ENABLING REAL-TIME COMMUNICATIONS IN RESOURCE-CONSTRAINED NETWORKS

by

Batuhan Mekiker

A dissertation proposal submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

December, 2023

DEDICATION

I dedicate this dissertation to my mother Neval, my father Menderes, my brother Çağatay, and my beloved partner in life Alè for their support, understanding, and encouragement.

TABLE OF CONTENTS

TABLE OF CONTENTS – CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES – CONTINUED

Figure Page

ABSTRACT

The Internet of Things (IoT) applications require flexible and high-performance data channels, but many IoT networks can only support single-use case applications, which limits their performance and flexibility for real-time and streaming applications. LoRa offers a flexible physical network layer but lacks the resource management needed in its link layer protocols to support real-time flows. My initial contribution, the Beartooth Relay Protocol (BRP), expands the performance envelope of LoRa, making it suitable for a wide range of IoT applications, including those requiring real-time and streaming capabilities, and aims to address the problem. However, the resource-limited nature of LoRa does not allow BRP to scale to multi-hop mesh network deployments while maintaining real-time streams. To address the limitations of BRP in supporting mesh network deployments and real-time streams beyond two hops, we focus on developing the second-generation Beartooth Radios, MKII, and the first-generation Beartooth Gateways. We utilize Commercially-available Off the Shelf Components (COTS) in the radios to provide a cost-effective, power-efficient, and compact solution for establishing real-time situational awareness. The self-healing mesh network provided with MKII and Gateways also enhances the reliability of the overall network, ensuring connectivity even in case of node failures. By incorporating military information brokers, such as the Tactical Assault Kit (TAK), the Beartooth Gateway establishes a hybrid network between Beartooth radios, gateways, and other TAK-capable devices, ensuring compatibility with existing IP networks. Building upon the premise that voice communications are an integral part of real-time SA, the last part of my research focuses on assessing audio quality and efficacy of audio codecs within bandwidth-constrained networks. Delving into voice communications in resource-constrained networks, my research contrasts the performance of Text-to-Speech (TTS) models with traditional audio codecs. I demonstrate that TTS models outperform audio codec compressed voice samples in quality while also effectively managing scarce resources and available capacity more efficiently. By combining flexible link layer protocol elements in BRP, Beartooth MKII radios, Gateways, and insights on integrating TTS systems for voice communication, my research demonstrates a versatile and flexible solution that provides real-time application streams and critical situational awareness capabilities in bandwidth-constrained networks and mission-critical applications.

INTRODUCTION

Motivation and Overview

The need for reliable and performant communication is essential in today's world. However, many regions lack the necessary infrastructure due to factors such as difficult terrain, remote locations, or low population densities. These factors often make it economically unviable for network service providers to establish connectivity infrastructure, resulting in a digital divide between connected and under-connected regions. Even in areas with connectivity solutions, mission-critical applications may face interruptions or failures because of natural disasters or man-made disruptions, further highlighting the need for robust communication networks.

In addition to supporting essential services, real-time communication networks are crucial for disaster response and recovery, and for promoting economic growth in remote areas. The lack of such connectivity hinders the progress of healthcare, and other critical services as well as recreational activities, maintaining a cycle of underdevelopment in disconnected regions [1]. Therefore, addressing the issue of inadequate real-time communication in bandwidth-constrained networks is the main objective that this dissertation seeks to achieve.

Real-time Data Dissemination Across Multiple Networks

Peer-to-Peer Networks To address these challenges, peer-to-peer networks have emerged as a viable alternative to planned network deployments. These networks are self-organizing, decentralized, and require no infrastructure, making them ideal for use in remote or off-grid regions. By enabling direct communication between devices, peer-to-peer networks offer increased resilience to failures and improved scalability compared to centralized systems. Furthermore, their decentralized nature makes them less vulnerable to targeted attacks or single points of failure.

Despite their advantages, peer-to-peer networks face certain challenges, such as limited bandwidth, and lack of standardization. These issues may hinder the widespread adoption and effectiveness of peer-to-peer networks, necessitating further research and development. Moreover, existing IoT technologies, mainly designed for IoT networks such as peer-to-peer sensor networks, may not be suitable for real-time user traffic as IoT applications are usually less resource-intensive and more delay-tolerant. This difference highlights the need for new solutions specifically created to support real-time user traffic in peer-to-peer networks.

<u>Problem Statement</u> Existing solutions within the LoRa MAC Layer domain, a flexible, low-power, and long-range IoT network solution, attempt to address the challenges of enabling reliable, real-time communication in infrastructure-less, unplanned, and bandwidth-constrained networks. However, they exhibit limitations in bandwidth and only single-hop capability that could impede their effectiveness in mission-critical applications. The growing need for real-time situational awareness (SA) in a self-forming, self-healing mesh network, and voice communication in such scenarios further brings the bandwidth issues into focus. Audio codecs, while efficient in transmitting voice samples across diverse network conditions, falter when network resources become scarce. They have to default to lower encoding rates, leading to a compromise in audio quality and the clarity of the message conveyed. These identified challenges are part of one overarching problem: The lack of a solution that enables reliable, real-time communication in infrastructure-less, unplanned, and bandwidth-constrained networks.

Research Design

The LoRa MAC Layer has emerged as a popular radio solution for off-grid communication. This technology is also affordable and easy to develop, making it accessible to a wide range of users. The LoRa MAC Layer's flexibility allows it to be used in various applications, from remote sensing to asset tracking, offering a versatile solution in unplanned, infrastructure-less networks.

However, the limited bandwidth of LoRa poses certain limitations, necessitating the development of new solutions like the Beartooth Relay Protocol (BRP). The BRP doubles the communication distance by utilizing relay nodes, allowing users to extend the range of their LoRa networks beyond the typical one-hop limit. The BRP effectively addresses the bandwidth constraints, marking a significant advancement towards better network solutions and solves the problem that it aims to solve. Although BRP is a step forward, the goal of enhanced performance and scalability in off-grid regions invites further improvements. Developing new solutions specifically designed for these areas will be crucial in enabling reliable and performant communication.

The growing importance of real-time situational awareness (SA) in mission-critical applications, particularly for small military units, has been underlined by recent conflicts. Conventional real-time SA solutions tend to be costly, energy-intensive, and cumbersome. This highlights the need for the development of more affordable, energy-efficient, and user-friendly alternatives that work in regions that lack infrastructure. The second-generation Beartooth Radios MKII, designed using commercially available off-the-shelf (COTS) components, strives to address these issues.

An integral component of SA, voice communications, is essential for its efficient and rich information delivery. However, in bandwidth-constrained scenarios, voice transmission becomes a significant challenge. Traditional audio codecs perform well in a variety of conditions but stumble when faced with limited bandwidth. This often leads to utilizing low-bandwidth audio codecs that are designed to work in resource-constrained networks. However, due to their highly lossy compression techniques and low encoding rates, low-bandwidth audio codecs significantly compromise audio quality and message clarity in such scenarios. This deficiency is especially pronounced in mission-critical contexts where accuracy and promptness are paramount. In search of a solution, recent advancements in neural networks and deep learning have spotlighted Text-to-Speech (TTS) models. These models, rather than transmitting comparatively larger encoded voice data, send concise text data, which is subsequently regenerated as voice on the receiver's end. This approach holds the potential to address the bandwidth constraints inherent in voice communications in real-time SA applications in resource-constrained networks.

Methodology The present dissertation is a comprehensive exploration of Internet of Things (IoT) technologies. The research is divided into three main parts, each contributing to a deeper understanding of bandwidth-constrained peer-to-peer networks' potential limitations. The first part of the thesis critically evaluates various IoT technologies, identifying their respective use cases, and highlighting their shortcomings. This evaluation aims to provide a solid foundation for the development of new solutions that address the limitations of existing IoT technologies.

Following the critical evaluation of IoT technologies, the study introduces a novel link layer protocol that enhances connectivity performance and mitigates the identified shortcomings. This new protocol, specifically designed for off-grid regions, is expected to improve the network reliability and performance of communication networks in these areas, measured with Packet Delivery Rate (PDR) and Throughput in Kbps respectively.

The next part of the thesis focuses on real-time Situational Awareness (SA) dissemination through the deployment of radios comprised of Commercial Off-The-Shelf (COTS) components. This section of the research investigates the integration of the aforementioned radios with existing mission-critical applications through gateways. This analysis contributes to the literature by providing valuable insights into the development and deployment of Peer-to-Peer communication technologies in complex and high-stakes environments.

Building on the exploration of real-time SA and the challenges inherent in bandwidth-constrained networks, the last section of the research discusses the domain of voice communication. Recognizing that voice communications are integral to real-time SA, the study conducts a comprehensive comparison between Text-to-Speech (TTS) generated audio and traditional audio codec encoded and decoded audio. The objective is to understand their respective performances, especially in low-bandwidth settings. By assessing the clarity, efficiency, and consistency of these audio communication methods, the research aims to shed light on the potential advantages of TTS model generated audio and how TTS systems might offer a robust solution for bandwidth challenges in voice transmissions.

My contributions are in three folds:

1. **Link Layer Solution - Beartooth Relay Protocol (BRP):** Introduction of BRP, a flexible novel link layer protocol, designed to enable real-time and streaming applications in bandwidth-constrained networks. The chapter includes details on the protocol's configuration, physical and link layer parameter tuning, and performance evaluation.

2. **Network Layer Integration - Cost-Effective Real-Time Situational Awareness:** Development of a network layer solution for efficient exchange of situational awareness data in bandwidth-constrained environments. The chapter includes creating a cost-effective, secure, bandwidth-efficient transport layer for Team Awareness Kit (TAK) messages over the XBee platform and a routing layer for communication between Beartooth Gateways and TAK servers, facilitating integration with existing IP network infrastructure.

3. **Application Layer Analysis - Real-Time Voice Communication in Bandwidth-Constrained Networks:** A comprehensive comparative analysis of Text-to-Speech (TTS) models with various vocoders versus traditional audio codecs. The study shows the trade-offs between voice intelligibility, clarity, and efficient network resource management of both TTS models and audio codecs.

Overview In Chapter 2, the thesis discusses the wireless connectivity problem and how enabling Device-to-Device connectivity solutions would address the wireless coverage gap. This analysis aims to provide a better understanding of the challenges and potential solutions in the context of wireless communication, especially in remote and off-grid regions.

Chapter 3 tackles the limitations of existing solutions for situational awareness. The chapter also proposes a solution to couple resource-scarce networks with higher capacity networks, creating hybrid networks in which situational awareness data can disseminate seamlessly. This examination offers insights into the potential improvements in real-time SA data dissemination across various networks.

Chapter 4 investigates the intricacies of voice communication in bandwidth-constrained networks. Specifically, it evaluates the comparative performance of TTS-generated audio and traditional audio codec processed audio in such networks. The chapter offers a comprehensive analysis to highlight the advantages and potential limitations of each method, shedding light on the evolving landscape of audio communication solutions suitable for bandwidth-constrained networks.

Finally, in Chapter 5, after delving into the comparison between TTS and traditional audio codecs in the previous chapter, the thesis concludes and discusses future work. This chapter summarizes the key findings of the research, encompassing both IoT technologies and audio communication advancements. It outlines potential avenues for further investigation and development, aiming to contribute to the ongoing evolution of IoT technologies and their role in SA scenarios such as mission-critical applications.

BEARTOOTH RELAY PROTOCOL: SUPPORTING REAL-TIME APPLICATION STREAMS WITH DYNAMICALLY ALLOCATED DATA RESERVATIONS OVER LORA

Problem Statement

People gravitate to convenient, one-fits-all solutions and expect products to work well for different applications in various scenarios. The same holds true for wireless networks. Bluetooth, for example, supports wireless music streaming, file sharing, and even control of mobile applications via car receivers. As of yet, networks in the Sub-GHz Industrial, Scientific, and Medical (ISM) band lack the same flexibility and primarily cater to IoT sensor communications [2–7].

One of the recent and exciting physical layer protocols in the ISM band is Long Range (LoRa) from Semtech [8]. LoRa's chirp spread spectrum (CSS) modulation makes possible long-range transmissions with low power consumption. LoRa's physical layer is also quite configurable in terms of bandwidth, transmit power, coding rate and spreading factor giving LoRa a broad performance envelope in terms of range and data rate [9]. However, link-layer protocols designed for LoRa do not fully take advantage of LoRa's flexibility and cater to specific applications in specific settings [3, 5, 6, 10–13]. As a result, LoRa remains a niche technology restricted to sensor networks, rather than broader Internet of Things (IoT) use cases.

To illustrate the above claim, LoRaWAN [14] is a link layer protocol base on LoRa for long-range, low-power, short-burst sensor communications. However, the collision avoidance mechanisms of LoRaWAN borrowed from ALOHA [15], lead to unpredictable message delay and loss. The resulting mean latency of 11 s and throughput of 28 bit/s make it unsuitable for real-time and reliable applications such as health monitoring [5, 14]. To provide predictable delay in LoRa networks several solutions proposed time-slotted medium access control (MAC) [3,4,10]. These protocols provide bounds on delay and high packet delivery rates (PDR). Yet, these protocols are hamstrung by the European Union (EU) regulatory duty cycle limitations on the ISM band, and so have low throughput and latency in the tens of seconds that limits their use for streaming applications [16]. A few approaches aim to sustain streaming data by abandoning PDR as a primary metric and focusing in improving LoRa throughput instead. Industrial LoRa [6] combines contention and contention-free transmissions to provide sustained throughput of 28 bit/s. DQ-LoRa [13] uses distributed queuing to provide throughput of 0.7 Kbit/s. Nevertheless, these protocols still achieve latency of more than 4 s limiting their use in real-time applications such as vehicular networks [17]. In summary, none of the existing LoRa MAC protocols support real-time data streams and moreover lack the flexibility to customize their parameters to meet the requirements of the specific application flows present in a network deployment

We propose Beartooth Relay Protocol (BRP), a flexible MAC protocol that supports, among others, real-time and streaming applications over LoRa. We also present the design of a frequency hopping LoRa radio developed by Beartooth that works in tandem with the BRP. This chapter offers the following contributions:

1. We describe BRP, a highly configurable gateway protocol supporting real-time and streaming applications. We control the BRP performance envelope with a configuration file distributed to nodes in a network deployment to match application performance requirements.

2. BRP provides latency under 500 ms making it suitable for real-time messaging, such as location updates within the Team Awareness Kit (TAK) tactical situational awareness application [18].

3. BRP provides flow throughput just shy of 0.8 Kbit/s letting it support bidirectional, real-time voice flows (a first in LoRa) encoded with Codec 2 [19].

4. We describe the mechanisms behind the Beartooth radio and BRP, specifically frequency hopping and scheduling of multiple transmission opportunities, that enable their performance while meeting Federal Communications Commission (FCC) regulations.

These results demonstrate that BRP has the potential to be adopted into many IoT applications with different performance profiles that go beyond sensor network communications.

Related Work

To frame the need for a flexible LoRa protocol that caters to both real-time and streaming data, we discuss the limitations of existing LoRa protocols in the commercial and research spaces.

Semtech introduced LoRa chirp spread spectrum (CSS) chip in 2009. LoRa encodes information with chirps, or transmissions of rising frequencies within the width of a channel (125 kHz or 250 kHz), where the starting frequency of chirp indicates a symbol [9]. The slope of a chirp is a function of channel width and the spreading factor, which defines the duration of each chirp and its resiliency to radio interference. The wider the channel and the longer the chirp the more resilient is the transmission to interference as these make it easier for the receiver to determine the starting frequency of the chirp. The CSS encoding gives LoRa resilience to multipath effects, fading, and Doppler frequency shifts [20]. However, the main advantage of LoRa is its range; radios based on Semtech chips support data rates between 0.3 and

37.5 Kbit/s and robust links at distances up to 9 km in urban and over 30 km in rural scenarios [14, 21].

The LoRa Alliance publishes a carrier sensing multiple access (CSMA) LoRaWAN protocol, which allows devices to communicate via gateways [22]. LoRaWAN utilizes an ALOHA based approach where end devices transmit contending frames, which leads to error rates as high as 70% depending on the frame size [10]. Although LoRaWAN utilizes other mechanism to boost performance such as Adaptive Data Rate (ADR) and Channel Activity Detection (CAD) collision avoidance protocol design is ill-suited for application streams with Quality of Service (QoS) requirements [23].

To address the unpredictable latency and unreliable delivery of LoRaWAN a number of protocols propose time-slotted approaches to medium sharing. Hoang et al. propose ST/CA – a slotted LoRa protocol with collision avoidance [10]. A gateway beacon synchronizes nodes to the start of a frame divided into transmission slots. Each transmission slot contains a number of delay slots. To transmit data a node engages CAD in a random delay slot and, if is does not detect a competing signal, proceeds to transmit data for the remaining delay slots and the rest of the transmission slot. While time synchronization and short delay slots reduce the impact of collisions, nodes still compete for each transmission opportunity. The evaluation shows maximum per node throughput of 11.3 bit/s under a PDR of 0.87. Frame duration is 25.7 s leading to the maximum latency of 25.65 s after a 500 ms beacon.

Piyare et al. propose an on-demand time division multiple access (TDMA) scheme for LoRa [11]. A gateway uses a separate wake-up receiver (WuRX) radio to wake up nodes and synchronize them to the start of a cycle. A node then chooses a transmission slot based on its node ID. The scheme assigns node IDs statically during network configuration, which makes it unable to deal with node churn and leads to higher latency for higher ID nodes. The evaluation shows a PDR of 100%, but high mean latency of over 2 s. The authors do not provide throughput performance.

Zorbas et al. propose TS-LoRa – a slotted LoRa protocol that allows nodes to autonomously compute their slot number [3]. TS-LoRA nodes operate in two stages, registration and transmission. During registration the gateway assigns a unique timeslot to a registering node, which the node then uses to transmit data. The gateway initiates data transmission with a `SACK` packet that also acknowledges previous transmissions. The scheme however leaves timeslots unused for nodes that chose not to transmit in their slot. The evaluation shows maximum per node throughput of 45.6 bit/s under a PDR of 0.9986.

Singh et al. propose a gateway-coordinated channel hopping scheme [12]. The protocol uses detailed on-air time calculation for beacon packets to synchronize node clocks. The gateway announces a schedule of timeslots and channels than nodes switch to for transmission. The paper evaluates the performance of scheduling and channel hopping scheme, but not of data transmission.

Leonardi et al. propose Industrial LoRa MAC that provides both contention and contention-free transmission slots within a frame [6]. Nodes compete for contention slots using ALOHA, while contention-free slots are scheduled. However, Industrial LoRa include time in the frame to communicate a schedule and the authors propose that the schedule should be specified offline, with dynamic scheduling left to future work. The authors evaluate Industrial LoRa in an OMNeT++ simulation showing best-case mean latency of 9.67 s and PDR of 0.34 for contention slots and 1.0 for contention-free slots.

Wu et al. propose DQ-LoRa based on distributed queuing (DQ) [13]. A DQ-LoRa frame is divided into a small number of contention slots, a data slot, and a gateway acknowledgement slot. Nodes compete for data slots by transmitting their

ID in one of the contention slots and the gateway acknowledges non-colliding IDs. Nodes that do not experience a collision use the data slots in subsequent frames to transmit, while other nodes continue to compete in contention slots. The work demonstrates that a small number of contention slots is sufficient to fully utilize data slots, even with a large number of nodes. The authors evaluate DQ-LoRa analytically showing best-case delay of around 4 s and throughput of 0.7 Kbit/s, a gain factor of 2.6 over LoRaWAN throughput.

Finally, Leonardi et al. underline the need for bounded end-to-end delay and higher reliability in IoT applications and introduce RT-LoRa to address delay and reliability with scheduling for real-time traffic [5]. The RT-LoRa is able to accommodate contention and contention-free transmissions for periodic and aperiodic data generation. Their results show RT-LoRa's packet loss rate and maximum end-to-end delay under two configurations. When compared to author's earlier approach Industrial LoRa, RT-LoRa has significantly higher PDR of 0.97 especially on shorter distances to the sink and latency of 20.7 s in best case scenario.

In summary, none of the approaches address the needs of latency sensitive real-time traffic, or streaming data flows. In this chapter, we propose a solution that applies time-slotted frame scheduling on a frequency hopping LoRa radio. Our approach not only noticeably reduces LoRa delay and increases throughput, but also provides the flexibility to configure network parameters to the performance needs of real-time and streaming applications present in a network deployment.

