Diss. ETH No. XXXXX

Seeking Autonomy for Low-Power Wireless Networks

A thesis submitted to attain the degree of

Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by
ANDREAS BIRI
MSc ETH EEIT, ETH Zurich

born on 01.03.1994
citizen of
Zug ZG, Switzerland

accepted on the recommendation of
Prof. Dr. Lothar Thiele, examiner
Prof. Dr. Carlo Alberto Boano, co-examiner
Prof. Dr. Laurent Vanbever, co-examiner

2023

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

TIK-SCHRIFTENREIHE NR. XXX

ANDREAS BIRI

Seeking Autonomy for Low-Power Wireless Networks

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

A dissertation submitted to
ETH Zurich
for the degree of Doctor of Sciences

DISS. ETH NO. XXXXX

Prof. Dr. Lothar Thiele, examiner
Prof. Dr. Carlo Alberto Boano, co-examiner
Prof. Dr. Laurent Vanbever, co-examiner

Examination date: July 14, 2023

DOI: 10.3929/ethz-b-000XXXXXXXX

Abstract

With the emerging Internet of Things (IoT), even everyday devices are increasingly expected to be interconnected. Low-power wireless networks are becoming ubiquitous and are anticipated to provide a broad range of services that are of pivotal societal and industrial importance. However, while the original vision of such networks was to freely distribute devices that may then self-organize and operate autonomously, practical deployments instead often require significant manual effort to plan, configure, and maintain wireless sensor networks (WSNs). In most instances, systems employ a highly centralized design oriented around a designated leader or fixed structures for network orchestrating, thereby strongly limiting their mobility, adaptivity, and fault tolerance. With recent developments in powerful sensing and processing capabilities, highly reliable and flexible networking schemes, and novel communication technologies, such constraints could soon be a remnant of the past.

In this thesis, we aim to boost the autonomy of low-power wireless networks and avoid dependencies on fixed infrastructure, pre-configurations, and manual interventions. In a variety of scenarios, we focus on how sets of nodes can autonomously form connected clusters and maintain reliable communication despite high mobility and a constantly changing network topology. In addition, we investigate novel methods to orchestrate networks without traditional leader-centric schemes and increase their fault tolerance by distributing decision-making and exploiting physical redundancy. To this end, this thesis makes the following main contributions:

- We introduce the concept of a dual-radio architecture for infrastructure-free interaction tracking. By combining energy-efficient discovery with high-fidelity sensing, we are the first to enable fully mobile nodes to self-organize the collection of pairwise distance information on a network level and accurately measure the physical network topology. We further show that this autonomy from fixed infrastructure enables new, mobile deployment options with unprecedented flexibility.
- We develop methods for robust network orchestration of mobile nodes despite arbitrary node and link failures. Based on a novel election scheme, we present the first low-power wireless network

protocol that can instantly handle network splits and find consensus on leadership. Through a cluster-wide discovery scheme, we merge autonomously operating clusters of nodes on the fly and maximize network coverage by quickly aggregating and integrating information on the current network topology.

- We propose to orchestrate dynamic clusters using purely event-based communication by directly exploiting the spatial and temporal correlation of signals from the sensed phenomenon. Leveraging the locality of a co-detection, where a physical event triggers multiple sensors quasi-simultaneously, we autonomously form an ad-hoc network and locally aggregate data for quick and efficient processing. Based on a novel symbiosis of short- and long-range radio links, centralized control and periodic overhead can be eliminated to achieve more energy-efficient communication with a lower, more consistent detection latency. We thereby propose a fundamentally new coordination approach to synchronous networking that avoids the elementary conflict between duty cycling and reactivity.
- We present the concept of concurrent coordination to enable fault-tolerant low-power wireless networking without any centralized coordination based on a novel distributed consensus and synchronization mechanism. With the underlying paradigm that each node is equivalent in protocol logic and configuration, we are the first to avoid any entity that maintains a unique state or serves a special purpose within such a persistent network. We further prove analytically that we can still guarantee freedom of harmful packet collisions, adaptivity to a dynamic network topology, and stable data exchange despite node and link failures.

Zusammenfassung

Mit dem aufkommenden Internet der Dinge (IoT) wird erwartet, dass selbst alltägliche Geräte zunehmend miteinander verbunden sind. Drahtlose Netze mit geringem Energiebedarf werden allgegenwärtig und werden voraussichtlich eine breite Palette an Diensten bereitstellen, die von zentraler gesellschaftlicher und industrieller Bedeutung sind. Während die ursprüngliche Vision solcher Netze die freie Verteilung von Geräten vorhersah, welche sich danach selbst organisieren und autonom arbeiten können, ist in der Praxis oft ein erheblicher manueller Aufwand für die Planung, Konfiguration und Wartung von drahtlosen Sensornetzwerken (WSNs) erforderlich. In den meisten Fällen sind die Systeme stark zentralisiert und orientieren sich an einem fremdbestimmten Anführer ("leader") oder an fixen Installationen, um das Netzwerk zu organisieren, wodurch ihre Mobilität, Anpassungsfähigkeit und Fehlertoleranz stark eingeschränkt werden. Mit den jüngsten Entwicklungen in Bezug auf leistungsstarke Sensorik und Datenprozessierung, hochzuverlässige und flexible Vernetzungskonzepte und neuartige Kommunikationstechnologien könnten solche Einschränkungen bald der Vergangenheit angehören.

Diese Doktorarbeit zielt darauf ab, die Autonomie von drahtlosen Netzwerken mit geringem Stromverbrauch zu erhöhen und Abhängigkeiten von statischer Infrastruktur, vorzeitigen Konfigurationen und manuellen Eingriffen zu vermeiden. In verschiedenen Szenarien untersuchen wir, wie Gruppen von Sensorknoten autonom zusammenhängende Ansammlungen ("clusters") bilden und trotz hoher Mobilität und einer sich ständig ändernden Netzwerktopologie zuverlässige Kommunikation aufrechterhalten können. Darüber hinaus untersuchen wir neuartige Ansätze zur Orchestrierung von Netzwerken ohne herkömmliche zentralisierte Methoden, welche auf einem festgelegten Anführer basieren, und erhöhen deren Fehlertoleranz durch die Verteilung der Entscheidungsfindung und die Ausnutzung physischer Redundanz. Zu diesem Zweck liefert diese Arbeit die folgenden Hauptbeiträge:

- Wir stellen das Konzept einer heterogenen Radio-Architektur für die infrastrukturfreie Verfolgung von Interaktionen vor. Durch die Kombination von energieeffizienter Entdeckung anderer Sensorknoten und hochgenauen Messungen sind wir die Ersten, die es

vollständig mobilen Knoten ermöglichen, das Messen paarweiser Distanzinformationen auf der Netzwerkebene selbst zu organisieren und die physische Netzwerktopologie präzise zu vermessen. Des Weiteren zeigen wir, dass diese Unabhängigkeit von statischer Infrastruktur neue und mobile Einsatzmöglichkeiten mit einer bisher unerreichten Flexibilität ermöglicht.

- Wir entwickeln Methoden für eine robuste Netzwerk-Orchestrierung von mobilen Knoten trotz beliebiger Knoten- und Verbindungsausfällen. Basierend auf einem neuartigen Wahlverfahren stellen wir das erste drahtlose Netzwerkprotokoll mit geringem Energiebedarf vor, das sofort mit Netzwerkaufteilungen umgehen und Konsens über die Führung der Ansammlung finden kann. Dank einem Entdeckungsmechanismus, welcher alle Knoten der Ansammlung miteinbezieht, vereinen wir autonom arbeitende Ansammlungen von Knoten zügig und maximieren die Netzwerkabdeckung durch eine rasche Aggregation und Integration von Informationen über die aktuelle Netzwerktopologie.
- Wir schlagen vor, dynamische Ansammlungen rein aufgrund ereignisbasierter Kommunikation zu orchestrieren, indem wir direkt die räumliche und zeitliche Korrelation von Signalen des erfassten Phänomens benutzen. Unter Ausnutzung der Lokalität einer Ko-Detektion, bei welcher ein physikalisches Ereignis mehrere Sensoren nahezu zeitgleich auslöst, bilden wir autonom ein Ad-hoc-Netzwerk und aggregieren Daten für eine schnelle und effiziente lokale Verarbeitung. Basierend auf einer neuartigen Symbiose von Kurz- und Langdistanz-Funkverbindungen können eine zentrale Steuerung und periodische Fixkosten eliminiert werden, um eine energieeffizientere Kommunikation mit einer geringeren, konsistenteren Latenz der Ereigniserkennung zu erreichen. Damit stellen wir einen grundlegend neuen Koordinationsansatz für synchrone Netzwerke vor, der den elementaren Konflikt zwischen Arbeitszyklus ("duty cycle") und Reaktivität vermeidet.
- Wir präsentieren das Konzept der gleichzeitigen Koordination basierend auf einem neuartigen verteilten Konsens- und Synchronisationsmechanismus, um fehlertolerante drahtlose Netzwerke mit geringem Energiebedarf ohne zentrale Koordination zu ermöglichen. Mit dem Grundsatz, dass jeder Knoten äquivalent ist bezüglich seiner Protokolllogik und Konfiguration, sind wir die Ersten, welche jegliches Element vermeidet, welches einzigartiges Wissen besitzt

oder einen speziellen Zweck innerhalb eines solchen dauerhaften Netzwerks erfüllt. Darüber hinaus beweisen wir analytisch, dass wir dennoch das Vermeiden von schädlichen Paketkollisionen, die Anpassungsfähigkeit an eine dynamische Netzwerktopologie und einen stabilen Datenaustausch trotz Knoten- und Verbindungsausfällen garantieren können.

Acknowledgments

To be included after the doctoral examination.

Contents

List of Figures

List of Tables

Acronyms

ADC analog-to-digital converter

BLE Bluetooth Low Energy

BOM bill of materials

CAD carrier activity detection

CRC cyclic redundancy check

CT concurrent transmissions

GNSS Global Navigation Satellite System

IoT Internet of Things

IR infrared

LP-WAN low-power wide-area network

MCU microcontroller unit

MSCC majority strongly connected component

NLOS non-line-of-sight

PCB printed circuit board

PRR packet reception rate

RAM random access memory

RNG random number generator

RSSI Received Signal Strength Indicator

RTC real-time clock

SCC strongly connected component

SPOF single point of failure

SWaP size, weight, and power

TDoA time difference of arrival

ToF time of flight

TSCH Time Slotted Channel Hopping

UWB ultra-wideband

WSN wireless sensor network

1

Introduction

Networked sensing has become a key driver for the Internet of Things (IoT). Automated building management, pollution and energy consumption monitoring, smart agriculture, as well as predictive maintenance lead to an estimated 16 billion currently active IoT devices [IoT23]. The ability of such systems to automatically gather detailed sensor data and collect them over low-power wireless links enables highly scalable, low-cost deployments for a plethora of applications. Along with the introduction of a variety of popular communication technologies, including IEEE 802.15.4, Bluetooth Low Energy (BLE), LoRa, and ultra-wideband (UWB), the development of energy-efficient and reliable wireless network protocols has enabled such devices to become ubiquitous. In tandem with powerful cloud computing for data analysis and tremendous progress in machine learning, the combination of distributed sensing and centralized data collection has established itself as the dominating structure of wireless sensor networks (WSNs).

However, while many such networks are still built around a traditional backbone infrastructure of static data sinks for connectivity to the Internet, a rapidly growing class of applications demands more autonomy. From a swarm of drones during a search-and-rescue operation [WTJ09, WH11] over the coordination of intersection crossings for self-driving cars [RPL20] and industrial high-frequency feedback control loops [MBJ+19] to the measurements of human interactions to study the spread of diseases [CAD+20, SPDG+20, RL22], local networks need to operate independently. Despite progress in researching more robust

connections, wireless links are notoriously variable [JFT+19] and sensor nodes often exposed to hostile environments [BISV08, WCPC18, ABC+20], which makes centralized systems prone to catastrophic failures [WC19]. Furthermore, due to privacy concerns, long and unreliable transmission latencies, and limited connectivity in indoor and remote settings, recent years have seen attempts to push intelligence back towards the edge through the rise of embedded machine learning [GHS+22, PAL22].

Therefore, it is essential that connected nodes can also exchange data *within* an autonomous cluster without external dependencies. For example, drones navigating tight indoor spaces must be able to fly in formation and prevent collisions [GET22] while sharing control and sensor data [HYM16]. An early-warning system that is monitoring volcanic activity [WAJR+05, WALJ+06], rockfalls [MFCP+19, WBDF+19], or wildfires [LWS06, YNL+12] should also be able to directly trigger warnings and restrict access to endangered areas. Similarly, human interactions must be automatically registered by mobile devices to research organizational behavior [OWK+09], childhood development [SPH+22], and pandemics [RLS+07] so infrastructure limitations do not severely constrain the system's coverage.

To attain such autonomy, a WSN must be able to orchestrate communication itself, i.e., by only depending on the sensor nodes it consists of. Traditionally, static deployments have benefited from fixed configurations such as a designated leader that synchronizes and schedules the network. However, mobile nodes may not be within reach of such a leader but still require coordination in order to communicate reliably and efficiently. In addition, even without voluntary movement, node and link failures can disrupt communication and fundamentally change the network topology at any time, thereby requiring network orchestration to be adjusted. Only by both automatically handling the formation of clusters consisting of connected nodes as well as by demonstrating resilience throughout its operation can a low-power wireless network become autonomous.

1.1 Aims of this Thesis

Based on the scenarios outlined above, this thesis aims to boost autonomy in two fundamental aspects of low-power wireless networks:

- Nodes in a WSN should be able to deal with the **autonomous formation** of persistent or ad-hoc clusters, including spontaneous splitting and joining, without having to rely on fixed infrastructure or manual pre-configurations and adjustments.

