IMPROVING RELIABILITY WITH BIT VOTING IN SYNCHRONOUS TRANSMISSION-BASED IOT NETWORKS

by

Burhanuddin Murtaza Rangwala

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
November 2024

© Copyright by Burhanuddin Murtaza Rangwala, 2024

Table of Contents

List of Tables

List of Figures

Abstract

Emerging Internet of Things (IoT) applications like industrial IoT require ultra-high reliability, strict latency, and low energy consumption. Balancing these requirements is a considerable challenge for traditional low-power wireless protocols. Recently, there has been considerable interest in addressing these challenges by utilizing flooding-based protocols based on the notion of Synchronous Transmissions (STX), where packets can take advantage of certain physical layer properties such as *non-destructive interference* and the *capture effect*. By fully embracing redundancy, such protocols offer a competitive edge over current flooding protocols – particularly in terms of dependability. However, STX-based protocols can suffer from the impact of the beating effect, a physical layer phenomenon caused by signal frequency offsets between simultaneously transmitting devices. This beating effect can corrupt packets and reduce reliability. The resulting errors can appear in isolation or in bursts, depending on the duration of the beating. Most physical layer (PHY) protocols employ error correction mechanisms, e.g., Forward Error Correction (FEC), or Direct Sequence Spread Spectrum(DSSS), but they are generally limited on how effective they are. Thus, we introduce `STX-Vote`, an error correction scheme that can handle both isolated and burst errors occurring due to the beating effect. The important observation is that, since STX protocols are by definition redundant, it makes sense to exploit the redundancy for the purpose of error correction, while *avoiding additional on-air overhead*. We implement `STX-Vote` on `nRF52840-DK` development kits as an extension of `OSF`, an open-source STX framework. Our evaluation was performed over networks that exhibited various beating effect patterns. The results confirm that `STX-Vote` improves the reliability of both Bluetooth Low Energy (BLE) version 5 and IEEE 802.15.4 PHYs, producing as much as a 8% improvement for BLE5.

List of Abbreviations Used

BLE	Bluetooth Low Energy
CFO	Carrier Frequency Offset
RFO	Relative Carrier Frequency Offset.
CRC	Cyclic Redundancy Check
STX	Synchronous Transmissions
MAC	Medium Access Control
OSF	Open Synchronous Flooding
PHY	Physical Layer
ROF	Robust Flooding
TDMA	Time Division Multiple Access
FDMA	Frequency Division Multiple Access
WSN	Wireless Sensor Network
PRR	Packet Reception Ratio
PDR	Packet Delivery Ratio
CSMA	Carrier Sensing Multiple Access

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Israat Haque, whose diligent guidance and unwavering support have been instrumental through my master's study. Her insights and encouragement allowed me to grow as a researcher, helping me navigate and overcome numerous academic challenges. I feel incredibly privileged to have had her mentorship at Dalhousie University.

I am equally grateful to Dr. Michael Baddeley, Principal Researcher at Technology Innovation Institute, Abu-Dhabi, United Arab Emirates. His invaluable expertise, thoughtful feedback were crucial to the work in this thesis. I deeply appreciate his mentorship and the time he devoted in guiding me. I would also like to extend my sincere thanks to Dr. Ioanis Nikolaidis for his valuable feedback and guidance in refining this work. I also extend my sincere thanks to PhD candidates Monzurul Ifath and Conrado Boeira for their mentorship, and other esteemed members of PiNET lab for their unwavering support in my academic journey at Dalhousie University.

My heartfelt appreciation goes to my beloved family - my mother, father, brother, grandparents, and my maternal uncle for their indispensable presence in my journey. The immeasurable support and boundless love they have bestowed upon me has helped me sail through the ups and down during the journey. Their belief in my aspirations has provided immense strength, and knowing that my work has made them proud is one of life's most fulfilling rewards.

Chapter 1

Introduction

A Wireless Sensor Network (WSN) is a collection of sensor devices which monitor and record the physical conditions of an environment and forward its recorded data to a central location using multi-hop wireless transmissions as shown in Figure 1.1. These sensor nodes are made up of several parts: a radio transceiver for communication, a micro-controller for processing, and a battery or energy harvesting systems to power the node. As the name suggests these sensors communicate using a wireless medium, and may have an ad-hoc nature where spontaneous network formation takes place to support the transfer of the data. These networks can be deployed in healthcare settings (patient monitoring, patient localization, etc.), in industrial settings (structure health monitoring, alarm systems, etc.) and in general in places where wired systems are either not possible or costly to deploy [4–18].

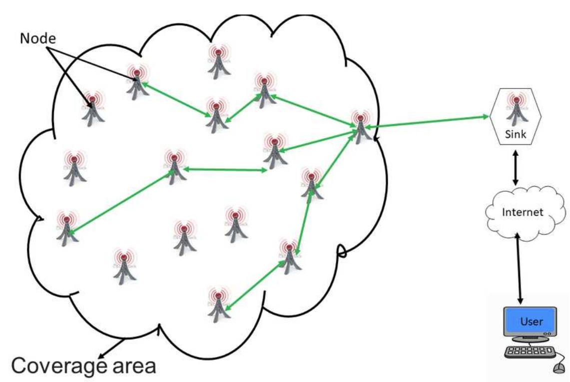


Figure 1.1: An example architecture of a Wireless Sensor Network (WSN). [1]

The sensor nodes can be organized in various topologies. The nodes or sensors in these devices either communicate information to a central location or the central location is sending information to the sensors (firmware updates, synchronization markers, etc.). The wireless medium is a shared medium which is accessible by all the nodes and if all the nodes simultaneously transmit using the medium, their packets may collide in air resulting in loss of packets. To tackle this loss, depending on the protocol, the sender may re-transmit the packet until it is successfully received at the receiver. Repeated retransmissions can overwhelm the wireless medium, especially when the number of WSN nodes is large.

Traditionally to access the wireless medium, wireless networks employ *Medium Access Control (MAC)* protocols, such as Carrier Sensing Multiple Access with backoff (CSMA/backoff) or Time Division Multiple Access (TDMA) which have been designed under the assumption that packet collisions are inherently destructive and should be avoided as much as technically possible. However, after Ferrari *et al.* [19] demonstrated that packet collisions are not inherently damaging in select physical layers standards, research started on promoting alternative protocol designs that allow collisions. The protocols following the pioneering work have demonstrated that for network flooding, allowing collisions can reduce latency and power consumption, partly because there is no need to monitor channels and routing overheads can be simplified or eliminated [20]. The family of the protocols promoting constructive interference via simultaneous transmissions of the same payload are called *Synchronous Transmission (STX)* protocols. STX caters specifically to flooding, i.e., the announcement of a message from a single node to the entire network. Flooding can be considered wasteful but at the same time increases the chances that a message will be received by its intended destination(s). As such, flooding can be employed as an elementary protocol on which to build more elaborate protocols.

The main idea behind STX-protocols is that, assuming the nodes are tightly synchronized, and assuming ideal carrier frequency agreement among nodes' transmissions, signals can constructively overlap and increase the probability of a packet's successful reception at its destination. However, in reality, the carrier frequency of the radios of transmitting devices are usually operating at a random offset from the intended carrier frequency due to the imperfections in their radio oscillator [2]. When transmissions with offsets from the ideal carrier frequency overlap in the air, they give rise to an effect called *beating effect*, named after the waveform that results due to alternating constructive (peaks) and destructive (valleys) interference pattern as seen in Figure 1.2. The destructive interference (beating valley) in the beating waveform weakens the received signal and can manifest as reception errors, negatively affecting network *reliability*.

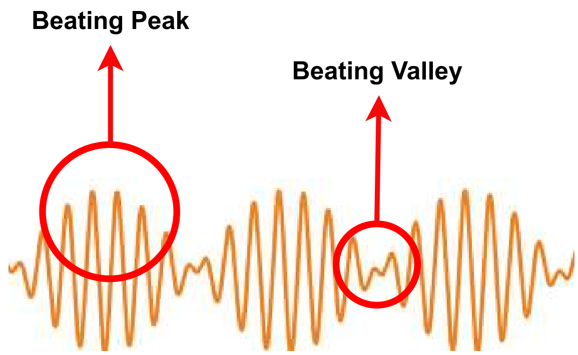


Figure 1.2: A sample beating waveform resulting from two transmitters [[2]].

STX-based schemes generally use re-transmissions [19, 21, 22], channel hopping [22–24], network coding [23–26] or a combination of the above methods to increase reliability. These methods generally exploit the *constructive interference* or *capture effect*. Constructive interference is not possible on its own [2] as what realistically is expected is to observe alternating patterns of constructive and destructive interference, as well as the impact of the capture effect. The problem is exacerbated when the number of simultaneous transmitters increases.

At the packet level there are methods such as Forward Error-Correction (FEC) or interleaving, but their impact on reliability in STX protocols is yet to be explored [20]. It was pointed out in [2] that errors occurring due to beating can be discreet (isolated, short bursts of errors) or bursty (long bursts of errors) depending on the frequency offsets among the simultaneous transmitters. Uncertain carrier frequency offset among the transmitting devices can, in turn, cause uncertain beating error bursts. FEC or Direct Sequence Spread Spectrum (DSSS) cannot handle burst errors beyond a number of contiguous corrupted bits [27] and [28]. Consequently, depending on beating conditions, coded BLE 5 and IEEE 802.15.4 PHYs can suffer an increased number of errors.

Unfortunately, besides [2], little research has been done to investigate the effect that beating has on different protocols at the packet level. The work in [2] shows that protocols would benefit from an error-correction mechanism capable of handling both bursty and discrete errors occurring due to beating, however, no such solution has been proposed yet.

This thesis fills the above-mentioned gaps and proposes a novel packet-level error correction scheme called `STX-Vote` to increase the reliability of STX-based protocols. In the event of an error, `STX-Vote` attempts to create a correct packet out of previously received error packets. Unlike FEC and DSSS, `STX-Vote` is not a-priori designed to handle a particular error length or type. Instead it relies on the fact that beating is periodic and that successive transmissions of the same packet will be received with different (locations of) bits in error. `STX-Vote` appears also able to improve the packet delivery ratio regardless of whether it is applied over a PHY that performs coding and error correction of its own (e.g., via FEC) or if it lacks such mechanism, i.e., uncoded transmissions.

1.1 Motivation

It has been pointed out [2] that beating effect has a detrimental effect on reliability of STX-based communication protocols. The beating effect can cause runs of consecutive errors of different lengths depending on the frequency offset between the simultaneous transmitters. Low powered physical layer standards such as IEEE 802.15.4 and BLE 5 are equipped with packet level error correction mechanisms such as Forward Error Correction(FEC) and Direct Sequence Spread Spectrum (DSSS). FEC relies on adding redundant parity bits in the transmission at the transmitter side and using this redundancy the receiver is able to detect and correct bit errors while DSSS provides robustness by spreading signal transmissions over frequencies.

FEC and DSSS schemes can correct packet errors caused up to a certain extent, for example, if the length of the error is less than the DSSS symbol length then DSSS can successfully correct the error [28]. To handle bursty errors caused by beating using these error correction mechanisms, one might suggest increasing the redundancy in FEC or the DSSS symbol length. However, the number of bits affected by beating cannot be quantified a-priori as the they depend on random frequency offsets. Also, the error handling steps can increase the decoding time at the receiver and the coding overhead at the sender [27]. Finally, the redundancy added by convolution codes in the coded BLE 5 PHYs might get corrupted in the packets themselves, leading to unsuccessful error correction attempts.

Using the capture effect is another way to mitigate beating, as suggested in [2]. To successfully decode it, the signal power of the desirable transmission must be higher than all other concurrent signals on the medium. However, [29] demonstrated that for the capture effect to work, other transmitters must be thirty times farther than the most dominant transmitter. Additionally, for a packet reception to be possible at the receiver, the power of the dominant transmitter must be at least 30dB higher than the sum of all the other simultaneous transmissions. These factors limit the scalability of relying on the capture effect, and negate the advantages of using STX-based protocols in the first place.

Overall, currently, the STX-based protocols are complex and cannot scale to mitigate the beating effect. On the other hand, FEC and DSSS have demonstrated limited performance under beating where there is a possibility of corruption of contiguous bits. The proposed `STX-Vote` fills the above gaps by introducing an error correction mechanism that does not add overhead while successfully mitigating the various, and unpredictable, burst errors caused due to the beating effect. Potentially, `STX-Vote` can also complement other existing solutions towards improving reliability.

1.2 Research Objective

Beating occurs due to relative *Carrier Frequency offset (CFO)* among devices that are transmitting simultaneously. Depending on the CFO the beating can be wide or narrow (Figure 1.3). The randomness of errors created due to the beating effect makes it challenging for physical layer standards to maintain reliability even though some of them are equipped with builtin error correction mechanisms such as FEC or DSSS. According to Baddeley *et. al.* [2], BLE 1M and BLE 2M do not include any error correction mechanism and suffer under narrow beating as they rely on their fast transmitting rates, which is what is currently exploited in STX protocols like Robust Flooding [22] and Glossy [19] by performing repetitive transmissions. On the other hand, BLE 125K, BLE 500K and IEEE 802.15.4 can survive the narrow beating with the help of their error correction mechanisms included in the standard, but experience a drop in reliability under wide beating.

Witnessing different behavior of different physical layers under beating, there needs to be an error correction mechanism which can help recover the packet corrupted due to beating. Through our research we present an error correction mechanism which can handle the bit errors caused by both narrow and wide beating

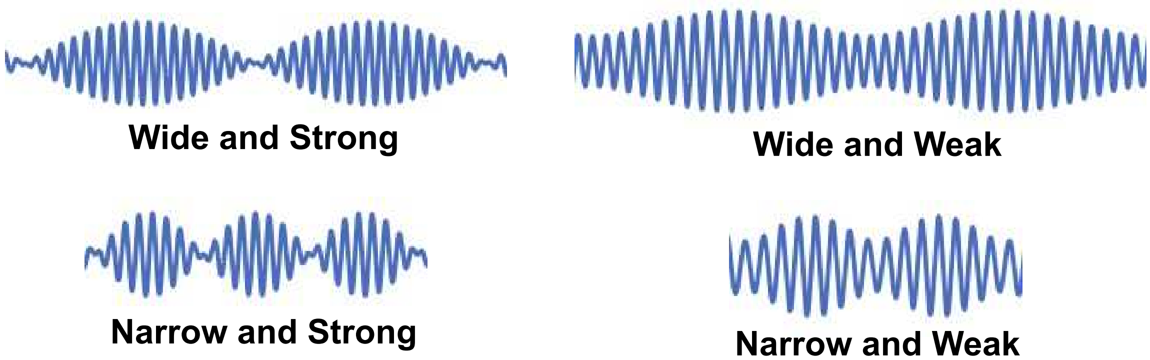


Figure 1.3: Types of beating [2].