Beartooth Radio

The Beartooth radio behind the BRP is a custom LoRa shield paired with a Raspberry Pi 4 controller, as shown in Figure 2.1. The shield includes a SX1276 LoRa chipset, which communicates with the BRP on the controller's CPU via the Pi's

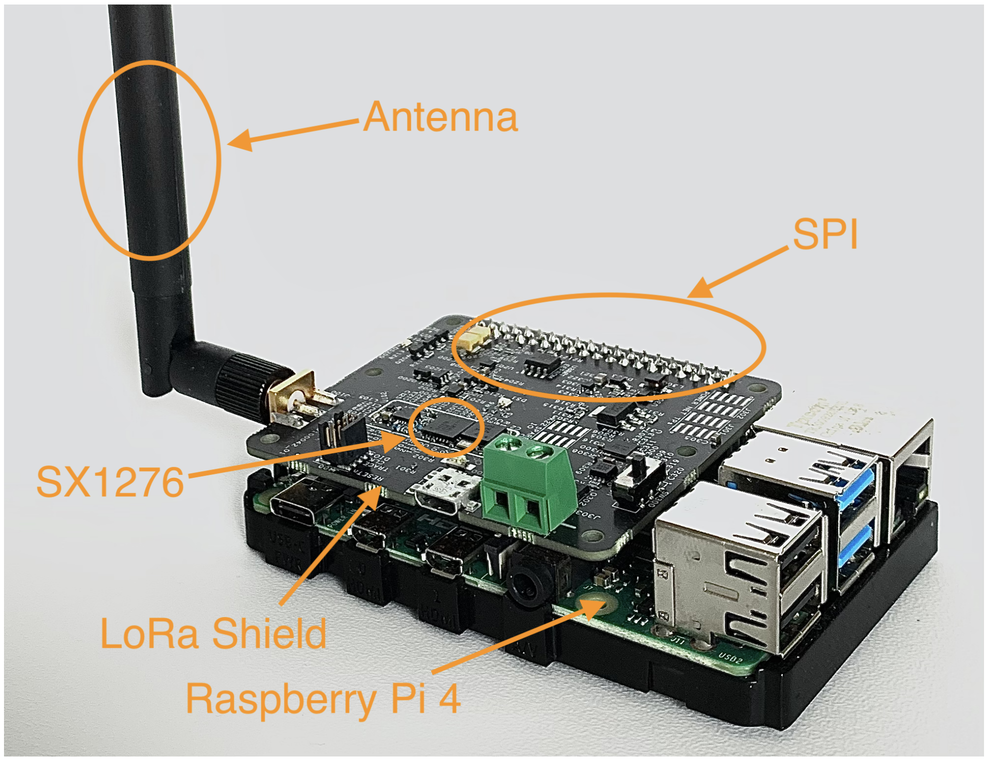


Figure 2.1: Beartooth radio LoRa shield paired with a Raspberry Pi 4.

Serial Peripheral Interface (SPI). The SX1276 chipset modulates a CSS radio signal in the 900 MHz band amplified to 30 dbm at the antenna. The SX1276 provides seven spreading factors, SF6 to SF12, with SF6 creating the the steepest slope providing the highest data rate, 37.5 Kbit/s, and SF12 creating the flattest slope provides the greatest robustness and therefore range with 0.26 Kbit/s [7]. LoRa also protects bits in transmission with a configurable error correction rates using Hamming codes and with a cyclic redundancy check (CRC).

One of our contribution is the frequency-hopping mechanism used by the Beartooth radio since 2017 [24]. The FCC Title 47 part 15 limits the maximum transmission duration on a channel, also known as dwell time, to 400 ms in the ISM band [25]. This regulation limits the throughput LoRa devices can achieve on a single frequency. However, SX1276 chipset supports internally timed frequency hops [20]

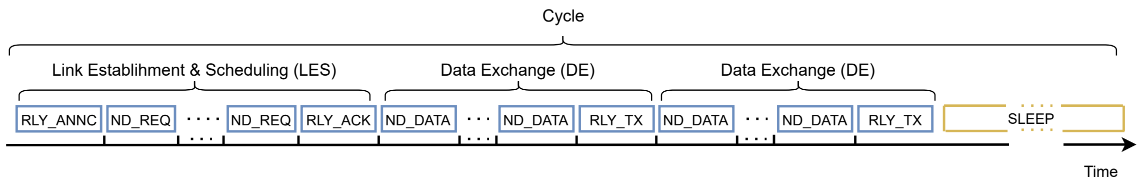


Figure 2.2: BRP cycle, stages, and frames.

and Beartooth radios change frequencies during message transmissions every 0.08 s, which effectively allows for indefinite transmission durations. Because LoRa supports 50 frequency channels, Beartooth radios use 50 semi-orthogonal hopping sequences. A relay and its clients agree on a hopping sequence and multiple sequences allow the coexistence of co-located network deployments. We handle occasional collisions on sequence timeslots with forward error correction (FEC) embedded in Beartooth frames. Beartooth radios and their approach to the use of the ISM band has been approved for operation by the FCC [24].

Beartooth Relay Protocol (BRP)

Protocol Requirements

In designing the BRP we had to consider Beartooth customer requirements, constraints of the LoRa chipset, and FCC regulations. Beartooth customers want to build networks that cover hundreds of square miles and support support two types of applications: situational awareness and team voice communications. The situational awareness application exchanges short messages that carry text, or GPS location. These messages are under 20 B and should be delivered in under 500 ms. The team voice application sends encoded voice streams that should be delivered to multiple recipients, again, in under 500 ms. We encode voice transmissions with Codec 2 for a throughput requirement of 700 bit/s [19]. We also explore a scenario to support generic sensor network compliant with EU duty cycle limitations in the ISM band, where duty cycle is restricted to 1% [6]. Messages in this network should achieve a PDR above 90% as in existing LoRa protocols [3, 5]. This application allows us to compare BRP performance to other EU-compliant LoRa protocols.

We aim to support these applications within the limits placed by the SX1276 chipset. The SX1276 chipset provides 50 10.9 Kbit/s channels at SF7, which forces a uniquely pithy BRP control signalling within the available bandwidth, as discussed in Section 2. Further, we determined experimentally that the SX1276 chipset faces limitations in transmitting back-to-back frames. For example, frames over 30 B transmitted every 150 ms result in the chipset becoming unresponsive until we cycle its power. The chipset, however, can transmit repeatedly frames under 30 B at that interval, or frames larger than 30 B are longer intervals. As a result of these hardware constraints, we configure node data frames under 30 B, but may still use larger frames for relay transmissions, as discussed in Section 2.

Finally, the FCC limits transmit power to 30 dbm [25]. The power limits restrict the communication range of Beartooth radios to 15.2 km as reported in our preliminary work [26]. Our transmissions also comply with FCC' dwell time regulations, as described in Section 2 where a radio cannot occupy one channel more than 400 ms [25].

Frames	Fields	Size (B)	Description
RLY_ANNC	type	1	Relay announces available LES control time slots and the network configuration.
	confid_id	1	
	ctrl_tbl	2	
ND_REQ	type	1	Node requests a number of DE time slots for data transmission.
	node_id	4	
	de_req_cnt	1	
RLY_ACK	type	1	Relay broadcasts DE time slot allocations and the number of DE stages.
	traffic_map	12	
	traffic_map_size	1	
	de_cnt	1	
ND_DATA	type	1	Node transmits their data.
	dest_id	4	
	data_len	1	
	data	20	
RLY_TX	type	1	Relay rebroadcasts received ND_DATA frames in RLY_TX.
	nd_data_len	1	
	nd_data_buffer	78	

Table 2.1: List of protocol frames with descriptions.

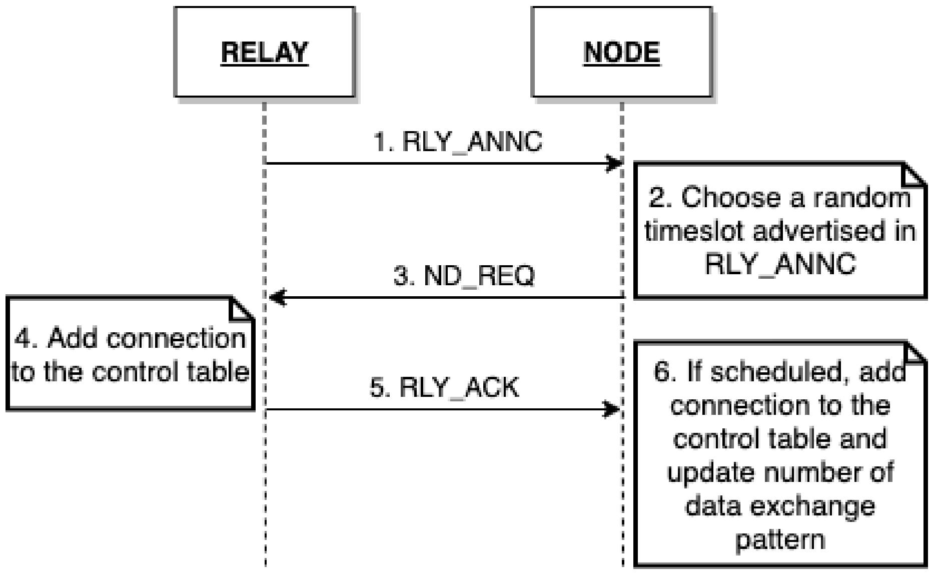

RELAY
NODE
1. RLY_ANNC
2. Choose a random timeslot advertised in RLY_ANNC
3. ND_REQ
4. Add connection to the control table
5. RLY_ACK
6. If scheduled, add connection to the control table and update number of data exchange pattern

Figure 2.3: Link Establishment and Scheduling message sequence.

Protocol Operation

At a high level BRP operates by repeating a transmission cycle of protocol frames shown in Figure 2.2. We list the details of details of the BRP frames in Table 2.1. Each cycle contains two types of stages, the Link Establishment & Scheduling (LES) stage and the Data Exchange (DE) stage. In the LES stage the relay synchronizes the nodes to the cycle start time and allows them to request transmission opportunities. In the DE stages nodes send data to the relay, which forwards the data the receivers via a broadcast. Depending on the amount of data nodes seek to transmit, the relay may allocate transmission opportunities in consecutive DE stages. While nodes compete in the LES stage and their request frames may collide, the data transmissions in the DE stages are collision-free.

Link Establishment Referring to Figure 2.3, to start a cycle a relay broadcasts the Relay Announce (`RLY_ANNC`) frame (Step 1), which includes configuration ID (`config_id`) and a LES control timeslot table (`ctrl_tbl`) with available timeslots for nodes to establish connections. The `config_id` specifies network parameters, discussed in Section 2.

A node may listen to frame preambles on different frequencies (in turn) to receive `RLY_ANNC` frames from different relays in an area. A node will chose the one with the strongest signal to noise ratio (SNR) and thereafter listen for `RLY_ANNC` frames on that channel. Nodes use the reception of `RLY_ANNC` to synchronize with a relay by establishing the start time of a cycle. If a node does not receive `RLY_ANNC` in some time, then it sequentially listens other channels and accepts announcements from other relays.

To connect with a relay, a node chooses a random available LES timeslot from `ctrl_tbl` (Step 2 in Figure 2.3) in which to send the Node Request (`ND_REQ`) frame containing the node ID (`node_id`) and how many DE stages (`de_req_cnt`) it needs for its traffic (Step 3). For example, to send a `ND_REQ` in timeslot 2, the node waits for the time it takes to transmit two `ND_REQ` frames after the reception of `RLY_ANNC`.

Scheduling The relay collects `ND_REQ`'s and adds `node_id`'s into a traffic queue used to schedule nodes in the DE stages. For simplicity, we implement a simple sticky scheduler, which gives priority to continuing flows from the previous cycle up to a limit. Otherwise the relay schedules new requests in the traffic queue randomly. The stickiness allows nodes to effectively reserve bandwidth for continued data streams across multiple cycles, while the limit and randomness provide fairness. The result of the scheduling process is a traffic map (`traffic_map`) containing `node_id`'s corresponding to DE timeslots of each scheduled node. For example, a traffic map `[215, 328, 328]` means that node `235` may transmit data in DE timeslot 0 and node `328` in timeslots 1 and 2, if it sent two `ND_REQ`'s.

Following the scheduling decision, the relay updates its `ctrl_tbl` (Step 4) by setting the bits in the LES timeslots used by `ND_REQ`s of the scheduled nodes.

The number of DE stages used by the relay provides a tradeoff between latency and throughput as discussed in Section 2. The number of DE stages may be fixed at the relay, though we implement a dynamic approach where the number of DE stages is the median of the `ND_REQ` `de_req_cnt` values to provide a balance between fairness and performance.

To announce the scheduling decision, the relay forms a Relay Acknowledgement (`RLY_ACK`) frame (Step 5) by including the `traffic_map` and the number of DE stages (`de_cnt`). A node receiving a `RLY_ACK` (Step 6) checks if its `node_id` is in the `traffic_map` and if so considers itself connected on the LES timeslot it used to sent its `ND_REQ`. The node also records its DE timeslot (the position of its `node_id` in the `traffic_map`) and the number of DE stages.

If two `ND_REQ`'s collide, and a node cannot find its `node_id` in the `traffic_map`, the node backs off the repeats the LES stage on a random available timeslot in the `RLY_ANNC` `ctrl_tbl` bitmap. It is important to note that as `RLY_ANNC` advertises only available control timeslots, nodes trying to connect to a relay will only consider available timeslots thus, `ND_REQ` can collide with only those of other, unconnected nodes.

To ensure that nodes do not need to repeat the connection process, entries in the `ctrl_tbl` on both nodes and the relay include a time to live (TTL) of 5 cycles. When a node stops receiving `RLY_ANNC`, it decrements the TTL of the connection. Similarly the relay decrements the TTL of a connection, if it does not receive a `ND_REQ` within a cycle. Reception of these frames resets the TTL to 5.

Data Exchange Referring to Figure 2.4, after a node, here Node A, receives `RLY_ACK` (Step 1) it looks up its data time slot(s) shared in `traffic_map` (Step 2). Then,

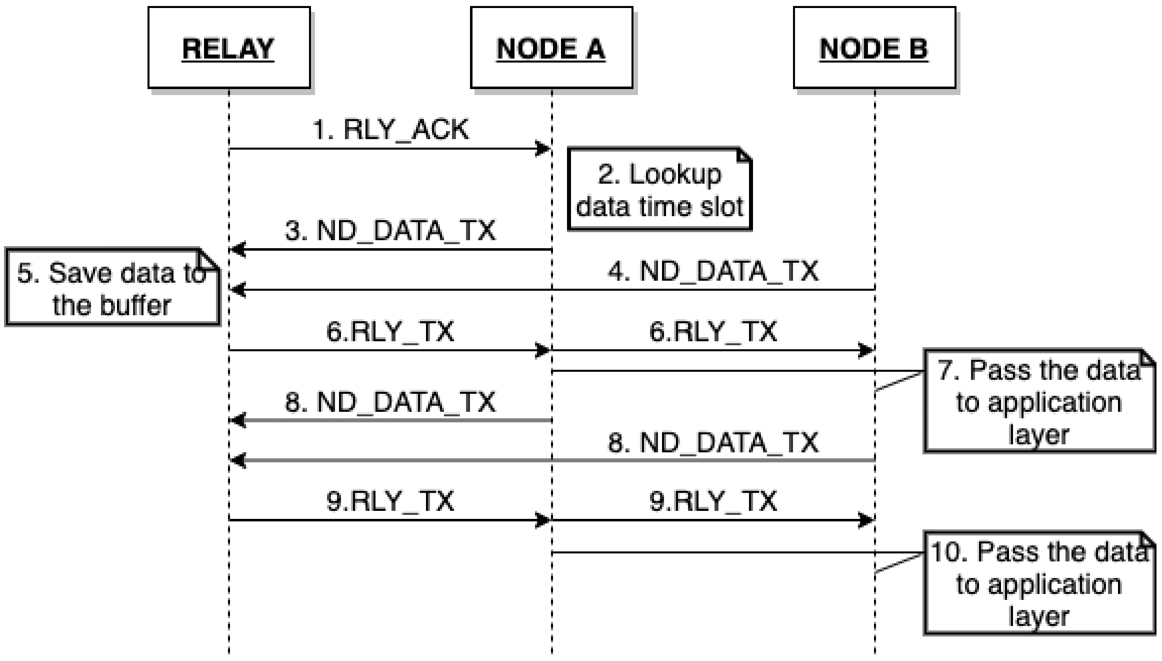


Figure 2.4: Data Exchange message sequence.

it forms Node Data (`ND_DATA`) frame and sends it in the assigned timeslot (Step 3). Another node follows the same pattern and sends its `ND_DATA` in (Step 4). `ND_DATA` contain destination address (`dest_id`), either `node_id` or `group_id`, the length of the data payload (`data_len`) and the data payload (`data`).

The relay collects all `ND_DATA` frames (Step 5), encapsulate them in a buffer (`nd_data_buffer`), and finally broadcast the `RLY_TX` frame (Step 6). Nodes receiving the `RLY_TX` pass onto the higher layer `data` if the encapsulated `ND_DATA` are addressed to their `node_id`, or a `group_id` subscribed to by the application layer (Step 7). Nodes and relay will repeat the DE stage `de_cnt` number of times (Steps 8-10).

Protocol Configuration

The BRP is quite flexible allowing Beartooth networks to support applications with different performance requirements. While the number of network parameters is quite large, we observe that the number of useful combinations is small. As a result, we let the relays to specify the set of parameters with a `config_id`, which then allows a node to look up a specific combination of parameters pre-loaded onto Beartooth nodes.

The configurable parameters include BRP parameters such as: the number of LES control timeslots, DE timeslot duration and count, and sleep time before a `RLY_ANNC` to reduce protocol duty cycle. The configurable parameters also include LoRa PHY settings such as spreading factor, channel bandwidth, coding rate, and whether or not the CRC is used.

Evaluation

To demonstrate BRP flexibility of performance, we evaluate its performance in three network scenarios, while measuring latency, cycle duration, PDR, and throughput. In all three scenarios, we have gathered hundred data points to plot the necessary figures.

Setup

We configure the Beartooth radios to use SF 7, coding rate of 4/5, channel bandwidth of 250 KHz, and the use of the CRC. The achievable channel bandwidth in this configuration is 10.9 Kbit/s and represents an attractive tradeoff between channel capacity and range [26].

Further, we configure three network scenarios as follows. Scenario 1 supports the exchange of short messages. The application generates 20 B messages every second with the goal of delivering them within 500 ms without relying on multiple DE stages. We vary the number of transmitting nodes between 1 and 3 while using 3 LES timeslots.

Scenario 2 supports real-time data streams. The goal of this scenario is to demonstrate BRP's ability to accommodate voice streams within 500 ms latency. We encode voice data with Codec 2 at 700 bit/s, which generates 20 B messages every 228 ms [19]. In the experiment we also vary the transmission interval to find the maximum flow throughput. We configure the number of DE stages to 3.

Finally, Scenario 3 supports BRP performance under the EU duty cycle regulations, which restrict nodes to transmit only 1% of the time [6]. The goal of this scenario is to demonstrate BRP's ability to meet EU regulations and compare its performance against other LoRa MAC protocols in terms of PDR and control overhead. To do so, we restrict the frequency of `RLY_ANNC` messages by adding a sleep interval to the cycle.

Results

Scenario 1: Short messages Our first experiment includes two devices: a relay and a sender/receiver. To measure the latency, we include the timestamp in the data packet and let node addresses the packets to itself. This approach allows us to measure the time difference between transmission (through the relay) and reception on the same node, without the need for synchronizing clocks between the sender and the receiver.

Figure 2.5 shows the CDF of cycle duration (blue solid line) and latency (orange dashed line) on the y-axis and time (ms) on the x-axis. We measure the latency at the sender between when the input data appears at the send buffer (from the application) and when it appears in the receive buffer (from the `RLY_TX`). We measure the cycle duration between the reception of `RLY_ANNC` and of `RLY_TX`.

We observe that latency ranges from 293 ms to 1.47 s with the mean of 478 ms. This result indicates that on on average the delay of short messages remains under

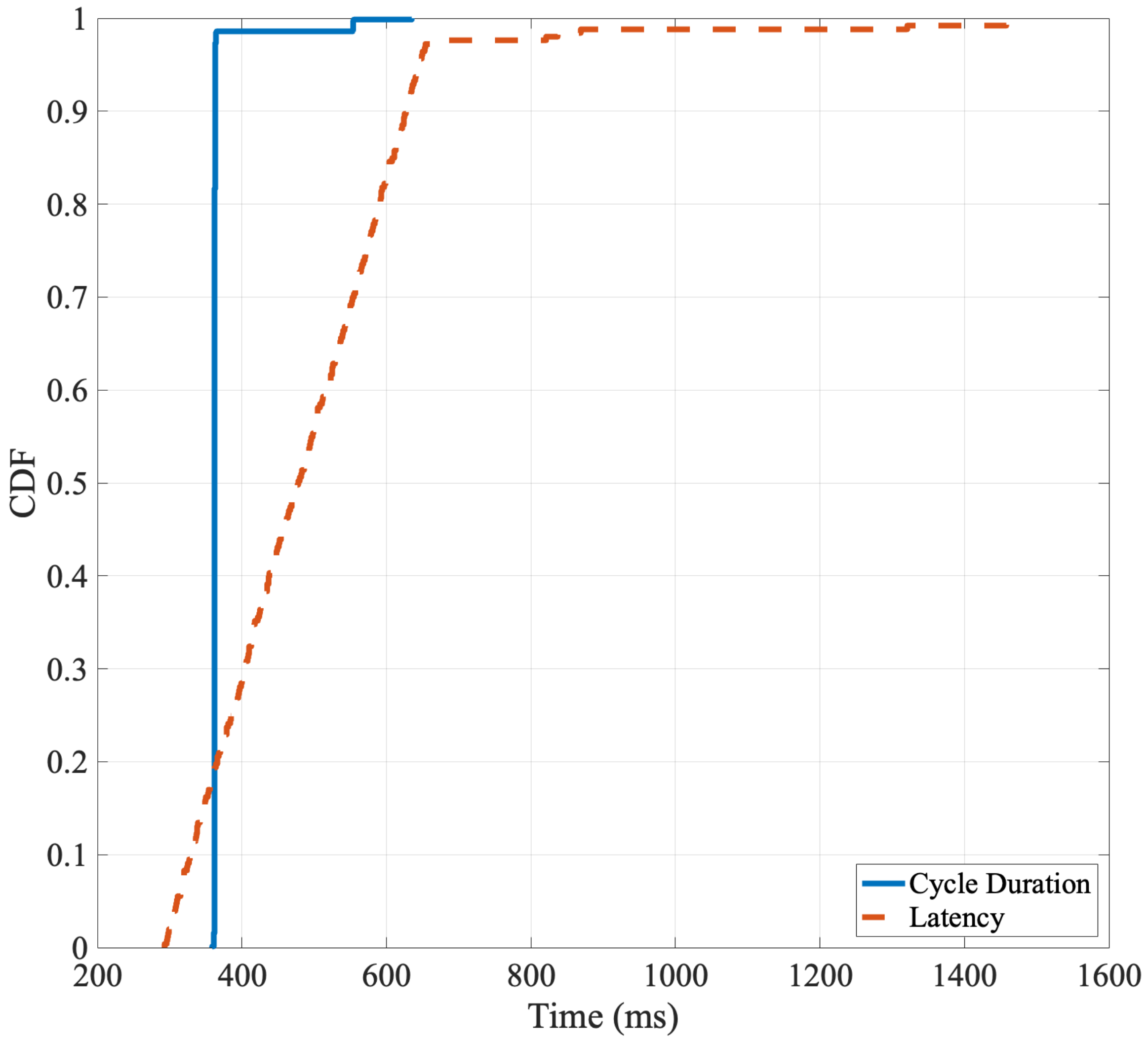


Figure 2.5: Two-hop latency and cycle duration.