- Due to hostile and volatile operating environments, a protocol for WSN should provide **autonomous resilience** to node and link failures so that the network can safely, adaptively, and persistently communicate despite high node mobility and drastically changing conditions.

Figure 1.1 shows the focus of the four chapters presented in this thesis in regard to these aims. *Autonomous formation* is investigated in Chapter 2 (formation of persistent clusters), Chapter 3 (splits and joins of persistent clusters), and Chapter 4 (formation of ad-hoc clusters). *Autonomous resilience* is covered in depth in Chapter 3 (robust orchestration of clusters) and Chapter 5 (fault-tolerant orchestration of a network).

1.2 System Requirements

To achieve both autonomous formation and autonomous resilience as introduced above, we identify requirements in the following three primary areas that a suitable WSN protocol must satisfy:

- *Network orchestration*: A network consists of a set of nodes that desire to interact with each other based on an underlying network topology. We define a *cluster* of nodes to contain a maximal set of nodes in which each member is physically able to communicate to any other node in the set across multiple hops. For each such cluster of connected nodes, orchestration can be separated into two parts. **Robust cluster coordination** must ensure that nodes can efficiently and reliably communicate with each other based on a shared notion of the cluster. This coordination must provide nodes with knowledge on when to send what and may only depend on devices that are part of the cluster. **Swift cluster formation** has to permit nodes to find and interact with all others within communication distance. The formation of a cluster must contain all connected nodes, requiring that nodes either continuously discover and merge clusters during operation or ensure that relevant nodes are already contained during bootstrapping.
- *Fault tolerance and adaptivity*: An autonomous WSN must be dependable and independently cope with changing conditions to leverage its full potential. **Resilient networking** requires that the safety and liveness of communication should be guaranteed even under arbitrary node and link failures. Safety demands that a cluster

agrees on when to communicate to prevent packet collisions even if a leader is not around to coordinate, and liveness seeks to preserve the exchange of data despite changes in the network topology. **Adaptivity to dynamic environments** requires a network to automatically adjust its communication to varying traffic demands and environmental conditions. This reactivity is one of the key indicators of autonomous, smart networks and is imperative to support node mobility and sustain long-term operation.

- *Efficient resource usage*: For a battery-powered sensor node with limited computational capabilities and bandwidth to be effective, its resources need to be used with care. This scarcity asks for the trade-off between low resource consumption and communication needs to be continuously re-evaluated and adjusted to operating conditions. **Energy efficiency** requires nodes to adapt their networking service according to current application demand by allowing the protocol to dynamically increase and decrease the intensity of communication, also referred to as duty cycle. Such flexibility calls for a system that is reactive to events by enabling hardware components on-demand and swiftly releasing obsolete resources instead of over-provisioning. **Channel efficiency** requires nodes to know precisely when they are permitted to send to avoid contention and corrupted communication. For protocols to scale well and support dense deployments as well as temporary spikes in traffic demand after an event occurred, communication within a cluster should be tightly synchronized and coordinated to achieve a dependable transfer of information.

1.3 Challenges

With requirements defined for (*i*) network orchestration, (*ii*) fault tolerance and adaptivity, and (*iii*) efficient resource usage, we next investigate distinct challenges that the design of an autonomous system entails. We thereby include a discussion on how current state-of-the-art systems tackle these issues and identify open needs to achieve autonomy.

1.3.1 Network Orchestration

Cluster coordination. Coordinating the communication of a cluster of multiple nodes requires each of them to share knowledge on which member is allowed to send at which time so packets do not collide. For a common

notion of the network state, most protocols rely on a centralized *leader*, i.e., a node that is assigned to perform a specialized and potentially unique role, to collect, process, and distribute this information [FZMT12, HL20]. This consolidation of all coordination tasks, including synchronization and scheduling, at a single point in the network is prone to both unavailability of the leader due to failures, discussed in more detail below, and high uncertainty as well as long delays for all nodes to reorganize after the leader changed. Distributed and autonomous schedulers [AVP+15, DANLW15, KKK19] help to mitigate these issues, but require routing information and synchronization that is once again based on a central root node [DWVT14, WTB+12]. Therefore, there is a need for mechanisms that robustly coordinate clusters despite failures and spread the required information throughout the network to mitigate their impact.

Initial cluster formation. Most protocols configure a fixed leader node to which others align during bootstrapping [HMZ18, LFZ13]. As some clusters may not include this designated leader, multiple backup leaders must be defined to take its place, usually in a hard-coded order [FZMT12]. To boost autonomy by avoiding such manual configurations, nodes may also independently perform leader elections on demand to find a suitable candidate from a dynamic set within the cluster [CRW11, SPZE20]. However, as both fixed leaders and leader elections do not prevent multiple parallel clusters, unintended splits of centralized systems may violate their assumptions and endanger system integrity due to conflicting leadership [RPL20, ANDL17b]. By demanding that a cluster consists of a majority of nodes, such incompatibilities may be prevented [ANDL17a, Lam01]. However, there is a need to permit the ad-hoc formation of independent clusters of any size without jeopardizing a system's integrity.

Cluster discovery. An orthogonal approach to forming a network during bootstrapping is to use neighbor discovery schemes [MB01, DC08]. By regularly sending beacons and listening for nodes nearby, these protocols enable nodes to find and merge efficiently and within bounded time [QLXL16, JLMP17]. As mobile clusters may encounter each other throughout the deployment, such discovery must run continuously and swiftly succeed once clusters are within range to prevent mutual interference due to colliding packets. However, discovery protocols are built on the implicit assumption that access to the radio is exclusive and are hence conflicting with the simultaneous use of a protocol for networking. Therefore, there is a need for designing systems that combine discovery and communication so that both can be pursued in parallel.

1.3.2 Fault Tolerance and Adaptivity

Consensus. The simplest method for all nodes to agree on a decision is if it is taken centrally by a single entity. For example, if a communication schedule needs to be adapted because of a change in traffic demand, the leader may simply adjust it locally and distribute a new version that is then adopted by the rest of the cluster [FZMT12, SDFG+17]. While effective, such centralized decision-making jeopardizes the resilience of a system if the source becomes unavailable. To reduce the dependence on a single entity, consensus protocols include all nodes in this process [BLG15, ANDL17b, PANL19] and make sure that essential information is available throughout the network [FZMT13]. However, as all existing consensus protocols still rely on a designated leader for synchronization, there is a need for a fully distributed consensus mechanism without any centralized service.

Link failure. As wireless links are known to be fluctuating and prone to interference, link-layer parameters can be adjusted on the fly to increase their robustness [PL21, BGS+22, YNB+22]. Some protocols even use a combination of communication techniques to temporally mitigate the influence of unstable links on one of them [MZL+20, SBDM20, IIPK19]. However, most protocols lack the capability to fundamentally change their network topology while preserving ongoing communication, such as when a network splits apart or clusters rejoin after a prolonged separation. Therefore, there is a need for clusters to autonomously and instantly react to network splits and merges without having to revert to bootstrapping first and cease communication in the meantime.

Node failure. The integrity of a system is often heavily challenged by node failures. For example, finding consensus requires input from all nodes and may otherwise block or become ambiguous [ANDL17b]. Even for protocols that are designed to cope with failures [MZL+18, PANL19, MZL+20, FZMT13], node failure at the leader can prevent reliable communication due to a loss of synchronization. For this reason, nodes must usually halt their data exchange until a new leader could be established [ANDL17a, FZMT12], which causes a catastrophic collapse of the network. As adversarial node failures may repeatedly delay the election of failover leaders and as failures are often correlated [WC19], thereby arbitrarily prolonging the network interruption, such systems cannot maintain a robust communication service. Therefore, there is a need for protocols that preserve safe and persistent communication for the remaining nodes in the cluster despite any type of failure.

1.3.3 Efficient Resource Usage

Energy-intensive communication. While low-power wireless technologies such as IEEE 802.15.4 and BLE have become highly energy-efficient, they are primarily limited to short-range communication. More specialized radios such as UWB for wireless sensing and LoRa for long-range communication induce significantly higher energy costs as they utilize more bandwidth and time per packet. Infrastructure-based systems have been able to alleviate some of these concerns through asymmetry by offloading the majority of the energy costs from the sensor nodes. For example, UWB is exclusively utilized with fixed installations to measure relative distances [GSR+19, PKD18] due to its high listening costs. This dependence on infrastructure is in direct conflict with a system's autonomy as it remains restricted to instrumented spaces. Similarly, LoRa gateways can continuously listen for incoming packets and may also schedule bidirectional communication [TDFB+23]. However, such systems cannot obviate the high intrinsic energy costs of a long-range packet exchange as long as they lack local filtering. Therefore, there is a need to extend such radios with onboard capabilities to focus their energy on when it is imperative to enable the technology and thereby boost their efficiency.

Mobility. Mobility endangers efficient communication, as it can cause nodes to search for disappeared leaders [FZMT12] or require the constant re-organization of routing structures that may take up to 25 min [LEG20]. In recent years, the introduction of concurrent transmissions (CT) has largely displaced tree-based routing by permitting reliable, efficient networking that is entirely topology-agnostic and hence perfectly suited for highly mobile nodes [ZMS20]. Instead of individually routing packets hop-by-hop, standard CT floods a packet concurrently through the network by having all recipients retransmit it simultaneously [FZTS11, LDFST17]. However, flooding by default requires all nodes to participate and spread messages throughout the network, even though a particular packet might only be of interest to a small subset such as the data sink. While methods have been developed to optimize this behavior for static networks [IMPR16, TDFB+23], networks with a constantly changing network topology spend a significant portion of their energy in vain. Therefore, there is a need to either leverage that flooded information is available everywhere to increase its effectiveness or restrict communication to nodes that are relevant to the current exchange.

Event-based communication. Most networks structure their exchange in periodic rounds [BvRW07, MELT08, FZMT12], as opportunistic communication is unpredictable and often does not scale well [Abr70]. Although protocols adapt their duty cycles or limit expenses in times of low activity [SDFG+17, SSB10], periodic overhead is pervasive. While such activity consumes energy without exchanging sensor data, it additionally occupies the channel and prevents other communication. Lastly, there is an inherent trade-off between the duty cycle of such protocols and the latency with which activity can be reported [SFBT19], resulting in lower energy consumption at the cost of a longer reaction time if the round period is increased. Additional hardware such as wake-up radios [SBBT15] and heterogeneous combinations thereof [PMMB18, BPS16] boost reactivity, but also require extra complexity and energy. Therefore, there is a need to only communicate based on an event when the demand to transmit sensor data exists and otherwise eliminate communication costs entirely.

1.4 Thesis Outline and Contributions

In the following, we present the main contributions of each individual chapter of this thesis as illustrated in Figure 1.1. We start in Chapter 2 by developing a system that permits nodes to autonomously form a network with a common leader while actively communicating and sensing in parallel, which we achieve through a heterogeneous hardware architecture that prevents cluster formation from interfering with coordination. As a network loses its coordination when the leader becomes unavailable due to changes in the network topology, we then improve the resilience of such network orchestration in Chapter 3 through the integration of coordination and discovery into a single protocol that efficiently finds consensus on leadership when the network splits and joins. However, as orchestration is only required when data exchange is actually needed and is otherwise redundant, we next explore in Chapter 4 how an ad-hoc network may only locally trigger leader election and additional networking based on sensed events. Lastly, we investigate in Chapter 5 how decision-making can be completely decentralized without requiring a leader at any point in time, thereby achieving fully distributed network coordination.

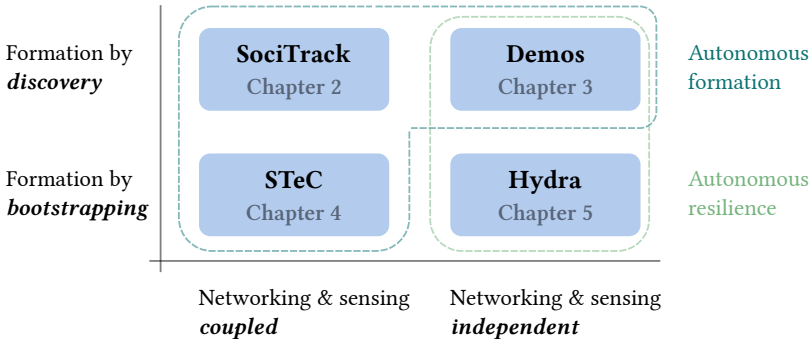

Figure 1.1: In this overview of the chapters and aims of this thesis, we distinguish chapters along two axes: How nodes form a network (either directly during bootstrapping or thereafter using discovery) and how networking and sensing are connected (as both can be conducted independently or directly coupled).

Infrastructure-Free Tracking through Mobile Sensor Networks (Chapter 2)

With SociTrack, we introduce a novel dual-radio architecture that separates cluster coordination and cluster formation in hardware to enable independent operation. This parallelization permits us to combine energy-efficient BLE discovery to build dynamic clusters with UWB to track highly mobile nodes in a constantly changing network topology. With this new platform, we offer autonomous social interaction tracking via wireless distance measurements without any supporting infrastructure. Unlike other systems, SociTrack does not have to choose between deployability and measurement fidelity. By decoupling the discovery and synchronization of other nodes from sensing, we are still able to employ highly accurate ranging and provide sub-second, decimeter-accurate ranging information.

Robust Orchestration for Autonomous Networking (Chapter 3)

With Demos, we present the first low-power wireless protocol that provides robust network orchestration for autonomous multi-hop clusters. Key to this robustness is Demos' ability to quickly find a new leader in case the previous coordinator becomes unavailable. We achieve such high adaptivity by introducing a set of novel election quorums to reliably re-elect the leader while it is available and instantly find consensus on new

leadership once the network topology changes. We further integrate a new, flexible cluster discovery scheme that efficiently finds neighboring clusters and swiftly merges them without disrupting the exchange of data. Demos is the first protocol that combines cluster coordination and cluster discovery to interchange and mutually leverage knowledge on connected nodes, thereby creating a symbiosis of both mechanisms.