. We show that by enabling our proposed error correction mechanism, the packet delivery rate is increased, sometimes substantially. Our proposed error correction mechanism does not require re-design of the PHY layer nor does it directly influence the signal generation. As such, it is placed above the PHY layer, as potentially part of an enhanced flooding-specific Data Link Layer.

1.3 Contribution

We propose `STX-Vote`, which we also refer to by the acronym BV (for Bit Voting), an error correction mechanism suitable for bit errors arising from the beating effect, which leverages the fact that STX protocols have to transmit the same packet multiple times. `STX-Vote` exploits the fact that multiple transmissions, each received with different bits in error, depending on beating, can be combined through a voting scheme to reconstruct an error-free packet. We first simulate the mechanism by extending an open source simulator [30] to study how it is affecting the reliability of transmissions over BLE 5 and IEEE 802.15.4 PHY. Furthermore, we implement `STX-Vote` as a driver-level extension in the state-of-the-art Open Synchronous-Flooding (OSF) framework [31], a multi-PHY synchronous-flooding framework that supports the `nRF52840-DK` System-on-Chip (SOC) platform [31]. OSF allows for easy addition of protocol-level and driver-level extensions to concurrent transmission protocols and has the ability to switch between PHYs on-the-fly. Then, we conduct an extensive hardware-based evaluation of `STX-Vote` on the D-cube public testbed hosted by the Graz University of Technology [32]. To assess the effectiveness of bit-voting used by `STX-Vote`, we do a comparison study of Packet Delivery Ratio (PDR) between transmissions with and without `STX-Vote`. Our extensive testbed experimentation campaign shows that using the `STX-Vote` under beating increases the packet delivery ratio of BLE 5 and IEEE 802.15.4 by 25-28% and 8%, respectively. In summary, the contributions of the thesis are as follows:

- We propose `STX-Vote`, a novel error correction mechanism that can increase the robustness of IoT networks employing STX-based communication protocols. An initial version of this work is published in [33].
- We simulate the beating effect experienced by the physical layers using MATLAB and evaluate `STX-Vote` by comparing the Packet Error Rate (PER) of the physical layers when `STX-Vote` is enabled to when it is disabled.
- We extensively evaluate `STX-Vote` in a local setup and on an over-the-air testbed and concluded that it could increase packet delivery ration up to 28%.
- To encourage reproducibility and further research we make the code of `STX-Vote` publicly available [34].

1.4 Thesis Outline

The remainder of this thesis is structured as follows: Chapter 2 is divided into two sections, the first of which, Section 2.1, lays the fundamental background required to better understand the work in this thesis. Next, Section 2.2 summarizes prior research work relevant to the thesis. Chapter 3 delves into a detailed methodology and design of the error correction mechanism. Here, the workflow and different steps involved in the error correction mechanism are explained in Section 3.1. Following this, Section 3.2 presents the evaluation of the proposed error correction mechanism in a simulator as well as a real world testbed. Chapter 4 discusses future research directions in Section 4.2 and provides concluding comments in Section 4.1.

Chapter 2

Background and Related Work

In this chapter, we begin by providing the essential background knowledge needed to understand the content presented in this thesis. Subsequently, we review in detail the recent literature relevant to our specific research questions.

2.1 Background

2.1.1 Mesh Networks

A mesh network is defined as a local area network topology in which nodes are wirelessly connected, directly, dynamically, and non-hierarchically to as many other peer nodes as possible. The extent to which nodes can connect is only limited by the reach of the wireless transmissions and, hence, the power of those transmissions. Mesh networks allow multi-hop data transfers between devices. As illustrated in Figure 2.1, mesh topologies allow multiple routes for the data to reach from an origin to a destination, which typically increases the resilience of the network in the event of occasional node failures. The mesh nodes do not need the support of any fixed networking infrastructure, e.g., a wired one, in order to reach each other. Usually, in a mesh network (as indeed in any sensor network), most of the time, the tasks performed are to collect data from any node to a collector/sink node and to announce data introduced at the collector/sink node to the entire network (a "broadcast"). Depending on the context, the collector/sink may be connected to the Internet and called a "gateway" node. More than one gateway may exist depending on design decisions taken.

Thus, a mesh network is an attractive solution for situations where a network needs to be deployed without the use of wired infrastructure and where many nodes are required to be placed due to the scale of the deployment. Nodes that are far away from the sink can send their data using the intermediate mesh nodes. This type of

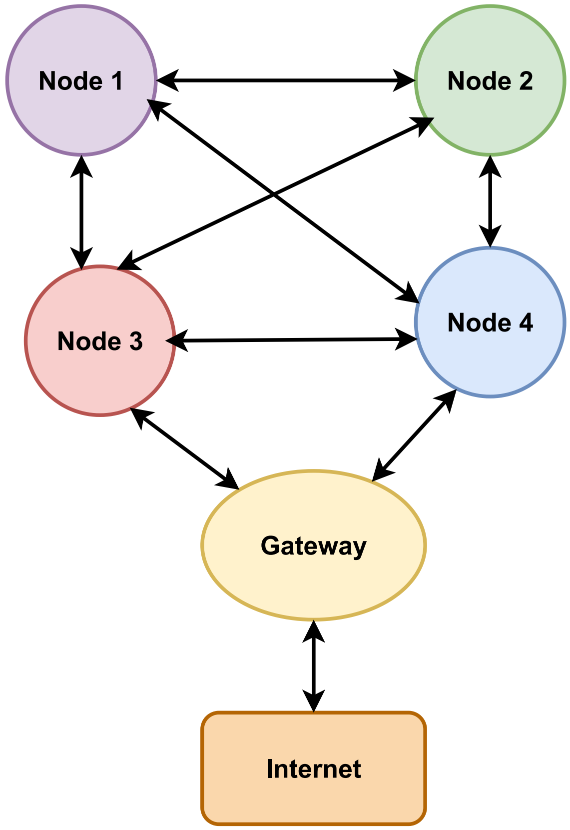


Figure 2.1: An example of a wireless mesh sensor network.

topology can facilitate uninterrupted data flow through the network in the event of a node failure and node mobility, as multiple routes to the destination exist.

Mesh network nodes use flooding or routing to communicate between origin and destination. Routing requires information about the hop-by-hop path to forward packets from origin to destination, whereas flooding works by forwarding the same data to every node in the network. For routing-based communication, routing tables are created among the nodes, based on which each node makes forwarding decisions as to how to forward data heading to a particular destination by choosing which of their neighbours to forward it to next. Routing is implemented by executing routing protocols responsible, e.g., for updating the routing tables at the nodes. Doing so can involve significant overheard, as the tables must be updated to accommodate node failures and mobility.

The overhead for maintaining routing tables involves re-computations and distribution of the updated routing tables across the network. This involves extra communication among nodes and increased energy consumption, which might be detrimental to networks using low-powered wireless devices [19, 35–37]. As an alternative to routing, a flooding mode of communication can be used, which obviates routing. Flooding operates by having nodes transmit a packet they have received with no specific destination in mind, i.e., as a form of broadcast. Hence, all their neighbors receive the packets and broadcast them. Some minor additional logic ensures that the broadcasting does not go on forever. The general idea with flooding is that, although it might appear wasteful, it increases the chances that a packet will be received at its destination. At the same time, its overhead, depending on the situation, might be competitive to the extra traffic a routing-based solution would have required. Also, flooding is simpler to implement than routing.

2.1.2 Synchronous Transmissions

In the interest of quickly delivering the data to the destination, some form of rapid flooding is required, i.e., nodes relay the traffic as soon as they receive it. However, rapid flooding can cause packet loss due to interference. The nodes perform retransmissions, which further congest the wireless medium and lead to a phenomenon called broadcast storm [38]. We can schedule broadcasts to overcome this congestion; however, determining an interference-free broadcast schedule is an NP-complete problem. Also, it needs recalculations often as topology changes due to node failures or mobility [39].

Furthermore, despite interference, a transmission can be received correctly because of a phenomenon called *capture effect*. The capture effect can be understood with an analogy: take a scenario where there are multiple people in a room, and there are multiple overlapping conversations happening; among these conversations, if you want to have your conversation, you need to speak louder so that the person listening to you can hear your voice among all the other conversations happening. This is known as the capture effect, where the receiver hears the transmission with relatively high power compared to other transmissions in the network, as shown in Figure 2.2. However, exploiting this phenomenon is not scalable as it becomes more complicated with the larger number of simultaneous transmitters [40].

To overcome those limitations and under the common belief that packet collisions

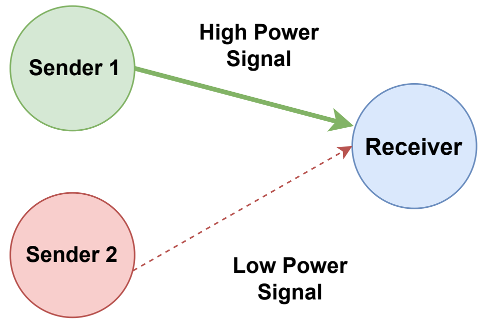


Figure 2.2: A high powered signal in the presence of a comparatively low powered signal resulting in a Capture Effect.

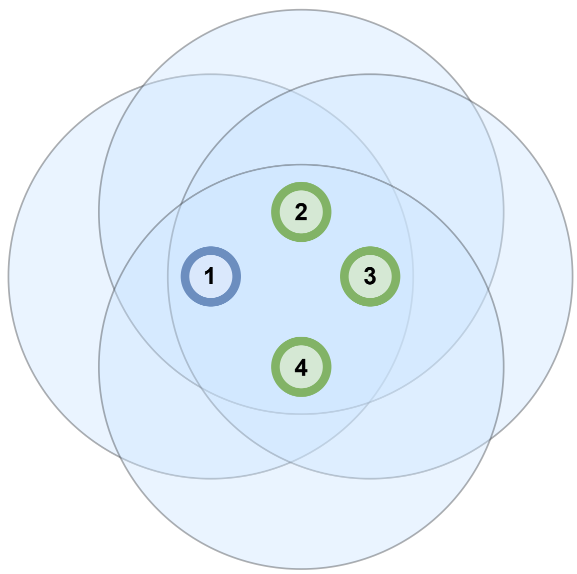


Figure 2.3: Nodes in a mesh network transmitting information synchronously.

are destructive, various medium access protocols were devised to avoid packet collisions, such as carrier sensing, handshaking, etc. However, a paradigm shift occurred when low-power wireless protocols were proposed based on the notion of synchronous transmissions. Specifically, Glossy [19] pioneered a new flooding scheme using interference to their advantage by synchronizing the transmissions of the participating nodes spatially and temporally and making them transmit the same packet on the same carrier frequency as shown in Figure 2.3 such that the transmissions interfere constructively in the air as shown in Figure 2.4. Thus, it increases the possibility of successful reception and decoding of the packet at the receiver.

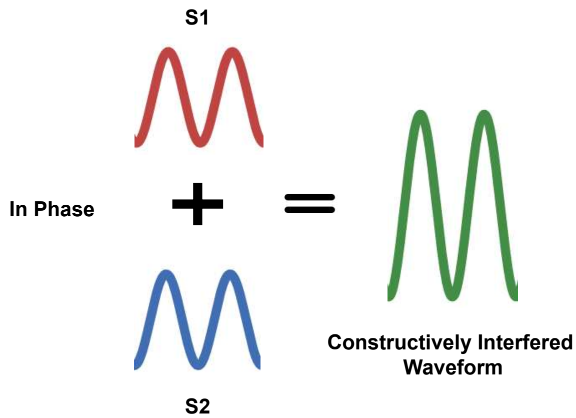


Figure 2.4: Constructive interference between two in-phase same frequency waveforms.

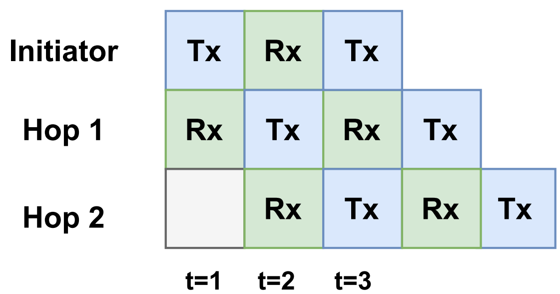


Figure 2.5: A sample Glossy flood for MAX_TX=2 per round adopted from [3].

2.1.2.1 Synchronous Flooding Protocol

This section presents Glossy [19] the de facto synchronous flooding protocol. In this protocol, a source node initiates a flood by transmitting a packet in the network that the nearby nodes hear due to the broadcast nature of the wireless medium. All the nearby nodes receive the packet at the same time. Subsequently, they start transmitting the packet at the same time, which triggers other nearby nodes to hear and relay the same packet further into the network, as shown in Figure 2.5 and Figure 2.6. Following this pattern of synchronous transmissions, it is able to flood a packet from the sender to the receiver, which a receiver receives due to the *constructive interference* (as all nodes are transmitting the same packet at the same time) and *sender diversity* (packet has multiple chances to reach the receiver due to multiple nodes transmitting the same packet.

Glossy defines *flooding rounds* as transmitting a packet multiple times controlled by a `MAX_TX` variable with the hope of successful repetition at the intended destination.

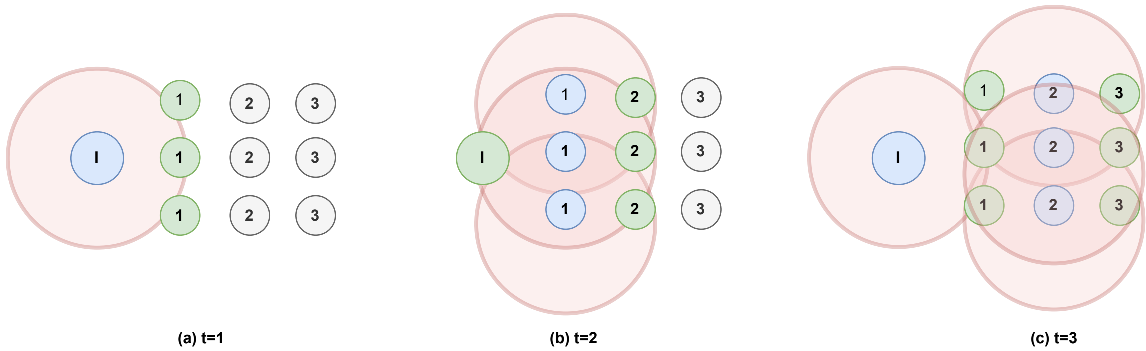


Figure 2.6: A sample STX flood generated during a round in Glossy for slots in Figure 2.5. At t=1 all first hop neighbors receive the transmission. At t=2 they simultaneously repeat the received transmission allowing further nodes to receive. Blue indicates transmission and green indicates reception as adopted from [3].

Multiple such rounds can increase the chances of a packet reaching the destination. Within each round, a specific time is designated for packet transmission or reception, which is called a *slot*. This division of a round into slots helps achieve precise temporal synchronization among the participating nodes. The nodes first receive a packet transmission in a slot, hence synchronizing itself with the sender using that reception event, and then transmit in the following slot, i.e., continuing the flood.