500 ms. The variation in latency comes from the timing of receiving application-layer data in relation to the state in the protocol cycle. Data received just prior to the transmission of `ND_DATA` achieves the lower latency of less than a full cycle, while data received just after needs to wait for the next cycle for transmission. When a `ND_REQ` is lost due to collision, the node may need to wait for the start of the next cycle, thus increasing the measured latency producing the tail in the CDF.

We also see in Figure 2.5 that the cycle duration is fairly stable averaging around 362 ms. However, when a node is unable to get a control timeslot in the first cycle, and the second data packet is queued, relay may schedule two DE stages consecutively resulting in longer cycle of 553 ms, shown as the step in the cycle duration CDF.

We also wanted to investigate how different numbers of LES timeslots in `RLY_ANNC` effect nodes' ability to schedule transmission, and so application layer latency. Our experiment consists of four devices, one relay and three nodes, where one to three nodes send periodic packets as in the previous experiment. In Figure 2.6 we show application latency on the y-axis and the number of LES timeslots on the x-axis. The different series in the plot show the number of active sender nodes.

With one and two active senders, we see an upward linear trend in latency with increasing number of LES timeslots. This increase is due to a longer cycle duration needed to accommodate the extra LES timeslots. However, with three sending nodes, we see latency decrease. This effect is due to the collisions of `ND_REQ` from simultaneous senders in randomly chosen control timeslots. Increasing the number of available LES timeslots decreases the probability of such collisions, thus the number of retries in subsequent cycles, and so message latency.

Thus, BRP cycle duration can be tuned by the number of LES timeslots to accommodate the number of simultaneous transmitters in the network. We note that BRP can accommodate a large number of nodes and the LES timeslot tuning applies

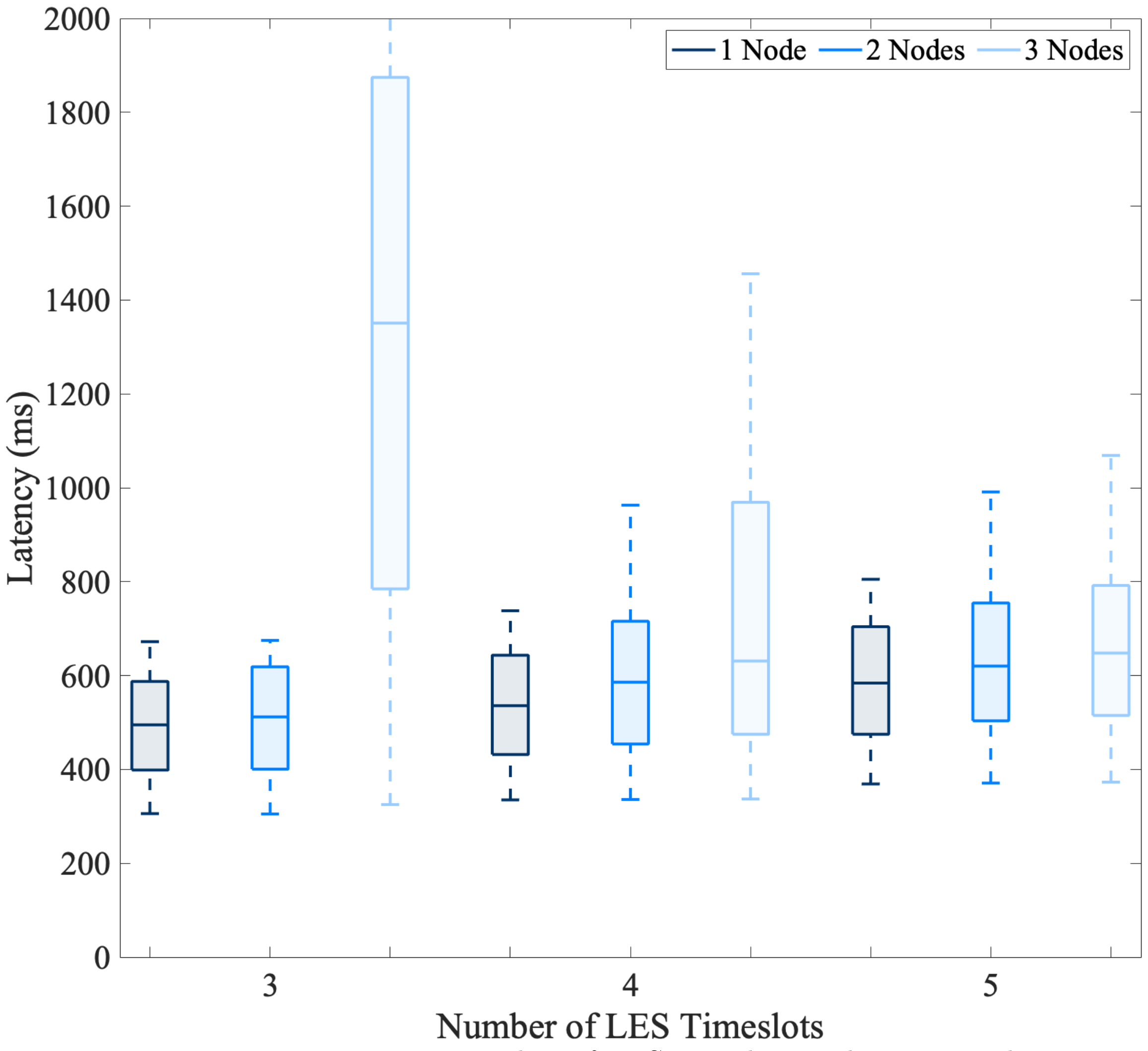


Figure 2.6: Latency versus number of LES timeslots and active senders.

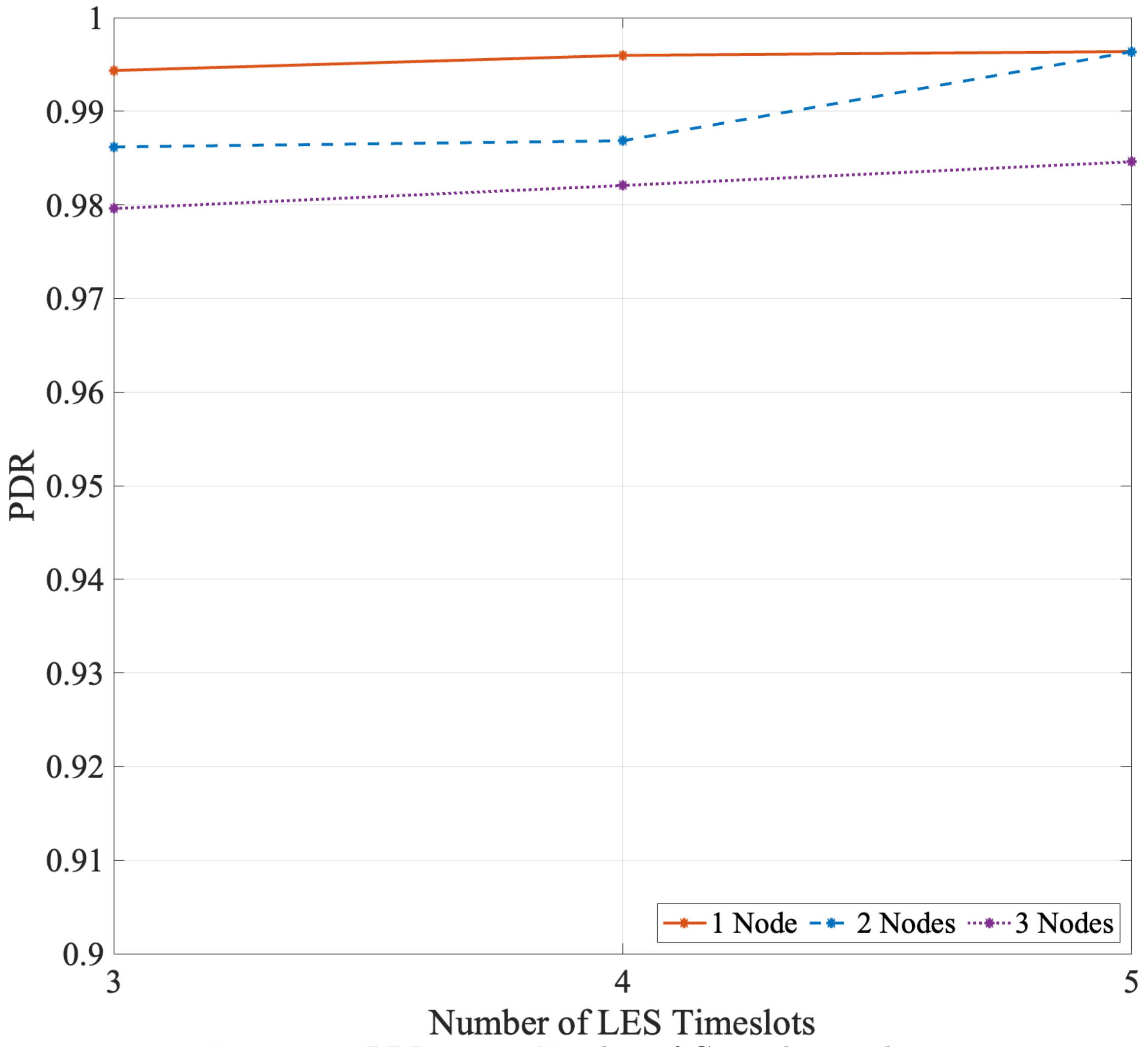


Figure 2.7: PDR versus Number of Control Timeslots.

merely to the number of simultaneous senders required by the application.

Figure 2.7 shows PDR, obtained in the previous experiment, on the y-axis and number of LES timeslots on the x-axis. We observe that PDR stays essentially the same regardless of the number of LES timeslots. This is because, number of LES timeslots has no effects on DE timeslots and when relay schedules three nodes within a cycle, nodes use all the available DE timeslots. We also observe that the PDR decreases slightly with additional simultaneous transmitters. Even though DE timeslots are strictly scheduled, a few milliseconds of shift can cause data frames to overflow into neighboring timeslot causing collisions on a very rare occasions. These rare collisions are due to the clock drift of the SX1276 chipset.

Scenario 2: Real-time flows We want to demonstrate BRP can accommodate data streams without compromising latency by enabling consecutive DE stages. In this experiment we utilize two devices; one node and one relay, configured to schedule 3, 5 or 7 data exchange stages consecutively.

Figure 2.8 shows latency on the y-axis and the number of consecutive data exchanges on the x-axis. We observe that the mean latency is 305 ms when relay schedules 3 DE stages back to back. Latency decreases to 283 ms when number of consecutive data exchanges are increased to 5. Finally at 7 DE stages, latency hits 278 ms and shows marginally diminishing returns, because the duration of LES stage remains undiminished and represents an increasingly large proportion of cycle duration.

To observe the effects of number of active nodes in scenario 2 we set up another experiment with four devices – three nodes and one relay. In this experiment we focus on throughput observed on the relay and average PDR observed on nodes. Figure 2.9 shows network throughput (blue solid line) in Kbit/s on left y-axis, PDR (orange dashed line) on the right y-axis and number of active senders on the x-axis. Experiments starts with 1 active node and the figure shows network throughput (and flow throughput) at just below 0.8 Kbit/s. When another node is activated we see a linear upward trend where network throughput reaches just below 1.6 Kbit/s with almost no drop in PDR, staying at 98%. The 3rd and the last node is activated, we still see an increase in network throughput reaching 2.25 Kbit/s however, the rate of the increase is diminished due to PDR getting a hit. As discussed earlier, the drop in PDR

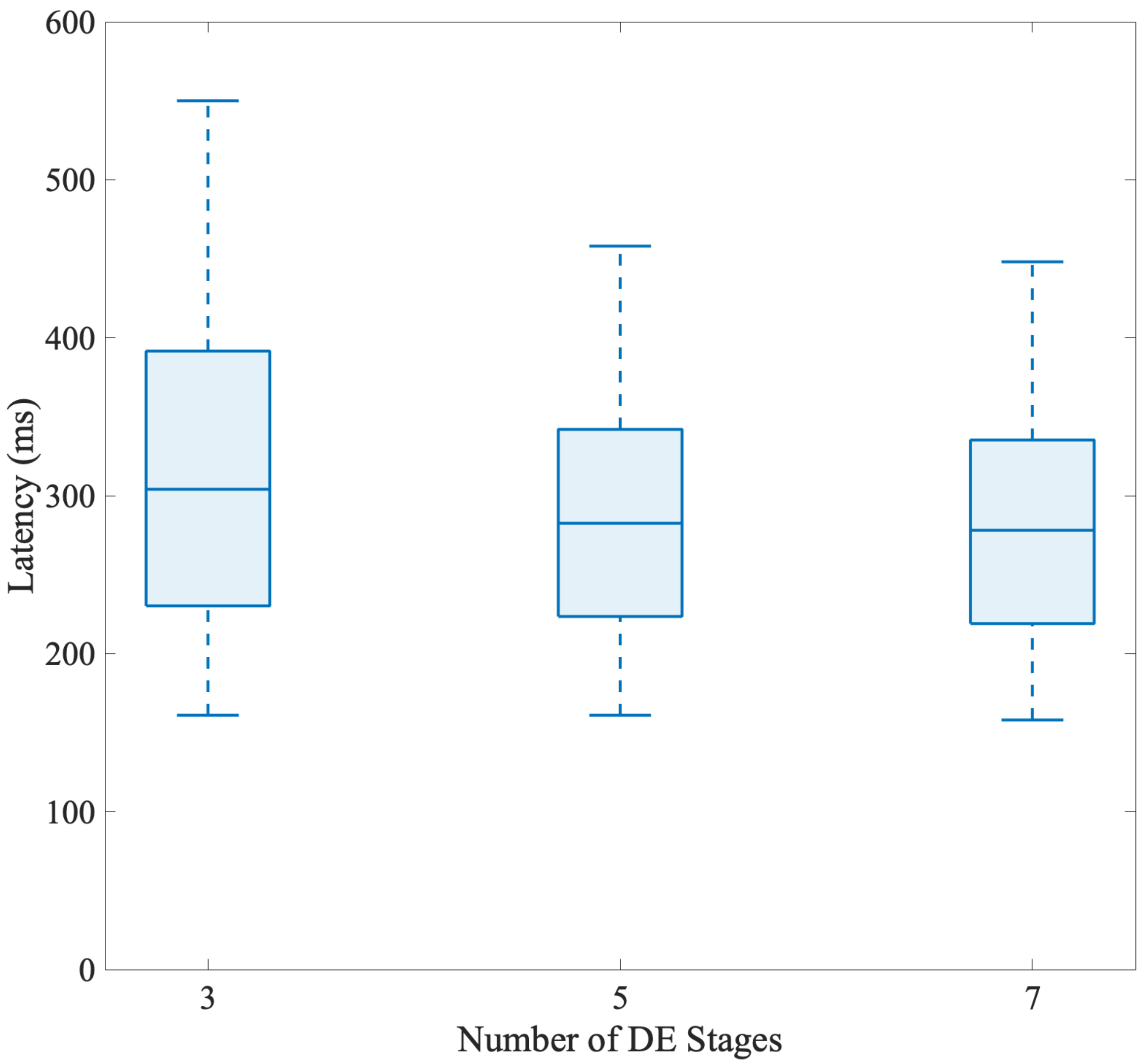


Figure 2.8: Latency versus Number of DE stages.

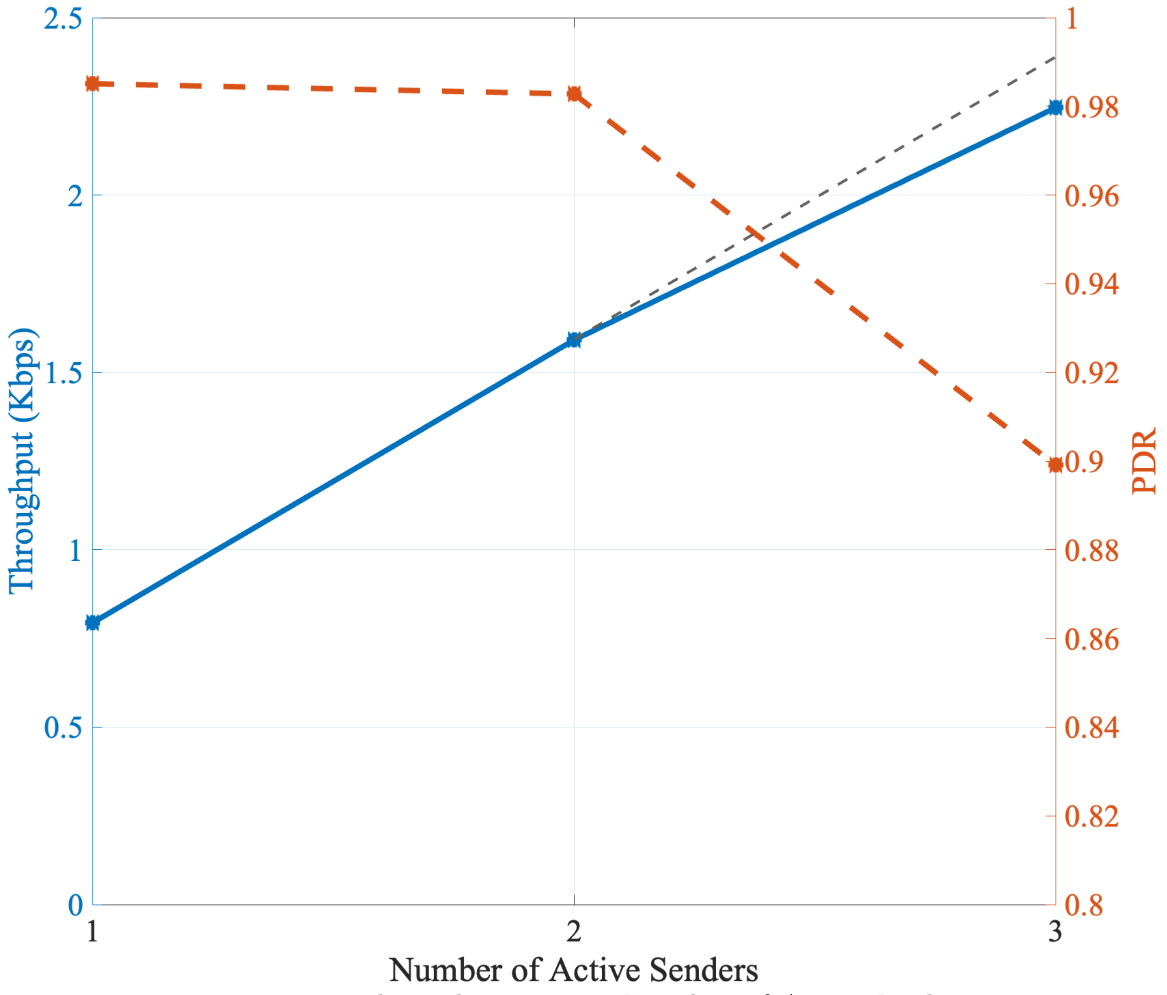


Figure 2.9: Throughput versus Number of Active Nodes.

is due to an occasional shift in ND_DATA transmissions and an overlap with neighboring DE timeslot resulting in ND_DATA collisions. Even under such collisions, the per flow throughput remains above 750 bit/s and allows the Beartooth network to deliver not one, but three simultaneous real-time voice flows encoded with Codec 2 [19].

Figure 2.10 displays network throughput from the previous experiment in Kbit/s on the y-axis and number of data exchange patterns per cycle on the x-axis. We observe that as the number of DE stages increases, so does network throughput. This effect is due to the fact that the single LES stage is amortized by multiple DE stages.