Exploiting Spatial and Temporal Correlation for Event-based Communication in WSNs (Chapter 4)

With STeC, we propose to incorporate event detection based on distributed sensors as an integral part of network orchestration. Instead of periodically exchanging data, nodes only activate their radios following a sensor trigger and form an ad-hoc cluster solely out of sensor-activated nodes in spatial and temporal proximity. By locally electing a leader out of the subset of nodes that have co-detected an event, coordination is limited to affected nodes and any periodic communication overhead can be eliminated. STeC is unique in that it thereby proposes a fundamentally new approach to the elementary conflict between duty cycling and reactivity. It further ensures that significant energy is only spent on events that are deemed relevant through additional local filtering. As its ad-hoc clusters only communicate to exchange data on the current event and disband thereafter, the discovery of other clusters can be avoided without impacting the system's integrity.

Concurrent Coordination for Fault-tolerant Networking (Chapter 5)

With Hydra, we introduce a protocol for low-power wireless networks that entirely decentralizes its network coordination through a novel distributed consensus and synchronization mechanism. Instead of having only a leader collect information and make decisions, all participating nodes exchange their current information on the set of nodes that they consider to be part of the network and the corresponding traffic demand. Based on a new consensus algorithm that provides provably unique decisions, network coordination remains consistent despite arbitrary node failures or network topology changes. As it completely avoids centralization and performs coordination tasks such as synchronization and scheduling concurrently, Hydra's concept of concurrent coordination constitutes the first fully distributed protocol to enable fault-tolerant low-power wireless networking and ensures safe, adaptive, and persistent communication.

2

Infrastructure-Free Tracking through Mobile Sensor Networks

Historically, localization and tracking has been a domain that has relied heavily on static infrastructure for reference points and network orchestration. This dependence on infrastructure severely limits the capabilities and autonomy of such systems, as their coverage is restricted to instrumented spaces and hence challenging to scale. In this chapter, we specifically address the problem of interaction tracking, where nodes within proximity trace each other and exchange information such as their pairwise distance. Typical example scenarios include the coordination within a swarm of drones as well as the tracking of human interactions to study social bonds and the spread of diseases. Until now, systems measuring interactions have been forced to choose between deployability and measurement fidelity—they operate only where infrastructure is available, under line-of-sight conditions, or provide coarse-grained proximity data. Therefore, a novel networking approach is needed that enables interaction tracking without infrastructure support, captures highly mobile individuals, and is energy-efficient to enable long-term deployments. In particular, dynamic clusters of sensors should be capable of reliably capturing all pairwise distance information with high fidelity. While we will again address the challenge of dynamic cluster formation in Chapter 4 based on sensor events, this chapter first explores how networked mobile sensors can autonomously form dynamic clusters to provide entirely infrastructure-free deployments.

This chapter is based on [BJT+20], [BPD19], and [Bir18].

We introduce SociTrack, a highly flexible platform for autonomous interaction tracking via wireless distance measurements. Based exclusively on the node-to-node communication within a network of mobile sensors, the system provides sub-second, decimeter-accurate tracking information over multiple days. The key insight that enables both deployability and measurement fidelity in one system is to decouple node mobility and network management from energy-intensive wireless sensing, resulting in a novel dual-radio architecture. SociTrack leverages an energy-efficient and scalable ranging protocol that is accurate to 14.8 cm (99th percentile) in complex indoor environments and that enables our prototype to operate for 12 days on a 2000 mAh battery. To validate its deployability and efficacy, SociTrack is used by researchers in early childhood development to capture interactions between caregivers and infants.

This chapter builds on [BJT+20] and [BPD19], which is partially based on the author's MSc thesis [Bir18]. The text and illustrative figures, the related work analysis, the described hardware, and many of the network discovery features and experimental results are a product of the author's doctoral studies and a prerequisite and stepping stone for the forthcoming chapters.

2.1 Introduction

A desire for networked high-fidelity tracking and interaction data — sub-meter accurate distances at sub-second sampling — can be found in a variety of applications. Drones can be used to localize human victims in a search-and-rescue scenario [CXC+19] or can leverage known pairwise interaction distances within the network to avoid collisions [GET22], coordinate a search grid [WTJ09, WH11], improve network coverage [HYM16], and fly in formation [LSD+20]. Further demands for rich interaction data include applications researching interracial and intercultural relations [MBYP09, Bea04], transit design [EW07], and workplace dynamics [MNMS17, KMO+12]. In our evaluation, we consider the successful deployment of this technology by childhood development researchers, who are seeking a better understanding of the impact of physical proximity between caregivers and infants [dB19, Zea18, SPH+22]. Hence, technologies and tools that flexibly facilitate proximal interaction tracking are likely to find many applications which differ in key parameters like detection latency, update frequency, spatial resolution, network density, and system lifetime. This breadth of applications motivates the design of an adaptable, general-purpose interaction tracking system.

Challenges. The biggest challenge in designing a measurement system for high-fidelity interaction tracking is meeting *in situ* deployment requirements. It must operate *wherever* a cohort travels, which precludes relying on supporting infrastructure (including so-called ambient infrastructure such as WiFi, cell towers, or Global Navigation Satellite System (GNSS)). Due to the highly dynamic nature of a mobile network, the system must continuously discover and form ad-hoc clusters to guarantee up-to-date information on connected nodes and ensure that all pairwise interaction data is collected, which requires a coupling of networking and sensing. Finally, a suitable system should work reliably without requiring manual interventions and be low-power to allow for lightweight, unobtrusive operation without the need to recharge during a long-term deployment.

In summary, we identify four challenges for networking and sensing:

C_{FIDELITY}	Sub-meter accuracy and sub-second sampling
C_{MOBILE}	Fast detection and frequent network changes
C_{INFRA}	Infrastructure-free (in situ) usage
C_{DEPLOY}	Reliable, unobtrusive, and long-lasting operation

We will encounter these four challenges again throughout this dissertation, thereby paying particular attention to one of them in each chapter. This chapter focuses on C_{FIDELITY} and enables high-fidelity sensing in an autonomous network. Chapter 3 centers on C_{MOBILE} and improves the reactivity to network splits as well as the communication between clusters of nodes, for which continuous discovery will also be a key element. Chapter 4 takes up C_{INFRA} again and demonstrates that the autonomy of sensor nodes may even be increased when infrastructure can be leveraged to offload energy costs, whereby we also couple networking and sensing. Lastly, Chapter 5 concentrates on C_{DEPLOY} and ensures that persistent, fault-tolerant networking is guaranteed under arbitrary circumstances.

System requirements. As the potential application space for an autonomous tracking system is broad, we start by distilling a common set of requirements. First, the collected data must be high-fidelity. While exact requirements are unique to specific applications, systems that can collect interaction data with sub-meter accuracy at sub-second sampling (C_{FIDELITY}) have been called for across various domains [BSLP13, Zea18]. Sensor nodes must be able to move freely, which requires them to form dynamic networks and continuously discover and update cluster membership throughout the deployment (C_{MOBILE}). To be able to drop into an unknown environment, a system cannot require deploying infrastructure nor can

it make assumptions about existing infrastructure (C_{INFRA}), which makes traditional instrumentation-based techniques not only impractical but inadequate. To measure reliably and sense natural behavior, the measurement system cannot rely on orientation (i.e., require line-of-sight between sensors) and must only have a low impact on the wearer (C_{DEPLOY}).

Status quo. No prior interaction tracking system fulfills all four requirements. Many systems build on low-power narrowband radios, such as IEEE 802.15.4 or more recently BLE, as these technologies have minimal deployment constraints and offer established solutions for infrastructure-free and highly mobile operation [API+11, CVdBB+10, MPGG+17, MFL+14, MNMS17, OWK+09, OGT+18, SLB+17]. However, narrowband radios have fundamental fidelity limits [Lan09], and in practice, these systems (even if leveraging information such as the Received Signal Strength Indicator (RSSI)) are restricted to coarse "near/far" estimates. In contrast, a plethora of indoor localization systems provide high-fidelity tracking, but they rely on infrastructure and cannot observe cohorts outside of instrumented spaces [LRS+15, GSR+19, SHH11, KPCD16, KPD15]. One recent system, Opo [HKPD14], achieves fidelity and mobility without infrastructure, but is restricted by the use of ultrasound to capture only short-distance (3 m), line-of-sight interactions, which imposes severe deployability restrictions.

Contribution. We introduce a new platform, *SociTrack*, designed for high-fidelity interaction tracking, that emphasizes autonomous operation across a range of scenarios. SociTrack is self-contained, with no requirement of infrastructure support or limitations on deployment methodology. The platform is designed to capture interactions among highly mobile sensor nodes. It builds dynamic networks capable of collecting pairwise distance information between all cohort members at up to 16 Hz with 14.8 cm 99th percentile accuracy. SociTrack can be worn by people of all ages during their normal day-to-day routines for up to two weeks. Further, the platform aims to constrain deployment scenarios as little as possible: it presents users with parameters to configure it to their needs, tools to predict its performance accordingly, and interfaces for real-time inspection and adaptation. With SociTrack, we make the following core contributions:

- Our design exploration in Section 2.4 shows that any single communication technology cannot provide high-fidelity, infrastructure-free interaction tracking of mobile nodes with high deployability on its own. To realize such a system, we introduce a heterogeneous architecture exploiting the individual strengths of BLE and UWB.

- We improve state-of-the-art UWB ranging protocols on two key dimensions: Section 2.5 explains how we reduce message complexity from quadratic to linear with regard to network size and shows that careful scheduling of broadcast packets can further result in a 50 % reduction of messages, which increases lifetime by 102 %.
- We demonstrate a configurable protocol that enables flexibility at deployment time with predictable, deterministic performance. Section 2.6 shows that this adaptivity permits usage across a breadth of applications and promotes exploration.

In Section 2.8, we evaluate SociTrack's performance on various micro-benchmarks and consider its efficacy for real-world studies of domain scientists by deploying it with psychology researchers for an infant tracking scenario [SPH+22]. Finally, we close with a consideration of upcoming hardware advances. While we develop and test SociTrack using custom hardware, Section 2.9 discusses emerging smartphones and wearables and shows how future global deployments could be as simple as a software update.

2.2 Motivation for Human Sensing

The need for high-fidelity interaction tracking data is well-known, but it might now be more pressing than ever when facing the impact of a global pandemic. In general, the study of proximal interactions is crucial for human sensing in many settings. Social scientists believe that changes in interaction behavior are an important indicator of cognitive decline and the onset of Alzheimer's disease [CZ10]. Patterns of social interactions are used to study informal informational networks and the underlying power dynamics in a workplace [OWK+09]. And of course, close physical interactions are of medical interest as they are a critical factor in the spread of diseases like SARS-CoV-2 [CAD+20, SPDG+20, MC20].

However, scientists lack *high-fidelity* data on contact networks in the wild [AZU18]. Current wide-area social interaction studies have to rely almost exclusively on self-reporting and public surveys [HKPD14, MFB15], which are highly subjective [MNMS17], subject to bias [BMW+16], and lack fine-grained quantitative metrics [OWK+09]. As a result, public health policy guidance on social distancing is based on limited studies taken in small populations and highly controlled settings. Due to these constraints, we lack an understanding of spreading distances at scale in real-world

environments to obtain improved, empirical models and take effective countermeasures.

In this chapter, we focus on the problem of long-term tracking of interaction distances among members of a cohort. These measures are important in the study of stress [EW07], human-robot interaction [WDTB+05], epidemiology [RLS+07], and child development [Zea18]. While small-scale experiments allow the collection of high-fidelity interaction data in laboratory settings, conscious observation can strongly influence behavior [CGR+09, TLKL+17] and consequently cause scientists to miss critical cases such as neglect of an infant by their caregivers [Zea18] or lead to the collection of data that misses the primary effects of caregiver-infant attachment relationships [dB19]. Therefore, it is critical to be able to measure behavior in people's natural environments, where interactions occur organically [AZU18, RDC18].

Example scenario. Early childhood development psychologists are interested in studying the interaction patterns of caregivers and infants [Zea18]. In proposed experiments, the reliable capture of short-lived interactions (e.g. check-ins) as well as sub-meter accuracy is important to infer behaviors (c_{FIDELITY}). Members are expected to continuously split up and rejoin over the course of the day (c_{MOBILE}), such as when family members go to work and school, pursue individual free-time activities, or do the chores together. Measurements must follow clusters of family members wherever they go (c_{INFRA}), covering environments that might be impossible to instrument due to their public nature, unavailability of electricity and connectivity, or sheer extent. Finally, devices must operate autonomously and reliably without requiring manual adjustments and should remain unobtrusive to participants (c_{DEPLOY}) to minimize the Hawthorne effect, where people change their behavior once they are conscious that they are under observation [Zea18, Gar00]. Section 2.8.1 evaluates SociTrack's performance in this and other real-world application scenarios.

2.3 Related Work

Interaction tracking and related technologies have a long history in mobile computing research. The primary differentiation of SociTrack is realizing high-fidelity tracking (sub-meter, multi-Hz) with no constraints on deployed infrastructure (i.e., anchors) or sensor placement (i.e., line-of-sight). Table 2.1 summarizes how SociTrack compares to existing systems and addresses the needs of various target application scenarios.

