reception rate. A single packet loss can add 5.042 seconds of latency for a spreading factor of 7, which quickly accumulates and results in very poor performance for LoRaWAN. However, the end-to-end latency stays reasonable for our approach, remaining below 10 seconds even when all nodes are in burst mode.

VII. Simulation

In this section, we evaluate the performance of Burst-MAC through large-scale simulations. We simulate the LoRa network in NS-3 with a single gateway and up to 1000 nodes, and present the results under various burst scenarios.

A. Setup

We use the LoRaWAN NS-3 module proposed in [43] for our simulation. The NS-3 module offers classes designed for simulating modulation and medium access technology of LoRaWAN network, supports simulations featuring a large number of devices and offers easy monitoring of network



(a) Packet Reception Rate (b) Energy Per Packet (c) Latency Per Packet

Fig. 8: Experimental results under varying percent of burst nodes.

performance. We use a single gateway and upto 1000 nodes, where nodes are positioned randomly across a disc centered at the gateway with radius up to 6 km. We use SF 7, 8, 9, 10 and do not use spreading factors greater than 10 due to the US dwell time regulation. The SFs are assigned to the nodes based on their distance from the gateway. The nearest node is assigned the smallest SF (7), while the farthest node is assigned SF 10. We use a bandwidth of 125 kHz and a coding rate of 4/5. We use $h(K) = k \ mod \ M$ as the hash function in simulations, where k is the node ID and M is the number of nodes in a group. The number of nodes in a group varies according to the number of VC used. We set the frequency of burst to 10s i.e., a packet is transmitted every 10s from each burst nodes. We employ the following burst scenarios: (i) varying the percentage of nodes experiencing burst in the network, (ii) varying the total number of nodes in the network, and (iii) varying the number of virtual channels for uplink transmissions.

B. Baselines and Metrics

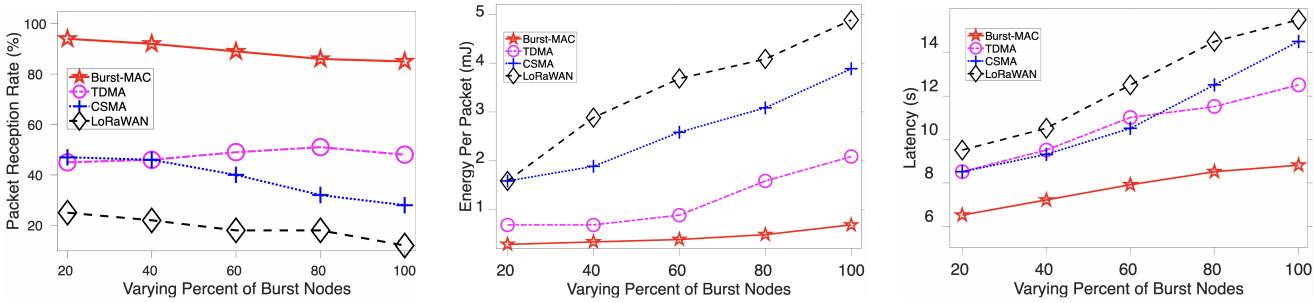
We compare Burst-MAC with LoRaWAN, as well as with CSMA and pure TDMA, which are commonly used approaches in the literature to reduce collision probability in densely deployed LoRa networks. We utilize the same metrics as our experiments: PRR, energy consumption, and latency for each scenario.

C. Results

1) Results under varying percent of burst nodes: Fig. 9 shows the results by varying the percentage of burst nodes. In this simulation, the number of LoRa nodes is set to 200 and we vary the percentage of burst nodes from 20% to 100%. The frequency of burst is set to 10 s and number of VC used is 32. It can be seen that Burst-MAC outperforms the other three baselines on all three metrics.