Multiple protocols [20] have been developed following Glossy and using *synchronous transmissions*, which exploits the constructive interference of physical layer transmissions and the sender diversity to deliver packets reliably. Also, this approach reduces latency compared to conventional protocols that use complex collision avoidance and routing schemes. One such protocol based on the Glossy is *Robust Flooding* [22], which is the basis of `STX-Vote`.

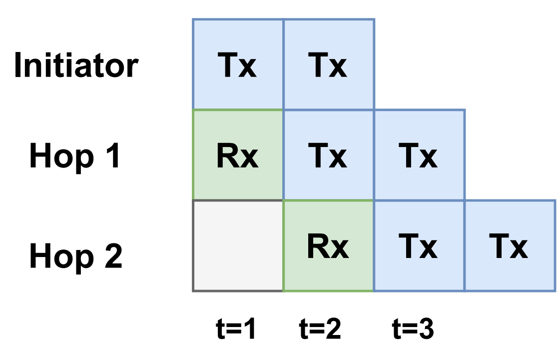


Figure 2.7: A sample RoF flood for `MAX_TX`=2 per round adopted from [3].

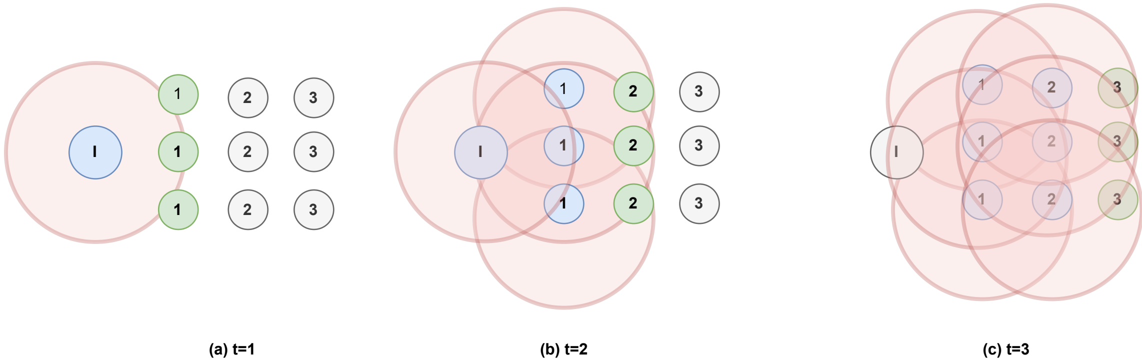


Figure 2.8: A sample STX flood generated by RoF according to the schedule from Figure 2.7. Once a node receives a packet, it immediately starts flooding packet for `MAX_TX` times, triggered by timeouts. Blue indicates transmission and green indicates reception as adopted from [3].

2.1.2.2 Robust Flooding (RoF)

Robust flooding is based on Glossy, where each packet is flooded in multiple rounds, and each round is divided into multiple slots. Nodes send the same packet in the same slots with tight temporal synchronization to reach the intended receiver with high reliability.

However, to achieve synchronous packet transmissions among nodes, each node in Glossy relies on a successful reception in every alternate slot. Suppose a packet reception is not successful in any slot due to the beating effect, external interference, multipath effects, etc. In that case, the next slot cannot be used for a transmission, resulting in a wasted slot. Wasting a slot can, in the worst case, halt flooding from propagating further. Robust flooding transmits packets that rely solely on timeouts to mitigate this, which triggers repeated transmissions. In particular, after the first correct reception of a packet, subsequent slots are triggered by timeouts and dedicated to repeating the packet that was received, as shown in Figure 2.7 and Figure 2.8. The nodes use local timeouts to schedule transmission in successive slots.

2.1.3 Beating Effect

The *beating effect* is a phenomenon that arises when transmissions from non-coherent transmitters with a slight shift from their intended carrier frequencies overlap in the air. The differences from the ideal frequency can be because of inherent inaccuracies in the crystal oscillator that provides the internal clock signals. The oscillation frequencies also vary over time but can be considered to have an almost constant value for the brief period of a single packet's transmission.

These overlapped waveforms exhibit a pattern of alternating constructive gain and destructive valleys resembling a "beat" [40]. This beating effect is said to arise due to *Relative Carrier Frequency Offset (RFO)*, i.e., the difference between the carrier frequencies used by the devices, as captured in Equation 2.1, assuming that the waveforms are synchronously transmitted. The beating effect can be modelled as a sinusoidal waveform of alternating periods of constructive *and* destructive interference (Figure 2.9) when two synchronous transmitters are involved. The shape of the beating effect becomes more complex when the number of overlapping transmissions increases.

$$\mathrm{RFO} = |\mathrm{CFO}_1 - \mathrm{CFO}_2| \tag{2.1}$$

The beating has a negative effect on the reliability of the network because the bits transmitted during the destructive valleys are likely to be received corrupted, causing packet reception errors. Along with the CFO, certain other factors dictate the type and intensity of the beating occurring in the real world. Research conducted by Baddeley *et al*. [2] demonstrates that the interaction of three key factors influences the packet error rate: the RFO between devices, the duration for which a packet is being transmitted on the wireless medium, and the difference in transmitting (and also received) power between synchronous transmission. They qualitatively classify the beating patterns encountered as belonging to one of the four classes shown in Figure 1.3.

- **Wide and Strong:** This type of beating is observed when the CFO between the devices is small, and the power difference between overlapping received signals is small. Depending on the duration of the packet in the air, destructive valleys in the waveform can last longer compared to the duration of packet transmission, possibly resulting in long busts of bit errors.

- **Wide and Weak:** This type occurs when the CFO between the devices is small, and the power difference between the simultaneously received signals is significant but less than the capture threshold. The destructive interference

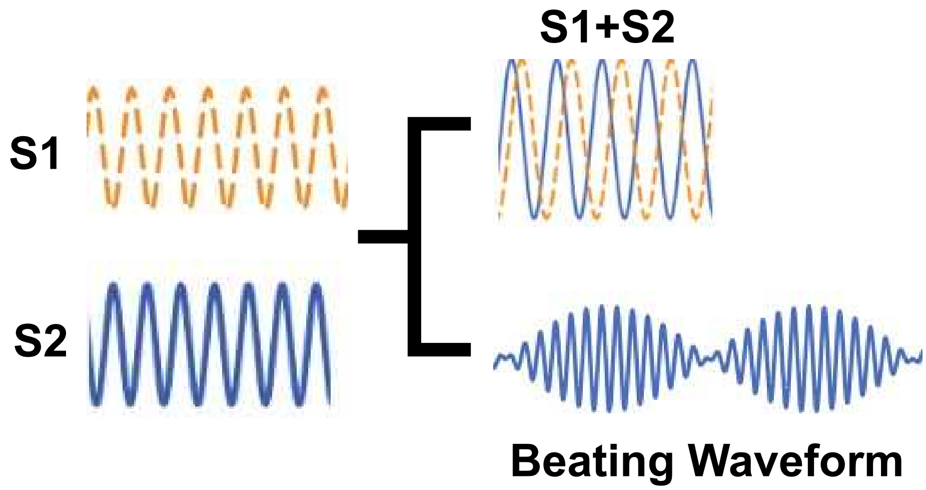


Figure 2.9: Beating effect when frequency offset transmissions overlap.

in the waveform is not as intense as in the case of wide–and–strong but can generate burst errors. Power differences larger than the capture threshold may lead to the capture effect.

- **Narrow and Strong:** In this type of beating, the CFO between devices is large, and the power difference between received signals is small. This type results in frequent but narrow destructive valleys, likely causing shorter bursts of errors compared to wide–and–strong beating.

- **Narrow and Weak:** This beating pattern is witnessed when there is a substantial CFO between transmitting devices, coupled with a significant power difference between overlapping received signals. The resulting waveform exhibits less intense destructive beating valleys compared to narrow–and–strong beating. Consequently, this pattern tends to produce isolated, sporadically distributed errors across the packet rather than continuous error bursts. The reduced intensity of destructive interference leads to a lower probability of bit errors. Legacy *Forward Error Correction (FEC)* schemes may be able to correct such errors as well.

Whether one observes any of those four patterns in the real world is completely random and depends on the exact conditions and circumstances in effect at the time of observation. The PHY layer properties of well known low–power physical layer standards such as IEEE 802.15.4, BLE 5 PHYs - BLE 125K, BLE 500K, BLE 1M and

BLE 2M respond differently to beating [2]. Ideally, we would like to have a means to correct errors no matter the particular class of beating pattern.

2.1.4 Effect of Beating on Low-Power PHY Transmissions

The most widely used physical layer standards used by low-power devices are IEEE 802.15.4, BLE 1M, BLE 2M, BLE 125K, and BLE 500K. They have varied properties: IEEE 802.15.4 is equipped with Direct Spread Spectrum Sequence (DS), and BLE 125K and BLE 500K are equipped with the *Forward Error Correction (FEC)* mechanism, whereas BLE 1M and BLE 2M have faster transmission rates but do not have any error correction mechanisms added.

BLE 1M is the standard physical layer mode in Bluetooth Low Energy (BLE), which operates at a symbol rate of 1MegaSymbol/second, and each symbol represents one bit. BLE 2M is another standard physical layer mode introduced in Bluetooth 5.0, which operates at a symbol rate of 2MegaSymbol/second, double the rate compared to BLE 1M.BLE 2M is a good candidate for high throughput applications such as firmware updates or streaming sensor data. Both use Gaussian Frequency Shift Keying (GFSK) modulation in the 2.4GHz ISM band and support a maximum payload of 256 bytes. They use Cyclic Redundancy Check (CRC) for error detection. These physical layer standards are also called uncoded BLE 5.0 PHYs and are capable of detecting errors but not correcting them [41].

BLE 125K and BLE 500K are referred to as coded physical layers, also introduced in Bluetooth 5.0, designed to increase the range of ble-based communication. These physical layers use Convolutional Codes as an error correction strategy to improve link reliability in noisy environments. They function by adding redundant bits to the original data before transmission. BLE 500K uses $S = 2$ mode, where each bit is encoded into 2 symbols. But BLE 125K uses $S = 8$ mode where each bit is encoded into 8 Symbols [41]. This added redundancy allows receivers to detect and correct errors without requesting re-transmissions.

IEEE 802.15.4 employs Direct Sequence Spread Spectrum (DSSS). This works by considering 4 bits into a single symbol and mapping it to a 32-chip Pseudo-Noise (PN) sequence, effectively spreading the signal by a factor of 8. This spreading makes it more resistant to interference and improves co-existence with other technologies.

The same PN sequence is used at the receiver to de-spread the signal and receive the original data. This signal spreading improves the signal-to-noise ratio, which leads to more reliable communication.

Certain studies [28, 42] have shown that for STX to function under beating requires packet-level coding schemes, as the errors caused by beating are similar to signal fading. These coding schemes can provide robustness against the fading. Understandably, BLE 1M and BLE 2M would suffer from errors caused by beating as they do not have any built-in error correction mechanisms. However, it is seen that BLE 125K, BLE 500K and IEEE 802.15.4 also suffer from errors generated due to beating [2], because due to the unpredictability of the burst error length, the errors might propagate over multiple symbols, making it difficult for the coding schemes to recover them.

A summary provided in [2] of the effects of beating on BLE 5 and IEEE 802.15.4 PHY is presented below.

- BLE 1M and BLE 2M can handle *wide beating* because fast transmission rates reduce the packet air time, and multiple re-transmissions increase the probability of one of the transmissions occurring during a constructive beating peak. However, they suffer during narrow beating as the constructive beating peaks are narrow, and they do not have any error correction mechanism that can help mitigate narrow beating.

- BLE 125K and BLE 500K, IEEE 802.15.4 can handle *narrow beating*, as it causes shorter bursts of errors, which their built-in error correction strategies, FEC and DSSS, can effectively handle. However, these physical layers suffer in the case of a wide beating as they cause larger bursts of errors compared to narrow beating and can span multiple symbols that hamper the functioning of the built-in error correction mechanisms [27].

In real-world scenarios, multiple types of beating can happen in different parts of the mesh network due to the unpredictability of the CFO of the devices and power-level relationships between the synchronously transmitting devices. In addition to beating, other channel impairments can contribute to packet errors while travelling through the network, such as noise, signal attenuation, multipath fading, etc., all of which can be time-varying. These impairments can also be tackled with Forward Error Correction (FEC) mechanisms [42, 43], but as Baddeley *et. al.* [2] remark, the coding schemes used for error correction have limited robustness against beating.

2.1.5 Open Synchronous Flooding (OSF)

Open Synchronous Flooding (OSF) [31] is an open-source framework which utilizes a novel radio driver and middle-ware architecture to enable synchronous flooding solutions for low-power wireless mesh networks in a Multi-PHY fashion using Nordic semiconductor's *nRF52840* chip. OSF supports an Application Programming Interface (API) that facilitates the development process of protocols without worrying about radio access and interrupt handling.

Unlike the existing synchronous flooding solutions, which are anchored to a single PHY, OSF exploits an interrupt-driven mechanism provided by nRF52840 to enable synchronous flooding operations. Figure 2.10 provides a block-level view of OSF components and their relation to the application and underlying hardware. The interrupt-driven nature of the framework reduces the attention the synchronous transmissions require from the microcontroller CPU; hence, it does not significantly impact other processes on the device. OSF also allows dynamic switching of PHYs at runtime. Technically, one can select the best PHY that suits the network conditions to improve reliability and efficiency. OSF also allows for the configuration of flooding round parameters like a maximum number of transmissions, the flooding protocol and the transmission power.

2.1.5.1 OSF Driver

OSF utilizes Robust Flooding [22] as its underlying synchronous flooding protocol, where the communication is divided into rounds and slots. Each slot is explicitly allocated for packet transmission or reception, and timers govern the transitions between the slots. Although independent, the nodes operate in synchronization with each other. For instance, as soon as the hardware issues an interrupt signalling the end of a packet reception, a timer is started, and if the reception is successful, then the packet is placed in a specific transmission slot, indicating that the packet will be transmitted at that particular slot at the expiration of the timer. Subsequent slots are similarly defined upon the expiration of the timer. In this way, utilizing interrupts and timers, OSF can orchestrate nodes to execute operations simultaneously as they are defined upon expiration of the same duration timeout from a well-defined point in time (the end of a packet arrival).

OSF achieves temporal synchronization as suggested in Robust Flooding [22] among nodes by also using linear regression across the instants of successive slots to compute the drift of a node's clock with respect to the node which initiated the flood and this way corrects the local clock of a node. Initially, at the start of a flooding round, it has to receive a packet to synchronize itself with the node that initiated the flood, and then subsequently, it can maintain synchronization by computing the clock drift. However, if the node does not receive a packet for a fixed number of rounds, then the node loses synchronization. Furthermore, it can only synchronize again when it receives a packet in a subsequent flooding round.