Scenario 3: EU duty cycle Finally, we want characterize the performance of BRP under EU duty cycle limitation and compare it against the performance of other LoRa MAC protocols. As mentioned earlier, the EU restricts duty cycle to 1% [6]. Doing so significantly reduces network throughput and increases latency. In the BRP the time-on-air (ToA) for relay transmissions at 27%. The relay ToA dominates node ToA, and so we add a sleep time of before each RLY_ANNC frame to bring relay (and node) ToA under 1% and into EU compliance.

Figure 2.11 shows the relationship between network throughput (blue solid line), maximum latency (orange dashed line) in seconds, and the number of consecutive DE stages with the appropriate cycle sleep time for each number of stages. The left y-axis marks network throughput for three simultaneous transmitters, while the x-axis shows the number of DE stages. While additional DE stages increase network throughput, the overall throughput remains constrained by the added sleep time. The right y-axis shows the maximum latency corresponding to the number of DE stages. We observe that the maximum latency increases with the number number of DE stages, because each DE stage requires an increase in sleep time to keep BRP duty cycle under 1%.

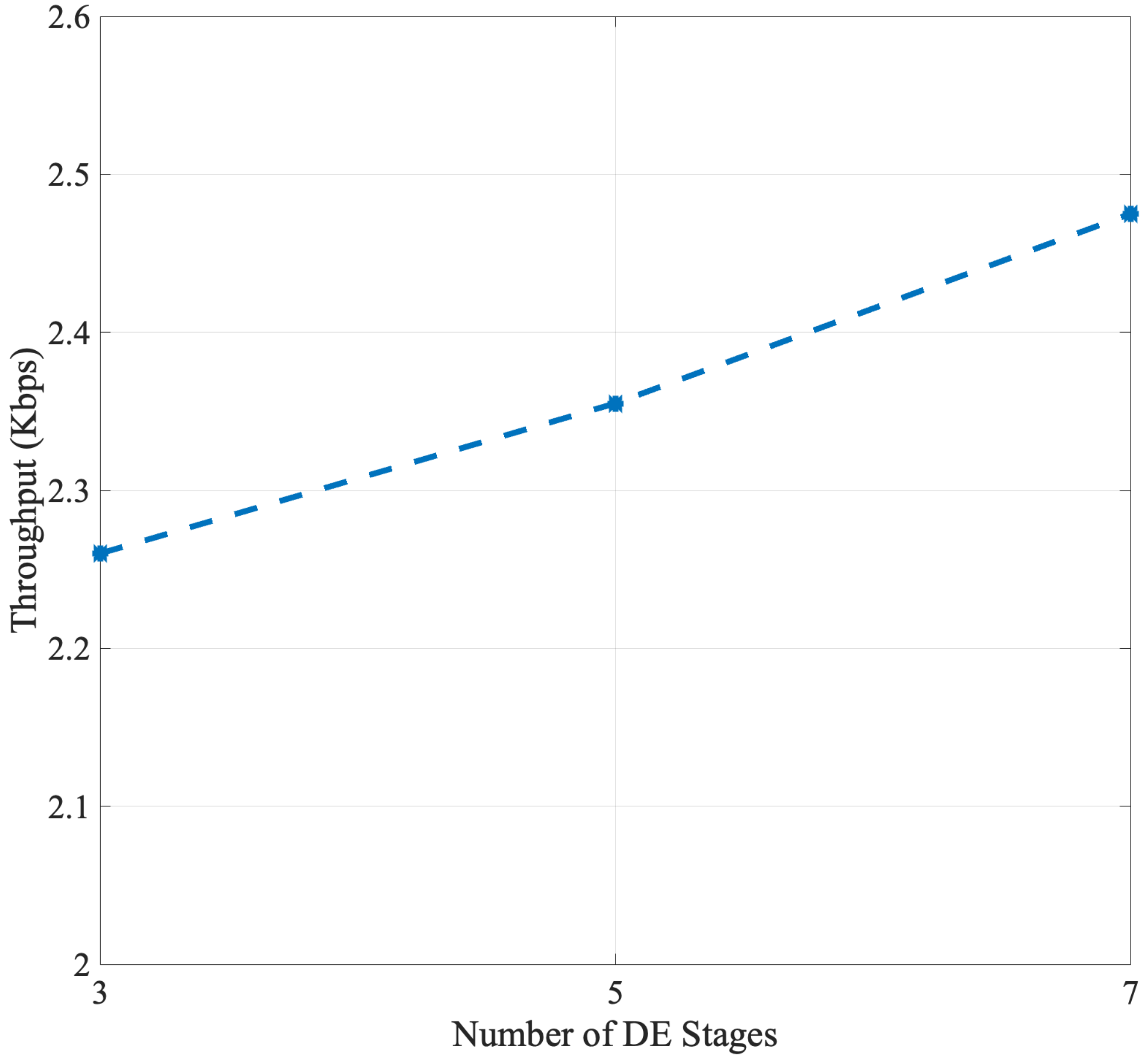


Figure 2.10: Throughput in Kbit/s vs. DE stages.

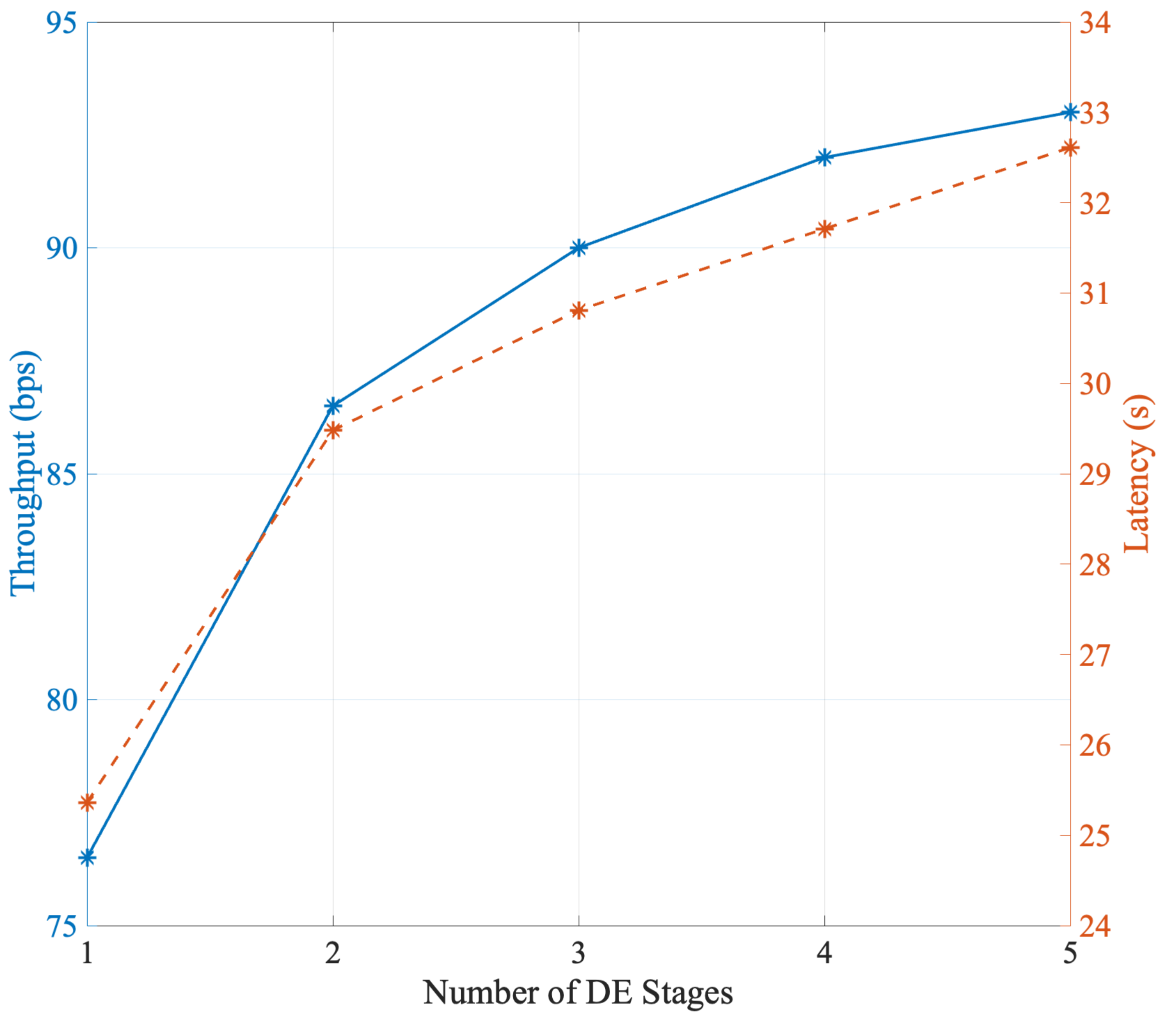


Figure 2.11: Analytical results of throughput and maximum latency observed under varying number of DE within a cycle.

Comparing BRP to Other LoRa MAC Protocols

In comparing BRP performance to that of other LoRa protocols discussed in Section 3, we observe that the combination of BRP and the frequency hopping Beartooth radio significantly outperforms existing approaches in terms of latency and throughput. The closest competitor to BRP is DQ-LoRa, which throughput gain of 0.7 Kbit/s, but a latency of around 4 s [13]. BRP delivers similar flow throughput, but with mean latency under 500 ms.

PDR values are comparable across the protocols and do not benefit from Beartooth radio's frequency hopping. We observe that BRP PDR of 0.98 is on par with that of TS-LoRa (0.9986) [3] and RT-LoRa (0.98) [5], and above that of ST/CA (0.87) [10].

Finally, we compare BRP's control overhead with the control overhead from other LoRa protocols. We calculate BRP control overhead ratio by counting the cycle bytes that represent protocol control (all the fields except `data`) and dividing it by total payload data (`data`). The smaller the ratio, the more efficient the protocol. In our calculations, we assume BRP has three active senders and three DE stages scheduled. Referring back to Table 2.1, in the LES stage, the control overhead comes from a `RLY_ANNC`, 3×`ND_REQ`, and a `RLY_ACK`, or

$$4 + 3 \times 6 + 15 = 37\,\mathrm{B}.$$

In the DE stage, the control overhead comes from 6 B in `ND_DATA` and 20 B in `RLY_TX`. With one LES stage for every three DE stages the total control overhead is

$$37 + 3 \times (3 \times 6 + 20) = 151\,\mathrm{B}.$$

Total payload data in the DE stage, on the other hand, includes 20 B in each of three `ND_DATA` frames and 60 B in the `RLY_TX`. Since there are three DE stages, the total payload over two hops is

$$3 \times (3 \times 20 + 60) = 360\,\mathrm{B}.$$

The the BRP control overhead ratio in this scenario is 151/360 or 41.9%. Of course a different protocol configuration, in terms of `ND_DATA` payload size, or the number of DE stages would change this ratio.

Using the scenario presented in the paper of Zorbas et al. we calculate the control overhead of TS-LoRa to 52% [3]. We were unable to compute the control overhead ratio for the other LoRa protocols listed in Section 3 due to insufficient information in their papers. Although not a direct comparison, we observe that BRP uses efficient signalling par with at least one competing LoRa protocol.

Conclusions and Future Work

In this chapter we presented the Beartooth Relay Protocol – a novel MAC protocol for LoRa. BRP provides the flexibility to meet various application performance requirements, notably under 500 ms latency for short message exchanges. BRP also supports real-time streams, specifically that of multiple, simultaneous voice flows, under the same latency bounds. BRP does so by leveraging frequency hopping mechanisms of the Beartooth radio and by making long-lived transmission opportunity reservations. We also demonstrate BRP's performance under EU duty cycle restrictions that are more stringent than FCC rules. The results indicate BRP's suitability to a range of IoT applications beyond sensor data collection.

In the future we plan to extend BRP beyond two-hop paths of relayed communications. Extending a single LoRa channel to support multiple hop is challenging due its limited bandwidth. To circumvent that problem we plan on equipping Beartooth relay nodes with additional LoRa radios to enable inter-relay communications. We will link the orthogonal channels used by these radios with data forwarding through the controller, supported by Address Resolution Protocol (ARP) and switched forwarding.

Additionally, we are working on extensions to connect BRP-like ad-hoc networks to the internet through gateways, bridging P2P connectivity with centralized infrastructure and creating hybrid networks.

The C++ implementation of BRP also allows us to move the protocols from Raspberry Pi onto the shield board micro controller. We evaluated micro controller options to enable Beartooth radio operation on standalone hardware.

Finally, Semtech introduced a LoRa Frequency Hopping Spread Spectrum (LR-FHSS) extension to LoRa in December of 2020 [27]. The LR-FHSS mechanism implements frequency hopping transmissions at the physical layer without changes to the interface presented to the link layer. The LR-FHSS mechanism does use an additional 3 B in the header and is able to provide added robustness at a modest impact to latency and throughput. We expect that the LR-FHSS will allow BRP retain most of its performance benefits while deployed on a generic LoRa hardware as opposed to a Beartooth radio. To verify that, we will perform an evaluation of BRP on LR-FHSS chipsets and compare it against the results presented in this paper.

The combination of BRP running on standalone hardware (without a paired Raspberry Pi) and a generic radio using the LR-FHSS mechanism will make it easier and cheaper to deploy BRP a variety IoT deployment scenarios.

COST-EFFECTIVE SITUATIONAL AWARENESS THROUGH COTS BEARTOOTH RADIOS AND GATEWAYS

Introduction

As we have seen in recent conflicts, the nature of warfare is evolving. The absence of encrypted communications generates an operational disadvantage for large military forces and enables small, highly mobile units to disrupt their advance [28]. On the other hand, accurate, up-to-date situational awareness (SA), whose definition can vary depending on the scenario, is crucial. For example, for troops on the battlefield, SA might refer to immediate access to markers of enemy troop locations, while for wildfire firefighters, it might entail rapid delivery of polygon-defined areas on the map to describe wildfire containment. Regardless of the specifics, SA allows these small units and their command centers to coordinate efforts. Consequently, a low-cost, scalable SA solution that can be tailored to different use cases becomes an increasingly important force multiplier that first responders and military commanders need in their arsenal.

Mission-critical or specifically battlefield SA relies on two integrated technologies: a connectivity layer that provides raw communication links between agents, and an information layer that disseminates location-based data and operational directives. Battlefield radios from Silvus [29], TrellisWare [30], and Persistent Systems [31] can establish direct as well as multi-hop connectivity among agents. These radios also integrate with mobile phones running the Android Tactical Assault Kit (ATAK) [32] to create an SA overlay. Networks built on these radios, however, have several shortcomings. First, these purpose-built radios are comparably more expensive than other radios built with commercially-available components, which makes it difficult to deploy them in large numbers. Unfortunately, the price per unit for these radios is not publicly available. However, discussions with users revealed that comparable network sizes would be 5-10 times more costly than networks built with commercial off-the-shelf (COTS) radio components. Second, they are power hungry which increases their weight, limits portability, and increases observability [33]. Finally, third, without an edge Tactical Assault Kit (TAK) [34] server in the field, SA data dissemination with SA communication systems that use these radios is geographically limited. SA data can only be exchanged within the coverage area of the radios, limiting SA to just one or a few local squads.

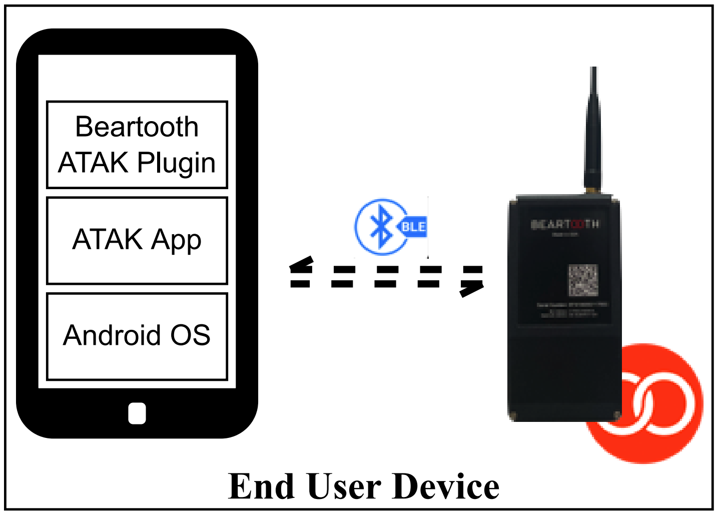
One way to reduce SA system costs and form factor is to build it with COTS components. Research on internet-of-things (IoT) communications has produced several multihop connectivity solutions [2, 35–37]. These low-cost radios provide low-bitrate links, which may nevertheless be suitable for SA applications. A key advantage of these radios is their low cost and spectral efficiency, which leads to low power requirements and small, lightweight form factors. Despite their benefits, these low-cost radios must also satisfy certain Quality of Service (QoS), such as low latency, delivery receipts for SA data, and resilience to network disruptions caused by high mobility, to be considered for SA systems and mission-critical applications. However, whether resource-limited IoT radios can meet these demands is unclear.

In this paper, we propose an SA solution based on IoT COTS radios. Specifically, we design the Beartooth MKII radios based on the XBee radio platform [38]. The XBee radio platform provides more than ten miles of range in urban areas and sixty miles in rural areas with a line-of-sight between transceivers. XBee also supports DigiMesh, a self-forming, self-healing link layer protocol with built-in frequency hopping and routing among more than a hundred nodes in a Low Power Wide Area Network (LPWAN) [39]. In addition, XBee radios are highly power-efficient and draw less than 3 mA while in standby mode, and 900 mA during data transmission at 30 dBm (1 W) transmit power [40]. Finally, the XBee platform is inexpensive with modules easily accessible commercially.

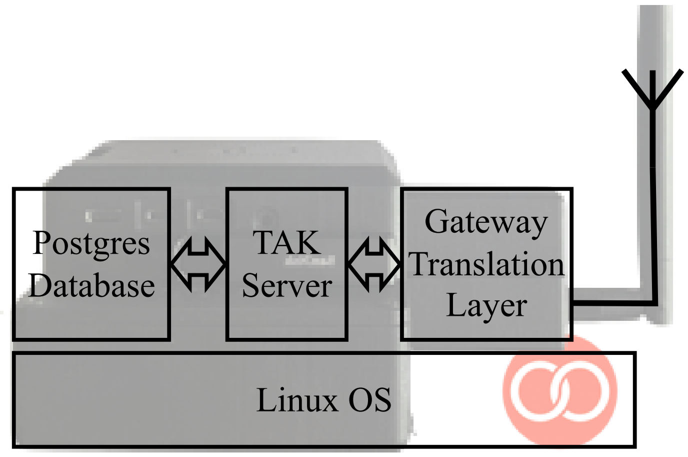
The Beartooth MKII radio couples an XBee radio with a Bluetooth module that allows it to speak to a mobile phone running ATAK. To address the bitrate limitation of the XBee platform, we design the Beartooth ATAK Plugin to rearrange the existing TAK data format known as Cursor-on-Target (CoT) events into a more compact representation. We compress larger data types such as sensor readings, images, and bulk data and we split them for transmission over multiple XBee frames. We also use unicast transmissions to reduce time-on-air for larger CoT events, which route frames directly to the destination to reduce the number of data and acknowledgment packets traversing on the network with respect to broadcast frames. While Beartooth MKII radios exchange our compact frames, the original CoT events still get recreated at the receiver then published to ATAK, allowing us to share SA in a resource-constrained COTS network. The resulting MKII form factor, shown in Figure 3.1a, is 4.09 x 1.21 x 0.7 inches and weighs in at 6 oz, including a battery. The Beartooth MKII radios are as much as 54% lighter than existing tactical radios discussed in Section 3.

Further, we provide shared SA between squads and commanders through the Beartooth Gateway. The Gateway is a handheld server as seen on Figure 3.1b that consist of a MKII radio and a translation layer to couple an XBee network to an IP network over cellular, satellite, or other radio technologies, such as the radios from Silvus, Trellisware, Persistent Systems, or Harris. IP connectivity allows the sharing of encrypted SA data among multiple squads as well as command centers running a TAK server [34].

Figure 3.2 illustrates a possible deployment scenario interconnecting multiple network technologies and operational areas into a shared TAK SA overlay. In Figure 3.2 we show two Beartooth networks, *A* and *B*, bridged together using



(a) Beartooth ATAK Plugin and MKII radio.



(b) Beartooth Gateway and its software modules.

Figure 3.1: Bearthooth system design.