Platform	Ranging method	Infrastructure	Size [cm^2]	Spatial resolution [cm]	Temporal resolution [s]	Maximal range [m]	Lifetime [d]
WASP [SHH11]	NB ToF	Yes	N/A	50	0.04	30	0.42
SurePoint [KPCD16]	UWB ToF	Yes	21.8	50	0.08	50	0.04
iBadge [CMY+02]	UL/RF TDoA	Yes	38.5	10	N/A	3	0.21
Opo [HKPD14]	UL/RF TDoA	No	14.0	5	2	3	3.9
Sociometric [WAO+11, KMO+12]	RSSI + IR	No	~50	100	2	10	1
Social fMRI [API+11]	RSSI	No	N/A	500	300	N/A	N/A
Smartwatches [MNMS17]	RSSI	No	21.2	600	10	6	0.71
SociTrack	**UWB ToF**	**No**	**21.4**	**15**	**0.06**	**64**	**12**
Scenario							
Office – Employee workplace dynamics		Yes		50	2	20	5
Elderly care – Habit monitoring		Yes		500	30	10	14
Family – Caregiver-infant patterns		No		20	0.5	10	7
Sports – Player tracking and analysis		Both		20	0.2	50	0.08

Table 2.1: Previous systems cannot cover the complete design space, particularly lacking in infrastructure-free solutions with high spatial resolution. SociTrack provides an adaptable solution that can be configured for various scenarios and which offers a wide trade-off space between network size, spatial and temporal resolution, and lifetime, explored in detail in Section 2.6. Note that the given performance figures showcase the individually obtainable results of target scenarios, as discussed in Section 2.8.3, and are not all obtainable simultaneously.

C_{FIDELITY}: *Interactions are more than proximity.* Many systems either provide only coarse-grained (often 2-5 m) distance resolution between members through signal-strength measurements [API+11, KLS+10, OWK+09] or simply report nearby presence [CVdBB+10, MFL+14]. Modern versions are often Bluetooth-based and highly deployable as they leverage smartphones and wearables [MNMS17]. While appealing in their accessibility, these proximity-based systems provide limited insight into the nature of interactions [SVB+11, SLB+17], which requires sub-meter accuracy [BM13, WDTB+05]. For example, they cannot provide key spatial insights such as the significance between 1-2 m distancing efforts for disease spread [BSLP13].

C_{DEPLOY}: *Technology limits interaction capture.* Some systems provide finer-grained distance estimation by leveraging the propagation speed of waves. However, the underlying technology can impose strong limitations on the deployability of a system. For example, Opo [HKPD14] exploits the time difference of arrival (TDoA) between an ultrasonic chirp and an IEEE 802.15.4 radio packet to achieve 5 cm ranging accuracy. However, ultrasonic signals are challenging in complex, indoor environments, as reflections can persist in the channel for over 50 ms [LRS+15] and therefore limit the sampling frequency. Reflections also reduce effective range: because Opo cannot distinguish the direct path from a reflection, it assumes all interactions beyond 3 m are reflections. Similar multipath issues impede other acoustic solutions [PSZ+07]. Finally, ultrasonic transducers are highly directional; in Opo, errors grow once the sensors are more than 45° off of direct line-of-sight. Because measurements rely on the successful reception of ultrasonic chirps, objects that obstruct line-of-sight, such as other people or wearing the sensor inside a pocket, prevent ranging altogether. In contrast, SociTrack's UWB-based solution is resilient to multipath interference and permits high-frequency, orientation-independent ranging despite non-line-of-sight.

C_{INFRA}: *Localization tied to infrastructure.* While not originally targeted at interaction tracking, many indoor localization systems can provide the needed accuracy from detailed position estimates. As narrowband radios are limited in fidelity and acoustic solutions are restricted in deployability, we focus on the localization systems that also use UWB for its accuracy and deployability potential. SnapLoc [GSR+19] emphasizes scale by supporting unlimited tags, SurePoint [KPCD16] highlights long-tail accuracy by introducing diversity ranging, and RFind [MSA17] and Slocalization [PKD18] reduce tag power to nanowatts via backscatter

techniques; but all of these systems require fixed infrastructure in every room to operate. A common trend in localization system design is the use of asymmetry: to improve performance and the size, weight, and power (SWaP) characteristics of nodes, complexity is pushed into infrastructure. Our applications, however, cannot afford this trade-off.

c_{INFRA}*: "Infrastructure-free" localization.* Several localization systems do not require additional deployment of infrastructure as long as they can take advantage of ambient infrastructure, such as WiFi [KGKR14, VKK16, XXLJ19]. While leveraging existing infrastructure can mitigate deployment costs in certain home and business settings, it is still an incompatible constraint for wide-ranging interaction tracking applications. For example, infant interaction tracking must operate in places where WiFi may be nonexistent, such as in the car, during a picnic at the park, or on a hike. While some early ideas leverage new wide-area infrastructure such as LoRa [IIKN19], evidence from both indoor and outdoor cellular localization suggest that neither coverage nor fidelity will be sufficient [RLS08, SDB16, Ope20]. Ultimately, to ensure reliable operation wherever a cohort may travel, an interaction tracking system cannot rely on infrastructure, neither intentionally deployed nor opportunistically used.

c_{MOBILE}*: Avoiding stochastic operation.* Unsynchronized networks require no coordination but use the available channel inefficiently and are limited in scale [Abr70]. Such stochastic operation is what limits Opo [HKPD14] to 2 s sampling. Systems like ALPS [LRS+15] and UPS+ [LAY19] demonstrate that it is possible to use the ultrasonic channel to realize sub-meter and multi-Hz tracking fidelity, but require infrastructure support to schedule ranging events. SociTrack is the first system to support all fidelity, deployability, and mobility challenges without relying on supporting infrastructure.

2.4 System Design

The goal of SociTrack is to provide high-fidelity interaction measurements while minimizing constraints on how and where the system can be deployed. First and foremost, SociTrack needs a reliable mechanism to measure the distance between cohort members. Across the array of acoustic, optical, and wireless ranging technologies, only radio-based measurement techniques allow sensors in pockets, cope with cluttered environments, or register room-wide interactions. Of the radio-based methods, only UWB

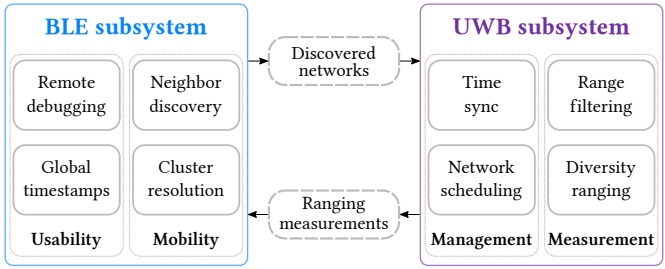

Figure 2.1: Task separation between BLE (left) and UWB subsystem (right). While the former handles external links for real-time data offload and neighbor discovery, UWB is used for network communication and distance estimation.

provides an accurate and robust ranging primitive [Lan09]. The historic drawback with UWB is its energy inefficiency in the face of mobile sensors.

Our key design insight is that while UWB is necessary to achieve high-fidelity ranging, we can mitigate its deficiencies on a modern platform by complementing it with another technology. A decade ago, an additional radio IC might have doubled the cost of a system—today, a Bluetooth chip and supporting passives represent just 8.7 % of our prototype bill of materials (BOM) costs. With access to multiple technologies in one system, the central design question then is how to best apportion tasks across the available subsystems. Naively, one might simply use the comparatively energy-expensive UWB radio exclusively for ranging (where there is no alternative), and Bluetooth for everything else. However, as we will show in this design discussion, a blended approach in which responsibilities are shared results in a more capable overall design.

2.4.1 Why is Mobility a Struggle for UWB?

UWB radios are generally not used in infrastructure-free designs with mobile nodes. The issue lies in UWB's high receive power draw, which results from the need to integrate energy below the noise floor over a wide bandwidth and the necessity to heavily process and despread the received signal [TSZ+07]. Discovery protocols require efficient listening however [DC08], and benefit from a balanced energy budget between transmission and reception of packets [QLXL16]. The UWB radio [Dec18] used in SociTrack averages up to 554.1 mW while listening, 24 × more power than its BLE radio [Nor20] at 22.7 mW. Furthermore, the receive (RX) and transmit (TX) power draw is less balanced; the RX/TX ratio of UWB

is 4.0 compared to 1.5 for BLE. This high, unbalanced energy consumption makes UWB particularly ill-suited for discovery, as a UWB-only system is forced to sacrifice lifetime or discovery latency. But quick discovery is critical, as short events such as a caregiver briefly checking in on an infant will be missed otherwise. By employing BLE instead of UWB, we can reduce discovery costs by 93 % and enable long-term mobile deployments.

2.4.2 SociTrack Protocol Design

Figure 2.1 gives an overview of the major components of the SociTrack protocol and how SociTrack partitions tasks. Before any devices are discovered, only the BLE subsystem searches for neighbors and the UWB subsystem is powered down. During subsequent steady-state operation, both the BLE and UWB subsystems remain active, as they handle different aspects of network maintenance. Designing a protocol atop two radio technologies can be both beneficial, as they operate independently without mutual interference (a concern we will have to approach in Chapter 3), and challenging, as the availability of BLE and UWB links between nodes may vary due to different propagation characteristics in complex, indoor environments. We will encounter similar design considerations in Chapter 4, where we will design a protocol atop two radio modulations with individual strengths and weaknesses.

Initial discovery. We must first design a discovery mechanism. Less mobile systems can treat discovery as a "startup" problem, a rare event before steady-state operation. However, in highly mobile settings, nodes can frequently switch networks and encounter new neighbors, thus discovery must occur continuously. Additionally, nodes may spend significant periods isolated from others, such as when an instrumented family member is at work, and must be able to rely on an efficient discovery mechanism to maintain lifetime. Finally, we also leverage the ubiquity of BLE on commercial devices to enable in-field inspection and debugging, requiring the discovery protocol to be compatible with standard BLE advertisements.

We select BLEnd, a state-of-the-art BLE discovery protocol, as it delivers highly predictable performance and supports deployment-time adaptation to trade-off discovery latency and energy use [JLMP17]. Off-the-shelf BLEnd assumes an isolated network communicating only with other BLEnd nodes and therefore exclusively sends standard *un*connectable BLE advertisements that it detects with periodic scans. With SociTrack, we are also interested in supporting communication with consumer devices (e.g. using smartphones for monitoring) and therefore opt for connectable

advertisements. In addition to real-time remote debugging, we leverage this gained connectivity to directly inform the sender of an advertisement of neighbors and achieve simultaneous bi-directional discovery. This enables us to halve the number of advertisements and reduce energy consumption by up to 50 % compared to the original design while maintaining the same discovery latency. We further incorporate empirical characteristics such as radio start-up costs into the design to improve the efficacy of our discovery optimization algorithm.

Cluster formation. Once nodes have discovered each other, they seek to form (or join) a cluster. To bootstrap such a local network, two orphans (i.e., nodes that have not yet joined a network) discovering each other will form a cluster and elect the node with the higher ID as the leader. Discovery advertisements are then updated to indicate an active cluster and include the current cluster lead's ID. If an orphan encounters an existing cluster, it will join as a regular member even if it has a higher ID to avoid churn.

Steady-state operation. Figure 2.2 shows initial discovery, followed by steady-state operation. During steady-state, the BLE radio continues to perform discovery to handle subsequent joins and departures. In addition, the UWB radio begins to manage cluster operation. The leader disseminates a schedule to all nodes, cluster members perform the scheduled ranging events, and then the cluster handles potential membership changes. Section 2.5 will examine ranging in detail, so it is sufficient to treat ranging as a black box for now.

Our network design is inspired by Glossy [FZTS11], the current state-of-the-art protocol for reliable multi-hop network communication among mobile nodes. Glossy uses concurrent transmissions (CT) to reliably synchronize and flood information through a network. Retransmission timing is controlled directly by the reception of other packets, which eliminates topology-dependent state and thus makes it well-suited to highly mobile sensors. While Glossy was originally implemented on top of IEEE 802.15.4, it has also been shown to work for both BLE [ANDL19] and UWB [KPCD16] networks. We will encounter CT in all chapters of this dissertation to provide reliable and topology-agnostic communication.

SociTrack employs flooding using Glossy on the UWB radio during the `Schedule` and `Contention` phase, seen in Figure 2.2. In principle, only the ranging strictly requires the higher energy radio. However, it is crucial that the ranging events are precisely scheduled, and establishing an accurate shared clock between the BLE and UWB radios would be non-trivial. More importantly, UWB and BLE can exhibit different propagation in complex,

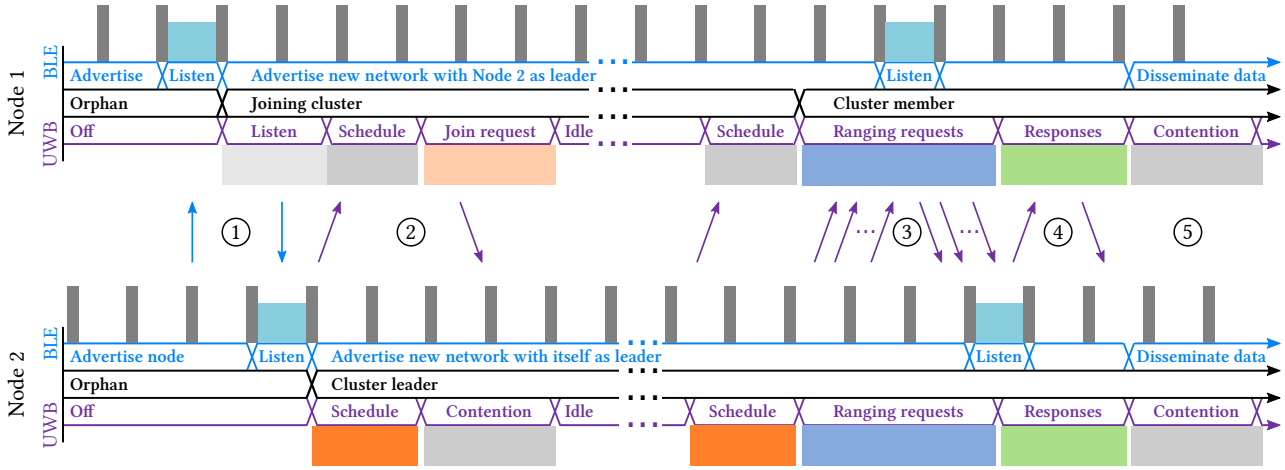

Figure 2.2: (1) Two nodes initially discover each other by periodically transmitting BLE advertisements and detecting the presence of other nodes and networks when listening. (2) Once a leader is determined, it initiates a UWB round consisting of schedule distribution and contention access so other nodes can register. Next, requesters and responders are scheduled and start ranging with each other. In contrast to prior diversity ranging protocols, ranging requests (3) and responses (4) are separated in time and occur entirely deterministically using broadcasts. (5) The gathered range information can then be disseminated to other devices such as laptops and smartphones over Bluetooth.

indoor environments. Disseminating the schedule on the same radio used for ranging ensures that all devices within ranging distance will be able to perform measurements, even when lacking BLE connectivity.