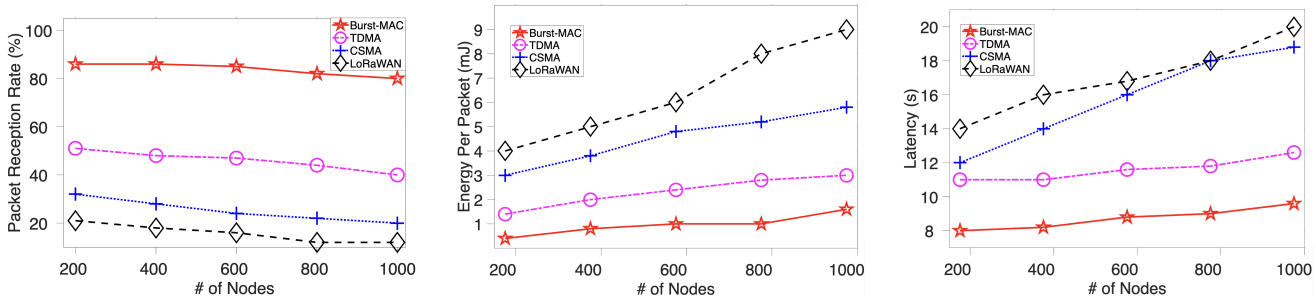
In Fig. 9(a), we compare the PRR of Burst-MAC with the baseline methods. We observe that as the percentage of burst nodes increases, the PRR achieved by Burst-MAC decreases slightly but remains above 85%, whereas the PRR of the other baselines drops significantly. Under LoRaWAN, we observe a low PRR of 10% when 100% of the nodes in the network are experiencing burst. This low PRR is due to LoRaWAN's use of a pure ALOHA protocol without collision avoidance, leading to a significant number of collisions when all nodes in the network are in a burst mode. When the percentage of burst nodes is relatively low (e.g. below 40%), CSMA performs slightly better than TDMA. However, when the percentage of burst nodes further increases, TDMA performs much better than CSMA, since the random backoff mechanism in CSMA is no more efficient than TDMA for reducing packet collisions in such highly bursty scenarios. In contrast, Burst-MAC surpasses both TDMA and CSMA by maximizing the utilization of time slots. It achieves this by assigning unused slots back to bursty nodes, ensuring that most packets are scheduled collision-free even in highly bursty scenarios. This approach reduces the need for retransmissions and increases the PRR for Burst-MAC. Overall, these results demonstrate that while the PRR of Burst-MAC decreases with an increase in the percentage of burst nodes, it still maintains a good PRR compared to the baselines. On average, Burst-MAC improves the PRR by 4.5× compared to LoRaWAN, 2× compared to TDMA, and 2.5× compared to CSMA.

In Fig. 9(b), we compare the per-packet energy consumption of Burst-MAC with the baseline methods. We observe that increasing the percentage of burst nodes in the network does not significantly impact the energy consumption of nodes under Burst-MAC compared to the baselines. Under Burst-MAC, the energy consumed per packet remains relatively stable, increasing from 0.2 mJ to 0.6 mJ even when all nodes in the network experience burst. This stability is attributed to our autonomous scheduling policy, which uses a simple hash function to compute schedules and disables ACK. In contrast, the energy consumed per packet for TDMA increases from 0.6 mJ to 2 mJ when the percentage of burst nodes increases from 20% to 100%. This increase is due to TDMA's centralized schedule distribution approach, which incurs significant energy overhead. CSMA and LoRaWAN exhibit high energy overhead, reaching up to 3.8 mJ and 4.8 mJ, respectively, when 100% of nodes are in burst. This high energy consumption demonstrates that these approaches are inefficient under highly bursty scenarios. Overall, these results



(a) Packet Reception Rate (b) Energy Per Packet (c) Latency Per Packet

Fig. 9: Simulation results under varying percent of burst nodes.



(a) Packet Reception Ratio (b) Energy Per Packet (c) Latency Per Packet

Fig. 10: Simulation results under varying number of nodes.

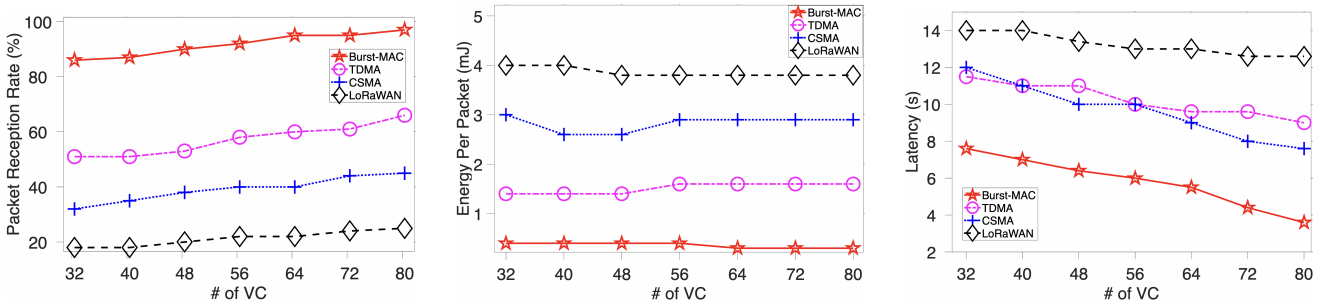
highlight the high energy efficiency of Burst-MAC compared to the baseline methods.