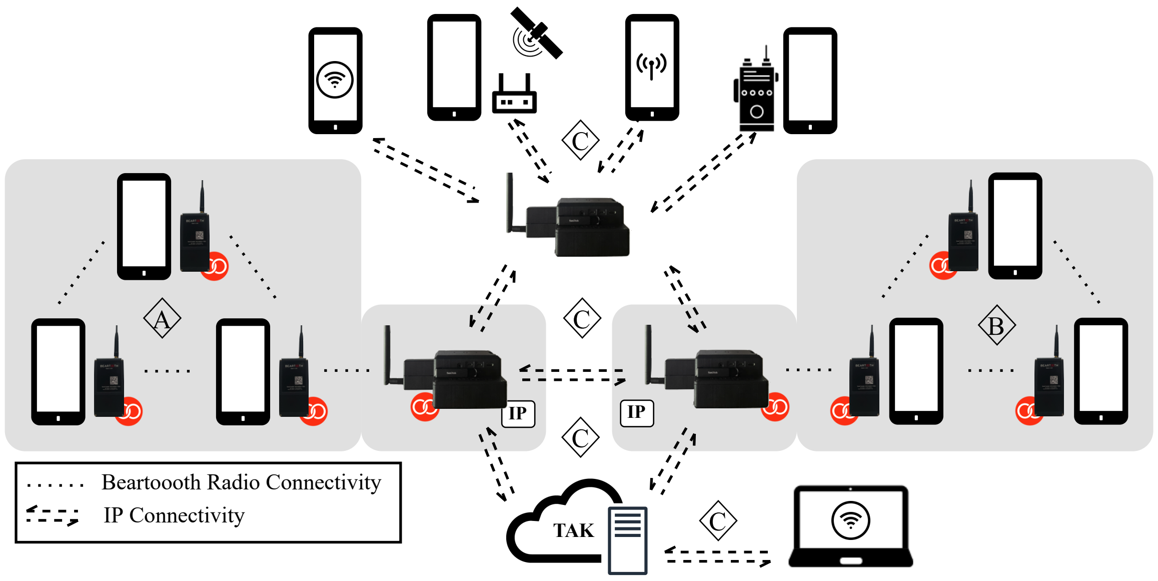


Figure 3.2: An example deployment of Beartooth network with MKII radios and Gateways, connecting many users spanning across WiFi, cellular (5G/LTE), satellite connectivity and battlefield radios.

two Gateways. Both Gateways are connected through an IP network such as the internet with the help of a Virtual Private Network (VPN) denoted as network *C*. An additional Gateway within network *C* serves the TAK information layer to multiple IP-connected devices through various network interfaces such as WiFi, cellular (5G/LTE), and satellite connectivity. All three TAK servers within Gateways are federated, allowing Beartooth SA data to flow into the IP network and SA data generated at network *C* to flow into Beartooth Networks *A* and *B*. A fourth TAK server sits in the cloud computing architecture, is also part of the network *C*, and is federated to Gateway *A* and *B*. Therefore both Beartooth network members and members of IP network can exchange SA and battlefield intelligence. Furthermore, commanding officers can observe the whole operation and even send orders through the TAK server in the cloud using a WinTAK [41] device, creating communication links that are encrypted between forces in the field and an operation command center.

This paper's main contribution is enabling SA over IoT COTS radios that are compatible with traditional SA systems through the following supporting technical contributions.

1. A secure, bandwidth-efficient transport layer for TAK messages over XBee platform

2. A routing layer for TAK messages among Beartooth Gateways and TAK servers

A Beartooth network composed of MKII radios and Gateways delivers a capable SA solution to coordinate squads and command centers over large areas of operation. The low cost and small form factor of the Beartooth radios make it possible to deploy SA at a large scale and is a powerful force multiplier.

Related Work

To illustrate the need for cheaper, smaller, and more flexibly deployable radios, we discuss the limitations of existing solutions to create connectivity and situational awareness. We consider commercially available tactical radios as well as COTS technologies from the IoT space.

In the commercial space, there are three predominant solutions Streamcaster, TSM Shadow, and MPU5 radios from Silvus Technologies, TrellisWare Technologies, and Persistent Systems respectively. All were developed to provide tactical connectivity and SA for first responders and military forces.

Streamcaster product line from Silvus Technologies uses a proprietary radio module capable of self-forming and self-healing a mesh IP network [29]. Streamcaster Lite radio has a maximum of 1 W transmit power and 20 Mbps data rate. The radio requires 4.8 W to 17 W power while transmitting at 1 W (30 dBm), which is a few times more than the Beartooth MKII radio. The Streamcaster Lite weighs in at 10.4 oz and takes up more than twice the physical space of a Beartooth MKII.

Similar to Streamcaster, TrellisWare's TSM Shadow handheld radio encloses a proprietary radio module [30]. It has a transmit power of up to 2 W with a maximum data rate of 16 Mbps. Although there is no publicly available information about power consumption, it would be safe to assume TSM Shadow would consume similar power as Streamcaster Lite. It weighs around 11.3 oz, an ounce more than Streamcaster Lite however, volume-wise, it sits between the Beartooth MKII and the Streamcaster Lite.

MPU5, compared to Streamcaster Lite and TSM Shadow, is a more complete and capable radio system with a full-fledged mobile CPU and Android Operating System on board [31]. Without the battery module, It weighs around 13 oz. MPU5 has a wide variety of radio modules for many frequency bands with data rates up to 150 Mbps. With varying radio modules for different frequency bands, its transmit power varies between 4 W to 10 W, and its power consumption varies between 30 W and 50 W.

All three radios have IP network capability and provide sufficient bandwidth to support hundreds of devices sharing real-time CoT events. However, their physical footprint and cost are significantly higher than Beartooth MKII radios. While all three radios do support IP networks and can handle TAK-formatted SA data, thus ensuring compatibility with the TAK ecosystem out-of-the-box, their functionality is geographically limited without an edge TAK server in the field. They can exchange local SA data within their coverage area, but to transmit SA data beyond this local network, an edge TAK server is required. Given this requirement for an edge server to extend coverage, it could be more beneficial to opt for physically smaller, easily concealable, cheaper, and therefore more expendable radios, such as Beartooth radios and Gateways.

We are interested in bringing cost-effective COTS devices to SA systems by adopting radio modules developed for a wide variety of IoT applications. The IoT solution space commonly utilizes solutions like Z-Wave [35], SigFox [36], DASH-7 [2], LoRa [35], and ZigBee [37], which provide comparatively low-cost and power-efficient options. Z-Wave and SigFox can support infrequent data transmission at a low data rate over long distances. However, due to their inability to meet the QoS requirements of SA and TAK traffic requires such as low latency and resilience to high mobility, these solutions are disqualified from use in mobile SA systems. DASH-7 and LoRa can sustain communication links with longer distances and sufficient maximum raw data rate of 200 Kbps and 37.5 Kbps respectively for single-hop networks while keeping power consumption low. However, no official or third-party link layer protocol implements a multi-hop self-forming, self-healing mesh network with performance SA systems require. For LoRa, in earlier work, we found building a mesh network challenging due to limited bandwidth, concluding that a second radio module is needed to extend coverage beyond two hops [42]. Finally, ZigBee's use in both sub-GHz and 2.4 GHz bands and compatibility with other ZigBee devices enhance its IoT use case. However, its network deployment is cumbersome, because it requires extensive planning of different radio roles.

Similar to ZigBee, DigiMesh, available on XBee platform, supports sub-GHz and 2.4 GHz ISM bands, with similar performance metrics like bandwidth, data rates, latency, and power consumption as discussed by Khalifeh et al. [37]. However, DigiMesh's key advantage is its simplified network role structure and its self-forming, self-healing mesh network which allows the network to be fluid and highly mobile. Unlike ZigBee, which requires distinct coordinator, router, and end-device roles, thereby adding complexity to deployment and maintenance, DigiMesh operates with a singular network role, easing deployment, maintenance, and network extension.

While DigiMesh and other IoT solutions hold certain advantages, they are not without drawbacks. The most notable limitations of DigiMesh are its inability to support data rates at Mbps levels and the lack of support for IP network connectivity. Despite these shortcomings, we show that core SA data types—such as text, geolocation, markers, polygonal shapes, and voice—can be effectively supported with proper on air data management. Furthermore, by using gateways, we can translate Beartooth SA data into traditional TAK formatted IP SA data, ensuring compatibility with existing infrastructure. Taking into account both the benefits and drawbacks, we have chosen the XBee platform and DigiMesh as our link layer protocol over other IoT COTS solutions.

Beartooth Network

To understand the effectiveness of the Beartooth network, we discuss the software components that both MKII and Gateway require for effective communication. We then elaborate on the types of situational awareness data that the MKII radios and Gateway can serve to the Beartooth Network and the IP-layer network.

Network Elements

The MKIIs are highly mobile and power-efficient handheld radios that forces in the field use to communicate within a local region. Gateways within Beartooth networks are edge servers bringing TAK capability to the field and integrating troops using MKII radios to a larger TAK SA overlay.

MKII

To limit the form factor of MKIIs we design them with limited computational power and place much of the transport layer logic on the connected phone.

There are a few reasons we opt to use this design decision. First, designing and building a radio without a powerful processing unit is cheaper and less complex. Second, having a power-hungry processing unit affects the power consumption and thus overall battery life of MKII radios. Last but not least, since the mobile phone does all the required processing, the protocol design is quite flexible and easily upgradable. If and when we decide to change how we encode data into packets, we can do so by modifying the phone plugin without the necessity to update the MKII radio firmware.

One of the key design decision involves efficiently serializing and transporting data in a resource-limited network. We use Google's Protocol Buffers (ProtoBuf) library [43], allowing platform-neutral serialization of any data structure. ProtoBuf uses schemas, simplifying the encoding of various data fields and types.

We define all Beartooth SA packet within a ProtoBuf schema where we defined fields such as `sourceUid`, `destinationUid`, `messageType`, `textPayload`, `messageUid` and `locPayload`. While designing the Beartooth packet schema, we focused on the limited network resources. CoT data format, being XML-based and designed for IP networks, is not optimally suited for resource-limited networks due to its inherent verbosity. Hence, we extract only the necessary data from the CoT event and form it into the Beartooth packet format, thereby minimizing its footprint. Once we have the proper packet format, ProtoBuf encodes necessary data into a series of bytes, then passes those bytes to the Beartooth radio interface.

The Beartooth MKII plugin for ATAK is a lightweight platform that connects to Beartooth MKII radio using Bluetooth Low Energy (BLE) to send various SA data such as text, location, markers, shapes, pictures, and voice messages. Together with the MKII radio, we design the plugin to be a robust and secure communications platform. As the default option, it auto-shares the user location with the entire network and a team leader to monitor the squad members. So that the team leader can take appropriate actions. In addition to the types mentioned above of SA messages, the plugin is also capable of sending casualty evacuation (casevac), navigation routes for walking, flying, and driving, icons for a wide variety of first responder missions, elevation, range and bearing. Once the plugin is connected to MKII through the BLE, the plugin configures the radio in the background to user preferences, where the user can set network channel and encryption settings.

Contrary to other radios, MKII radios coupled with the Beartooth ATAK plugin do not directly transmit CoT events due to limited network resources. When the user triggers a SA event, such as a marker on the map in ATAK, the plugin puts necessary data to a Beartooth SA frame using the ProtoBuf schema, serializes it, and sends it to the BLE connection. Then, BLE queues the data for the radio in MKII to be transmitted. End-user devices receiving the Beartooth SA frame pass the data to the plugin. Using the ATAK API, the plugin only then recreates CoT events at the receiver device to be shown on the ATAK user interface.

We also implemented a network scanning tool enabling end-users to plan and monitor their Beartooth network deployment. The tool uses MKII radios to ping all devices within a local DigiMesh network, visualizing RSSI values, distances, hops, and overall network health on the Beartooth ATAK plugin. This helps users optimize radio deployment for stronger communication links and manage the Beartooth network with ease.

Gateway The Gateway, shown in Figure 3.1b, consists of three individual modules: Virtual Private Network (VPN), TAK Server, and Gateway Translation Layer. We use ZeroTier VPN [44] which is lightweight, easy to set up peer-to-peer VPN solution that provides static IP addresses for TAK servers within the Gateways

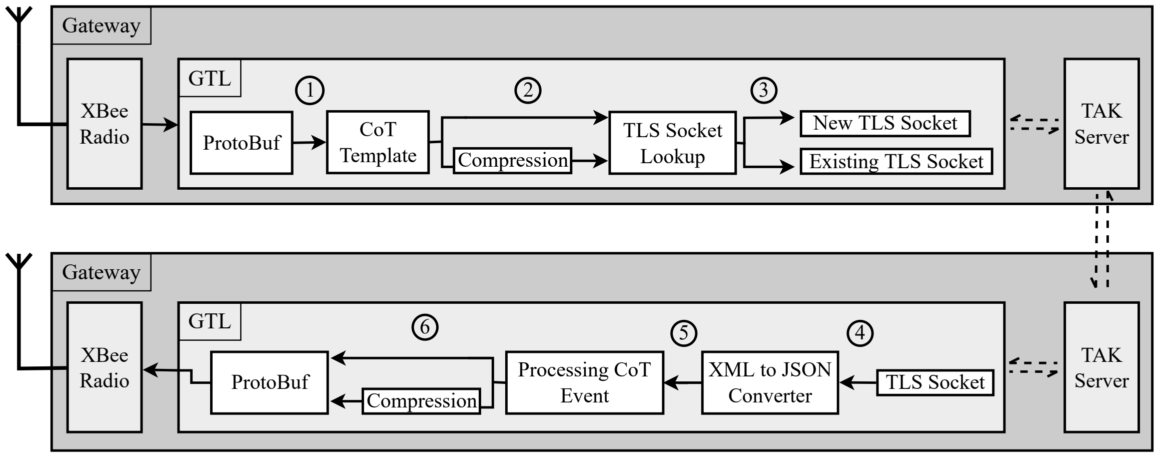


Figure 3.3: Data progression within Gateway and GTL showing various steps and software modules.

and IP-connected end user devices. Furthermore, we use TAK server version 4.8 optimized for Single Board Computers (SBC). It allows us to bring the TAK server to the edge of the network and connect multiple local networks through the TAK's federation protocol. While the federation protocol's details fall outside this paper's scope, it is important to note that it facilitates the secure exchange of all, or selected, SA data between authenticated TAK servers through encrypted TLS links. If there is any connection interruption, it is handled within the TAK.

Gateway Translation Layer

We design Gateway Translation Layer (GTL) which is the novel solution that links Beartooth MKII devices to the TAK Server and the IP network through an XBee radio connected to its serial port. GTL enables bi-directional SA data communication, as illustrated in Figure 3.3. The following sections describe GTL's processing of incoming and outgoing data for both directions in detail.

Data Flow: Beartooth to IP-network We show the data flow from Beartooth MKII devices to an IP network through a Gateway in Figure 3.3. The GTL, using the pre-determined Beartooth's ProtoBuf schema, parses the incoming SA data frame from the Beartooth network and determines the data and CoT event type. Depending on the type, the GTL process SA data within the DigiMesh frame as shown in *step 1* and forms a valid CoT event out of one of the pre-configured CoT event formats representing the original SA data frames. The pre-configured CoT events are templates that help us generate valid CoT events by pluging in relevant Beartooth SA data.

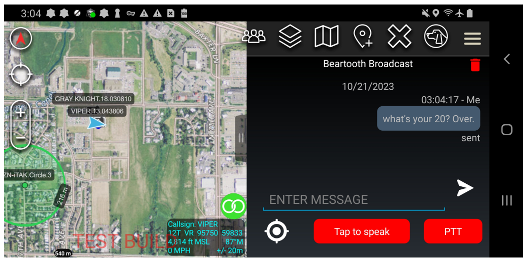
After forming a valid CoT event, depending on the CoT event type the GTL can route fully formed CoT event to the compression engine in *step 2* to reduce message time-on-air. Only medium-sized CoT events such as marker, routes and shapes and CoT events carrying bulk data such as pictures and compressed data packets go through the compression engine. In *step 3* the GTL looks up for active TLS sockets based on sender address. If there is an established TLS socket, the GTL recycles that active TLS socket to submit CoT event to TAK. Otherwise, the GTL creates a TLS socket from available client certificates generated during the build process and stored at the Gateway. Having multiple certificates helps us keep latency low and TLS connections sticky for ongoing transmissions and recently active users. Once GTL establishes a TLS connection with the TAK server, it submits the generated CoT event and stores the connection for a brief period. Depending on the type and destination of the event, TAK Server triggers a set of procedures determining how CoT events get processed and routed.

Data Flow: IP to Beartooth Network Conversely, as seen on bottom section of Figure 3.3, when the data path is from the IP network to Beartooth Network, SA data is either routed from a TAK Server as seen in Figure 3.3 or it is received by the bottom TAK server via a device such as an ATAK that is connected to the this TAK server through a TLS connection. Then TAK relays SA data received in CoT format to GTL. In *step 4* the GTL converts XML-based CoT event into a JSON object for ease of accessibility. GTL then processes the JSON CoT events by categorizing them by the event type on *step 5.* As we discussed, if the CoT event type is what we classify as medium-sized or bulk data, the SA data goes through a compression procedure. Similarly, GTL forms the Beartooth SA data frame from either compressed or uncompressed SA data using Beartooth's ProtoBuf schema in *step 6.* Then the GTL passes the serialized data in bytes onto to the radio module using the serial port. The radio module then transmits data either by unicasting or broadcasting depending on the data type.

Supported Data Types

The Gateway supports two-way communication between the Beartooth network and IP network. Therefore the Gateway support data flows from IP-network to the Beartooth network, the Beartooth network to IP-network, or the Beartooth network to the Beartooth network with Gateways in between. The Gateway also supports various CoT event types and meets the required QoS for the SA through the GTL. The next couple of paragraphs shows SA data types Beartooth network and the Gateway supports.

Small-sized packets (up to 256B) are three distinct data types: text, geo-location, and acknowledgment as seen in Figure 3.4a. Text and acknowledgment messages are CoT events used for communication and delivery receipt confirmation within the TAK overlay. Geo-location messages, another type of CoT event, contain latitude, longitude, and altitude data, enabling end-users to share their location.


Beartooth Broadcast

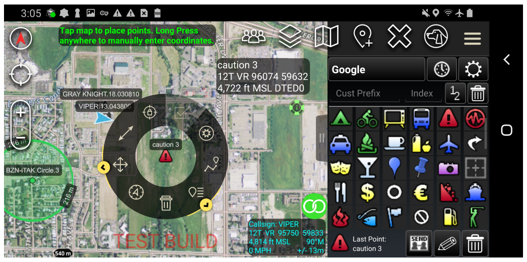
10/21/2023

03:04:17 - Me

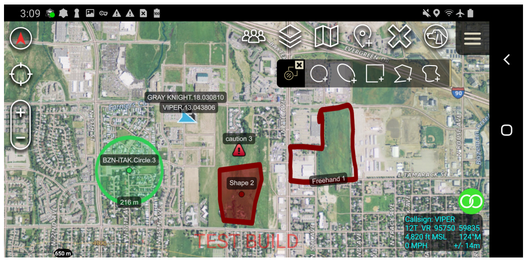
what's your 20? Over.

sent

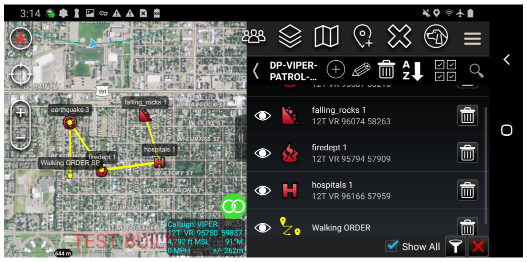
(a) Small-sized Packets: Text and Location Data.



(b) Medium-Sized Packets: Markers and Shapes.



(c) Medium-Sized Packets: Shapes and Routes.



(d) Bulk Data Packets: Any SA overlay bulked together in a `.zip` file.

Figure 3.4: Supported Data Types.

Medium-sized packets (256B to a few KB) are CoT events that are map overlays and can be represented similarly such as markers, routes, and shapes as seen in Figure 3.4b and Figure 3.4c. Due to DigiMesh's 256B frame payload limit, we transmit larger CoT events in a series of frames. A compression engine deflates XML-based CoT events to minimize spectrum usage and frame count. The GTL stores received frames, compiling complete CoT events once all frames are received. With built-in retransmission protocol, if a frame is lost during transmission, GTL can request retransmission through a selective acknowledgment mechanism to ensure reliability and accurate CoT event compilation.

Bulk Data Packets (up to 25KB) are `.zip` files that include an XML-based manifest, guiding TAK on the enclosed data type and processing requirements. The Beartooth network supports various data formats, such as images, map overlays, icon sets, and TAK server configurations as seen in Figure 3.4d, given the data does not exceed 25KB. The transmission strategy mirrors that of medium-sized packets, using a series of frames for data delivery including the packet loss and retransmission protocol.

Evaluation

In this section, we evaluate the application layer performance of Beartooth Network as a whole including GTL. Beartooth devices utilize the DigiMesh protocol from the XBee platform. As a result, they inherit its performance. A detailed performance evaluation of DigiMesh is beyond the scope of this paper (see work by Khalifeh et al. [37] for an overview).

Latency To demonstrate Beartooth MKII radios and Gateways' capabilities in the SA use case, we evaluated their performance on supported data types by taking end-to-end message latency measurements. These measurements encapsulate the entire SA data life-cycle, from dissemination in the Beartooth Network to processing by the GTL and TAK server. Our setup included one MKII radio, a phone (Samsung Galaxy XCover) with the Beartooth ATAK plugin, an IP-connected phone (Samsung Galaxy XCover) with ATAK, and a Gateway. The latency measurements, reflecting single-hop latency for each data type, were based on timestamps from transmission start till SA data received by the IP-connected phone, with each experiment repeated a hundred times.