Re-discovery. Because the UWB network is based on flooding and the BLE network is point-to-point, they have different connectivity graphs. This means that when a new node discovers a cluster via BLE, it must somehow let the cluster *lead* know of its presence, even though it may not be in direct range of the leader. To address this issue, the cluster lead schedules a contention-access window in each UWB round during which new nodes may flood requests. The same contention-based technique will also be used in Chapter 3 to notify the cluster lead of discovered clusters. Naively, it would be highly energy-expensive for an orphan node to continuously listen on its UWB radio for a potential contention window. However, because it was first initiated through BLE discovery, the consequent UWB registration mechanism succeeds immediately.

Adaptive contention. Reactivity is essential for mobile networks, which requires SociTrack to gracefully handle the case when many nodes appear in a short interval. For example, during a merge operation, ranging is effectively paused for nodes of the joining cluster until they are scheduled in the new cluster. To mitigate substantial measurement gaps, the cluster lead adjusts the contention duration depending on the count of join requests in the previous round using multiplicative gains (by doubling, maintaining, or halving the number of available contention slots). This mechanism manages to quickly satisfy a spiking network demand, as seen in our evaluation in Section 2.8.3.

Cluster interference. A ubiquitous problem in traditional WSNs is collision avoidance between two active clusters, as nodes usually listen only when scheduled to reduce energy consumption. SociTrack's heterogeneous radio architecture is uniquely suited to effortlessly detect and resolve such a conflict. During steady-state, inter-network communication occurs exclusively over UWB. The BLE subsystem meanwhile continuously attempts to discover new nodes. When a cluster member discovers another active cluster with a higher leader ID than its current cluster, it will attach to the new cluster and start advertising it. Because this discovery occurs out-of-band, we enable collision detection and cluster resolution without influencing ongoing measurements. In Chapter 3, we will extend this concept and introduce a cluster discovery mechanism that permits entire clusters to join simultaneously, enabling even quicker merges.

Detecting departures. In a multi-hop network, it is often the case that a node is not in one-hop range of the leader. When flooding packets, all nodes retransmit identical payloads concurrently, so it is impossible for the leader to detect whether a specific node failed to participate during schedule dissemination. During the ranging events (which are detailed in the next section), packets do contain IDs; however, these packets cannot be flooded as they are used for time of flight measurements, so the leader cannot observe ranging packets of a node that is more than one hop away. As a consequence, the leader cannot easily detect whether a node may have left the cluster. For this reason, nodes are automatically dropped from the schedule after a fixed number of rounds and must re-register in a dedicated contention-access window (see Figure 2.2). Cluster members can track their own network dwell time and proactively notify the leader when they are close to expiration. If a former leader deschedules all its members, it becomes an orphan again. On the other hand, when the leader leaves the network, we avoid bootstrapping a new network from scratch by promoting the second-highest node to automatically take over if too many rounds have passed without receiving a schedule from the leader. However, we will see in Chapter 3 and Chapter 5 that such a failover strategy cannot maintain persistent communication for arbitrary node and link failures and will improve autonomous resilience through explicit leader election and the decentralization of cluster coordination.

External communication. BLE provides the unique advantage to permit connections to many commercial devices such as laptops and smartphones. We leverage this ability to disseminate measurements in real time for deployment monitoring and in-situ remote debugging. Additionally, we exploit the time-awareness of such devices to enrich our data with global timestamps which a leader distributes inside the network to facilitate data analysis.

2.5 Ranging Protocol Design

Pairwise ranges compose interactions, which makes their accurate and efficient collection critical. Optimizing the performance of the UWB channel for high-fidelity ranging is well-studied in prior work [KPCD16]. Instead, our protocol contributions focus on making collecting ranges efficient in the absence of powered infrastructure nodes, reducing redundant messages, and incorporating flexibility depending on available energy.

Overview. The goal of the ranging protocol is to get an accurate estimate of the time of flight (ToF) of radio packets between every pair of nodes in a cluster. In the simplest case, one node starts a ranging event by transmitting a *request* to which another node *responds*; based on the timing of the response and its contents, the *requester* can estimate the ToF to the *responder*.

Diversity improves ToF ranging. To reduce variance in ToF estimates, state-of-the-art UWB ranging protocols [KPD15, KPCD16] employ diversity by *requesting* on multiple physical channels. This diversity is achieved by alternating between different transmit or receive antennas and switching frequencies. In our implementation, each node has three antennas with a polarization offset of 120° and can select from three non-overlapping frequency channels, offering up to $3 \times 3 \times 3 = 27$ individual measurements. These measurements are converted to a single range estimate by taking a system-dependent percentile out of the samples, generating a result that is both accurate and precise [KPCD16]. Note that while the *requester* transmits a series of packets, the corresponding reception energy consumption is much higher at responders. This asymmetric energy cost during a ranging event is one of the peculiarities of UWB as mentioned in Section 2.4.1. Traditional infrastructure-based designs exploit this asymmetry by offloading responding to powered "anchor" devices. As all SociTrack nodes are identical, any combination of devices must be able to perform ranging. They therefore act as *hybrids*, each rotating through both requester and responder responsibilities.

Broadcasts enable scale. For infrastructure-based systems with fixed "anchors", mobile nodes need only range with known reference points to establish their position. Because we assume all nodes are mobile, the relative position to a subset of nodes is insufficient to obtain a full pairwise interaction graph. To ensure coverage, all nodes must therefore range with one another. In a straight-forward unicast design, a cluster with n nodes will require $O(n^2)$ packets to capture all pairwise ranges which would not scale efficiently. Instead, in SociTrack, both requests and responses are sent as broadcasts, which requires only $O(n)$ messages. In contrast to previous approaches in which all nodes immediately respond to a request, we achieve linear complexity by sending aggregated batch responses after all requests have been transmitted. Notice that this is only possible because we apply network-wide scheduling of all packets.

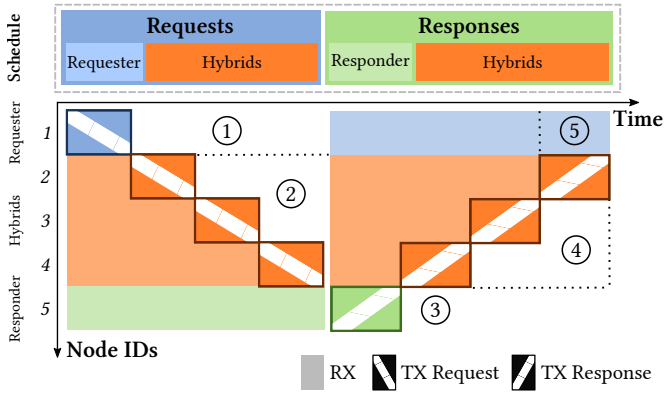

Figure 2.3: (1) Exclusive requesters transmit and then wait to receive responses. (2) Similarly, hybrids stop listening after their own transmission. Therefore, the number of active hybrids is steadily declining. (3) While exclusive responders continuously listen during the requesting phase, they afterward respond and return to sleep first. (4) For responses, hybrids are scheduled in reverse order to balance energy consumption. (5) If nodes have gathered sufficient responses, they can independently stop listening to save energy.

Ranging pyramid. A range is a symmetric measure: for every pair, only one node needs to request while the other responds. We leverage this insight to reduce energy consumption by an additional 50 %. The very first hybrid acts as the requester for all pairs, so it can sleep during all other request transmissions. Consequently, it must listen at the end when all others send their responses. The last hybrid scheduled for requests will be the responder for all pairs, so it must listen to all requests, but can sleep immediately after sending its response. Through this reversing of the scheduling order of hybrid nodes for the responses, we balance energy consumption across the network. This results in a "pyramid schedule" as shown in Figure 2.3. Using our system model introduced in Section 2.6, we find this protocol increases the lifetime of a network of 10 nodes ranging at 1 Hz by up to 102 % compared to the state-of-the-art [KPCD16].

Beyond hybrids. In some deployments, full pairwise connectivity may not be required. For example, if infrastructure nodes ("anchors") can be deployed, they do not need to range with one another. Instead, such a resource-rich node can act as an *exclusive responder*, enabling other nodes to range with them at minimal cost and offloading listening costs to mains-

powered devices. Similarly, if some nodes are severely energy-limited, they may act as *exclusive requesters*, sending requests, but never responding.

2.6 Supporting the Design Space

Previously, we considered the system design from the bottom up, identifying and addressing the key technical challenges. Now, we take a top-down view and consider the axes available to researchers who use SociTrack as a platform for their studies. In our discussions with stakeholders and a literature review [HKPD14, SVB+11], we identify four key dimensions that drive experimental design:

- **Temporal fidelity:** How quickly is a new interaction discovered, how long must it last to be detected, and what is the corresponding sampling rate of distance measurements? *Ranges from sub-second to several minutes.*
- **Ranging quality:** How accurate and precise do the distance measurements between cohort members have to be? *Ranges from decimeter to room-level.*
- **Deployability:** How bulky are the sensors, where can they be worn, and how long must their battery last? *Ranges from infants to adults, and from hours to a month or more.*
- **Usability:** How do we monitor the system's health and data, and how do we recover the data? *Ranges from paid staff on-site to remote access only.*

The next step is to identify how to parameterize SociTrack to expose the desired experimental dimensions, thereby enabling platform adaptation and design exploration. As hardware capabilities are fixed, we cater to deployability constraints (c_{DEPLOY}) and usability considerations from the start and ensure that restrictive SWaP goals and remote accessibility demands are met. We find that through four configuration mechanisms, we can provide the needed experimental design controls:

- **BLE parameters:** Dictate discovery latency. This drives quick network joins and cluster resolution time, at a trade-off with an increased energy use and corresponding lifetime.
- **UWB update interval:** Controls ranging sampling rate, maximal network size, and energy use.

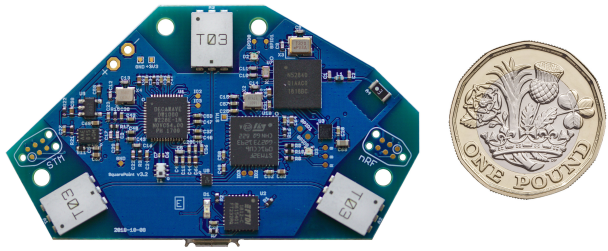

Figure 2.4: SociTrack's compact size makes it easily wearable for all age groups. It further reduces the awareness of being under observation, as it can be hidden underneath clothing. A plastic case shields the board and battery from external influences.

- **Diversity sampling:** Configures how many antennas and frequency channels are used for each range sample, which affects sampling rate, ranging quality, and energy use.
- **Maximum network size:** Bounds the sampling rate and impacts system lifetime, as the duration of an update increases linearly with the network size.

Bridging top-down and bottom-up. Mapping experimental design constraints to a selection of parameters can be challenging and hard to estimate, especially for non-experts. Furthermore, parameter interactions can be subtle. For example, less diversity requires fewer samples per range event which leads to shorter rounds, increases the maximal sampling rate and network size, and reduces energy consumption—but all at the expense of ranging quality.

Therefore, we support pre-deployment exploration of the design space through simulation. For this, we characterize our system using detailed ranging, power, and timing measurements. The resulting open-sourced system model allows scientists to validate system capabilities and tune parameters in advance, as we will demonstrate in Section 2.8.3. This reduces deployment risks, eases adoption, and promotes time-efficient exploration of novel applications in a diverse set of environments. For experimenters, SociTrack shifts trade-off decisions from *design*-time to *deployment*-time.

2.7 Implementation

We implement the proposed dual-radio architecture, which consists of BLE and UWB subsystems, on a custom printed circuit board (PCB) shown in Figure 2.4. The BLE subsystem uses the Nordic Semiconductor nRF52840 as a combination of BLE radio and discovery management microcontroller unit (MCU) and adds a separate real-time clock (RTC) for timestamping, an SD card for permanent storage, and an accelerometer for activity detection. The UWB subsystem employs an STMicroelectronics STM32F091CC MCU as a ranging controller and DecaWave's DW1000 UWB radio.

Figures of merit. As one of our motivating applications includes instrumenting infants, we pay particular attention to the size, weight, and safety of our implementation. The PCB measures 61×35 mm and weighs 7.7 g. We pair this with a 2000 mAh battery, weighing 34.1 g, and package both in a custom-built, 3D-printed case. This results in a compact, portable system that measures $67 \times 48 \times 19$ mm with a weight of 59.8 g.The realized tracker is small enough to be worn on the wrist, on a lanyard around the neck, on a running belt, or inside a shirt or pants pocket. Infants can easily carry them in the front pocket of a specialized vest, as is common for such experiments. Data collection occurs via an SD card as well as remotely by collecting the data in real-time through BLE advertisements sent by the tracker, which can be visualized or forwarded by a smartphone.

Deployment considerations. As SociTrack is designed to be deployed by non-experts in real-world settings, we provide multiple mechanisms for deployment assistance. Each node can be easily monitored visually via two RGB status LEDs that display the states of the BLE and UWB subsystems separately. For more in-depth monitoring and debugging, a BLE interface exposes characteristics which can be interacted with using a smartphone app for Android and iOS as well as JavaScript applications that we built and provide to researchers. The interface further enables experimenters to configure the device on the fly and adjust parameters without requiring direct physical access to hardware in a running experiment, as well as in situ data aggregation and verification.