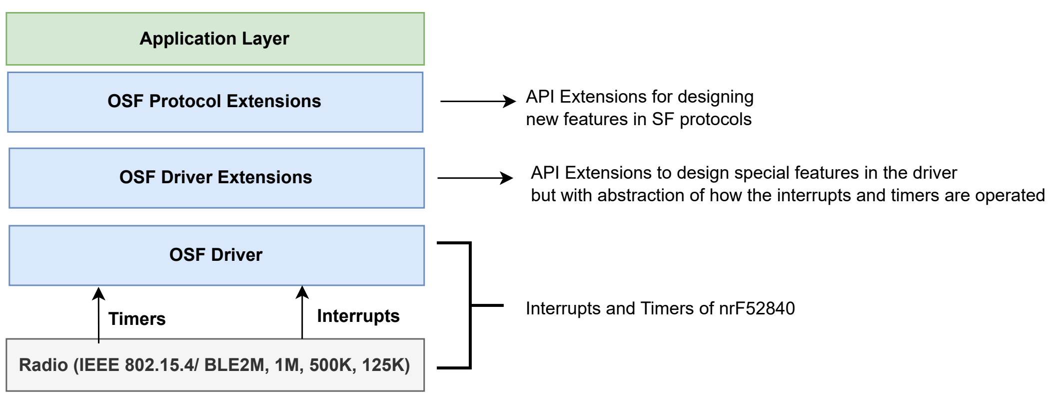


Figure 2.10: The OSF software stack.

2.1.5.2 OSF Driver and Protocol Extensions

OSF provides protocol and driver-level extensions (API), which are used to design enhancements without worrying about the tight synchronization requirements the OSF driver handles. Protocol extensions are used to design functional enhancements, such as specific enhancements for data collection scenarios, added to existing synchronous flooding protocols. OSF also supports redesigning the complete protocol using the underlying driver functionality. Driver-level extensions are used for designing enhancements without worrying about radio access and interrupt handling. We design our solution as a driver-level extension, as all the operations we introduce can be performed upon packet reception events.

The authors of OSF, Baddeley *et. al.* [31], have designed a broadcast protocol for a data dissemination application where a node acting as source broadcasts data to all the other nodes in the network using the underlying Robust Flooding Lim *et. al.* [22], which we adopt in our study as well. Thus, due to the unprecedented customization properties of OSF and the central role of transmission repetitions in the OSF stack, we can develop our proposed error correction mechanism by employing RoF. Of particular importance to our solution is the ability to collect packets of the same flooding round that were received with bit errors, ostensibly due to the beating effect, and to subsequently correct their bit errors.

2.2 Error Mitigation Strategies

We present a brief overview of four general classes of error mitigation strategies and the extent to which they are relevant to `STX-Vote`.

2.2.1 Channel Hopping

Channel hopping is a method used to avoid interference caused by other devices transmitting or operating (e.g., microwave oven present in the environment) on the same channel by switching channels for each transmission in a predefined sequence of channels. Several works use channel hopping to increase network resilience as it increases the probability of reception on one of the channels.

Lim *et al.* [22] employ consecutive synchronous transmissions up to a predefined number of slots until the packet is received at the receiver. This protocol leverages spatial redundancy by utilizing all possible links from source to destination. It also leverages frequency diversity by transmitting each slot on a different channel to avoid any possible interference that a previous transmission slot might have encountered. The authors of [44] focus on improving the resiliency of the CRYSTAL protocol using channel hopping and noise detection. The Transmission and Acknowledgment (TA) pairs of the protocol transmit on different channels to overcome the limitation of the longer awake time caused by their noise detection method. RedFixHop [21] is another STX-based protocol that quickly and reliably transmits packets from source to destination in Wireless Sensor Networks (WSN) by tightly coupled flooding to create constructive interference. To further increase reliability, it repeats the flooding a predefined number of times and switches channels for each flood.

A node in these protocols performs channel assessment to implement adaptive channel-hopping or abides by a fixed channel-hopping sequence. Even if channel hopping is implemented and constructive interference is exploited, beating can still occur because of the Relative Frequency Offset (RFO) between the transmitting devices, leading to packet corruption. Thus, even if channel hopping is utilized, it is still essential to address the beating effect.

2.2.2 Network Coding

Network coding uses advanced mathematical algorithms to aggregate multiple messages from different nodes that want to share information into a single transmission toward the receiver. STX protocols based on the network coding take advantage of the capture effect, where a strong signal can overpower others at the receiver amid multiple transmissions and the sender diversity offered by it as multiple nodes participate in the communication. This enhances the probability of a successful reception by providing multiple possibilities for the messages to reach the destination.

Splash [45], an STX-based protocol, uses XOR coding in conjunction with fast-forwarding in a hierarchical architecture, which utilizes constructive interference and XOR coding to recover packets. Ripple [23] builds upon Glossy [19] by transmitting multiple distinct packets in a round-based communication to increase the network's throughput. This increase in throughput has a trade-off with decreased chances for any specific packet to be received at the receiver. To compensate for this issue, Reed-Solomon FEC is used to recover all the transmitted packets. Mixer [25] uses Random Linear Network coding (RLNC) in many-to-many communication, where a node combines multiple messages in a single packet and propagates it further. This makes the network resilient to network changes as the information can travel through multiple routes, and RLNC facilitates quick recovery in case of loss of information from other coded packets.

Since they transmit different packets, network coding solutions rely on the capture effect for a successful reception in case of multiple transmissions happening at the same time. However, some of them use constructive interference in addition to network coding, do not consider the beating effect, and can suffer from the negative effects of beating.

2.2.3 Re-transmissions

Re-transmissions are a practical, widely used method in low-power networks to deal with packet losses. In case of a loss of packet, either the receiver requests the packet to be transmitted again or the sender will re-transmit the packet when a timer expires.

A protocol like Glossy [19] uses *Rx-Tx, Rx-Tx...* patterns of flooding and achieves high reliability. Robust Flooding [22] builds on Glossy, where a node requires a successful reception before it can start another transmission, which can be a bottleneck due to network conditions. Robust Flooding uses aggressive re-transmission of packets following *Rx-Tx-Tx-Tx...* pattern to exploit the possibility that at least one of the transmissions will reach the destination. All nodes utilizing these communication protocols can hear packets being transmitted across the network and can mimic their transmission patterns, which increases the chances of successful reception using either constructive interference or the capture effect.

In a sense, protocols like Glossy, particularly Robust Flooding, have already adopted re-transmissions as the error recovery strategy. However, because of the beating effect, if a node fails to receive the correct transmission during the communication slot, it may experience synchronization drift, which impacts the entire flooding. Therefore, it is preferable that an additional error correction mechanism be implemented in addition to the repeated transmissions to handle the impact of beating while also exploiting the abundance of re-transmissions.

2.2.4 Packet Combining

Packet combining exploits multiple incorrect packet receptions of the same packet, e.g., by performing packet merging of the correctly received parts, assuming a sufficient number of receptions has occurred. As an example, [46] perform XOR operations on the packets deemed in-error by the receiver hardware to find the bit error positions and then use a brute-force approach to determine their bit values, followed by a CRC

Reference	Physical Layer Impairment	Handles Beating Effect?	Reliability Mechanism	Packet–Level Mechanism
[46], [47], [48]	None	N/A	EC	Yes
[22], [44], [21]	CI	No	CH	No
[19], [22]	CI	No	RT	No
[23], [25], [45]	CE + CI	No	Network Coding	No
STX-Vote	**CE + CI**	**Yes**	**EC**	**Yes**

Table 2.1: Summary of related work (CI = Constructive Interference, CE = Capture Effect, CH = Channel Hopping, RT = Re-Transmissions, EC = Error Correction).

check to confirm they recovered the correct packet. A Received Signal Strength Indicator (RSSI)-based packet combining approach in [47] considers the symbols with the highest RSSI from the incorrectly received packets to assemble a new packet followed by error checks to confirm correct reconstruction. Finally, [48] gradually builds a database of recently received packets, recording the error positions and uses the database to produce plausible correct packet candidates.

Packet combining using multiple incorrect receptions of the same packet to correct errors is effective in mitigating errors in wireless sensor networks [46]. Packet combining belongs to the class of *repetition coding* strategies. To the best of our knowledge, this technique has not been applied to synchronous flooding protocols and could address errors resulting from beating because *it is unlikely that beating will corrupt the exact same bit locations in every reception of the same packet*. We use this observation in `STX-Vote` by reconstructing each bit value of a packet based on a per-bit voting scheme, assuming an adequate number of receptions of the same packet to make this voting meaningful. The key observation is that STX protocols introduce, by design, redundancy in transmissions. Hence, we introduce in `STX-Vote` a mechanism that treats this repetition as if it were repetition coding. Summarizing, Table 2.1 compares previous works, highlighting the gap that `STX-Vote` addresses. `STX-Vote` can mitigate the bit errors due to the beating effect, which has been shown to negatively impact STX protocols [2].

Chapter 3

Design and Evaluation

This section presents the operation of `STX-Vote`, i.e., different working parts of the error correction mechanism that work in concert to provide robustness against the beating effect. To demonstrate the effectiveness of `STX-Vote`, we present a multi-faceted methodology to evaluate its performance, which includes simulations and experiments. At first, simulations are conducted to generate errors due to beating and inspect whether `STX-Vote` can correct those errors under various Packet Error Rate (PER) conditions and different physical layers. The study then proceeds to real-world experiments using nRF52840-DK devices, measuring the impact of `STX-Vote` on Packet Delivery Ratio (PDR) in both local setups and on the D-cube public testbed. This evaluation helps assess the efficacy of `STX-Vote` in mitigating beating-induced errors across diverse beating conditions and physical layer configurations.

3.1 Error Correction through Bit-Voting

3.1.1 Overview of `STX-Vote`

`STX-Vote` is based on the notion that the phase of the beats can vary between different receptions of the same packet, meaning that a particular block of bits might be corrupted in one reception but received correctly in another. Based on this insight and leveraging the inherent repetition of robust flooding (the underlying STX communication protocol), STX-Vote performs error correction using a form of repetition coding by voting on the received bit values. Following the voting process on all the packets received with bit errors during a communication round, it takes on a per-bit basis the majority of the votes collected during the voting process to attempt to reconstruct the packet. The majority of votes at a particular bit location decide the reconstructed bit value. Specifically, STX-Vote operates at the receiver by maintaining an array of signed integers. The size of the array is determined by the

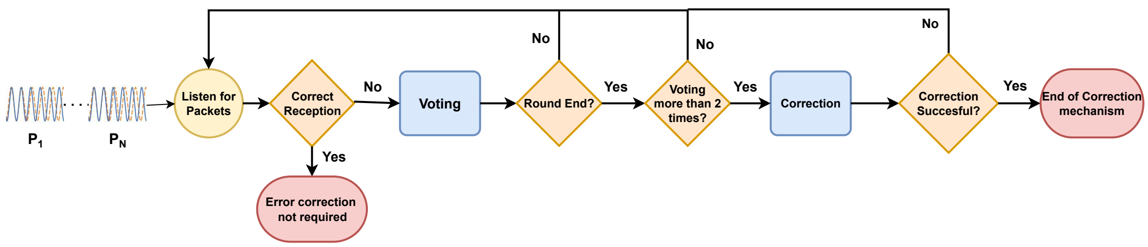


Figure 3.1: Overview of STX Vote operations at the receiver.

packet length. For example, if the packet length is 255 bytes, then the packet will contain (255 x 8 = 2040) 2040 bits, therefore, the length of the voting array would be 2040. Each entry in the array tracks the accumulated votes for each bit position across the received packets transmitted during the same round (and hence containing the same payload). The values stored at the voting array are used to reconstruct the intended packet, and the reconstructed packet is checked using a special CRC appended by the sender.

3.1.2 Sender Operations

The sender of the information will generate a payload to be transmitted, constructing a packet in the format shown in Figure 3.2, which follows the packet format of OSF. The fields are:

- **Slot**: the number (index) of the communication Slot in which the packet is being transmitted,
- **Source**: the Source address, i.e., the originating node,
- **Destination**: the Destination address,
- **Epoch**: the number of epochs elapsed until the packet is created, a fixed time interval between packet creations within which a Synchronous Flooding protocol executes a sequence of operations.
- **ID**: a unique identifier for the packet,
- **Payload**: the data to be delivered to the destination,

- **BV_CRC**: (optional) a special CRC appended specifically to enable `STX-Vote` to determine the correct reconstrution of the packet.

`BV_CRC` is added if our error correction mechanism specific to NAME is enabled. It is different from the one that may (depending on the used protocol) be present at the link layer. For the computation of `BV_CRC`, the fields Epoch and Slot are mutable across transmission slots and rounds and are assumed to be equal to zero as shown in Figure 3.3.

Slot	Source	Destination	Epoch	ID	Payload	BV_CRC

Figure 3.2: OSF packet structure.

0	Source	Destination	0	ID	Payload

Figure 3.3: Packet contents as used for `BV_CRC` computation.

3.1.3 Voting Process

The voting mechanism is the cornerstone of the error correction mechanism. During a communication round, if the transmission was successfully detected at the receiver during each of the repeated transmissions during the communication slots of the round, it may contribute to the voting process. First, it is checked whether, using the CRC, it has been received successfully. If yes, then there is no need for further steps. If the CRC check fails, the packet contributes to the voting process. A zero in a bit position of the packet received with errors results in a reduction by 1 of the votes for that bit position. Conversely, a one results in an increase by 1 of the votes for that bit position. Note that the votes array is initialized by zeroes and reset to zeroes whenever a new packet's transmission is seen starting. An example of the vote aggregation process is shown in Figure 3.4.

3.1.4 Error Correction

After a communication round ends without a successful packet reception, the receiver undertakes error correction, as illustrated in Figure 3.5. As votes aggregate, a large

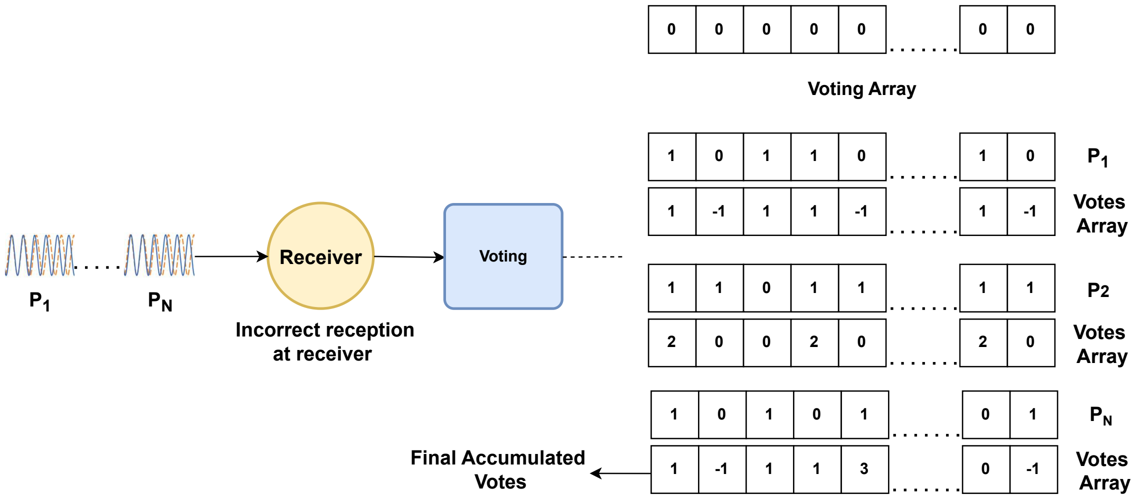


Figure 3.4: The voting operation of `STX-Vote` across N receptions of the same packet.

positive number increases the confidence that the particular position has a bit value of 1. By contrast, a larger negative value is strong evidence that the bit position has a value of 0. Also, a zero aggregate vote is assumed by convention to be interpreted as a zero at the specific bit position (but the confidence behind this being correct is lower than if the aggregate votes were negative). Reconstruction occurs if the receiver has received two or more incorrect packets during the communication round.