In Fig. 9(c), we compare the latency of Burst-MAC with the baseline methods. We observe that the latency increases for all approaches as the percentage of burst nodes increases. This increase is attributed to the high number of collisions and retransmissions in burst scenarios. The latency increase is almost exponential for CSMA and LoRaWAN, while it remains reasonable for Burst-MAC, staying below 8 s even with 100% nodes under burst. This result demonstrates that Burst-MAC outperforms the baselines in maintaining a high packet reception rate.

2) Results under varying number of nodes in the network: In this simulation, we vary the number of LoRa nodes from 200 to 1000. The percentage of nodes under burst is set to 80%. The frequency of burst is 10 s and number of VC used is 32. Fig. 10 presents the results under varying number of nodes in the network.

In Fig. 10(a), we observe a decrease in PRR for all approaches as the number of nodes increases. However, Burst-MAC maintains a significantly better PRR compared to the baseline methods. Even with a network of 1000 nodes, out of which 80% are under burst, Burst-MAC achieves an 80% PRR. In contrast, for the same scenario, the PRR of LoRaWAN drops to 12%, while CSMA and TDMA achieve 20% and 40%, respectively. This result is attributed to the efficient grouping of nodes under Burst-MAC, which localizes collisions within a group. This grouping ensures that even with an increasing number of nodes in the network, only a small number of nodes remain within a group, reducing collisions and increasing overall PRR. In contrast, collisions in all other baseline methods occur across the entire network, leading to a decline in PRR as the number of nodes increases. Burst-MAC outperformsTDMA, CSMA, and LoRaWAN by approximately $2\times$, $3.5\times$, and $4\times$, respectively. This demonstrates that Burst-MAC effectively handles burst traffic and is scalable. In Fig. 10(b), we observe that as the number of nodes in the network increases, the average energy consumption per packet for all approaches also increases. TDMA and Burst-MAC exhibit similar trends since Burst-MAC is based on TDMA. However, Burst-MAC significantly outperforms TDMA in energy consumption, using 0.4 mJ per packet for 200 nodes and 1.6 mJ per packet for 1000 nodes. In contrast, TDMA's energy consumption increases from 1.4 mJ per packet for 200 nodes to 3 mJ per packet for 1000 nodes. This is because TDMA requires the gateway to dissipate schedules to all the nodes, increasing energy consumption. Burst-MAC overcomes this drawback by implementing a hash-based semi-distributed scheduling. With CSMA, energy consumption reaches 5.8 mJ per packet for 1000 nodes, whereas LoRaWAN shows the highest consumption at 9 mJ per packet for the same number of nodes. Overall, these results highlight improved energy efficiency with Burst-MAC in dense deployment scenarios.

In Fig. 10(c), we compare the per-packet latency of Burst-MAC with the baselines. We find that Burst-MAC maintains a reasonable per-packet latency, remaining below 10 s even



(a) Packet Reception Ratio (b) Energy Per Packet (c) Latency Per Packet

Fig. 11: Simulation results under varying number of Virtual Channels.

in highly bursty scenarios. The latency for TDMA remains nearly constant because, even with an increased number of nodes. However, the latency for TDMA is higher than that of Burst-MAC because TDMA is susceptible to collisions in highly bursty scenarios. Burst traffic is unpredictable and transient, leading to retransmissions and increased latency for TDMA, which is inherently best suited for networks with predictable high traffic. Notably, CSMA performs similarly to LoRaWAN. This is because carrier sensing becomes inefficient when packets are generated at a high frequency during bursts for a large number of nodes, leading to packets waiting while nodes sense the channel.

3) Results under number of virtual channels: In this simulation, we vary the number of virtual channels for uplink transmission. We set the number of LoRa nodes in the network to 200, the percentage of burst nodes to 80%, and the frequency of burst to 10 s. Fig. 11 presents the results across various metrics under varying number of virtual channels.

In Fig. 11(a), we note that increasing the number of virtual channels leads to an increase in PRR for all approaches. For Burst-MAC, the PRR reaches 97% when the VCs are increased to 80. This improvement is attributed to the higher number of virtual channels, which results in fewer nodes in each group. Consequently, collisions within a group are reduced, leading to more packets being successfully transmitted to the gateway. While a similar trend is observed for all other approaches, the increase in PRR for LoRaWAN is minimal, only 25% with 80 VCs. Overall, these results demonstrate that Burst-MAC performs, on average, 3.5× better than LoRaWAN and 2× better than TDMA and CSMA.