Research artifacts. All hard- and software artifacts for SociTrack are open source and available on GitHub [BCKP20]. In addition, we provide both our modified version of BLEnd (as described in Section 2.4.2) and a detailed system model (see Section 2.6). The platform is actively used by collaborators for longitudinal research on caregiver-infant interactions [SPH+22].

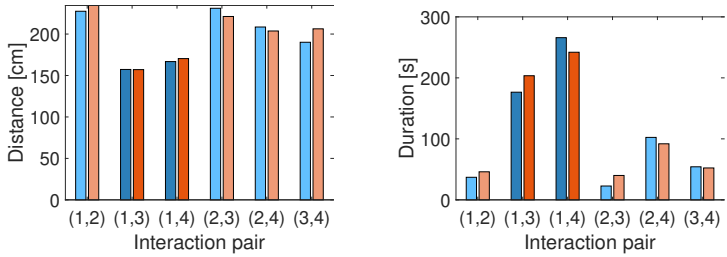

Figure 2.5: Comparing both the average duration and distance, we observe that Individual 1 encounters distinctly longer interactions over a shorter distance with both Individuals 3 and 4. Ground truth diaries confirm that the pair sat together to meet. All other interactions occur in the hallway with significantly larger distances and briefer duration. Notice that interactions are detected reciprocally (blue and orange bar), suggesting high measurement reliability.

2.8 Evaluation

To demonstrate the efficacy of our system, we test SociTrack in both controlled experiments and real-world deployments. We first validate the overall functionality of the platform using three of the application scenarios discussed in Section 2.3. Thereafter, we evaluate the measurement fidelity at varying ranges and show that we achieve decimeter ranging accuracy by leveraging a combination of antenna and frequency diversity. We then demonstrate through simulations using our system model that the system can be easily adapted to diverse scenarios and that our protocol enhancements significantly reduce join latency and message complexity compared to the state-of-the-art. Finally, we investigate SociTrack's deployability and environmental influences on the measurements.

2.8.1 Validating the Deployment Scenarios

Office interactions. To validate the platform with different groups of people, we first conduct experiments with 4 participants over a period of 10 hours, covering a complete working day. During this period, the platform proves to be reliable for data collection. Despite the significant movement of up to 10 people inside a busy open-plan office space spanning 15 m, nodes on average receive 96.1 % of all schedules and accomplish 85.7 % of rangings.

Throughout the day, the system automatically identifies 81 interactions between the four individuals. Figure 2.5 shows that pairwise relations vary significantly in both average duration and average distance. In particular,

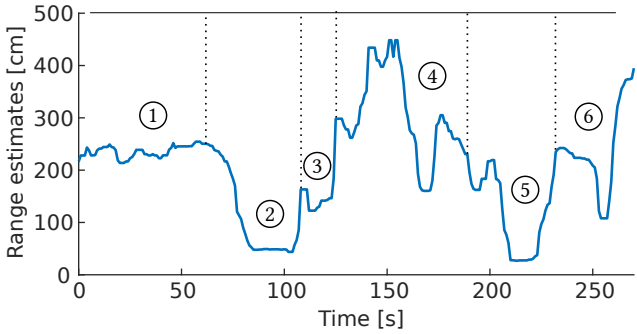

Figure 2.6: (1) A toddler is initially playing on his own while his caregiver is sitting in the corner of the room. (2) As soon as the infant spots a stranger entering, he quickly seeks comfort in the caregiver's proximity. Once the toddler realizes that the stranger and caregiver are on friendly terms, (3) he hesitantly greets the unknown person. (4) The infant then explores the room. As soon as the stranger tries to interact with him and join the play, (5) the toddler once again retreats to the caregiver. After having been comforted, (6) the infant accepts the stranger and continues exploring.

two of the combinations involving Individual 1 show distinctly closer physical relations over a considerably longer period of time. Leveraging our ability to globally timestamp data via synchronization over BLE using smartphones, we correlate the data with a manual log. This enables us to deduce that these pairs correspond to two meetings, during which the individuals were seated. Note that all interactions are automatically detected during post-processing of the data and result in detailed spatial and temporal data which can be further investigated by social scientists. While we only display interactions closer than 250 cm here, the analysis can be extended to include multiple, fine-grained proximity zones [WDTB+05]. By comparison, a proximity-based system cannot support such post-hoc distance-dependent analysis.

Family dynamics. To evaluate the efficacy of SociTrack for social scientists, we give nodes to partners in developmental psychology. They deploy them to execute the "Strange situation" procedure [AB70], which observes the attachment relationships of an infant and their caregiver. Based on how the infant reacts to unexpected disturbance, activity is categorized into one of four behavioral patterns. For this experiment, a toddler (wearing the node in a small vest) and their father (with the node in his pants pocket)

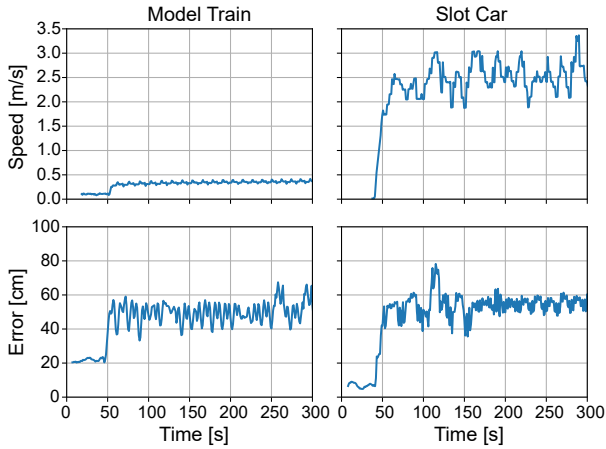

Figure 2.7: We examine the error of moving nodes through low- and high-speed experiments, with a SociTrack device attached to a model train and to a slot car respectively. Error and speed are presented as a moving average over 10 seconds.

are placed in a room where scientists can observe the infant's reaction to an unknown person entering it. Figure 2.6 shows the data from SociTrack measurements, which match the results from visual observation through domain experts. Here, the infant fits the *secure* pattern, happily exploring the room on his own and retreating to the caregiver as a "secure base" in times of distress. While previously only possible in instrumented rooms, SociTrack permits such experiments in the wild.

Sports tracking. We are also motivated to support applications such as sports analysis where high-speed tracking and up-to-date network management are essential. We will later in this section show that the combination of multiple frequency channels and antennas allows us to achieve sub-decimeter range measurements in stationary environments. As SociTrack performs 30 different range measurements, the estimates are gathered over a duration of 60 ms and are essentially smeared across a node's vector of motion, potentially impeding the tracking performance of fast objects.

To *investigate the effect of motion* on system accuracy, we design a controlled motion rig that allows the evaluation of measurement accuracy at different speeds and compare it against ground truth of an optical tracking system. We explore two scenarios in Figure 2.7: a device attached to a model train and one attached to a slot car, both traveling around

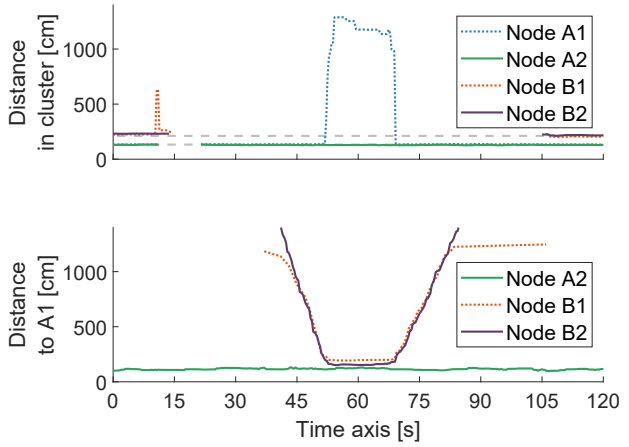

Figure 2.8: The bottom plot displays the view of node *A1* which correctly discovers the approach of a new cluster of nodes and merges both clusters. When lacking discovery, clusters heavily interfere: In the top plot, two clusters without this feature repeat the exact same experiment. While they are outside each other's range, they successfully collect pairwise data. Once the two clusters come closer, they start to interfere with each other and observe severely impeded (cluster *A*) or completely blocked (cluster *B*) measurements.

elliptical tracks. We find that the presence of movement, regardless of speed, introduces an average error of about 50 cm. Despite speeds matching a jogging human at 10 km/h (2.8 m/s), SociTrack provides stable results.

To *investigate the impact of our network management scheme*, we compare a version of SociTrack that mimics traditional single-radio designs (network management feature disabled) with our dual-radio architecture in Figure 2.8. For this, two clusters (*A* and *B*) of two subjects each approach one another, briefly chat, and then separate again. We observe that SociTrack successfully detects and quickly merges two clusters well before the four subjects are within interaction distance. Furthermore, we also find that without our interference-avoiding discovery feature, ranging would be severely impeded. Indeed, collected measurement data would not only miss interactions between different clusters; due to the interfering schedules of the two clusters which result in jammed communication, intra-cluster interactions would be lost as well.

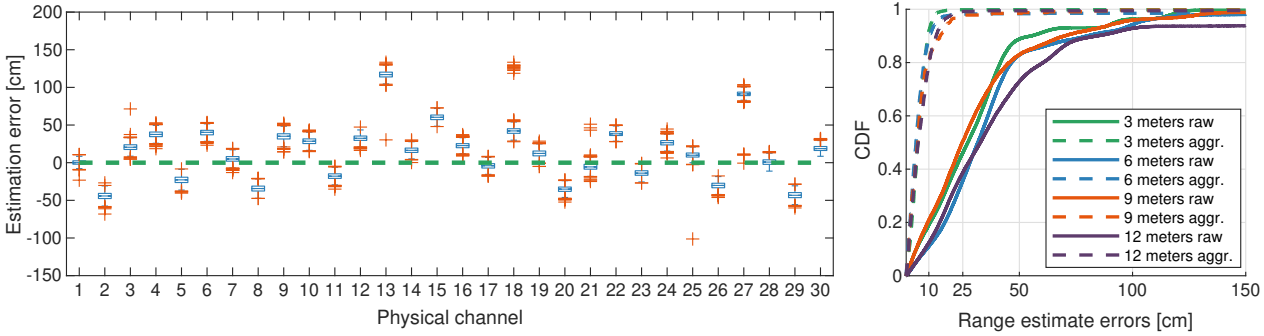

Figure 2.9: Combinations of frequency bands and antennas result in 30 different observations of the same distance. While each offers precise measurements thanks to UWB's stability, cross-polarization and small-scale fading can result in systemic accuracy error (left). By *aggregating* over all settings, we obtain results that are both precise and accurate as shown by the dashed curves (right). We observe that aggregation boosts the average accuracy and significantly reduces the long-tail error: the 90th percentile error is decreased by 82.6 % at a distance of 3 m, while the 99th percentile error is limited to only 14.8 cm.

2.8.2 Characterizing Measurement Fidelity

Next, we analyze whether our system delivers the required measurement fidelity. To begin, we analyze the gain of our ranging protocol, which combines 30 range measurements using antenna and frequency diversity, compared to a single estimate. We gather 34,680 estimates over a distance of 3 m in a static indoor line-of-sight setting. In Figure 2.9, we observe that each physical channel is precise, with an average empirical 90 % confidence interval of 14.7 cm. However, medians exhibit an average error of 30.2 cm. Aggregating the 30 range measurements by taking the 30th percentile of the distance estimates reduces the overall interval to 11.9 cm and the median error to 9.3 cm. Analyzing combinations of antenna and frequencies over various ranges, we find that average precision is boosted by more than 19 % from antenna diversity, while average accuracy increases by up to 66 % using multiple frequency channels.

A second experiment investigates whether distance impacts accuracy. For this, we conduct stationary measurements over four distances (3, 6, 9, and 12 m) as shown in Figure 2.9. We gather more than 30,000 ranges per distance. While the number of raw outliers slightly increases with distance, we see that the aggregation of measurements improves the empirical 90 % confidence interval by at least 82.1 % when compared to raw ranges and reduces the 95th percentile error by at least 79.7 %. Aggregating channel measurements cuts off the long tail of the distribution of range estimates.

Throughout our experiments, UWB offers excellent connectivity indoors, operating in a cluttered, multipath-rich environment as well as through walls and objects. SociTrack achieves a maximal line-of-sight range of 64 m outdoors and at least 50 m in our longest indoor hallway. The BLE radio communicates up to 28 m, which ensures room-level discovery. With its shorter range, BLE discovery inherently averts accidental triggering while UWB is out of range. Note that after initial discovery, the system's effective measurement distance is only limited by the UWB range.

2.8.3 Exploring the Parameter Space

Finally, we leverage our detailed system model to investigate the deployment parameters of the protocol, discussed in Section 2.6, to predict system lifetime. Figure 2.10 shows that small networks of a few nodes can exceed a week of operation and that larger networks, such as school classes, can attain over two days of constant ranging.

We find that energy usage depends on the modes of *exclusive requester*, *exclusive responder*, and *hybrid* as introduced in Section 2.5, while the

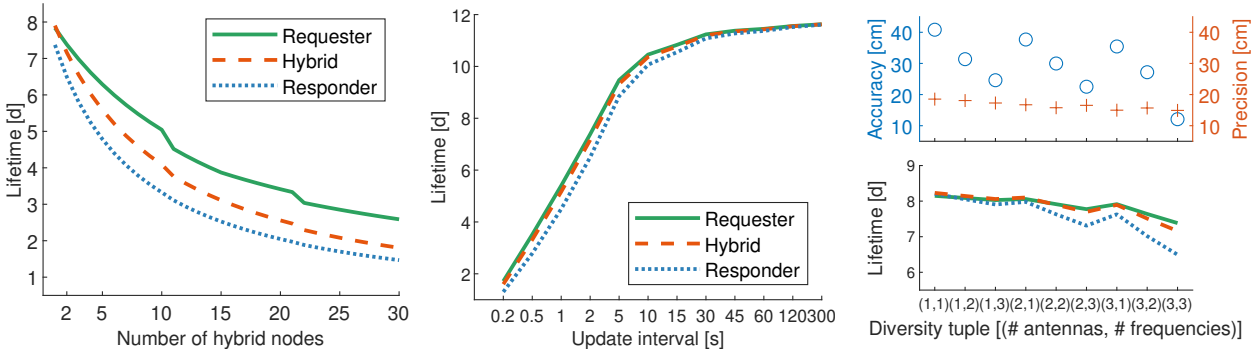

Figure 2.10: While small networks support more than one week of operation with an update interval of 2 s, even large networks of more than 30 nodes endure for multiple days (left). The update interval offers scientists the ability to adjust the expected lifetime and run experiments for up to nearly two weeks for a network of two nodes (middle). Decreasing diversity can further boost deployment duration by almost two days for two nodes with an update interval of 2 s, but trades off accuracy and precision, which decrease by up to 237 % and 24 % respectively (right).