Figure 3.5 shows latency in seconds on the y-axis and supported data types on the x-axis. First, we evaluate small-sized packets such as text and location data. We observe the latency is 0.73 s on average with variance around 0.02 s. For medium-sized packets where CoT event types such as markers, routes, and shapes we have two scenarios. In the first scenario, markers and simple polygonal shapes (e.g. 2-3

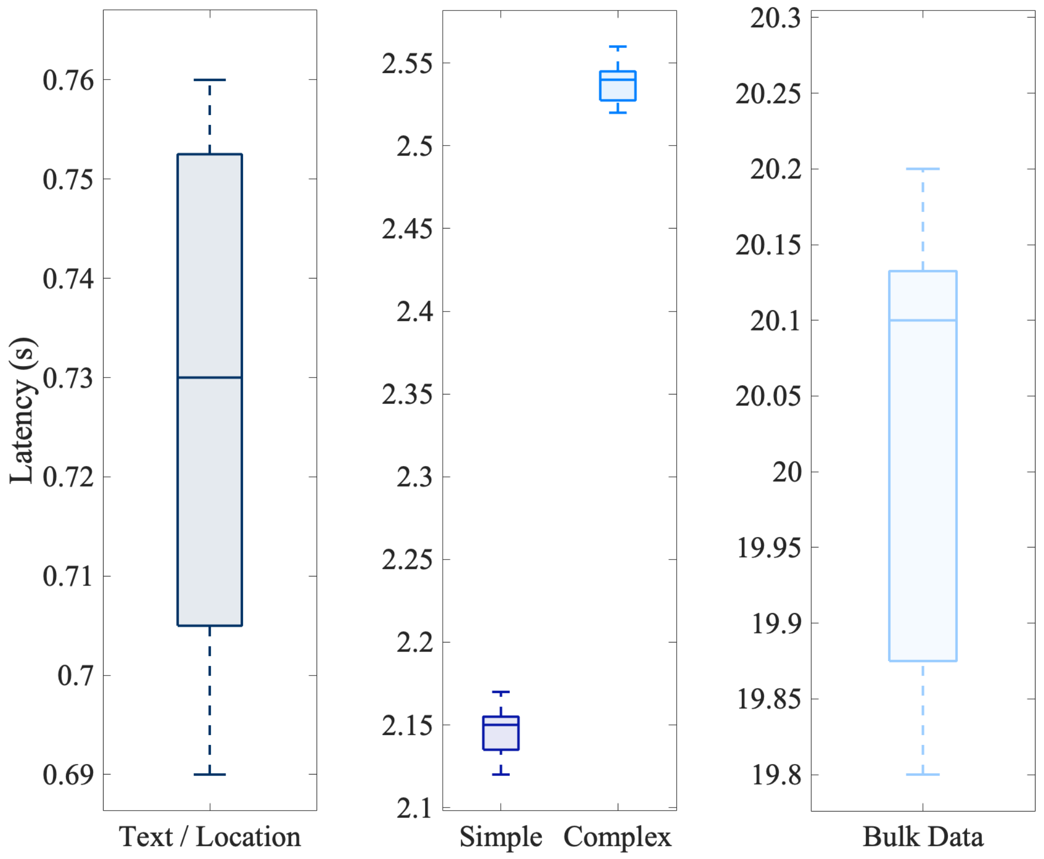


Figure 3.5: Latency with varying situational awareness data types

vertices), the latency is 2.15 s on average. In the second scenario markers and shapes have increased detail and complexity (e.g. *casevac* marker with all the attributes or polygonal shape with 4+ vertices), the latency reaches upwards of 2.5 s. Finally, for bulk data, we had a compressed file enclosing an image sized 6 KB. We observe that the latency is around 20.1 s on average. These numbers met the user QoS requirements of field trial evaluations.

Scalability We evaluated the application layer scalability of our Beartooth network in several military exercises, where it underwent significant stress, demonstrated by a large number of active devices and SA data packet transmissions. In one instance, we used around 100 Beartooth end-user devices, each updating their location every five seconds and exchanging SA text messages. Both field forces and authorities evaluating our network solution reported no issues with reliability or performance throughout the exercise, and we observed no service interruptions. These results highlight the Beartooth network's scalability and its ability to maintain functionality and dependability under high-demand conditions.

Conclusion and Future Work

In this paper we presented a COTS network deployment with Beartooth MKII radios and Gateways to provide SA to first responders and military forces spanning multiple Beartooth and IP networks. We believe extending the usage of COTS radios in the form of Beartooth MKII radios and introducing Gateway to interconnect IP networks through TAK can provide scalable SA at low-cost, with a compact form factor. Our experiments show Beartooth devices can support real-time text, location as well as more complex SA data types such as markers, shapes and bulk data with tolerable latency and QoS. Users have provided overwhelmingly positive feedback, with satisfaction regarding the currently supported network size and SA data types.

With several active war zone deployments and military training, the MKII radios and Gateways are currently being battle tested.

Our future plans include integrating the Beartooth Plugin with various systems and ATAK plugins, increasing compatibility by integrating diverse sensor applications. We aim to enhance Gateway performance via multi-threading and parallel processing. Developing a web interface is already underway for simplified Gateway management and WebTAK use, expanding SA capabilities to all web browser-capable platforms.

VOICE COMMUNICATION IN BANDWIDTH-CONSTRAINED NETWORKS: TEXT-TO-SPEECH VS. AUDIO CODECS

Problem Statement

Voice communications over networks are pivotal in many scenarios, largely due to their inherent efficiency and the rich information they can transmit. Unlike text-based methods, a voice message conveys both the identity of the user and the content of the message without the need for recipients to physically engage with their devices, such as typing or navigating through apps. This direct and easily discernible method of communication is crucial where quick and unmistakable understanding is vital. However, maintaining the clarity and reliability of voice communications, especially in settings with limited bandwidth, introduces a technical challenge that warrants detailed investigation and improvement.

Traditionally, audio encoding has been the go-to solution for voice communication needs over networks, serving as the conventional method for transmitting voice, even amidst various network environments. While audio codecs are optimized for many scenarios, and may even function in low-bandwidth networks, their dependency on substantial bandwidth for real-time communication becomes a glaring limitation in bandwidth-constrained networks, especially when congestion happens in urban areas or disruption occurs due to terrain-induced effects on RSSI and link quality degradation in rural wireless networks. The compromise then is often on the audio quality, delay, and disruption tolerances. These trade-offs not only jeopardize the clarity and accuracy of the message but also become notably detrimental in scenarios such as mission-critical applications where the accuracy and prompt delivery of the information are crucial.

However, the landscape of audio communication is rapidly evolving. The advancements in neural networks, deep learning techniques, and the pace of hardware development are not just pushing the boundaries of what is possible but also enabling the personalization and faithful recreation of the original audio at the receiver. The evolution in technology introduces a novel method that capitalizes on Text-to-Speech (TTS) models to address the challenge of clear voice communication in resource-constrained networks. Instead of transmitting larger packets of encoded voice data, the strategy of utilizing TTS models involves sending only text and basic user information. Once received, the voice can be regenerated at the receiver, capitalizing on the fact that text data is significantly leaner compared to its encoded audio counterpart. While traditional audio codecs don't require excessive bandwidth, their data packets are considerably larger and demand more bandwidth to transmit compared to the data needed to regenerate TTS audio. Employing TTS for audio communication offers an efficient means to manage scarce resources in limited-bandwidth scenarios, bolstering both the audio quality and information accuracy. This approach heralds a robust, resource-efficient alternative in the domain of voice communication over a network under constricted conditions.

The main contribution of this chapter is to provide insight into the comparative nuances of utilizing Text-to-Speech (TTS) models with varying vocoders versus traditional audio codecs in low-bandwidth networks. The insight derived provides a valuable perspective toward improving voice intelligibility, quality, clarity, and managing valuable network resources.

In this research, through a meticulous comparison of speech audio generated by neural models, vocoders, and traditional audio codecs using metrics such as Frechet Distance (FD), Intelligibility Score (IS), Contrastive Language-Voice Pretrained (CLVP) [45], and Perceptual Evaluation of Speech Quality (PESQ) [46] scores, we aim to provide comprehensive insights into the multifaceted relationships and impacts that various model and vocoder combinations can have on audio and network behavior. These metrics allow us to delve into the nuanced performance attributes of different combinations in varying network conditions. The CLVP model [45], inspired by OpenAI's Contrastive Language-Image Pre-Training (CLIP)-like architecture [47], distinctly measures the similarity between text and voice clips, thereby serving as a potential quality metric for Text-to-Voice models. Moreover, the derived CLVP Score evaluates the alignment between text and spoken audio, while FD provides a contrast between genuine spoken text and TTS outputs and IS, based on Audio Speech Recognition (ASR) and wav2vec [48] unsupervised speech recognition model, measures the intelligibility of a TTS system's output. By employing these metrics, the research aims to show how diverse model and vocoder combinations affect audio quality and network performance in low-bandwidth networks, promoting a comprehensive understanding crucial for enhancing clear and reliable communication.

This paper investigates the potential of TTS models to effectively replace traditional audio codecs, particularly under bandwidth constraints. Our initial findings indicate a promising trend. Specifically, the VITS [49] model delivers remarkable clarity, closely mirroring the original recordings, while Fastspeech2 [50] impresses with its rapid sample generation. As this research unfolds, it becomes evident that TTS systems might not just be alternatives, but potentially perform better in limited bandwidth networks. There's immense potential in refining these TTS models further, discovering more effective vocoder pairings, and, crucially, transitioning from controlled environments to real-world, mission-critical applications. This not only highlights the validity of our current findings but also signifies our commitment to elevating audio communication standards in the face of bandwidth constraints in wireless networks.

Background

In this chapter, we explore key components and concepts of our research on voice communication in limited networks. We discuss Text-to-Speech (TTS) models and their main components, describe key concepts and features of the audio codecs we used, and describe the metrics that guide our evaluation.

Text-to-Speech

Speech synthesis, often referred to as Text-to-Speech (TTS), is the process of turning text into understandable and natural-sounding speech using natural language processing, signal processing, and machine and deep learning techniques. Previous work combines techniques to develop various TTS models focusing on different aspects of speech, like fast inference, high quality, low compute complexity, etc. A TTS model consists of three main components; Text Analysis, Acoustic Modeling, and Vocoding.

Text analysis translates the raw text into linguistic features [51]. It lays the groundwork for speech synthesis by addressing the text's pronunciation, normalization, and segmentation [52]. While recent end-to-end neural TTS methods simplify much of this module, tasks such as text normalization and grapheme-to-phoneme conversion remain essential in managing diverse text formats and extract specific phonemes [53].

Acoustic modeling turns linguistic features into a spectral representation, setting the stage for vocoding, or actual speech generation [54]. Several models have emerged to address different challenges in TTS. For instance, Tacotron utilizes a sequence-to-sequence model with attention mechanisms to map text to mel spectrograms [55]. FastSpeech offers a non-autoregressive approach, emphasizing faster speech synthesis by eliminating recurrent computations [50]. VITS, on the other hand, is unique in its hybrid approach, integrating both Variational AutoEncoders (VAEs) and adversarial training from Generative Adversarial Networks (GANs) with a sequence-to-sequence model [49]. This blend aims to produce high-fidelity and varied speech.

The last component, vocoding, is responsible for the act of generating the audible speech waveform. Utilizing various techniques differentiates vocoders. One of the vocoder types Autoregressive vocoders, operates sequentially which potentially leads to slower audio speech generation [56]. Flow-based vocoders harness normalizing flows for parallel waveform generation [57]. GAN-based vocoders, such as Parallel WaveGAN and MelGAN, employ Generative Adversarial Networks, optimizing the quality of the generated waveforms [58].

Audio Codecs

Traditionally, when transmitting voice across networks, an audio codec applies a process called encoding on recorded audio samples. This process involves lossy compression, which reduces the size of the audio but can also diminish its clarity and quality. While numerous proprietary compression techniques and audio codecs exist, our selection focuses on open-source codecs. Specifically, we prioritize codecs capable of compressing audio to levels suitable for low-bandwidth networks.

Starting with more traditional audio codec, Codec 2 is a low-bitrate speech audio codec that utilizes sinusoidal coding, a technique tailored for human speech [19]. It operates at bit rates ranging from 450 bps to 3.2 Kbit/s, ideal for low-bandwidth networks like Mobile Ad-Hoc Networks (MANETs). Sinusoidal coding models speech by summing harmonically related sine waves atop a speaker's fundamental frequency. It encodes the pitch and amplitude of the harmonics and exchanges them across channels in a digital format. This approach, closely related to multi-band excitation codecs, relies on periodicities in overtone frequencies to recreate spoken audio efficiently.

Transitioning from the traditional methods employed by Codec 2, Google's Lyra illustrates a stride towards integrating machine learning with audio compression [59]. Like its traditional counterparts, Lyra extracts distinctive speech features but leverages a generative model to recreate the speech signal, enhancing the audio quality while maintaining low bitrates ranging from 3.2 Kbit/s to 9 Kbit/s, suitable for real-time communications in bandwidth-constrained environments.

Similarly, Facebook's Encodec embraces neural vocoding techniques, akin to Google's Lyra. It employs a neural network-based encoder-decoder architecture for real-time, high-fidelity audio compression and supports encoding rates of 1.5, 3, 6, 12, and 24 Kbit/s [60]. The utilization of neural networks illustrates a common feature between Lyra and Encodec, showcasing a modern approach toward achieving efficient audio compression while maintaining high audio quality. This approach caters to the demands of real-time audio streaming and communication in resource-limited networks.

Metrics

Next, we delve into the metrics we used for evaluating both TTS models and audio codecs in low-bandwidth networks. These metrics offer a measurable insight into the quality and efficiency of the audio compression techniques as well as TTS efficacy.

Fréchet Distance

The first metric we utilize is the Fréchet Distance (FD), a concept described in detail in work by Alt et al. [61]. In the realm of audio and Text-to-Speech (TTS) models, FD plays a pivotal role in quantitatively assessing model performance. Specifically, FD is employed to compare the reference audio and the audio that is in question. However, it is crucial to clarify that in our approach, we do not directly compute Fréchet Distance on voice signals but rather extract feature vectors from the audio to compute FD. Extracting feature vectors offers significant advantages: it reduces the complexity of voice signals, therefore enhancing computational efficiency, and normalizes the data, ensuring comparability between different samples. The comparison between reference audio can either be with synthesized speech generated by a TTS model or an audio codec-processed voice sample. Computing the FD score hinges on the computation of the Fréchet Distance between two feature vectors generated by the last layer of the Contrastive Language-Voice Pre-trained (CLVP) model. To compute FD, we first use feature vectors from the last layer of the CLVP model for both real and generated/processed speech samples. Then, we calculate the mean and covariance of these feature vectors for each set of samples. The FD score is finally obtained by measuring the Fréchet Distance between the two Gaussian distributions represented by these statistical measures. Mathematically, this distance is given by:

$$FD = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \tag{4.1}$$

where, μ_r, μ_g represent the means, and Σ_r and Σ_g denote the covariance matrices of the feature vectors from the real and generated speech samples, respectively. Tr is the trace of a matrix, representing the sum of its main diagonal elements.

A lower FD score indicates a closer resemblance between the two distributions, signifying a higher fidelity of the generated speech in mirroring real spoken text, thereby reflecting superior model performance.

CLVP Score Inspired by OpenAI's Contrastive Language-Image Pretraining (CLIP), the Contrastive Language-Voice Pre-trained (CLVP) model is specifically designed for audio-text pairs, extending the concept beyond CLIP's image-text focus [47].

CLVP employs contrastive learning to align matching audio-text pairs closely and distinguish non-matching pairs, using datasets like LJ Speech [62]. This approach enables the model to learn from a diverse range of audio-text examples, enhancing its understanding and accuracy.

The architecture of CLVP model features a dual-encoder setup: an audio encoder processes audio clips, and a text encoder manages textual descriptions. The model expects audio and its matching transcript. Through each corresponding encoder, the model transforms the input pair into low-dimensional data called embeddings to store extracted features, which are then projected to a multi-dimensional space called shared latent space to facilitate comparison.

In the shared latent space, the model computes the CLVP score, a measure of the similarity between the text and speech embeddings by performing a dot product calculation between each corresponding text and audio pair called Einstein Sum [63]. This calculation yields a value that directly correlates with the pair's similarity: a greater dot product indicates higher degrees of similarity between the pairs thus a higher CLVP score. The CLVP score is central to evaluating Text-to-Speech (TTS) systems, as it quantitatively assesses how closely the generated speech matches the input text. A higher CLVP score indicates a more accurate alignment, reflecting the model's effectiveness.

Through its innovative architecture and contrastive learning approach, the CLVP model emerges as a significant tool for evaluating and improving TTS systems and Automatic Speech Recognition technologies.

Intelligibility Score The Intelligibility Score (IS) is a metric in evaluating Text-to-Speech (TTS) systems, leveraging the Wav2Vec model, particularly trained to discern correct audio snippets from distractors using the contrastive loss function [48].

In the context of TTS evaluation, the IS employs the `Wav2Vec2ForCTC` variant to analyze TTS-generated audio in comparison to the original text. Given the original transcription and the audio in question, `Wav2Vec2ForCTC` transcribes the audio into text, and this transcribed text is then compared to the original text for evaluation. This analysis is grounded in the principles of Connectionist Temporal Classification (CTC) loss, a method widely used in automatic speech recognition (ASR) for aligning continuous audio input with its corresponding discrete text output [64].

CTC loss addresses the challenges of sequence alignment in ASR without requiring a predetermined alignment. It introduces a 'blank' character to manage the discrepancies in sequence lengths and to accommodate the variability in speech patterns as seen in Figure 4.1. By considering all possible alignments and their cumulative probabilities, the CTC loss function calculates the likelihood of the target transcription given the input audio, guiding the model to maximize this likelihood during training.

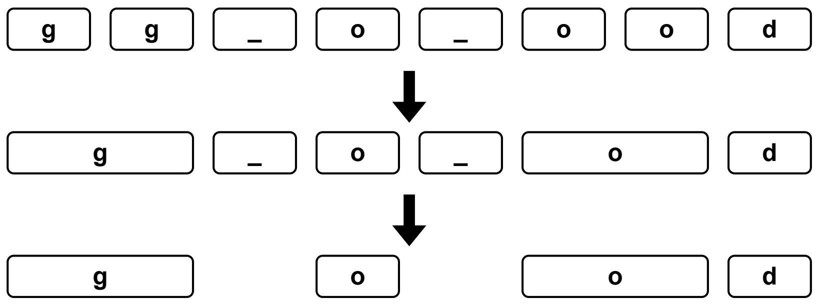


Figure 4.1: Steps taken by CTC method discern the word 'good'.

In TTS systems, the Intelligibility Score (IS) is calculated using the CTC loss to evaluate the accuracy of ASR models in transcribing TTS-generated speech compared to real human speech. The process starts with normalizing audio samples from both TTS and real speech to standardize input levels. The ASR model, specifically `Wav2Vec2ForCTC`, then computes the CTC loss by comparing its transcription to the actual text, pinpointing differences. When real speech is present, this loss is adjusted to factor in natural speech intelligibility variations. The final IS, reflecting the mean of these adjusted CTC losses, indicates how closely the TTS-generated speech resembles the original text in content and clarity. Lower scores suggest better intelligibility and naturalness of the TTS model.

Perceptual Evaluation of Speech Quality The Perceptual Evaluation of Speech Quality (PESQ) serves as an objective tool for estimating the Mean Opinion Score (MOS), a standard method used for evaluating the quality of voice recordings [46]. While MOS traditionally relies on subjective assessments from numerous participants, PESQ offers an efficient, objective alternative that can assess the quality of telephone networks and audio codecs without direct user feedback.

PESQ operates by analyzing the degradation and noise in audio that has been transmitted over a telephone network or encoded by an audio codec. It employs a process that begins with the alignment of a reference voice signal with a degraded one, followed by an auditory transformation simulating the human hearing process. The algorithm then calculates perceptual differences, using a psychoacoustic model to emphasize critical aspects of speech as perceived by human listeners [46].

However, PESQ is not ideally tailored for TTS-generated audio evaluations. TTS systems are evaluated on criteria beyond mere signal degradation, such as tone, natural sounding, and pronunciation clarity. Consequently, PESQ yields lower scores for TTS systems, approaching its baseline of 1. This discrepancy primarily may arise from the variations in duration and subtle acoustic differences between the TTS-generated voice signal and the reference voice signal, highlighting the divergence in specific attributes such as tone and pronunciation which results in lower PESQ score.

Although PESQ is not ideally suited for evaluating TTS-generated audio, research by Cernak et al. and Bhattacharjee et al. suggests a correlation between PESQ scores and MOS in the context of TTS models [65, 66]. Both work indicates that PESQ can still offer valuable insights into the perceptual quality of TTS systems once it is scaled to default MOS scale, providing a potential avenue for approximating TTS quality in scenarios where traditional MOS testing is impractical.

Inference Time In the context of real-time voice communication over bandwidth-constrained networks, evaluating TTS models using the inference time metric is crucial because it directly affects real-time data delivery in real-time voice communication scenarios. In this study, inference time was precisely measured from the timestamp the input text was provided to the TTS model until the audio output file was generated. The duration a TTS model takes to translate text into natural-sounding speech is a key performance indicator in environments where network resources are limited, and keeping latency as low as possible is essential. For such applications, it is imperative that the TTS model not only generates clear and understandable audio but does so with minimal delay. This requirement is vital in maintaining effective communication, ensuring that the generated speech is delivered promptly without taxing the limited network resources, especially in mission-critical applications. The challenge lies in optimizing TTS models to achieve a balance between swift response times and maintaining speech clarity, all within the constraints of limited bandwidth. This balancing act is especially vital in domains like mission-critical applications where real-time data delivery is as important as the quality and clarity of the voice signal output.