In Fig. 11(b), we compare the per-packet energy consumption of Burst-MAC with baselines. We observe that Burst-MAC maintains the lowest energy consumption among all the methods. Average energy consumption per received packet in Burst-MAC remains almost constant at 0.4 mJ and even slightly decreases to 0.3 mJ as VCs increases. This is because the increase in VCs allows nodes to be grouped more efficiently, with fewer nodes per group, leading to a higher PRR. The addition of more VC does not incur any energy overhead in Burst-MAC. For TDMA, energy consumption slightly increases to 1.6 mJ as the number of VCs increases, due to the gateway needing to manage more schedules. CSMA shows a slight decrease in energy consumption, due to reduced collisions and shorter wait times with more VCs. Energy consumption for LoRaWAN also decreases slightly for similar reason. Overall, these results demonstrate that Burst-MAC is the most energy-efficient approach, performing approximately 6× better than LoRaWAN.

In Fig. 11(c), we observe that latency decreases for all approaches as the number of virtual channels (VCs) increases. Burst-MAC exhibits the lowest latency at 3.6 s with 80 VCs. This reduction in latency is because more VCs allow more nodes to transmit simultaneously, resulting in less contention and fewer collisions.

VIII. Conclusion

In this paper we present ***Burst-MAC***, a low-overhead MAC protocol for LoRa to handle burst traffic. Burst-MAC confines collision domain of each node within a small group of nodes, allowing concurrent transmissions from the groups. It employs a semi-distributed TDMA approach limited within a small group of nodes, where nodes determine their transmission time slots using a hash function, eliminating the need for centralized schedule distribution, thereby significantly minimizing overheads associated with traditional TDMA approaches.

By considering mixed criticality scenarios, our approach provides flexibility in handling different levels of node priority during burst and non-burst periods. We can efficiently manage packet assignments, prioritizing burst nodes when necessary and dynamically adjusting the schedule as per the network's changing requirements. Overall, our elaborate approach addresses the complexities of burst and non-burst periods in LoRa networks. By integrating synchronization mechanisms, power considerations, and dynamic scheduling, we achieve robust and efficient communication while minimizing the chances of collisions and maximizing network performance.

Acknowledgement

The work was supported by NSF through grants CAREER-2306486, CNS-2306745, CNS-2301757, and by ONR through grant N00014-23-1-2151.

REFERENCES

[1] J. P. Bardyn, T. Melly, O. Seller, and N. Sornin, "Iot: The era of lpwan is starting now," in *ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference*, 2016, pp. 25–30.

[2] https://www.i-scoop.eu/internet-of-things-guide/iot-network-lora-lorawan/.

[3] J. Petajajarvi, K. Mikhaylov, R. Yasmin, M. Hämäläinen, and J. Iinatti, "Evaluation of lora lpwan technology for indoor remote health and wellbeing monitoring," *International Journal of Wireless Information Networks*, vol. 24, 06 2017.

[4] Y. Li, Z. Wang, and Y. Song, "Wireless sensor network design for wildfire monitoring," in *2006 6th World Congress on Intelligent Control and Automation*, vol. 1, 2006, pp. 109–113.

[5] L. Li, J. Ren, and Q. Zhu, "On the application of lora lpwan technology in sailing monitoring system," *2017 13th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pp. 77–80, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:16850262

[6] 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.

[7] S. Benaissa, P. A. David, E. Tanghe, J. Trogh, L. Martens, L. Vandaele, L. Verloock, F. Tuyttens, B. Sonck, and W. Joseph, "Internet of animals: characterisation of lora sub-ghz off-body wireless channel in dairy barns," *Electronics Letters*, vol. 53, pp. 1281–1283, 2017.

[8] 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.

[9] J. Ortín, M. Cesana, and A. Redondi, "Augmenting lorawan performance with listen before talk," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3113–3128, 2019.

[10] ——, "How do aloha and listen before talk coexist in lorawan?" in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1–7.

[11] R. Oliveira, L. Guardalben, and S. Sargento, "Long range communications in urban and rural environments," in *2017 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2017, pp. 810–817.

[12] T. Voigt, M. Bor, U. Roedig, and J. Alonso, "Mitigating inter-network interference in lora networks," in *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*, ser. EWSN '17, 2017, pp. 323–328.

[13] M. A. Haque and A. Saifullah, "A game-theoretic approach for mitigating jamming attacks in lpwan," *EWSN*, 2023.

[14] 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 *ACM/IEEE IPSN '18*.