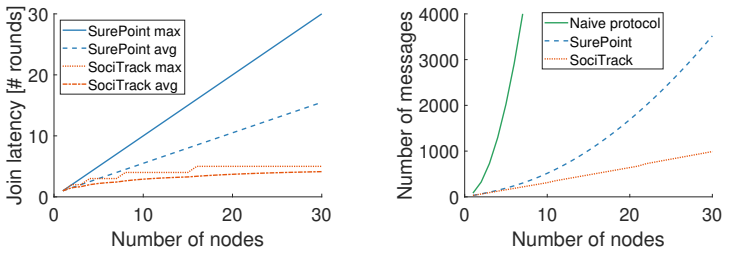

Figure 2.11: SociTrack automatically adjusts contention slots when a group of nodes joins the network and handles 30 instantaneously appearing nodes within maximally 5 rounds (left). In contrast, SurePoint requires 6× longer. We further see that SociTrack realizes a 3.6× reduction in message complexity compared to the state-of-the-art ranging protocol for such a network of 30 nodes (right).

overhead of cluster management is negligible. As exclusive requesters primarily send packets, they can largely avoid listening for packets; notice however the drop in lifetime for networks exceeding 10 nodes due to packet length limitations of UWB, which requires a split of response broadcasts and prolongs listening. On the other hand, exclusive responders listen for all requests, drawing 435 mW while doing so. As a fusion of requester and responder, hybrids would naively require their combined energy costs; exploiting our scheduling scheme shown in Figure 2.3, we can significantly reduce these costs and achieve infrastructure-free operation.

Scalability. Figure 2.11 demonstrates the effectiveness of the protocol's dynamic contention adjustment. Compared to SurePoint [KPCD16], SociTrack can quickly handle the simultaneous appearance of large groups, such as when students arrive for a class. Furthermore, we observe that the join latency difference between nodes is reduced by 88 %. When comparing message complexity, we find that SociTrack outperforms SurePoint as well as a naive, pairwise ranging implementation without broadcasts [KPD15] by a factor of 3.6× and 73.6× respectively for a network of 30 nodes.

2.8.4 Investigating System Deployability

Directionality. Systems that rely on ultrasound for TDoA measurements, like Opo [HKPD14] and iBadge [CMY+02], are limited by the field of view of their ranging sensor. They require near face-to-face orientation to detect interactions and acquire ranges. While the UWB antennas used by SociTrack have a significantly more omnidirectional radiation pattern than

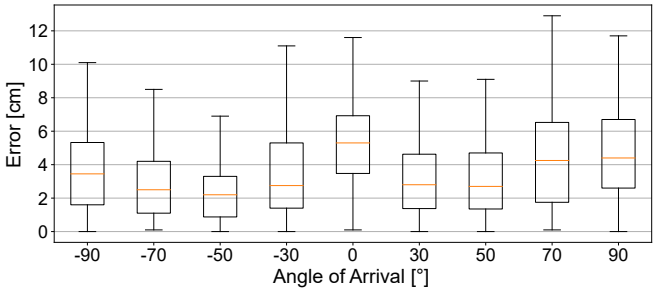

Figure 2.12: To explore the effect of the angle of arrival on performance, we position one device at the center of a 1 m semicircle, and place another at the edge at predefined angles. We perform 120 ranges at each orientation and find that the mean, 25th, and 75th percentiles fall below one decimeter of error for all orientations. Experimental deployments can therefore likely ignore device orientation.

ultrasonic frontends (e.g., Opo reports a detection angle of 60°), they still exhibit poles of decreased performance. Figure 2.12 shows that orientation contributes negligible error and hence is not a deployment concern for SociTrack.

Some applications may desire orientation data, which can be challenging to obtain with radio frontends. The Sociometric badge [WAO+11], nominally just an RSSI-based proximity tracker, adds infrared (IR) transceivers to detect when proximate cohort members are facing one another. A similar orientation detector could easily extend SociTrack, but is out of scope for this work.

NLOS from human bodies. The human body is a well-known attenuator, which might reduce the effective range or fidelity in crowded environments. While we did not observe any sustained connectivity losses in our deployments, we also explore the possible impact of through-body transmission by placing two nodes 5 m apart and then having additional volunteers stand directly in the line-of-sight path. When the first individual steps in, the variance in range estimates increases insignificantly. When the second individual steps in, the range estimation error occasionally spikes to around 4 m, which likely indicates that ranges are being recovered from a non-line-of-sight (NLOS) reflection. These measurement spikes can however be effectively eliminated in post-processing through median filtering with filter widths of 3-5 samples.

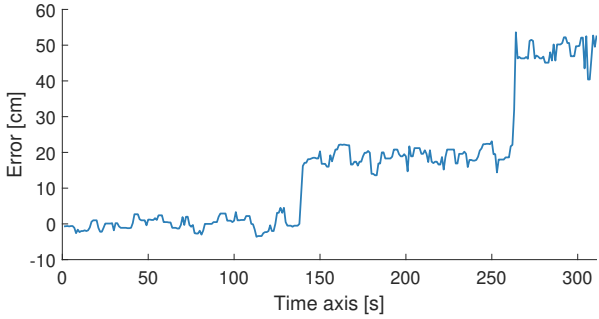

Figure 2.13: Compared to the ground truth at the start, we find that the blocked line-of-sight through one (after 140 s) or two (after 260 s) people leads to an overestimate of the physical distance.

Next, we investigate whether the distance estimation is still being correctly represented if people are sitting with their backs to each other, a common situation encountered in office spaces and public transport. For this, we position two subjects 2 m apart with devices attached to their chests and compare the data gathered with them either facing each other, them setting behind each other, or with their backs to each other. We find that in all three cases, the measurements are stable over time. We observe that distance estimates slightly increase, possibly due to the decreased propagation speed of radio signals in tissue, which can be as much as 8 times slower for muscles [VZA+18]. The presence of a single person between nodes results in an average overestimation of 19 cm, which increases to 46 cm for two people as shown in Figure 2.13.

Future work could utilize recent advances in UWB connectivity graphs [DFPA+17] and ML-based UWB channel analysis [MGWW10, ZLW18, SH18, BEGNMC19, GRS+17] to automatically detect NLOS conditions and correct errors with ranges from other links. In addition to UWB ranging, one could augment the distance estimations with traditional, coarse-grained BLE RSSI data that is gathered through the periodic BLE advertisements in case the primary ranging system should fail.

2.9 Discussion

SociTrack is a first-of-its-kind platform. Next, we consider potential extensions and place our contributions in a broader context. We close with

a look at how future systems might leverage SociTrack's insights and the impact potential of the availability of high-fidelity interaction tracking.

2.9.1 External Adoption Experiences

We find during our collaboration with partners in psychology [SPH+22] that seemingly minor features, unrelated to the core sensor design, can significantly improve SociTrack's efficacy for non-technical users in the field. For example, at a small scale, unscrewing cases to remove boards for charging is feasible, but when deployed in the homes of volunteers unaffiliated with the research team, it is undesirable. While splash-resistant charging ports exist, we have included wireless charging for the future generation of sensors. New chips make the implementation of such technology near-turnkey, and a port-free case design significantly reduces potential vulnerabilities of devices in the wild, particularly for small-scale research prototypes. Similarly, initial designs included a switch to disable the sensors while not in use. In practice, this introduced the risk that end-users would accidentally disable measurements during a study. The less error-prone solution for real-world deployments is to use an RTC that allows study administrators to filter for times of interest (e.g., shutting down at night or exclusively sensing during work days).

Finally, to support deployments, we designed and tested feature-rich wireless debugging and system monitoring through BLE connections. While this was useful for testing with the research team, when deployed on volunteers, our "backup debugging" of color-coded LEDs has proven to be much more practically relevant—it is easier for non-technical users to verify that the sensor is "all green" or to report an "orange flashing" than to manage a companion smartphone application. Encapsulating meaningful state into easily identifiable colors and patterns is hence critical to facilitate user feedback. We have implemented and extended these features in a future hardware iteration in combination with acoustic feedback.

2.9.2 Benefits of the Dual-radio Architecture

Enabling infrastructure-free ranging. While reliable (ToF) ranging used to be a long-standing unanswered problem for mobile systems, UWB radios offer a practical solution to many real-world problems. However, due to their high energy consumption if used as a stand-alone system, the technology was not applicable to wearable systems without relying on externally powered infrastructure. Our addition of a separate discovery radio enables long deployments for low-power systems in a wearable form factor.

Integrating network de-fragmentation. While network bootstrapping is a part of almost any low-power network protocol, the fragmentation of networks due to mobile nodes most often remains an open issue. As nodes avoid idle listening for energy efficiency and only enable their radios at precisely synchronized points in time, progressive network decomposition into clusters and the resulting undetected self-interference is unavoidable in the face of node mobility. In practice, many protocols do not address de-fragmentation and will continue to create additional, smaller clusters throughout a deployment. In addition to increasing self-interference over time, this separation restricts members of these clusters to interact with the few nodes they know. This chapter demonstrates that a dual-radio architecture provides a solution that enables the detection and resolution of such clusters without interfering with regular operation or compromising core measurement fidelity.

Handling mobility. Supporting both range-based interaction tracking with high node mobility and complete autonomy (i.e. infrastructure-free deployments) requires solving four distinct problems. First, the severe energy restrictions demand a design that limits the use of UWB, a task we solve through a complementary discovery mechanism over BLE. Second, nodes cannot rely on the presence of fixed anchors, thus they must act as hybrids that send both requests and responses to ensure successful ranging. Third, a highly dynamic network structure requires topology-independent multi-hop communication, which we provide using a CT-based protocol. Fourth, wide-area mobility leads to network partitioning over time and requires continuous neighbor discovery as well as an efficient de-fragmentation mechanism, which is enabled by the non-interfering, independent dual-radio architecture. Only with each one of these four solutions are we able to build a mobile system that supports infrastructure-free high-fidelity ranging in real-world settings.

2.9.3 Looking Forward

Evolving hardware. Today, SociTrack relies on custom hardware as UWB radios are not ubiquitously available. This is quickly changing, as new smartphones and wearables include UWB to support location attestation features such as user authentication to devices and spaces. Indeed, Apple's iPhone 11 contains a UWB radio with three antennas [iFi20] and could support our diversity ranging scheme out-of-the-box. As systems with heterogeneous radios become more common, low-cost, large-scale deployments of SociTrack could be started through a simple iOS software update.

Evolving interrogatives. Lacking sub-meter resolution, it would be impossible for behavioral analysis to identify the immediate reactions of the infant in Figure 2.6 and automatically distinguish between an infant seeking close contact for comfort or playing happily near their caregiver. SociTrack enables us to detect even detailed movement such as when the infant briefly turns away from the caregiver to scan the environment (visible as brief peaks). The ability to collect high-fidelity interaction data *in situ* fundamentally changes the research capacity of social scientists and paves the way for them to investigate questions with unprecedented detail.

Evolving societal needs. In the face of a worldwide health crisis, effective and actionable interaction information is critical. One of the key techniques for mitigating disease spread is limiting contact through social distancing. As SARS-CoV-2 containment failed in early 2020, governments directed people to stay 1 m,[1] 1.5 m,[2] or 2 m[3] apart in an effort to reduce the infection rate. The rationale behind this discrepancy is unclear; it turns out that many administrative healthcare policies are backed by surprisingly little evidence [SPDG+20, fDPC19], and as a result, inconsistent policies are common [CSM13, CAD+20].

Scientists need tools that enable them to collect representative data in real-world settings to better understand behavior. Such data provides the foundation for fact-based public policies. While both government agencies and private companies are developing smartphone applications for contact tracing [App20, Goo20], these technical means to detect and limit the spread *after* infections have already occurred must be accompanied by controlled scientific experiments to study precautionary measures that can prevent them. Our hope is that SociTrack can act as research-enabling research and serve as a platform to address urgent questions in social science and beyond.

2.10 Summary

This chapter demonstrates that it is possible to capture pairwise interaction data in situ with sub-meter accuracy and sub-second sampling. It identifies four pillars that make up a measurement system that meets the needs of

[1] WHO [Wor20], European Union [Eur20], France [Phi20]
[2] Australia [Aus20], Netherlands [Nat20], Switzerland [Fed20b], Germany [Fed20a]
[3] United States [Cen20], United Kingdom [Pub20b], Canada [Pub20a], India [Min20]

real-world longitudinal studies for social scientists: it must collect high-fidelity samples—sub-meter, sub-second ranging data—, it cannot rely on supporting infrastructure, it must be robust to participant mobility, and it cannot restrict how and where participants wear measurement devices. We show that constraining a platform to any single communication technology requires sacrificing at least one of these pillars. We then demonstrate that a novel, *heterogeneous* BLE and UWB radio architecture can succeed in satisfying all system requirements. As we will see in Chapter 4, the combination of two radio modulations can be used to similar effect and also enable the autonomous formation of a network. Our experiments illustrate that the ubiquitous use of broadcast packets reduces message complexity for pairwise ranging from quadratic to linear in the number of nodes, and that in concert with careful scheduling, this enables scalable, long-term deployments.