Methodology

I conducted our experiments on a computer, featuring an Intel Core i9-9900K CPU and an NVIDIA GeForce RTX 2080 SUPER GPU, with 3072 CUDA cores [67]. The setup includes 32 GB of RAM and an INTEL 660P series SSD [68], facilitating swift data processing essential for model training purposes. I opted to use the LJ Speech dataset [62], a widely-utilized resource in Text-to-Speech research. The dataset offers around 24 hours of single-speaker English recordings with sampling rate of 22.05 KHz and serves as the experimental foundation for this study. Its notable usage in prior research provides various pre-trained vocoders, enabling a robust comparison of TTS models and traditional audio codecs within our evaluation.

For this study, I utilized three prominent TTS models: FastSpeech2 [50], Tacotron2 [55], and VITS [49]. Utilizing the ESPNet 2 framework [69], a comprehensive toolkit for exploring a variety of TTS models and vocoders, I coupled TTS models with an array of vocoders such as Parallel WaveGAN [70], HiFiGAN [71], Style MelGAN [72], Fullband MelGAN, and Multiband MelGAN [73]. Table 4.2 lists the vocoder output Mel Range, training constraints, and overall model checkpoint size after respective number of iterations [74]. I selected audio codecs, Codec 2 [19], Lyra [59] and Encodec [60], that are particularly suited for networks with constrained resources, emphasizing low encoding rates. I gathered our data by randomly picking audio samples from the LJ Speech Dataset and generating corresponding speech samples from transcripts across all TTS systems. Then, I encoded and decoded these original recordings using the chosen audio codecs and encoding rates. Finally, I compared the TTS-generated samples with the decoded audio samples derived from the original recordings.

In my investigation, I employed Analysis of Variance (ANOVA) test as the primary statistical method to recognize and validate significant differences in the mean scores of various metrics across distinct groups of TTS model and vocoder pairings and audio codecs. The ANOVA yielded p-values substantially lower than the conventional significance level of 0.05 for each metric, suggesting apparent differences in model performances. However, the reliability of ANOVA hinges on certain assumptions – namely, the normality of data and homogeneity of variances across groups [75].

To verify these assumptions, I conducted the Shapiro-Wilk test for normality and Levene's test for homogeneity of variances [76, 77]. The results, unexpectedly, indicated a departure from normality and unequal variances among the groups. This meant that we could not utilize ANOVA test, as these tests have fundamentally shown.

Given these violations, I shifted my analytical approach to the Kruskal-Wallis test, a non-parametric alternative to ANOVA. The Kruskal-Wallis test is particularly suited for data that do not meet the normality and homogeneity prerequisites of ANOVA [78]. This test evaluates the differences in medians rather than means and is less sensitive to the distribution of the data, thereby providing a more appropriate analysis framework for our dataset. The test affirmed the initial indications from the ANOVA, revealing significant differences across TTS models and audio codecs for all considered metrics as detailed in Table 4.1. This reinforced the conclusion that the variances observed in the metrics across models were not mere coincidences but statistically significant. Such findings underscore the criticality of selecting appropriate statistical tests and validating their underlying assumptions, especially in the context of comparative analyses.

Table 4.1: ANOVA Validity and Kruskal-Wallis Test Results

Metric	Test	Test Statistic	p-value	Conclusion
FD	Shapiro-Wilk	0.74	7.35×10^{-19}	Normality violated
	Levene's	2.35	1.40×10^{-3}	Homogeneity violated
	Kruskal-Wallis	158.77	1.30×10^{-23}	Statistically significant
IS	Shapiro-Wilk	0.55	5.08×10^{-24}	Normality violated
	Levene's	2.09	5.40×10^{-3}	Homogeneity violated
	Kruskal-Wallis	52.25	1.05×10^{-4}	Statistically significant
CLVP	Shapiro-Wilk	0.80	2.00×10^{-16}	Normality violated
	Levene's	1.12	3.30×10^{-1}	Homogeneity not violated
	Kruskal-Wallis	154.09	1.03×10^{-22}	Statistically significant
PESQ	Shapiro-Wilk	0.66	3.32×10^{-21}	Normality violated
	Levene's	1.92	1.24×10^{-2}	Homogeneity violated
	Kruskal-Wallis	178.88	1.61×10^{-27}	Statistically significant

Vocoder	**Mel Range (Hz)**	**# of Iterations**	**Checkpoint Size**
Parallel WaveGAN (v3)	80-7600	3M	214.9MB
Fullband MelGAN (v2)	80-7600	1M	138.4MB
Multiband MelGAN (v2)	80-7600	1M	105.3MB
HiFiGAN (v1)	80-7600	2.5M	968.9MB
Style MelGAN (v1)	80-7600	1.5M	108.5MB

Table 4.2: Training and checkpoint information for vocoders used in the evaluation.

In this section, we strictly compare TTS results to traditional audio codecs. To achieve objective comparison we use the following metrics.

<u>Frechet Distance</u>

Figure 4.2 presents the distribution of FD values among 15 unique combinations of TTS models and 6 audio codecs. The x-axis denotes the FD values, while the various pairings of TTS models and audio codecs are outlined on the y-axis.

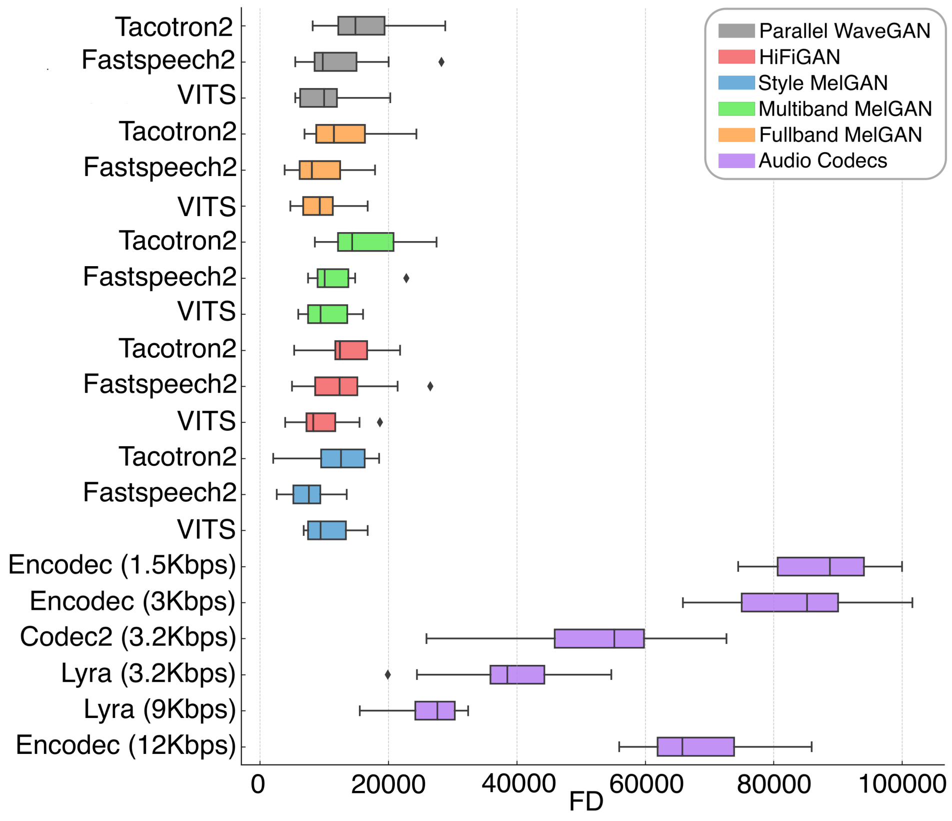


Figure 4.2: Frechet Distance between the original sample from LJ Speech dataset and synthesized/decoded speech.

The Figure 4.2 reveals that both FastSpeech2 and Tacotron2 exhibit higher FD values compared to VITS, indicating that VITS maintains a closer resemblance to the original recording during its speech synthesis. It is also evident that Tacotron2 displays a broader range of values, suggesting some level of inconsistency in its output. In the realm of vocoders, Style MelGAN and Fullband MelGAN consistently demonstrate lower FD values, outperforming their counterparts when integrated with all three models.

In comparing audio codecs, it is clear that the combined output from any TTS model and vocoder more closely mirrors the original sample than that produced solely by the audio codecs. As anticipated, a reduction in encoding rate is associated with a compromise in quality. This correlation is pronounced in the Encodec with a 1.5 Kbit/s encoding rate, which exhibits the highest FD, diverging most from the original sample.

When drawing parallels among audio codecs with proximate encoding rates – Encodec at 3 Kbit/s, Codec2, and Lyra at 3.2 Kbit/s – Encodec is discerned to produce speech with a higher FD. While Codec2 and Lyra showcase comparable efficacy, Lyra slightly edges out Codec2, possibly due to its unconventional audio encoding approach. Notably, despite boasting a loftier encoding rate, Encodec at 12 Kbit/s still registers a higher FD than Lyra at 9 Kbit/s. This observation clearly shows the optimized nature of the Lyra audio codec, marking it as a better choice over Encodec.

Intelligibility Score (IS)

Figure 4.3 depicts the distribution of IS across 15 different combinations of TTS models paired with 6 distinct audio codecs. Similar to Figure 4.2, on the x-axis we have IS, whereas the y-axis represents the various combinations of TTS models and audio codecs.

An obvious observation emerging from the data is the pronounced spread of VITS model in its distribution relative to the other two models. Specifically, when paired with Parallel WaveGAN, this combination yields results with notable variability and it suggests that the remaining models offer a more consistent mapping to the original transcript. Furthermore, Fastspeech2 manifests the narrowest distribution, leading

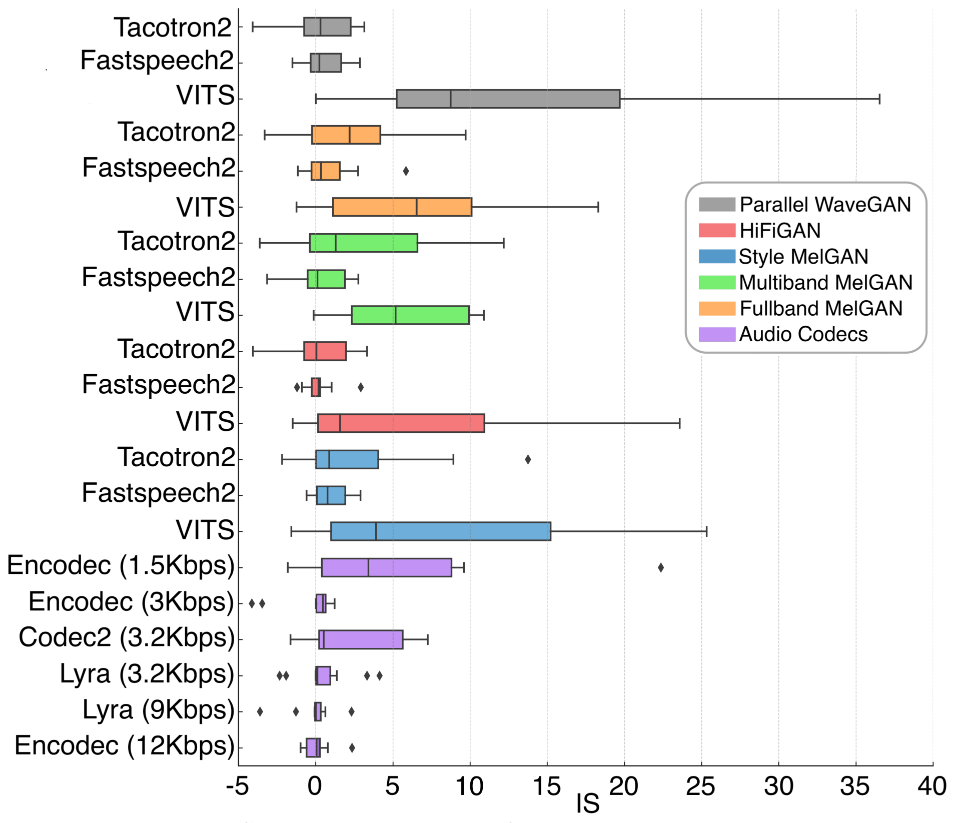


Figure 4.3: Intelligibility Score based on LJ Speech transcripts and synthesized/decoded speech.

to highly consistent outcomes. In contrast, Tacotron2, despite its broader spread, consistently reports the lowest IS across all vocoder pairings.

Overall, audio codecs and TTS models seem to showcase comparable performance. However, a subtle performance improvement is discernible in favor of audio codecs when considering the spread of their distributions.

A crucial point of consideration that we need to make is that the CLVP model is trained on the LibriSpeech [79] and Common Voice datasets [80], followed by fine-tuning on libriTTS [81]. Given that our evaluation uses the LJ Speech dataset, we encounter results that appear counterintuitive, suggesting superior intelligibility over the actual ground truth. This anomaly can be explained by the ideal recording conditions of the LJ Speech dataset and its single-speaker nature. Consequently, certain TTS model pairings might appear to synthesize speech surpassing the original quality.

CLVP Score

Figure 4.4 displays the CLVP score in a descending order on the x-axis, with combinations of TTS models and vocoders, as well as the related audio codecs, on the y-axis. Traditionally, a lower CLVP score should indicate a closer representation of text within the audio according to [45]. However, our results challenge this premise. Audio codecs with lower encoding rates, which would be expected to have greater losses, curiously produce lower CLVP scores. This counterintuitive finding suggests that a higher CLVP score might actually offer a more accurate representation of text in the audio. Furthermore, it's clear that TTS model and vocoder pairings generally outperform audio codecs in fidelity. The standout is the Fastspeech2 model paired with the Style MelGAN vocoder, achieving a CLVP score close to 15, while other TTS models hover between CLVP scores of 13 and 14. Among audio codecs, Lyra with 9 Kbit/s consistently achieves the highest CLVP score, yet it still lags behind the average performance of TTS models.

PESQ Score

Figure 4.5 displays the PESQ scores on the x-axis, contrasted against various model and vocoder combinations on the y-axis. A distinct separation is observable between TTS models and audio codecs. While audio codecs with higher encoding rates often achieve higher PESQ scores, it is important to note that PESQ primarily assesses degradation and noise in audio quality rather than elements like pronunciation, tone, and punctuation. Consequently, this nature of PESQ does accurately portrays the actual quality and leads to TTS-generated speech scores

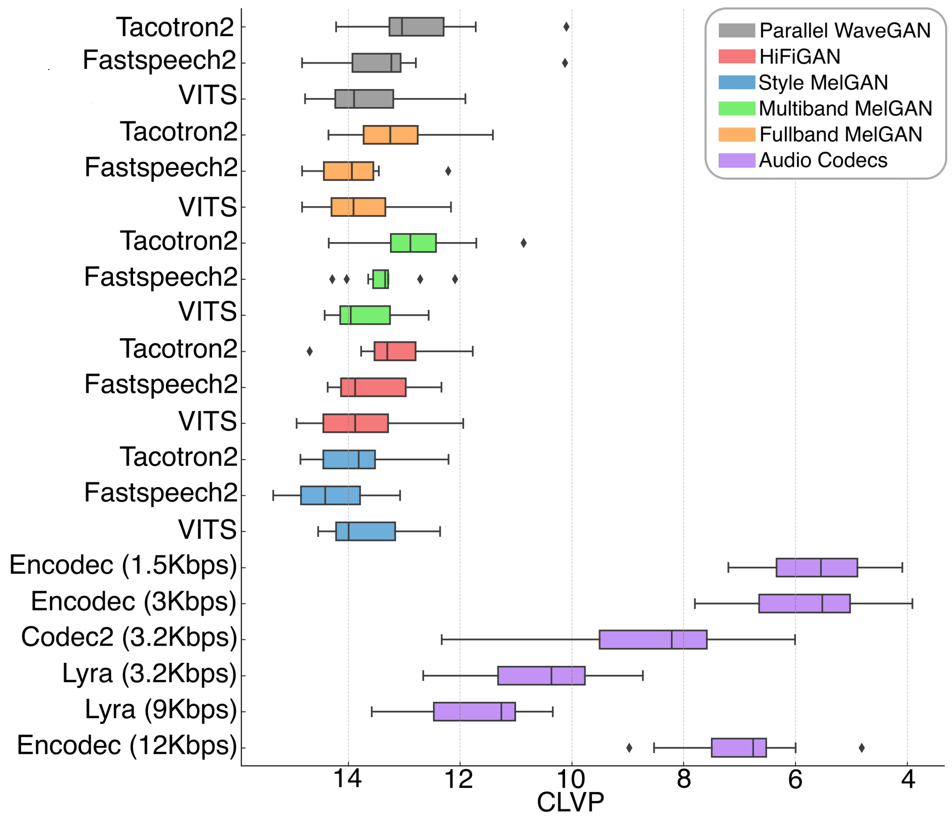


Figure 4.4: CLVP Score based on LJ Speech transcripts and synthesized/decoded speech.

typically ranging between 1 and 1.5. Another trend worth mentioning is that Encodec and Lyra, with encoding rates close to 3 and 3.2 Kbit/s, outperform Codec2.

Inference Duration

In assessing hardware performance and computational complexity, I focused on the inference duration required by TTS models to generate speech samples. Compared to the simpler decoding process of audio codecs, TTS models exhibit varying inference durations based on model and vocoder combinations. Figure 4.6 displays the inference duration, with the x-axis denoting duration in seconds and the y-axis showing model and vocoder combinations. The results were consistent; VITS, which produced more

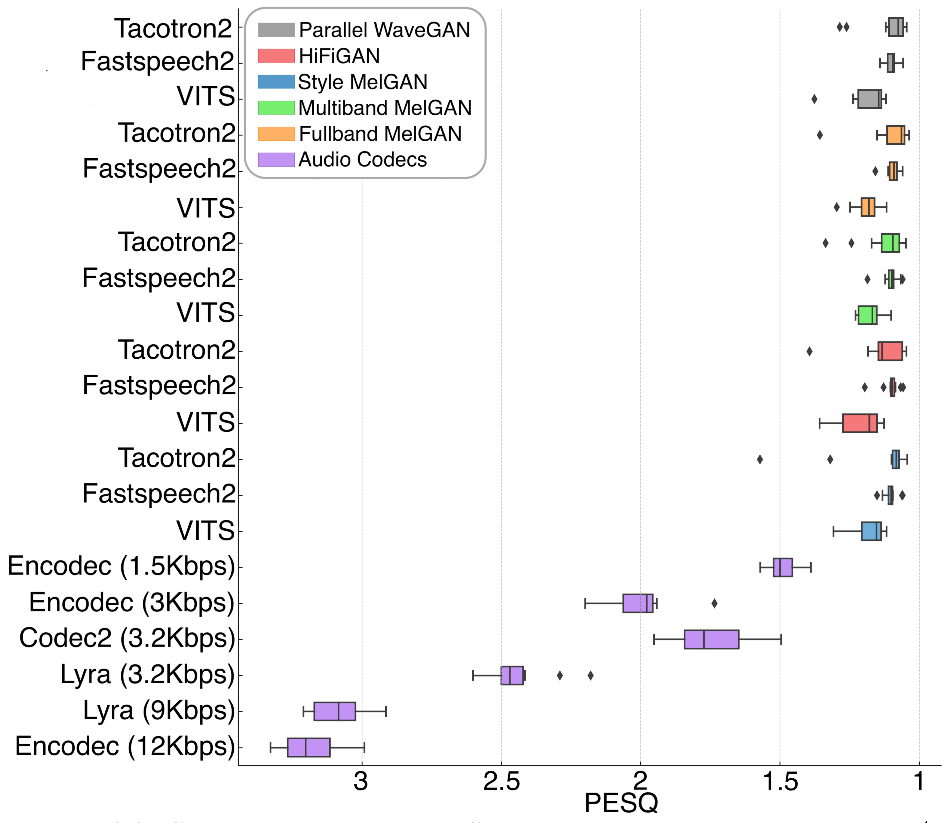


Figure 4.5: PESQ Scores based on LJ Speech dataset and synthesized/decoded speech.

accurate speech samples, took about 300 to 700 ms longer per sample. Interestingly, the choice of vocoder in VITS did not affect the duration. On the other hand, Fastspeech2 generated samples fastest as the name suggests. Among vocoders, HiFiGAN had the longest inference duration for both Tacotron2 and Fastspeech2 correlated with its checkpoint size as shown in Table 4.2, aligning with its higher quality and clarity as well as complexity.