The voting process also results in the reconstruction of `BV_CRC`. As illustrated in Figure 3.6, once the packet is reconstructed, to verify the correctness of the reconstructed packet, the BV_CRC is calculated the way it was outlined in Section: 3.1.2. The packet is considered successfully recovered if this calculated BV_CRC matches the reconstructed BV_CRC. If this check fails, the packet recovery is considered failed, and the receiver continues to wait for the next round of receptions, potentially leading to further voting opportunities or a correct reception. If a correct reception is received, all error correction processes are halted, i.e., the receiver does not perform any further error correction, any vote aggregation is abandoned, and votes are reset to zero.

3.2 Evaluation and Results

This section presents the performance evaluation of `STX-Vote` across the widely used IEEE 802.15.4 and BLE physical layers in low-powered IoT devices. At first,

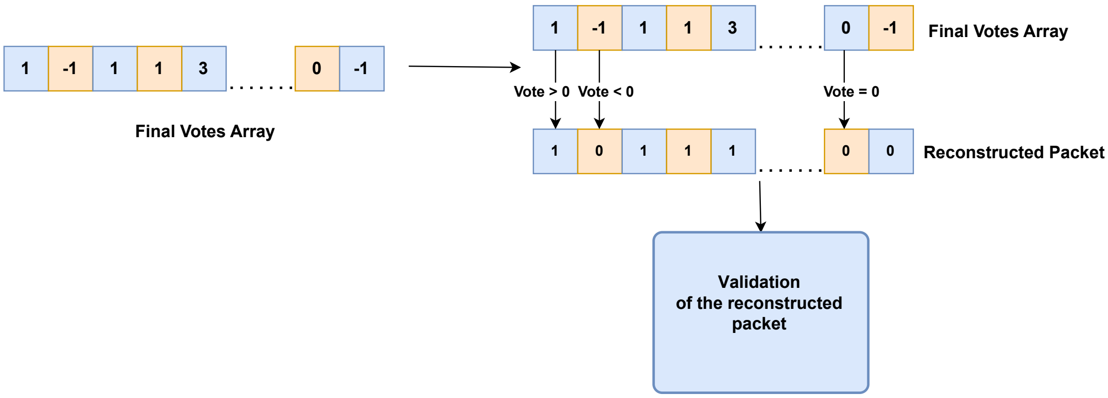


Figure 3.5: The reconstruction operation of `STX-Vote` at the receiver.

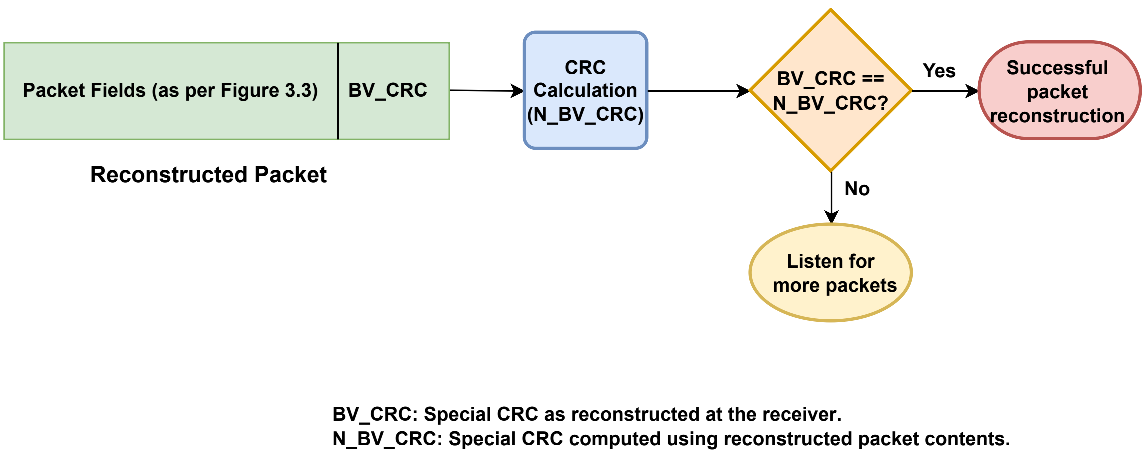
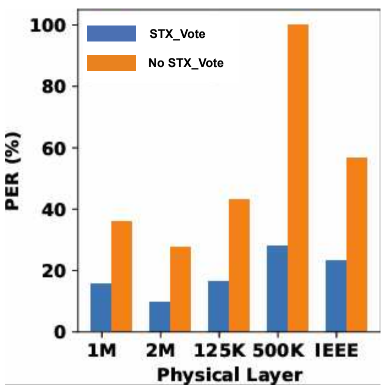


Figure 3.6: Validating a reconstructed packet.

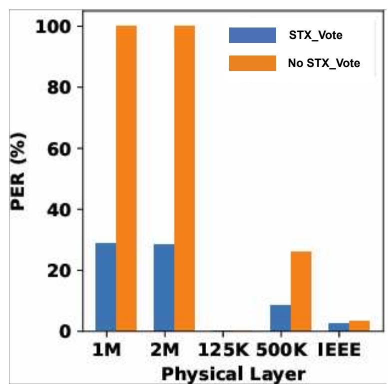
we showcase the robustness of `STX-Vote` under beating using simulation, as it is possible to simulate different CFO between the synchronous transmitters resembling the oscillator inaccuracies and the power differences, resulting in different beating types (wide and narrow, strong and weak) which will result in varying bursts of errors. We measure the Packet Error Rate (PER) for different physical layers under different types of beating for two scenarios: 1) when `STX-Vote` is enabled and 2) when `STX-Vote` is disabled. Following simulation evaluation, we demonstrate how `STX-Vote` mitigates beating errors in STX-based communications through experimental results from real-world platforms, where we measure the Packet Delivery Ratio (PDR) to illustrate the effectiveness and practical deployability of `STX-Vote` as a complement to existing error correction mechanisms.

We consider a three-node topology where two nodes act as synchronous transmitters, and one of the nodes acts as a receiver. In the case of more than two synchronous transmitters, the amplitude distortions become complex, making it challenging to explain analytically [42]; therefore, two synchronous transmitter topologies facilitate simple sinusoidal beating patterns which is easy to understand.

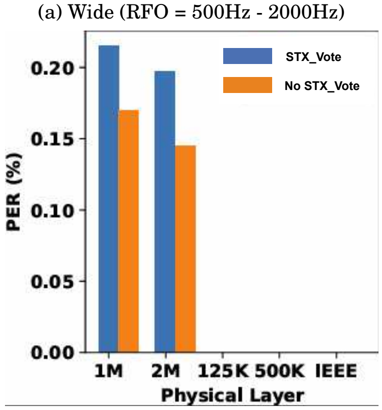
3.3 Simulation Results



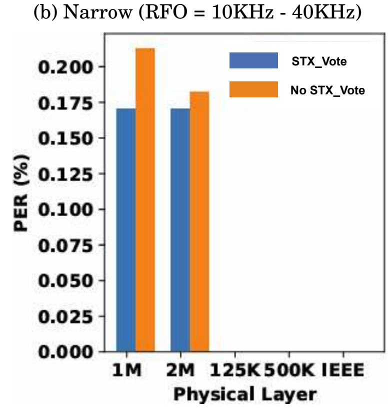
(a) Wide (RFO = 500Hz - 2000Hz)



(b) Narrow (RFO = 10KHz - 40KHz)



(c) Wide (RFO = 500Hz - 2000Hz)



(d) Narrow (RFO = 10KHz - 40KHz)

Figure 3.7: The average PER for strong and weak beatings.

3.3.1 Setup

We modify the simulator[1] [2] to assess the packet error rate of `STX-Vote`. We consider the scenario of two synchronous transmitters and analyze the impact of beating across four BLE 5 PHYs and the IEEE 802.15.4 by varying the beating frequencies.

[1] `https://github.com/ADEscobar/ct-simulator`

The simulation assumes additive white Gaussian noise and no synchronization errors while mimicking different CFOs between the transmitters arising due to oscillator inaccuracies. The simulated receiver for the Synchronous Transmissions is realized as a non-coherent Binary Frequency shift Keying (BFSK) receiver used in the PHYs used for the study [40]. We vary the CFOs to simulate wide or narrow beating and introduce power delta to influence the intensity of beating, leading to weak or strong beating. We simulate a packet stream containing a random payload with eight samples per symbol following [40] to characterize the beating effect, where two synchronous transmissions of identical packets undergo amplitude distortion if they overlap in the air. Lastly, we implement the proposed bit voting, error detection, and error correction mechanism of `STX-Vote`, validating its ability to correct packet errors caused by beating by comparing the transmitted packets with the ones reconstructed using `STX-Vote`.

We compute the average Packet Error Rate (PER) as the total number of packets correctly decoded (including the packets corrected using error correction) over the total number of packets originating at the sender. We simulate the flooding of 200 packets, a number chosen to produce statistically significant results while minimizing experimentation time. The packets have a length of 255 bytes for BLE 5 PHYs and 125 bytes for IEEE 802.15.4 as these packet lengths give us the flexibility to fit all the headers required for functioning of synchronous flooding protocols, payload and BV_CRC while being within the limits of the packet length support provided by each PHY and also provides enough time to witness beating to allow us to evaluate the effectiveness of `STX-Vote` over each physical layer. We take the average PER over 10 simulation runs for the cases when `STX-Vote` is enabled and for the instances when `STX-Vote` is disabled. The standard deviation in PER across these 10 experimental runs was found to be minimal and therefore not reported. For each simulation run, we set the maximum number of transmissions in a flooding round to 6.

3.3.2 Simulation Results

Figure 3.7 presents the average PER across different beating frequencies, giving rise to wide and narrow beating with a 25dBm Signal-to-Noise ratio. Along with beating frequency, we adjust the transmission power between the synchronous transmissions, simulating strong and weak beating. We first present the results for the strong beating, which is shown in Figure 3.7a and Figure 3.7b. Uncoded PHYs (BLE 1M and BLE 2M) result in increased PER as the beating frequency increases, inducing beating that changes from wide to narrow.

In the case of narrow beating, i.e., when the RFO between the synchronous transmitters is high, PER escalates as seen in Figure 3.7b as packet transmission spans one or more destructive valleys due to high CFO between the transmitters, leading to frequent errors. Despite their high transmission rates, uncoded PHYs cannot handle those errors without an error correction mechanism. However, when `STX-Vote` is enabled, we observe a significant drop in PER from 67.88% to 21.39% for BLE 1M and from 63.57% to 18.78% for BLE 2M, confirming its effectiveness against strong narrow beating.

For wide beating, i.e., when the RFO between synchronous transmitters is not high, the PER is not as elevated compared to narrow beating, as seen in Figure 3.7a. A potential explanation is that the fast transmission rates of uncoded PHYs increase the probability that at least one of the transmissions occurs during a constructive beating peak. Nevertheless, `STX-Vote` is still beneficial for transmissions during beating valleys under strong beating conditions. Consequently, we observe a 15% reduction in PER for BLE 1M and 18% reduction in PER for BLE 2M when `STX-Vote` is applied.

According to the specifications of coded PHYs (BLE 125K, BLE 500K and IE-EE80 2.15.4), they are equipped with error correction mechanisms such FEC for BLE 125K and BLE 500K and DSSS for IEEE 802.15.4. These mechanisms are crucial in handling beating-induced errors, as illustrated in Figure 3.7b. These mechanisms can sustain reliability under narrow beating, providing similar performance to `STX-Vote`, which indicates that the coding used by BLE 125K, BLE 500K and IEEE 802.15.4 can correct the short error bursts in the packet caused by narrow beating. However, the situation changes in *wide beatings* as shown in Figure 3.7a. This shift may be due to the distortions that extend across multiple symbols, requiring additional error correction mechanisms like `STX-Vote`. With `STX-Vote` enabled, errors are reduced by 15% for BLE 125K, by 23% for IEEE 802.15.4 and by 63% for BLE 500K. We can witness a difference between the performance of BLE 125K and BLE 500K because of the difference in the redundancy added by their codes. This difference makes

BLE 500K more susceptible to beating errors compared to BLE 125K.

We conclude that *weak beating does not have a profound impact on PER*, as demonstrated in Figures 3.7c and 3.7d. In the scenarios of weak beating, the distortion caused by the differences in transmission power between the synchronous transmitters is not strong enough to corrupt individual bits. As a result, uncoded PHYs do not require error correction mechanisms to tackle weak beating. On the other hand, coded PHYs already equipped with built-in error correction techniques can handle the distortions caused by weak beating without degrading reliability. Their existing mechanisms, such as FEC for BLE or DSSS for IEEE 802.15.4, effectively mitigate minimal interference, ensuring stable communication even when weak beating occurs.

3.4 Real-World Testbeds

The aim of this set of experiments is to evaluate `STX-Vote` in a real-world setting where the reliability of STX protocols is not only affected by the beating effect but also by other real-world factors such as multipath, external interference, etc. This also demonstrates the deployment feasibility of `STX-Vote` in STX-based communications where pilot experimentation is performed to verify the functioning of `STX-Vote` on real-world devices, followed by extensive evaluation on the D-Cube testbed which has been used in various competitions [2]. We use the same three node topology as used in simulations. For all of our real-world experiments, we use *nRF52840-DK* devices flashed with OSF firmware that facilitates synchronous flooding protocols and allows the configuration of flooding round parameters. Specifically, the maximum number of transmissions per flooding round is set to 6 as it has been demonstrated to offer dependable performance for the functioning of synchronous flooding protocols [19].

Additionally, we plot the bit error distribution across the length of the incorrectly received packets during multiple flooding rounds to see the beating taking place due to the RFO between the synchronously transmitting devices. We are able to provide this information by comparing the bits in the received packet against the bits that a correctly received packet would have delivered.