[15] S. Fahmida, V. P. Modekurthy, M. Rahman, A. Saifullah, and M. Brocanelli, "Long-lived lora: Prolonging the lifetime of a lora network," in *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. IEEE, 2020, pp. 1–12.

[16] A. Jain, P. Modekurthy, and A. Saifullah, "Control over low-power wide-area networks," in *2024 ACM/IEEE 15th International Conference on Cyber-Physical Systems (ICCPS)*, 2024, pp. 192–201.

[17] 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.

[18] 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.

[19] P. Marcelis, V. S. Rao, and R. V. Prasad, "DaRe: Data recovery through application layer coding for loRaWANs," *IoTDI '17*, 2017.

[20] R. Eletreby, D. Zhang, S. Kumar, and O. Yağan, "Empowering low-power wide area networks in urban settings," in *SIGCOMM*, 2017, pp. 309–321.

[21] C. Li, H. Guo, S. Tong, X. Zeng, Z. Cao, M. Zhang, Q. Yan, L. Xiao, J. Wang, and Y. Liu, "Nelora: Towards ultra-low snr lora communication with neural-enhanced demodulation," in *SenSys*, 2021, pp. 56–68.

[22] Z. Xu, P. Xie, and J. Wang, "Pyramid: Real-time lora collision decoding with peak tracking," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–9.

[23] X. Xia, Y. Zheng, and T. Gu, "Ftrack: Parallel decoding for lora transmissions," in *SenSys*, 2019, pp. 192–204.

[24] M. O. Shahid, M. Philipose, K. Chintalapudi, S. Banerjee, and B. Krishnaswamy, "Concurrent interference cancellation: Decoding multi-packet collisions in lora," ser. SIGCOMM '21, New York, NY, USA, 2021, p. 503–515.

[25] X. Wang, L. Kong, L. He, and G. Chen, "mlora: A multi-packet reception protocol in lora networks," in *ICNP*. IEEE, 2019, pp. 1–11.

[26] S. Tong, Z. Xu, and J. Wang, "Colora: Enabling multi-packet reception in lora," in *INFOCOM*. IEEE, 2020, pp. 2303–2311.

[27] Z. Xu, S. Tong, P. Xie, and J. Wang, "Fliplora: Resolving collisions with up-down quasi-orthogonality," in *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2020, pp. 1–9.

[28] M. A. Haque and A. Saifullah, "Handling jamming in lora," *IoTDI*, 2024.

[29] B. Hu, Z. Yin, S. Wang, Z. Xu, and T. He, "Sclora: Leveraging multi-dimensionality in decoding collided lora transmissions," in *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, Oct 2020, pp. 1–11.

[30] A. Gamage, J. C. Liando, C. Gu, R. Tan, and M. Li, "Lmac: efficient carrier-sense multiple access for lora," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3372224.3419200

[31] H. Cho and S. W. Kim, "An anti-collision algorithm for localization of multiple chirp-spread-spectrum nodes," *Expert Syst. Appl.*, vol. 39, no. 10, p. 8690–8697, aug 2012. [Online]. Available: https://doi.org/10.1016/j.eswa.2012.01.213

[32] N. El Rachkidy, A. Guitton, and M. Kaneko, "Collision resolution protocol for delay and energy efficient lora networks," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 2, pp. 535–551, 2019.

[33] 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.

[34] L.-H. Shen, C.-H. Wu, W.-C. Su, and K.-T. Feng, "Analysis and implementation for traffic-aware channel assignment and contention scheme in lora-based iot networks," *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11 368–11 383, 2021.

[35] 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, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/11/3718

[36] C. Gu, R. Tan, and X. Lou, "One-hop out-of-band control planes for multi-hop wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 15, no. 4, jul 2019. [Online]. Available: https://doi.org/10.1145/3342100

[37] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, "Using lora for industrial wireless networks," in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, 2017, pp. 1–4.

[38] https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/.

[39] https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/44000000MDnR/Et1KWLCuNDI6MDagfSPAvqqp.Y869Flgs1LleWyfjDY.

[40] "Dragino gps/lora shield," https://www.dragino.com/products/lora/item/102-lora-shield.html.

[41] "Arduino uno rev3," https://store-usa.arduino.cc/products/arduino-uno-rev3.

[42] "Ettus research," https://www.ettus.com/product/.

[43] D. Magrin, M. Centenaro, and L. Vangelista, "Performance evaluation of lora networks in a smart city scenario," in *2017 IEEE International Conference on communications (ICC)*. ieee, 2017, pp. 1–7.