The results further indicate the performance in a bandwidth-constrained network scenario, for instance, a network with a bandwidth of 10 Kbit/s, an audio codec with an encoding rate of 3.2 Kbit/s can facilitate three concurrent real-time voice streams. Link layer protocol can partition encoded real-time voice samples into smaller chunks

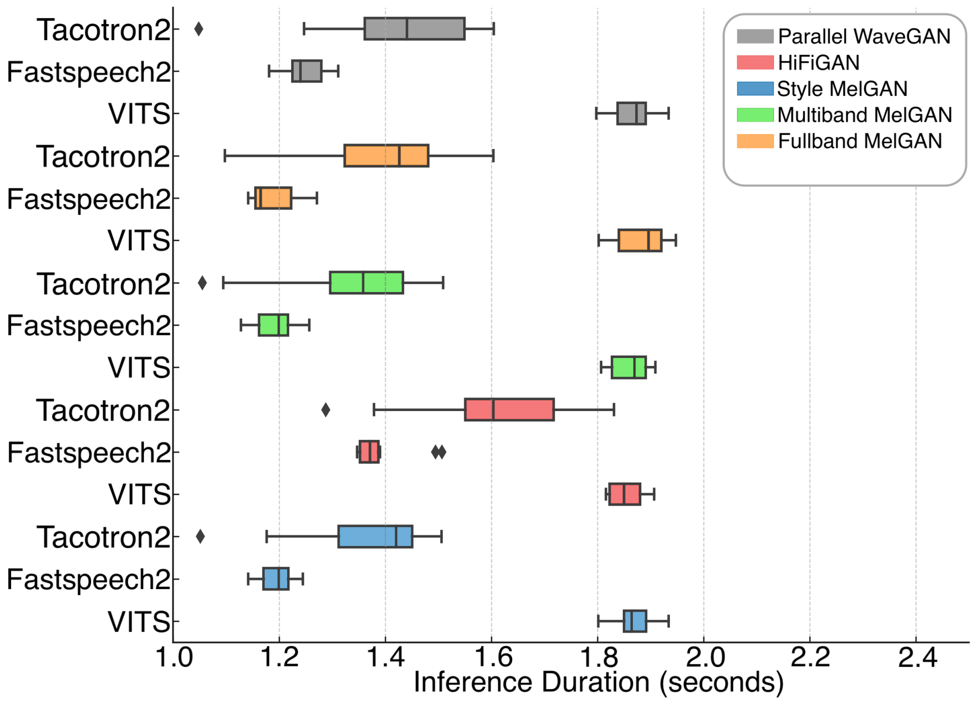


Figure 4.6: Inference time required for each TTS model and vocoder combinations.

for transmission over the network. To ensure smoother playback, voice samples can undergo a buffering process at the receiver. The overall latency, including the buffering latency, typically remains below the range of 500 ms, as the evidence shown in an earlier research [42]. In a network with similar resources and configuration, TTS can handle not just three concurrent data streams, but a significantly larger number. I determined the median text size from the LJ Speech dataset samples used for TTS evaluation by counting characters and character memory allocation which is approximately 128 B or 1.028 Kb. With the median duration of generated TTS voice samples at 8.74 s, we can calculate the encoding rate (ER) using the formula:

$$ER = \frac{Data\,Size}{Duration} \tag{4.2}$$

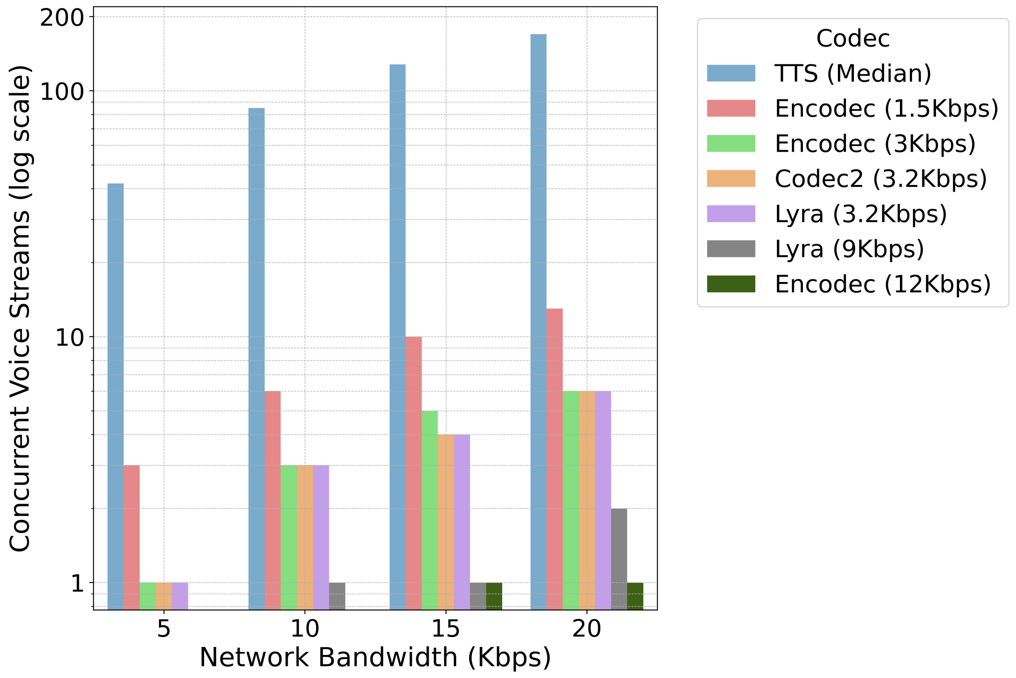


Figure 4.7: Concurrent voice streams possible within varying network bandwidth.

Substituting the given values yields:

$$ER = \frac{1.028\,Kb}{8.74\,s} \approx 0.117\,Kbit/s \tag{4.3}$$

This translates to eighty-five concurrent transmissions, a significant contrast to the three allowed by audio codecs. Figure 4.7 offers a more detailed understanding of concurrent transmissions in logarithmic scale on the y-axis against diverse network bandwidths on the x-axis. It is evident that TTS utilization greatly boosts the potential for concurrent transmissions, outpacing other audio codecs by a wide margin. However, it is crucial to recognize that the inference process at the receiver does slightly increase latency compared to traditional audio codecs, resulting in delays between 1.2 s to 1.9 s as shown in Figure 4.6 [42].

Conclusion and Future Work

In this study, we investigated the efficiency of Text-to-Speech (TTS) models in comparison to traditional audio codecs in low-bandwidth conditions. Utilizing metrics such as Fréchet Distance, Intelligibility Score, CLVP, and PESQ scores as well as inference time, we discerned the performance characteristics of various models, vocoders, and audio codecs in bandwidth-constrained environments. The study initially employed the ANOVA test to validate statistical significance, but due to violations in normality and homogeneity assumptions, we shifted to the Kruskal-Wallis test due to the non-parametric approach. This methodological pivot, not only validated the significant differences across TTS models and audio codecs but also highlighted the importance of selecting appropriate statistical tests in comparative analyses. While audio codecs consistently performed well in the Intelligibility Score, TTS models, especially when paired with the appropriate vocoders, demonstrated superior audio clarity as evidenced by metrics like Fréchet Distance, Intelligibility, and CLVP Scores. Notably, VITS emerged as the leading model in terms of audio fidelity, whereas Fastspeech2 excelled in processing speed, as indicated by the inference duration metric. We further delved into the implications of inference duration in resource-constrained networks. In such settings, TTS systems offer efficient resource management, allowing a network to support a higher number of concurrent TTS-generated playbacks, provided the application can tolerate the inherent latency associated with inference. Future endeavors will center on enhancing the assessed TTS models, investigating diverse vocoder combinations, and broadening the evaluation to include real-world, mission-critical applications. This will further validate our findings and contribute to advancements in real-time voice communication in bandwidth-constrained networks.

CONCLUSION

The integration and advancement of wireless communication technologies have played a crucial role in shaping modern applications, from consumer products to military applications. This thesis presented two innovative approaches to address the challenges in diverse areas of wireless communication by proposing the Beartooth Relay Protocol (BRP) and the Beartooth MKII radio system.

BRP, a flexible MAC protocol designed for LoRa, aims to enhance the performance of real-time and streaming applications. By employing BRP with an advanced LoRa radio, the protocol can address the limitations of previous LoRa MAC protocols, such as constrained flexibility, high latency, and difficulties with real-time data transmission. This development expands LoRa's applicability to a more diverse range of IoT scenarios in a wider range of applications.

Further, the Beartooth MKII radio system, based on the low-cost and power-efficient XBee radio platform, provides a scalable and cost-effective situational awareness (SA) solution for military applications. The Beartooth MKII radio's integration with the Android Tactical Assault Kit (ATAK) and the Beartooth Gateway enable seamless and secure communication among squads, commanders, and command centers. Additionally, the compact form factor of the Beartooth MKII radio offers significant weight reduction compared to existing solutions.

In the concluding chapter, I delved into an essential component of real-time SA: voice communications within bandwidth-limited networks. The examination centered on the efficacy of Text-to-Speech (TTS) systems versus the conventional method of transmitting audio codec compressed speech data. The results indicated that voice samples generated by TTS not only provide superior clarity but also necessitate less data transmission. This, in turn, enables the network protocols to allocate resources more wisely.

In summary, the proposed solutions demonstrate the potential for advancing the capabilities of wireless communication technologies in both civilian and military contexts. By leveraging the flexibility and configurability of these systems, it is possible to address the challenges associated with real-time data streaming and situational awareness in a wide range of applications. Future research should focus on evaluating on further optimizing these systems for specific use cases and evaluating their performance in real-world scenarios and mission-critical applications.

REFERENCES CITED

[1] “2019 Broadband Deployment Report,” Federal Communications Commission (FCC), Washington, D.C., Tech. Rep. 19-44, 2019.

[2] P. Mach, Z. Becvar, and T. Vanek, “Internet of Mobile Things: Overview of LoRaWAN, DASH7, and NB-IoT in LPWANs Standards and Supported Mobility,” *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, Oct. 2018.

[3] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, and D. Pesch, “TS-LoRa: Time-slotted LoRaWAN for the industrial internet of things,” *Computer Communications*, vol. 153, Mar. 2020.

[4] M. Bor, J. Vidler, and U. Roedig, “LoRa for the internet of things,” in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, Feb. 2016.

[5] L. Leonardi, F. Battaglia, and L. L. Bello, “RT-LoRa: A medium access strategy to support real-time flows over LoRa-Based networks for industrial IoT applications,” *IEEE Internet of Things J.*, vol. 6, no. 6, pp. 10 812–10 823, Dec. 2019.

[6] L. Leonardi, F. Battaglia, G. Patti, and L. L. Bello, “Industrial LoRa: A novel medium access strategy for LoRa in industry 4.0 applications,” in *IEEE Industrial Electronics Society*, Dec. 2018.

[7] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, “A Study of LoRa: Long Range & Low Power Networks for the Internet of Things,” *MDPI Sensors*, vol. 16, no. 9, Sep. 2016.

[8] “LoRa — Platform for IoT,” [Online]. Available: https://www.semtech.com/lora.

[9] J. C. Liando, A. Gamage, A. W. Tengourtius, and M. Li, “Known and Unknown Facts of LoRa: Experiences from a Large Scale Measurement Study,” *ACM Trans. Sensor Netw.*, vol. 15, no. 2, Feb. 2019.

[10] Q. L. Hoang, H. P. Tran, W.-S. Jung, S. H. Hoang, and H. Oh, “A slotted transmission with collision avoidance for LoRa networks,” *Procedia Computer Science*, vol. 177, Nov. 2020.

[11] R. Piyare, A. L. Murphy, M. Magno, and L. Benini, “On-demand tdma for energy efficient data collection with LoRa and wake-up receiver,” in *Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct. 2018.

[12] R. K. Singh, R. Berkvens, and M. Weyn, “Time synchronization with channel hopping scheme for LoRa networks,” in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*, Oct. 2020.

[13] W. Wu, Y. Li, Y. Zhang, B. Wang, and W. Wang, “Distributed queueing-based random access protocol for LoRa networks,” *IEEE Internet of Things Journal*, vol. 7, no. 1, Oct. 2020.

[14] R. E. Chall, S. Lahoud, and M. E. Helou, “LoRaWAN Network: Radio Propagation Models and Performance Evaluation in Various Environments in Lebanon,” *IEEE Internet of Things J.*, vol. 6, no. 2, pp. 2366–2378, Mar. 2019.

[15] N. Abramson, “Development of the ALOHANET,” *IEEE Transactions on Information Theory*, vol. 31, no. 2, Mar. 1985.

[16] H. T. Yew, M. F. Ng, S. Z. Ping, S. K. Chung, A. Chekima, and J. A. Dargham, “IoT based real-time remote patient monitoring system,” in *IEEE International Colloquium on Signal Processing Its Applications (CSPA)*, Apr. 2020.

[17] M. Hadded, P. Muhlethaler, A. Laouiti, and L. A. Saidane, “A centralized TDMA based scheduling algorithm for real-time communications in vehicular ad hoc networks,” in *Software, Telecommunications and Computer Networks (SoftCOM)*, Dec. 2016.

[18] “Team Awareness Kit: Tactical Situational Awareness Solution,” Fact Sheet, Department of Homeland Security, Mar. 2020, [Online]. Available: https://tinyurl.com/TAKInfoSheet.

[19] “Codec 2,” Jul. 2019. [Online]. Available: http://www.rowetel.com/?page_id=452

[20] “AN1200.22 LoRa^{TM} Modulation Basics,” Application Note, Semtech Corporation, May 2015, [Online]. Available: https://tinyurl.com/LoRaModBasics.

[21] J. Petajajarvi, K. Mikhaylov, A. Roivainen, T. Hanninen, and M. Pettissalo, “On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology,” in *IEEE ITS Telecommunications (ITST)*, Dec. 2015.

[22] LoRa Alliance Technical Committee, “LoRaWAN^{TM} 1.1 Specification,” Contribution, LoRa Alliance^{TM}, Oct. 2017, [Online]. Available: https://tinyurl.com/LoRaWAN11Specs.

[23] “Symphony Link ^{TM} vs. LoRaWAN ^{TM}: A Guide for Engineers and Decision Makers,” White Paper, LinkLabs, 2016, [Online]. Available: https://tinyurl.com/SymphonyVsLoRaWAN.

[24] “Certification Test Report For Bluetooth and LoRa Cellphone Connection Assistant,” Apr. 2017. [Online]. Available: https://tinyurl.com/FCC-BTR-REP

[25] “FCC Code of Federal Regulations Title 47 Part 15 Radio Frequency Devices,” Jun. 2020, [Online]. Available: https://tinyurl.com/FCCTitle47P15.

[26] B. Mekiker, M. P. Wittie, J. Jones, and M. Monaghan, “Beartooth relay protocol: Supporting real-time application streams over LoRa,” Jul. 2020, [Online]. Available: https://arxiv.org/abs/2008.00021.

[27] G. Boquet, P. Tuset-Peiro, F. Adelantado, T. Watteyne, and X. Vilajosana, “LR-FHSS: Overview and performance analysis,” Dec. 2020.

[28] “Startegy for the Long Haul: The US Marine Corps Fleet Marine Forces for the 21st Century,” Report, CSBA Online, 2008. [Online]. Available: https://tinyurl.com/CSBA-Report

[29] “STREAMCASTER RADIOS ,” Spec. Sheet, Silvus Technologies, 2021, https://tinyurl.com/silvus-scaster-specs.

[30] “TSM Shadow Product Datasheet,” Spec. Sheet, TrellisWare Technologies, 2021. [Online]. Available: https://tinyurl.com/trellis-tsm-shadow

[31] “MPU5 Spec. Sheet,” Spec. Sheet, Persistent Systems, 2021. [Online]. Available: https://tinyurl.com/mpu5-specs

[32] K. Usbeck, M. Gillen, J. Loyall, A. Gronosky, J. Sterling, R. Kohler, K. Hanlon, A. Scally, R. Newkirk, and D. Canestrare, “Improving situation awareness with the Android Team Awareness Kit (ATAK),” in *(C3I) Technologies for Homeland Security, Defense, and Law Enforcement XIV*. SPIE, 2015.

[33] “Lighter, faster, smaller equipment needed for soldiers to win, says Gen. Townsend,” News Article, US Army, 2018. [Online]. Available: https://tinyurl.com/army-Gen-Townsend

[34] “Team Awareness Kit: Tactical Situational Awareness Solution,” Fact Sheet, DHS, 2020. [Online]. Available: https://tinyurl.com/tak-fact-sheet

[35] S. Al-Sarawi, M. Anbar *et al.*, “Internet of Things (IoT) communication protocols: Review,” in *IEEE Information Technology (ICIT)*, May 2017.

[36] “Sigfox Technology Overview,” Jul. 2019. [Online]. Available: https://tinyurl.com/what-sigfox

[37] A. Khalifeh, H. Salah, S. Alouneh, A. Al-Assaf, and K. Darabkh, “Performance evaluation of DigiMesh and ZigBee wireless mesh networks,” in *Wireless Communications, Signal Processing and Networking*, 2018.

[38] "Digi XBee Ecosystem," Online Resource, DIGI. [Online]. Available: https://tinyurl.com/digi-xbee

[39] "Wireless Mesh Networking: ZigBee vs. DigiMesh," White Paper, DIGI, 2018. [Online]. Available: https://tinyurl.com/zigbee-vs-digimesh

[40] "Digi XBee SX 900 RF Module," 2022. [Online]. Available: https://tinyurl.com/xbee-specs

[41] "Team Awareness Kit (TAK) - Enhancing Homeland Security Enterprise Collaboration on the Mobile Edge," White Paper, DHS and S&T, 2019. [Online]. Available: https://tinyurl.com/tak-dhs

[42] B. Mekiker, M. P. Wittie, J. Jones, and M. Monaghan, "Beartooth relay protocol: Supporting real-time application streams with dynamically allocated data reservations over LoRa," in *Computer Communications and Networks (ICCCN)*, 2021.

[43] "Protocol Buffers - Google's data interchange format," GitHub Repo, Google. [Online]. Available: https://tinyurl.com/protobuf-github

[44] "Protocol Design Whitepaper," White Paper, ZeroTier, 2018. [Online]. Available: https://tinyurl.com/zerotier-protocol

[45] J. Betker, "TTS-scores," Apr. 2022. [Online]. Available: https://github.com/neonbjb/tts-scores

[46] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *Acoustics, Speech, and Signal Processing. Proceedings*, 2001.

[47] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021.

[48] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," 2019.

[49] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," 2021.

[50] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," 2022.

[51] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura, "Speech synthesis based on hidden markov models," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1234–1252, 2013.

[52] N. Xue, “Chinese word segmentation as character tagging,” *Computational Linguistics & Chinese Language Processing*, vol. 8, no. 1, pp. 29–48, Feb. 2003.

[53] K. Yao and G. Zweig, “Sequence-to-sequence neural net models for grapheme-to-phoneme conversion,” *CoRR*, 2015.

[54] X. Tan, T. Qin, F. Soong, and T. Liu, “A survey on neural speech synthesis,” 2021.

[55] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” in *Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[56] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” 2016.

[57] D. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37. PMLR, Jul. 2015, pp. 1530–1538.

[58] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” 2014.

[59] “SoundStream: An End-to-End Neural Audio Codec,” Oct. 2023. [Online]. Available: https://blog.research.google/2021/08/soundstream-end-to-end-neural-audio.html

[60] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High fidelity neural audio compression,” 2022.

[61] H. Alt and M. Godau, “Computing the fréchet distance between two polygonal curves,” *International Journal of Computational Geometry & Applications*, vol. 05, no. 01n02, pp. 75–91, 1995.

[62] K. Ito and L. Johnson, “The LJ Speech dataset,” 2017. [Online]. Available: https://keithito.com/LJ-Speech-Dataset/

[63] K. Åhlander, “Einstein summation for multidimensional arrays,” *Computers & Mathematics with Applications*, vol. 44, no. 8, pp. 1007–1017, 2002.

[64] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML ’06, 2006, p. 369–376.

[65] P. Bhattacharjee, R. S. Raju, A. Ahmad, and M. S. Rahman, "End-to-end bangla speech synthesis," in *Science & Contemporary Technologies (ICSCT)*, 2021.

[66] M. Cernak and M. Rusko, "An evaluation of a synthetic speech using the pesq measure," in *Forum Acusticum*, 2005.

[67] "Compare 20 series specs," Oct. 2023. [Online]. Available: https://www.nvidia.com/en-us/geforce/graphics-cards/compare/?section=compare-20

[68] "Finally, pcie* and intel® qlc 3d nand in one ssd." Oct. 2023. [Online]. Available: https://www.intel.com/content/www/us/en/products/docs/memory-storage/solid-state-drives/consumer-ssds/660p-series-brief.html

[69] T. Hayashi, R. Yamamoto, T. Yoshimura, P. Wu, J. Shi, T. Saeki, Y. Ju, Y. Yasuda, S. Takamichi, and S. Watanabe, "Espnet2-tts: Extending the edge of tts research," 2021.

[70] R. Yamamoto, E. Song, and J. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," 2020.

[71] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," 2020.

[72] A. Mustafa, N. Pia, and G. Fuchs, "Stylemelgan: An efficient high-fidelity adversarial vocoder with temporal adaptive normalization," 2021.

[73] G. Yang, S. Yang, K. Liu, P. Fang, W. Chen, and L. Xie, "Multi-band melgan: Faster waveform generation for high-quality text-to-speech," 2020.

[74] T. Hayashi, "Parallel WaveGAN implementation with Pytorch," Aug. 2023. [Online]. Available: https://github.com/kan-bayashi/ParallelWaveGAN

[75] H. Scheffé, *The Analysis of Variance.* Wiley, 1959.

[76] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[77] M. B. Brown and A. B. Forsythe, "Robust tests for the equality of variances," *Journal of the American Statistical Association*, vol. 69, no. 346, pp. 364–367, 1974.

[78] "Use of ranks in one-criterion variance analysis," *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, 1952.

[79] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[80] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, “Common voice: A massively-multilingual speech corpus,” in *Language Resources and Evaluation*, 2020.

[81] H. Zen, R. Clark, R. J. Weiss, V. Dang, Y. Jia, Y. Wu, Y. Zhang, and Z. Chen, “Libritts: A corpus derived from librispeech for text-to-speech,” in *Interspeech*, 2019. [Online]. Available: https://arxiv.org/abs/1904.02882