3.4.1 Lab Experiments

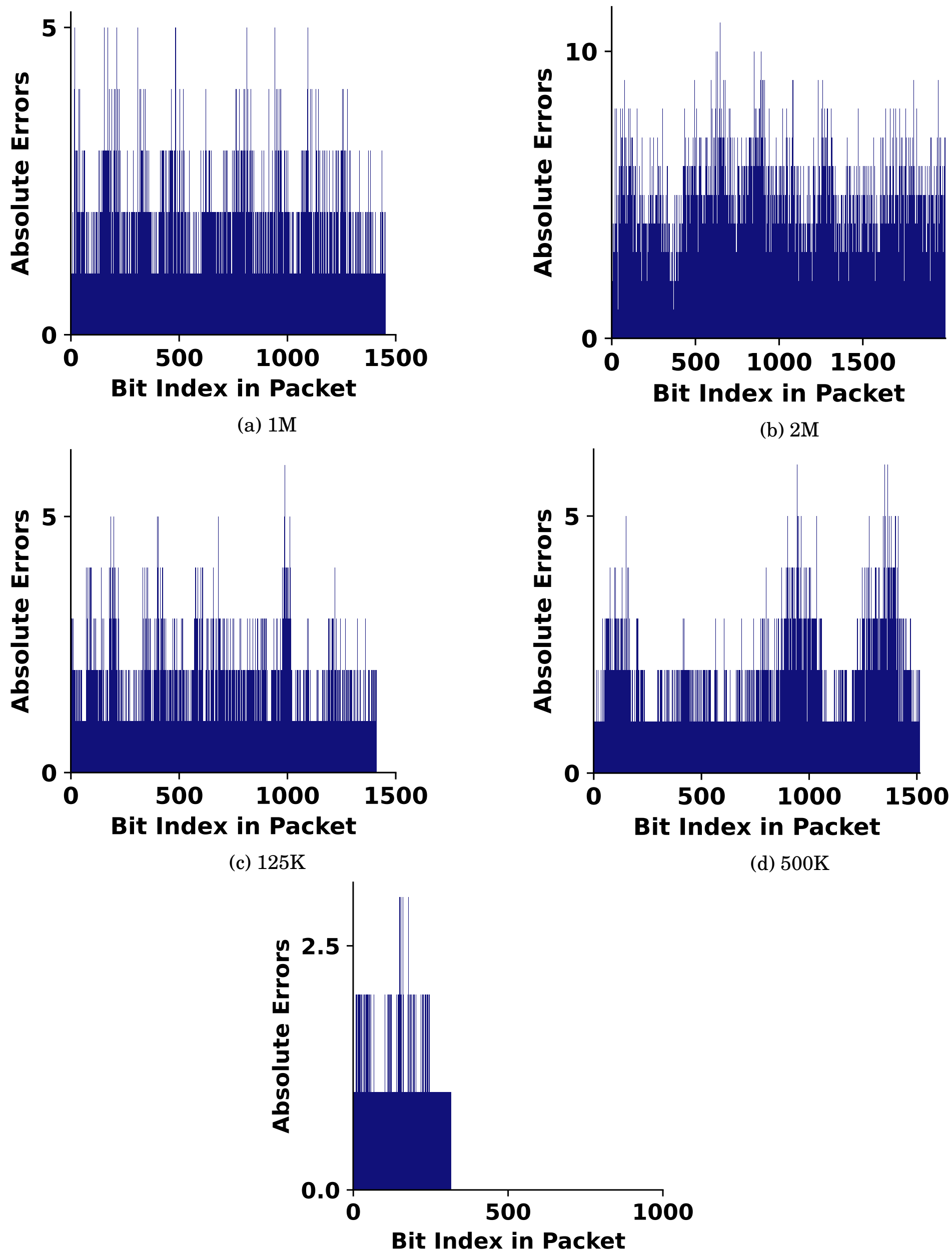
3.4.1.1 Setup

The lab setup provides pilot experimentation to witness beating occurring in a real-world setting and to identify how different PHYs handle beating and the impact of `STX-Vote` in recovering from beating-induced errors. We utilize three `nRF52840-DK` devices which support all BLE 5 PHYs and IEEE 802.15.4, positioning them at random distances on a desk to establish a single-hop communication between the nodes, assuming no synchronization errors. Each node is assigned a specific role and performs actions according to the role assigned; for example, the node assigned as a source will generate a packet of a specified length. For our experimentation, we choose 255 bytes for BLE and 125 bytes for IEEE 802.15.4, as these lengths allow sufficient time to capture the effects of beating on the packet [2]. The node assigned the role of forwarder listens for packet transmission, and as soon as it successfully receives a packet, it starts forwarding that packet, functioning as another synchronous transmitter. Finally, the third node acts as a receiver node.

We set the transmission power as -40dBm as the effects of beating at higher powers (-20dBm, -16dBm, -8dBm) were not observed, possibly due to the capture effect prevalent in STX-based communications. These parameters enable us to observe the impact of beating and identify its different types. During the experiment, the source is programmed to generate 100 packets for each PHY; subsequently, to evaluate the performance of `STX-Vote`, we calculate the Packet Delivery Ratio (PDR), defined as the total number of packets successfully received out of the total number of packets originated (100 in this case) and denotes the reliability of the 1-hop link between the sender and receiver. We take an average PDR for each physical layer over five runs since this was an exploratory run to check the manifestation of beating and the working of the error correction mechanism before deploying it to the D-Cube testbed.

3.4.1.2 Lab Results

Figure 3.9 depicts the average PDR for two scenarios: 1) with `STX-Vote` enabled and 2) with `STX-Vote` disabled. We witness the overall improvement in PDR for



(a) 1M

(b) 2M

(c) 125K

(d) 500K

(e) IEEE

Figure 3.8: The observed beating patterns for each physical layer during the lab experiments.

all BLE 5 PHYs and IEEE 802.15.4 operating under beating and using `STX-Vote` as an error correction mechanism. From the results, it is seen that uncoded PHYs, BLE 1M and BLE 2M experienced a performance gain of approximately 40% (with a standard deviation of 9.12) and 55% (with a standard deviation of 13.09), respectively, when `STX-Vote` is enabled. This behavior can be explained by analyzing the beating histograms in Figures 3.8a and 3.8b. We can see that the uncoded physical layers may experience narrow beating, and under narrow beating, uncoded PHYs suffer the most due to the absence of an error correction mechanism. Furthermore, it is seen that out of BLE 1M and BLE 2M (both uncoded PHYs), BLE 2M suffered the most from the beating effect due to its faster transmission rate. However, `STX-Vote` can successfully reconstruct the incorrectly received packets.

In particular, because of the manifestation of narrow beating, multiple smaller discrete portions of the packet get corrupted. Due to the periodic beating, this corruption spreads throughout the packet. Using this to its advantage and leveraging the inherent repetition of synchronous flooding protocol, `STX-Vote` can correct in BLE 2M the packets by voting on multiple such incorrect packets based on the correct portions of the packet. Similarly, BLE 1M benefits from `STX-Vote`, with the performance improvement being proportional to its transmission rate.

In contrast, coded PHYs BLE 125K, BLE 500K, and IEEE 802.15.4 paired with `STX-Vote` show performance increases of approximately 50% (with a standard deviation of 4.3), 20% (with a standard deviation of 5.36), and 20% (with a standard deviation of 3.37), respectively. From Figures 3.8c to 3.8e, we can infer that the coded physical layers may be experiencing wide beating as there are no distinct peaks visible in the histograms. Wide beating results in a larger burst of errors than narrow beating. Out of all the coded PHYs BLE 125K suffers the most due to its lower transmission rate, which results in the packet being in the air for a longer time and, in conjunction with wide beating, results in a larger burst of errors, which the built-in FEC cannot recover from. However, by incorporating `STX-Vote`, BLE 125K can successfully correct the packets its FEC mechanism misses. Similarly, `STX-Vote` enhances the performance of FEC employed by BLE 500K and DSSS employed by IEEE 802.15.4. `STX-Vote` proves beneficial for all the PHYs under beating as it offers an increase in PDR, especially for uncoded PHYs.

3.4.2 D-cube Experiments

3.4.2.1 Setup

Experiments were performed using the D-Cube public testbed [32], which is equipped with `nRF52840-DK` devices supporting all BLE 5 PHYs and IEEE 802.15.4. D-Cube has 50 `nRF52840-DK` devices deployed at a building in TUGratz, Austria, spread

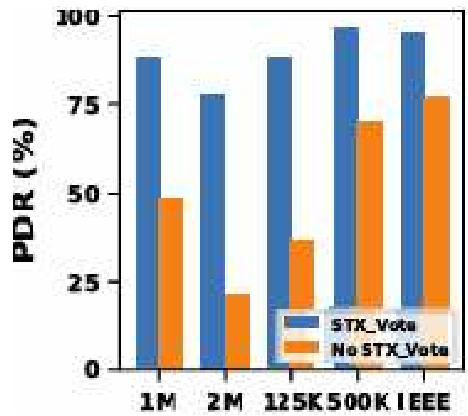


Figure 3.9: The average PDR for lab experiments under beating.

across two floors of the building in multiple rooms such as offices and seminar rooms. The experiments were conducted over random combinations of the three node topology, all a single-hop away from each other.

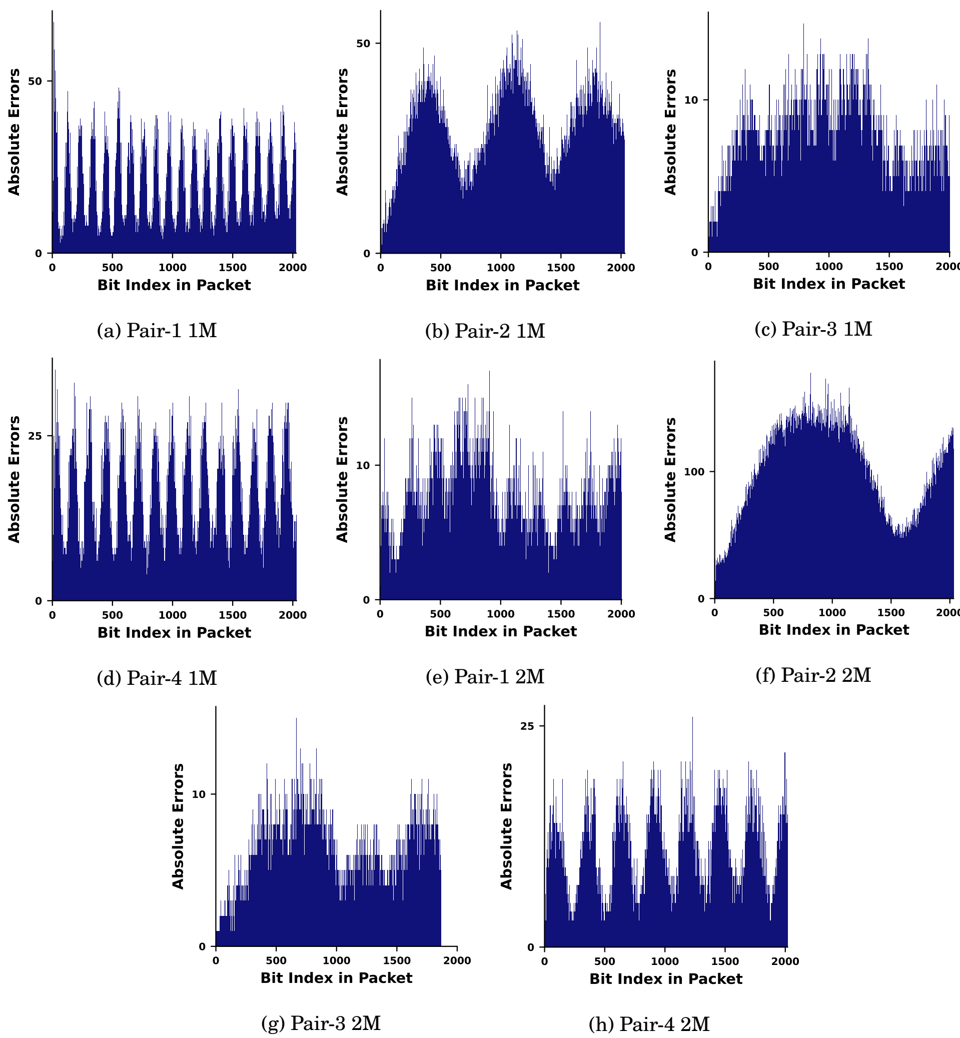
We adopt the same three-node single-hop topology we used in the lab experiments. The source node generates and transmits packets of length 255 bytes for BLE 5 PHYs and 125 bytes for IEEE 802.15.4 for the same reasons as in the lab experiments. Similarly, there is a receiver node and a node acting as a forwarder (another synchronous transmitter) in the network. We conducted an exploratory run to determine the node combinations among the myriad of node combinations possible in the D-Cube and to find the ones whose difference in carrier frequency appeared to result in beating destruction enough for the errors to occur in the packet receptions. We ended up with 10 of those three-node random combinations.

The low transmission powers supported by `nRF52840-DK` are -8dBm, -16dBm, -20dBm, and -40dBm and the synchronous transmitters transmit 200 unique packets for each physical layer for each of the 10 node combinations. For transmission power -8dBm, no beating was encountered between the source and the forwarder of the selected pairs. We suspect that due to higher transmission power, the capture effect impacted correct reception at the receiver. The transmission power was further decreased to -16dBm and -20dBm to increase the probability of beating and its resulting errors. As transmission power was reduced, 4 out of 10 combinations demonstrated beating. This is shown in Figures 3.10 and 3.12, where there are indications of different types of beating occurring for different node combinations. For

-40dBm, the nodes could not receive any packets due to extremely low transmission power.

After discovering the four 3-node combinations demonstrating beating, we ran tests using transmission powers of -16dBm and -20dBm. For every transmission power and PHY choice, the Packet Delivery Rate (PDR) is measured to evaluate the effectiveness of `STX-Vote`.

3.4.2.2 D-cube Results



(a) Pair-1 1M (b) Pair-2 1M (c) Pair-3 1M (d) Pair-4 1M (e) Pair-1 2M (f) Pair-2 2M (g) Pair-3 2M (h) Pair-4 2M

Figure 3.10: The observed beating pattern for *uncoded* PHYs at -16dBm.

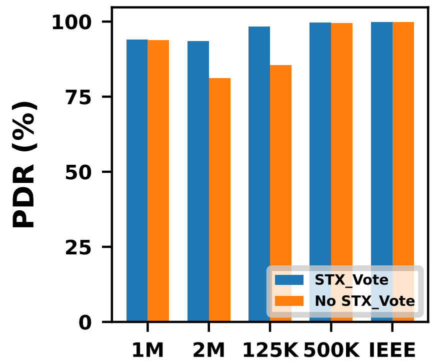


Figure 3.11: The average PDR for all BLE 5 and IEEE 802.15.4 PHYs at -16dBm.

3.4.2.3 D-cube Results at -16dBm

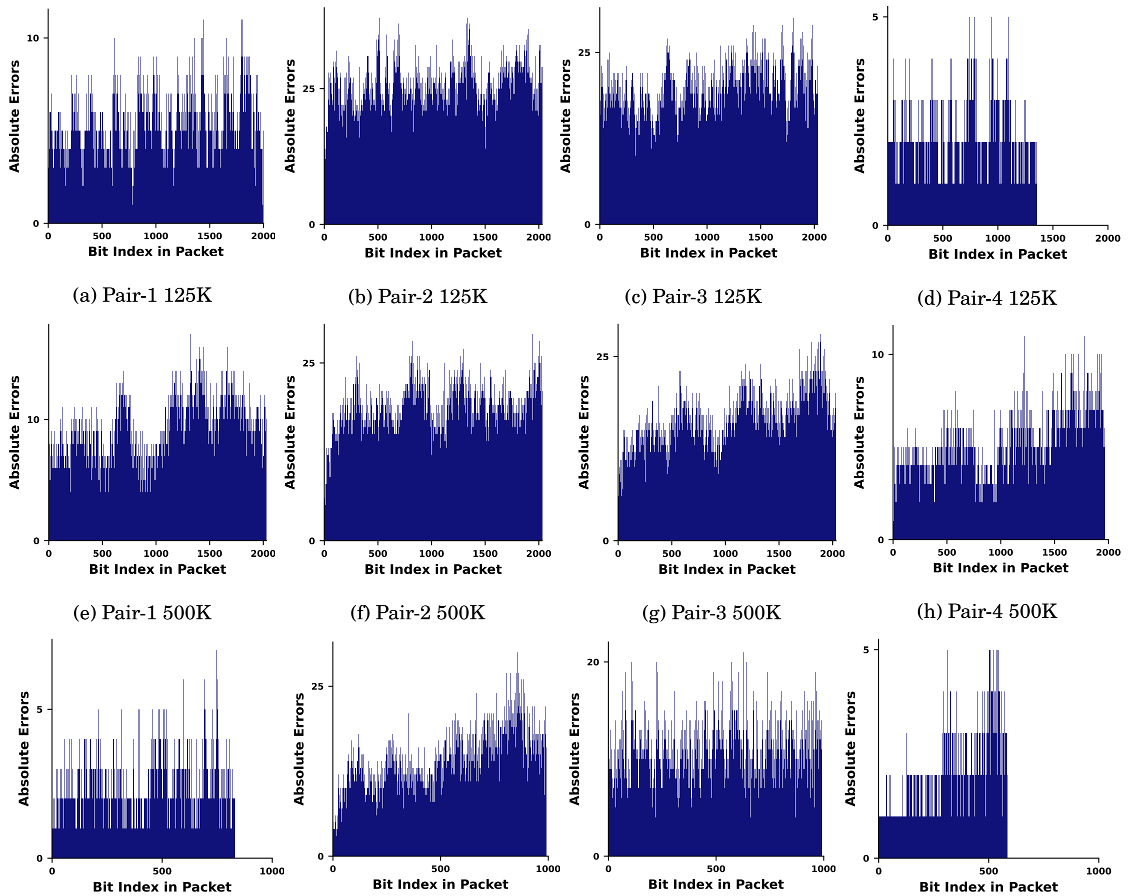
From Figure 3.11, we can see no improvement for uncoded PHY BLE 1M and an approximately 15% (with a standard deviation of 25) increase for BLE 2M. BLE 2M suffered because the beating must have been narrow, as seen in Figures 3.10e to 3.10h. Also, its transmission rate is high, and the lack of an error correction mechanism resulted in a high packet corruption probability compared to BLE 1M.

On the other hand, among the coded PHYs, BLE 125K suffered the most from the wide beating exhibited by the selected pairs as shown in Figures 3.12a to 3.12d. A distinct envelope cannot be seen, suggesting that the transmissions are affected by beating valleys. Because of wide beating and slower transmission rate, the FEC in BLE 125K cannot handle the errors occurring in the transmissions, and `STX-Vote` increases the PDR by 15% (with a standard deviation of 13) compared to when it is disabled.

For BLE 500K and IEEE 802.15.4, even though they encounter wide beating as shown in Figures 3.12e to 3.12l, their underlying error correction mechanisms, FEC and DSSS, respectively, are potent enough to correct the errors and may not require the inclusion of `STX-Vote`.

3.4.2.4 D-cube Results at -20dBm

The histograms in Figure 3.13 show a significant improvement of almost 20% (with a standard deviation of 6) in PDR for BLE 1M, but no improvement for BLE 2M. The reason behind this behaviour can be supported by analyzing the beating histograms



(a) Pair-1 125K (b) Pair-2 125K (c) Pair-3 125K (d) Pair-4 125K

(e) Pair-1 500K (f) Pair-2 500K (g) Pair-3 500K (h) Pair-4 500K

(i) Pair-1 IEEE (j) Pair-2 IEEE (k) Pair-3 IEEE (l) Pair-4 IEEE

Figure 3.12: Wide beating encountered by coded PHYs at -16dBm.

from Figures 3.14a to 3.14f. From the histograms, there are indications that the beating encountered by BLE 1M is narrower as compared to the beating encountered by BLE 2M. We notice the distinct envelope in the case of BLE 1M, whereas for BLE 2M, there is no distinct beating envelope, suggesting that BLE 2M is experiencing wide beating. Thus, it can leverage its fast transmission rate to fit a transmission during constructive beating crest [2]. Because of this, BLE 2M is giving a similar performance irrespective of `STX-Vote` being enabled or disabled.

Next, we analyze the behaviour of coded PHYs. Coded PHYs have built-in error correction mechanisms; however, these can only withstand beating up to a certain extent. From Figures 3.15a and 3.15h, we notice that there is no distinct beating envelope, which indicates that they are likely experiencing wide beating. Thus, their reliability decreases as their built-in error correction mechanisms fail to correct the

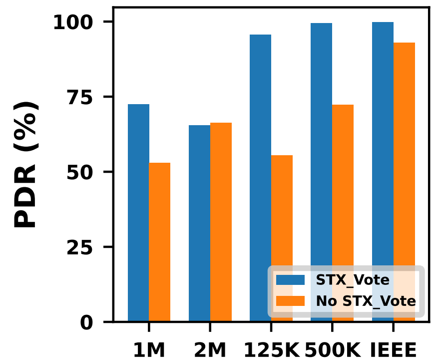
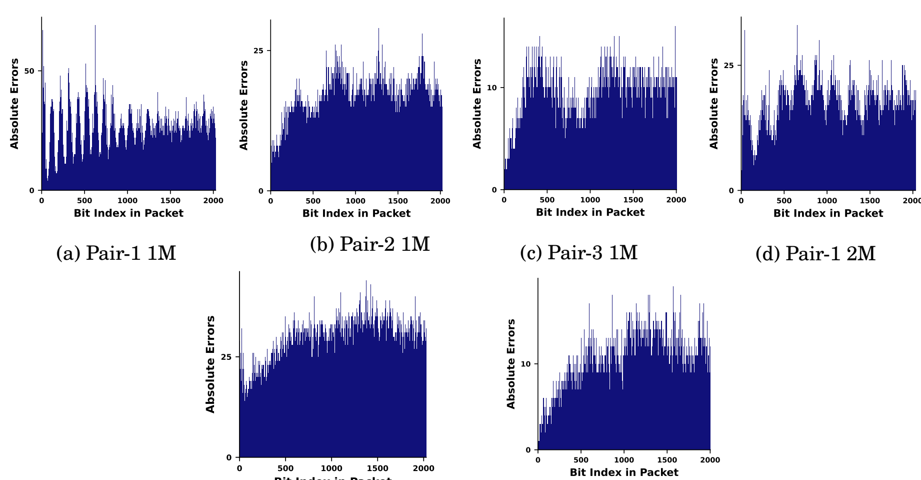


Figure 3.13: Average PDR for all PHYs at -20dBm with STX-Vote enabled or disabled.

errors due to wide beating. However, when these physical layers leverage STX-Vote, BLE PHYs 125K and 500K witness around 28% (with a standard deviation of 27) and 25% (with a standard deviation of 47) increase in their PDR, respectively. Along with coded BLE 5 PHYs, IEEE 802.15.4 also experiences wide beating, as seen from Figures 3.15i to 3.15l. We can assert that even though its DSSS coding scheme is robust to errors when the burst of the error caused by beating exceeds the DSSS coding depth, it cannot recover such packets. However, using STX-Vote shows an ~8% increase in PDR. Overall, STX-Vote provides robustness against wide and narrow beating by relying on the fact that beating across multiple packets is unlikely always to corrupt the same bit locations. Therefore STX-Vote can increase the probability of a successful reception through the voting-based reconstruction it performs.



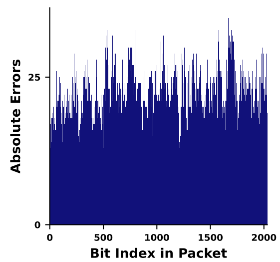
(a) Pair-1 1M (b) Pair-2 1M (c) Pair-3 1M (d) Pair-1 2M

(e) Pair-2 2M (f) Pair-3 2M

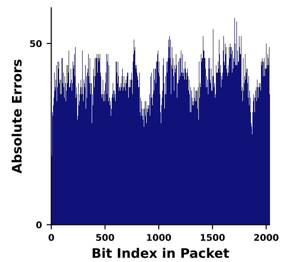
Figure 3.14: Observed beating patterns for uncoded physical layers at -20dBm.

Overall, we attribute the packet errors to primarily occurring due to the *beating effect*, which is noticeable in the modulation schemes of the PHYs we studied. Changing the specific modulation scheme to something more advanced, e.g., the modulation used in recent WiFi standards, can reduce or eliminate the impact of beating. However, many modulation schemes used by BLE 5 and IEEE 802.15.4 are typical of low-cost embedded sensor devices and are likely to remain relevant and widely used for years to come.

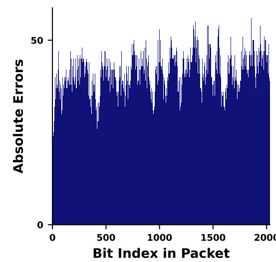
Furthermore, we cannot ignore the effects of other real-world phenomena, such as multipath effects, path loss attenuation, external interference, etc., which could also cause packet errors. For `STX-Vote`, while its voting approach is based on the unlikely corruption of the exact same bit locations across successive receptions, it can just as easily correct bit errors at random bit positions caused by other factors. That is, `STX-Vote`'s handling of beating-induced errors cannot be separated from handling any error in general, including any spurious errors unrelated to the beating effect. This lack of specificity of `STX-Vote` can be viewed as an advantage rather than a shortcoming.



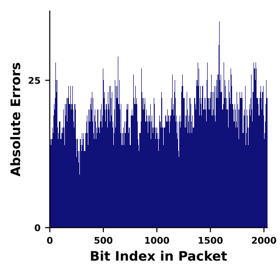
(a) Pair-1 125K



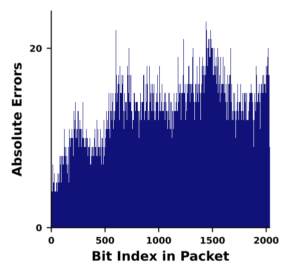
(b) Pair-2 125K



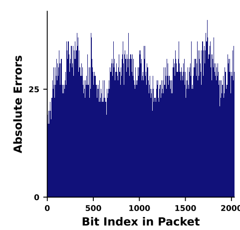
(c) Pair-3 125K



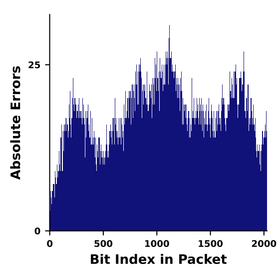
(d) Pair-4 125K



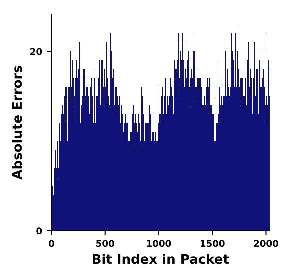
(e) Pair-1 500K



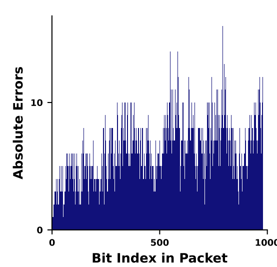
(f) Pair-2 500K



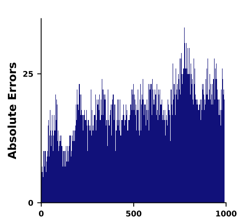
(g) Pair-3 500K



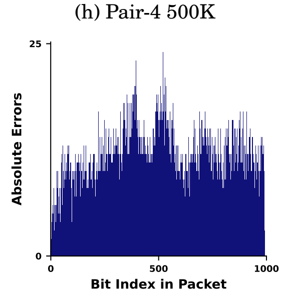
(h) Pair-4 500K



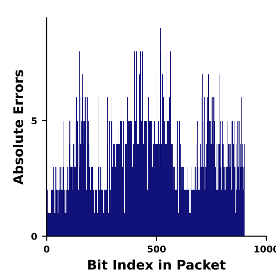
(i) Pair-1 IEEE



(j) Pair-2 IEEE



(k) Pair-3 IEEE



(l) Pair-4 IEEE

Figure 3.15: Indications of beating patterns across all coded BLE and IEEE 802.15.4 PHYs at -20dBm transmission power. No distinct envelopes are evident for BLE 125K and BLE 500K, suggesting that the receptions fell under beating valleys resulting due to wide beating between devices.

Chapter 4

Conclusion

4.1 Summary

This thesis introduced `STX-Vote`, an innovative error correction scheme designed to mitigate beating-induced errors in synchronous transmissions, a persistent challenge in the field of synchronous transmission-based communication paradigm. Through both simulation and real-world experimentation, we demonstrated that `STX-Vote` improved reliability by leveraging the unlikely corruption of the same bit positions in the received packets by periodic beating. `STX-Vote` exploited the inherent repetitions present in STX-based protocols through a form of repetition coding. `STX-Vote` applied majority voting to determine each bit's intended value, thereby reconstructing the correct packet at the receiver. We have also shown that `STX-Vote` is effective in various beating conditions where devices experience either 'narrow' or 'wide' beating depending on conditions and the underlying physical layer. In particular, we demonstrated that uncoded BLE PHYs achieve up to a ~20% increase in reliability in the D-Cube testbed environment. At the same time, `STX-Vote` increases the reliability of coded PHYs by up to ~28%. These encouraging results suggest that the simple bit voting technique in `STX-Vote` is a low overhead, effective error mitigation strategy in STX-based protocols.

4.2 Future Work

The proposed error correction mechanism can withstand a beating and can be evaluated under two synchronous transmitters and a single receiver. It is natural to wish to expand the evaluation to more complex real-world scenarios. However, in most real settings, multiple (more than two) synchronous transmitters might be employed to exploit the sender diversity offered by STX-based protocols, which increases the probability of a transmission being successfully received at the receiver, as it offers multiple routes and opportunities for the transmission to reach the receiver. Different transmitters participating in the communication will also have different CFOs due to oscillator inaccuracies, resulting in various deviations from the ideal carrier frequency. This CFO diversity and varied transmission power may lead to complex beating patterns as seen in Figure 4.1, which may be difficult to model [42]. Complex beating patterns could lead to various burst error patterns that are more challenging to mitigate.

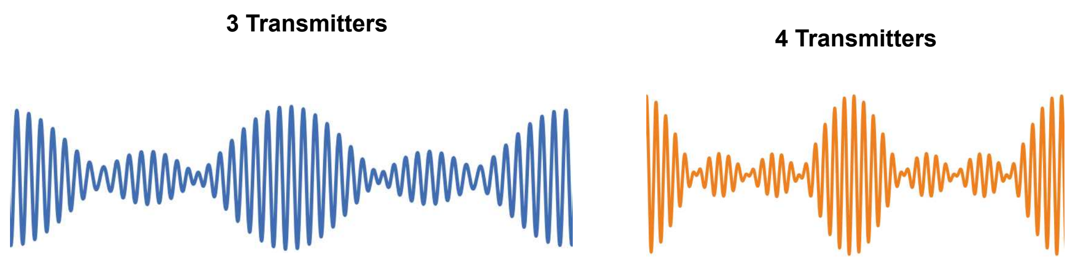


Figure 4.1: Complex beating pattern when more than two transmissions are simultaneously received.

As part of future work, we would like to study the performance of `STX-Vote` under those complex beating scenarios where the beating peaks and valleys could differ depending on the collective distortion caused by the overlapping signals. Given the design of `STX-Vote` to not depend on particular patterns, as long as the errors are spread at different bit locations across repeated transmissions, there is hope that `STX-Vote` can also address those more elaborate forms of beating-induced bit corruptions. Furthermore, we would like to study the impact of combining `STX-Vote` with STX schemes other than robust flooding to identify the combined effects of packet-level error correction like `STX-Vote` with other mechanisms employed at higher layers of the protocol stack.

We also plan to utilize network programmability by leveraging software-defined networking (SDN) [49–51] in deploying `STX-Vote`. Specifically, there are existing SDN-based packet forwarding methods to improve reliability in low-power IoT networks. We will explore the possibility of integrating these existing approaches in `STX-Vote` to make it agile in various networking conditions while retaining its high reliability [52–60].

Bibliography

[1] A. C. Djedouboum, A. A. Abba Ari, A. M. Gueroui, A. Mohamadou, and Z. Aliouat, "Big data collection in large-scale wireless sensor networks," *Sensors*, vol. 18, no. 12, p. 4474, 2018.

[2] M. Baddeley, C. A. Boano, A. Escobar-Molero, Y. Liu, X. Ma, V. Marot, U. Raza, K. Römer, M. Schuss, and A. Stanoev, "Understanding concurrent transmissions: The impact of carrier frequency offset and RF interference on physical layer performance," *ACM Transactions on Sensor Networks*, vol. 20, no. 1, pp. 1–39, 2023.

[3] M. Baddeley, "Software Defined Networking for the Industrial Internet of Things," Ph.D. dissertation, University of Bristol, 2020.

[4] I. T. Haque, I. Nikolaidis, and P. Gburzynski, "On the impact of node placement and profile point selection on indoor localization," in *Joint IFIP Wireless and Mobile Networking Conference*. Springer, 2009, pp. 220–231.

[5] ——, "Expected path length for angle and distance-based localized routing," in *2009 International Symposium on Performance Evaluation of Computer & Telecommunication Systems*, vol. 41. IEEE, 2009, pp. 137–141.

[6] ——, "A scheme for indoor localization through RF profiling," in *2009 IEEE International Conference on Communications Workshops*. IEEE, 2009, pp. 1–5.

[7] I. T. Haque, M. S. Islam, and J. Harms, "On selecting a reliable topology in wireless sensor networks," in *2015 IEEE International Conference on Communications (ICC)*. IEEE, 2015, pp. 6376–6382.

[8] I. T. Haque, "On the overheads of ad hoc routing schemes," *IEEE Systems Journal*, vol. 9, no. 2, pp. 605–614, 2014.

[9] I. T. Haque and C. Assi, "Profiling-based indoor localization schemes," *IEEE Systems Journal*, vol. 9, no. 1, pp. 76–85, 2013.

[10] I. T. Haque, "A sensor based indoor localization through fingerprinting," *Journal of network and computer applications*, vol. 44, pp. 220–229, 2014.

[11] I. Haque, I. Nikolaidis, and P. Gburzynski, "On the benefits of nondeterminism in location-based forwarding," in *International Conference on Communications (ICC)*, 2009.

[12] I. T. Haque, "Non-deterministic geographic forwarding in mobile ad hoc networks," in *2008 IEEE Asia-Pacific Services Computing Conference*. IEEE, 2008, pp. 1144–1149.

[13] I. T. Haque and C. Assi, "Localized energy efficient routing in mobile ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 7, no. 6, pp. 781–793, 2007.

[14] I. Haque, I. Nikolaidis, and P. Gburzynski, "On the pitfalls of directional location-based randomized routing," *SPECTS 2007, July 16-18, San Diego, CA, USA*, 2007.

[15] I. T. Haque and C. Assi, "OLEAR: Optimal localized energy aware routing in mobile ad hoc networks," in *2006 IEEE International Conference on Communications*, vol. 8. IEEE, 2006, pp. 3548–3553.

[16] I. T. Haque, C. Assi, and J. W. Atwood, "Randomized energy aware routing algorithms in mobile ad hoc networks," in *Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, 2005, pp. 71–78.

[17] T. Fevens, I. T. Haque, and L. Narayanan, "Randomized routing algorithms in mobile ad hoc networks," in *Mobile and Wireless Communication Networks: IFIP TC6/WG6. 8 Conference on Mobile and Wireless Communication Networks (MWCN 2004) October 25–27, 2004, Paris, France*. Springer, 2005, pp. 347–357.

[18] T. Fevens, I. Haque, and L. Narayanan, "A class of randomized routing algorithms in mobile ad hoc networks," *AlgorithmS for Wireless and mobile Networks (A SWAN 2004), Boston*, 2004.

[19] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*. IEEE, 2011, pp. 73–84.

[20] M. Zimmerling, L. Mottola, and S. Santini, "Synchronous transmissions in low-power wireless: A survey of communication protocols and network services," *ACM Computing Surveys (CSUR)*, vol. 53, no. 6, pp. 1–39, 2020.

[21] A. Escobar, F. J. Cruz, J. Garcia-Jimenez, J. Klaue, and A. Corona, "RedFixHop with channel hopping: Reliable ultra-low-latency network flooding," in *2016 Conference on Design of Circuits and Integrated Systems (DCIS)*. IEEE, 2016, pp. 1–4.

[22] R. Lim, R. Da Forno, F. Sutton, and L. Thiele, "Competition: Robust Flooding using Back-to-Back Synchronous Transmissions with Channel-Hopping." in *EWSN*, 2017, pp. 270–271.

[23] D. Yuan and M. Hollick, "Ripple: High-throughput, reliable and energy-efficient network flooding in wireless sensor networks," in *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2015, pp. 1–9.

[24] X. Ma, P. Zhang, Y. Liu, C. A. Boano, H.-S. Kim, J. Wei, and J. Huang, "Harmony: Saving concurrent transmissions from harsh RF interference," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1024–1033.

[25] C. Herrmann, F. Mager, and M. Zimmerling, "Mixer: Efficient many-to-all broadcast in dynamic wireless mesh networks," in *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, 2018, pp. 145–158.

[26] M. Mohammad and M. C. Chan, "Codecast: Supporting data driven in-network processing for low-power wireless sensor networks," in *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 2018, pp. 72–83.

[27] S. Chandrasekhar and X. Liu, "Experimental study on 42.7-Gb/s forward-error-correction performance under burst errors," *IEEE Photonics Technology Letters*, vol. 20, no. 11, pp. 927–929, 2008.

[28] C.-H. Liao, Y. Katsumata, M. Suzuki, and H. Morikawa, "Revisiting the so-called constructive interference in concurrent transmission," in *2016 IEEE 41st Conference on Local Computer Networks (LCN)*. IEEE, 2016, pp. 280–288.

[29] A. Escobar-Molero, "Improving reliability and latency of wireless sensor networks using concurrent transmissions," *at-Automatisierungstechnik*, vol. 67, no. 1, pp. 42–50, 2019.

[30] ——, "GitHub - ADEscobar/ct-simulator — github.com," https://github.com/ADEscobar/ct-simulator, 2021, [Accessed 09-12-2024].

[31] M. Baddeley, Y. Gyl, M. Schuß, X. Ma, and C. A. Boano, "OSF: An Open-Source framework for Synchronous Flooding over multiple physical layers," in *Proc. of the* 19^{th} *EWSN Conf.*, Oct. 2022.

[32] M. Schuß, C. A. Boano, and K. Römer, "Moving beyond competitions: Extending D-cube to seamlessly benchmark low-power wireless systems," in *2018 IEEE Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench)*. IEEE, 2018, pp. 30–35.

[33] B. Rangwala, A. Powelson, M. Baddeley, and I. Haque, "STX-Vote: Improving reliability with Bit Voting in Synchronous Transmission-based IoT networks," 2024. [Online]. Available: https://arxiv.org/abs/2405.02022

[34] ——, "GitHub - PINetDalhousie/CTX_Vote — github.com," https://github.com/PINetDalhousie/CTX_Vote, 2024, [Accessed 15-12-2024].

[35] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, 2002, pp. 194–205.

[36] M. Ahyar, M. Yani, R. F. Sari *et al.*, “Comparison of energy efficiency and routing packet overhead in single and multi path routing protocols over S-MAC for wireless sensor network,” in *2012 Sixth UKSim/AMSS European Symposium on Computer Modeling and Simulation*. IEEE, 2012, pp. 406–411.

[37] R. Zhang, J.-M. Gorce, O. Berder, and O. Sentieys, “Lower bound of energy-latency tradeoff of opportunistic routing in multihop networks,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, pp. 1–17, 2011.

[38] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, “The broadcast storm problem in a mobile ad hoc network,” in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, 1999, pp. 151–162.

[39] A. Ephremides and T. V. Truong, “Scheduling broadcasts in multihop radio networks,” *IEEE Transactions on communications*, vol. 38, no. 4, pp. 456–460, 1990.

[40] A. D. Escobar Molero, “Using concurrent transmissions to improve the reliability and latency of low-power wireless mesh networks,” Ph.D. dissertation, Dissertation, Rheinisch-Westfälische Technische Hochschule Aachen, 2020, 2020.

[41] M. Woolley, “Bluetooth core specification version 5.0 feature enhancements,” *Bluetooth SIG*, 2021.

[42] B. A. Nahas, A. Escobar-Molero, J. Klaue, S. Duquennoy, and O. Landsiedel, “BlueFlood: Concurrent transmissions for multi-hop Bluetooth 5—Modeling and evaluation,” *ACM Transactions on Internet of Things*, vol. 2, no. 4, pp. 1–30, 2021.

[43] M. E. Pellenz, R. D. Souza, and M. S. P. Fonseca, “Error control coding in wireless sensor networks: Trade-off between transmission and processing energy consumption,” *Telecommunication Systems*, vol. 44, pp. 61–68, 2010.

[44] T. Istomin, M. Trobinger, A. L. Murphy, and G. P. Picco, “Interference-resilient ultra-low power aperiodic data collection,” in *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 2018, pp. 84–95.

[45] M. Doddavenkatappa, M. C. Chan, and B. Leong, “Splash: Fast data dissemination with constructive interference in wireless sensor networks,” in *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 269–282.

[46] H. Dubois-Ferriere, D. Estrin, and M. Vetterli, “Packet combining in sensor networks,” in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, 2005, pp. 102–115.

[47] T. Sakdejayont, C.-H. Liao, M. Suzuki, and H. Morikawa, “Beating the Beat: RSSI-Based Packet Combining in Concurrent Transmission Sensor Networks,” in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, 2016, pp. 356–357.

[48] S. Kothapalli, S. Ningthoujam, S. R. Sahoo, R. K. Batchu, and S. K. Chakraborthy, “Extended packet combining schemes for error control in wireless networks,” in *2022 2nd International Conference on Artificial Intelligence and Signal Processing (AISP)*. IEEE, 2022, pp. 1–7.

[49] I. T. Haque and N. Abu-Ghazaleh, “Wireless software defined networking: A survey and taxonomy,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2713–2737, 2016.

[50] I. Haque, M. Nurujjaman, J. Harms, and N. Abu-Ghazaleh, “SDSense: An agile and flexible SDN-based framework for wireless sensor networks,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1866–1876, 2018.

[51] D. Saha, M. Shojaee, M. Baddeley, and I. Haque, “An energy-aware SDN/NFV architecture for the internet of things,” in *2020 IFIP networking conference (networking)*. IEEE, 2020, pp. 604–608.

[52] V. Kolar, I. T. Haque, V. P. Munishwar, and N. B. Abu-Ghazaleh, “CTCV: A protocol for coordinated transport of correlated video in smart camera networks,” in *2016 IEEE 24th International Conference on Network Protocols (ICNP)*. IEEE, 2016, pp. 1–10.

[53] U. Lekhala and I. Haque, “PiQoS: A programmable and intelligent QoS framework,” in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2019, pp. 234–239.

[54] I. Haque and M. Moyeen, “Revive: A reliable software defined data plane failure recovery scheme,” in *2018 14th International Conference on Network and Service Management (CNSM)*. IEEE, 2018, pp. 268–274.

[55] M. Darianian, C. Williamson, and I. Haque, “Experimental evaluation of two openflow controllers,” in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*. IEEE, 2017, pp. 1–6.

[56] M. Moyeen, F. Tang, D. Saha, and I. Haque, “SD-FAST: A packet rerouting architecture in SDN,” in *2019 15th International Conference on Network and Service Management (CNSM)*. IEEE, 2019, pp. 1–7.

[57] F. Tang and I. Haque, “Remon: A resilient flow monitoring framework,” in *2019 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2019, pp. 137–144.

[58] F. Tang, M. Shojaee, and I. Haque, "ACE: an Accurate and Cost-Effective Measurement System in SDN," 2021. [Online]. Available: https://arxiv.org/abs/2108.12849

[59] I. Haque and D. Saha, "SoftIoT: A resource-aware SDN/NFV-based IoT network," *Journal of Network and Computer Applications*, vol. 193, p. 103208, 2021.

[60] Z. Wang, Y. Zhou, M. Qiu, I. Haque, L. Brown, Y. He, J. Wang, D. Lo, and W. Zhang, "Towards fair machine learning software: Understanding and Addressing Model Bias Through Counterfactual Thinking," 2023. [Online]. Available: https://arxiv.org/abs/2302.08018