SociTrack is the first system to enable autonomous tracking based on UWB, a feat that was previously exclusively reserved for infrastructure-based systems. Furthermore, it already introduces continuous neighbor discovery as an essential service in parallel to networked communication. However, we will show in Chapter 3 and Chapter 5 that its centralized scheduling does not provide persistent networking in the face of node and link failures. When deploying SociTrack, we observed that a network split can cause a leader-less cluster to stop sensing until orphans could re-discover each other individually after prolonged periods of time, which significantly impacted its tracking reliability for highly mobile nodes. To improve the reactivity to changing conditions, we therefore explore in the next Chapter 3 how we can boost autonomous resilience by improving the protocol's performance when facing network splits and joins through cluster-wide coordination and discovery.

3

Robust Orchestration for Autonomous Networking

While specialized WSN protocols perform well in the setting they were designed for, they often lack the ability to quickly adapt once operating conditions change drastically. Chapter 2 has shown how a dual-radio architecture can be employed to enable autonomous formation and prevent network fragmentation after a change in the topology. However, the hardware costs and complexity of using two radios renders this platform costly for most application scenarios; while separation in hardware simplifies the parallelization of continuous discovery and cluster coordination, most systems do not offer such diversity. In addition, as cluster discovery and coordination are entirely separated, nodes are slow to find and merge entire clusters, which delays adaptivity. More generally, the resilience to node and link failures is of particular importance when considering the robustness of network orchestration, as clusters of nodes that lost their leader or split apart need to re-organize and find each other again. As we will also see in Chapter 5, a centralized scheduler like the one in Chapter 2 is prone to intermittent communication when suffering from such failures. However, for a network of mobile nodes that is expected to frequently split and rejoin, network orchestration should be able to quickly adjust to new conditions.

With Demos, we present a low-power wireless protocol that ensures robust network orchestration despite node and link failures by intertwining cluster coordination and cluster discovery. Demos rapidly finds consensus

This chapter is based on [BZT23] and [BDFK23a].

on leadership even if the set of nodes fluctuates by introducing new election quorums. In addition, a novel cluster-wide discovery scheme enables autonomous clusters to merge on the fly and maximize network coverage. Experiments with controlled mobility on a multi-hop network of 24 nodes demonstrate that Demos maintains reliable data exchange despite severe disruptions and adapts to changes within seconds. We further find that Demos' ability to continuously coordinate and discover achieves highly robust orchestration of fully autonomous clusters.

3.1 Introduction

While many traditional WSNs rely on static nodes and external infrastructure such as central data sinks, a rapidly growing class of applications features mobile nodes and autonomous operation. Examples that are of particular societal and industrial importance range from human sensing [ILM+21, RL22] to multi-agent systems [HYM16, LSD+20]. Advances in re-configurable hardware [ADY+19], flexible physical layers [BGS+22], and data-driven media access [GLG+21, PL21, ZJGD22] provide the necessary robustness and adaptivity for such applications on the link layer. However, current network-layer solutions still fall short of these emerging requirements.

Problem. Imagine a swarm of drones in a search-and-rescue scenario. The swarm requires continuous coordination and data exchange among drones over a wireless multi-hop network, for example, to jointly localize human victims [CXC+19, WTJ09, WH11] or to safely navigate complex spaces [GET22, LSD+20, MBH+22]. During a search-and-rescue mission, the network may split into multiple clusters voluntarily (e.g., when multiple areas of interest are identified) or unexpectedly (e.g., due to sudden non-line-of-sight conditions). Using current network-layer solutions, such a network split would result in a loss of coordination for a subset of drones [DWVT14, FZMT13, ITMP18, PANL19, YNB+22] or render the drones unable to reconnect and merge into one complete network [MZL+18, MZL+20, MBTZ22], thus endangering mission success.

The root of this problem is that nearly all existing network-layer solutions orchestrate the nodes in a centralized fashion. Within a *cluster* of connected nodes, which may comprise all or only a subset of the nodes in a network, a *coordinator* computes and distributes information to synchronize the nodes and instruct them on when each may communicate.

In case the coordinator becomes unavailable, orchestration is halted and the data exchange breaks down. While such a disruption may be caused by node mobility [ILM+21, RL22], static clusters also frequently suffer from unexpected disturbances due to node or link failures [ABC+20, WCPC18]. Despite its relevance, current network-layer solutions [FZMT13, ITMP18, MZL+18, MZL+20, PANL19, YNB+22] cannot quickly and safely appoint a new coordinator after arbitrary node and link failures. While recent distributed synchronization protocols in principle enable nodes to continue exchanging data after a network splits into clusters, they provide no means for these clusters to discover each other and merge [FZMT12, MBTZ22].

For robust orchestration despite arbitrary node and link failures, a network-layer protocol needs to be able to (*i*) accurately identify nodes and their traffic demands in a cluster, (*ii*) maintain reliable multi-hop communication in the face of frequent network topology changes, and (*iii*) discover and merge with other clusters. These abilities enable autonomous networking as required by emerging applications.

Contribution. We present *Demos*, the first low-power wireless protocol that provides robust network orchestration for autonomous multi-hop networking. Demos features a novel consensus mechanism to reliably determine a cluster coordinator and to ensure that cluster information remains up-to-date at all times. By minimizing network state and temporal dependencies, Demos swiftly adapts its data exchange and instantly reacts when the network topology changes. Key to this robustness is Demos' ability to preserve orchestration in case the current coordinator becomes unavailable based on newly introduced election quorums. Combined with a novel cluster discovery scheme, nodes always remain in exchange with connected nodes and quickly merge clusters to maximize coverage. By building on concurrent transmissions (CT) [FZTS11, LFZ13], Demos propagates information reliably while being agnostic to the underlying network topology. This flexibility enables the protocol to inherently support frequent network fluctuations and high node mobility.

This chapter makes the following major contributions:

- We introduce Demos, the first low-power wireless protocol that achieves robust network orchestration despite arbitrary node and link failures.
- We propose novel methods for cluster coordination and discovery, enabling autonomous clusters that can dynamically split and merge to increase network coverage.

- We provide an open-source implementation of Demos and demonstrate its ability to robustly orchestrate nodes under challenging conditions in real-world experiments.

Based on the application and protocol requirements given in Section 3.2, we present an overview of Demos in Section 3.3. We describe the protocol in more detail in Section 3.4 and provide a formal analysis in Section 3.5. In Section 3.6, we test Demos in various mobility and failure scenarios and demonstrate its ability to adapt on the spot and ensure maximal connectivity.

3.2 Motivation and Challenges

We first discuss the application requirements, deduce prerequisites for robust network orchestration, and then identify the challenges we need to overcome in the design of Demos.

3.2.1 Application and Protocol Requirements

Application requirements. Autonomous systems typically feature different traffic categories (e.g., control, coordination, and application data) [HYM16]. As each category differs in bandwidth and temporal resolution, the application must be able to specify and adapt its traffic demands. Especially in multi-agent systems, reliable real-time traffic is indispensable, with requirements changing over time depending on the developing scenario (e.g., switching from searching for a victim to streaming video feeds after localization [HYM16]).

To support mobile networks such as drone swarms, consistently maintaining connectivity such that no node remains isolated is paramount [WTJ09, WH11] and should be temporally and spatially robust. However, due to large coverage areas and precarious environments, multi-hop communication is often a must [HYM16] and needs to cope with a dynamic topology [LBL+21].

Most crucial, however, is the ability of nodes to self-organize into an autonomous network by detecting nearby nodes and establishing communication. Nodes may frequently encounter network fluctuations [WH11] and seek to dynamically exchange information whenever possible [WTJ09].

Protocol requirements. From the preceding application requirements, we deduce the following requirements for a suitable network-layer protocol. To cater to strict and challenging traffic demands, communication should

be scheduled to prevent contention and ensure efficient use of the restricted bandwidth [DWVT14]. Reliable scheduling requires the collection of traffic demand from the currently participating nodes and uniquely assigning time slots to avoid packet collisions.

To cope with time-varying communication links, network orchestration should have minimal dependence on the current topology. As a consequence, the protocol should be sufficiently robust to ensure that the exchange of data is preserved despite unexpected node and link failures.

Lastly, to autonomously and rapidly coordinate communication without potentially making interfering decisions, each cluster should choose a unique cluster coordinator. The election of a coordinator must fulfill agreement (i.e., all connected nodes elect the same node), termination (i.e., if there are valid candidates, one must eventually be elected), validity (i.e., the elected node must be part of the cluster), and stability (i.e., a new coordinator is only elected if the previous one has become unavailable) [AKNR13, CRW11].

3.2.2 Challenges

For a protocol design that must autonomously schedule traffic, support reliable multi-hop networking, and robustly orchestrate a network despite arbitrary node and link failures as outlined above, Demos has

(i) to elect a coordinator in a network of unknown size,

(ii) to preserve communication when a cluster splits away,

(iii) to continuously discover other nearby clusters, and

(iv) to merge clusters and maximize network coverage.

Next, we investigate each of these challenges in more detail.

Election of cluster coordinator. Most leader election algorithms rely on knowing the exact network size [SPZE20] and only consider it as a start-up problem [ANDL17a] or after long periods of inactivity (e.g. 2 min [FZMT12]). However, our application continuously requires elections because node and link failures may cause a leader to suddenly disappear. For stability, elections must only occur if the current coordinator is unavailable and depend on votes from an unknown set of nodes, with the result required to be unique for agreement.

Network split. Two clusters that separated could continue to use the same schedule. For at least one cluster, however, the coordinator is no longer available. So far, protocols then usually fall back to bootstrapping before

restarting from scratch [FZMT12]. Instead, the remaining nodes may swiftly initiate an election and establish the new traffic demand.

Cluster discovery. After re-establishing cluster coordination, clusters may operate independently. However, such separations lead to a gradual decay into highly fractured patches of limited individual coverage. To recombine, a secondary radio could be used in parallel [BJT+20, ILM+21] to continuously look for other clusters in the vicinity without affecting the data exchange on the primary radio. To avoid such an increase in complexity and costs, a holistic protocol design may instead combine data exchange and discovery. However, most discovery protocols depend on a fixed structure [DC08, QLXL16] as they are not co-designed with a scheme to exchange application data. Furthermore, they focus exclusively on the pairwise discovery of nodes and do not cater to multi-hop networks.

Cluster merging. Once another cluster has been discovered, a node could independently leave its previous cluster and synchronize to the new one. While this straightforward approach based on individual pairwise discovery and decision-making is predominant [CSG+16, WSC+18], such a design only gradually increases network coverage. Even worse, as discovery can be bi-directional, merging could be simultaneously attempted in both directions and data exchange may collapse as nodes attempt to synchronize to disbanded clusters. Alternatively, nodes may leverage cluster-internal communication and exchange their discoveries to take a collective decision, but long delays for merging clusters could remain an issue.

3.3 Designing Demos

Demos aims to achieve one of the early visions of WSNs: To be able to distribute sensor nodes in a deployment area, which would then automatically coordinate and communicate by themselves without detailed preconfiguration [MB01]. To this end, Demos provides robust network orchestration which enables nodes to constantly exchange data and maximize their coverage despite a fluctuating network topology. By introducing dynamic cluster coordination, the protocol tackles the first two challenges presented in Section 3.2.2 and ensures that nodes maintain coordination even after a network splits into multiple clusters. With its cluster-wide discovery scheme, Demos overcomes the remaining challenges and offers an autonomous, restorable communication service among all connected

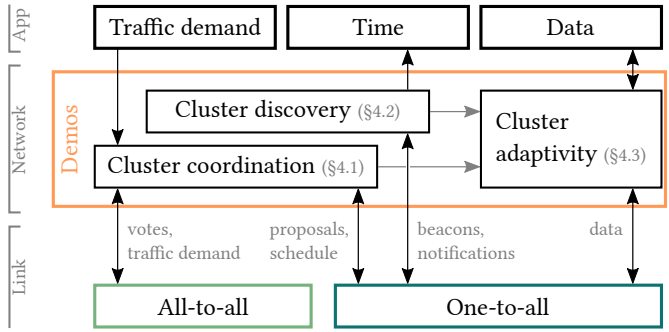

Figure 3.1: Demos provides network orchestration for the application. Based on two link-layer primitives, it establishes coordination, discovery, and adaptivity on the network layer.

nodes. In the following, we first present the overall design principle before giving an overview of Demos' structure and protocol phases.

3.3.1 Demos in a Nutshell

At its essence, Demos is a network orchestration protocol that continuously re-elects a cluster coordinator and aggregates information on all connected nodes. During this election, the coordinator periodically collects votes and the traffic demand of nodes in the cluster and then assigns exclusive time slots through scheduling. For this communication, the cluster uses a frequency channel that directly depends on its coordinator to prevent interference with others. However, if no coordinator is available, the nodes follow a shared randomized sequence of node IDs to deterministically elect a new leader. After the exchange of application data, a distributed cluster discovery scheme detects neighboring clusters and notifies the cluster coordinator. The entire cluster can then merge simultaneously with its neighbor.

To support such abrupt changes in the network topology and provide a robust data exchange despite high node mobility, Demos builds on a link layer utilizing CT [ZMS20], as displayed in Figure 3.1. Such network flooding is topology-agnostic and permits reliable communication due to its inherent redundancy. Based on these primitives, Demos provides novel mechanisms for the coordination (Section 3.4.1) and discovery (Section 3.4.2) of clusters as well as for adaptivity to changing conditions (Section 3.4.3).

3.3.2 Protocol Structure

Demos is structured into rounds of period T which all nodes of a cluster $C \subseteq \mathcal{N}$ execute synchronously, where $\mathcal{N} = \{i \in \mathbb{N} \mid 1 \leq i \leq N\}$ is the total set of nodes in a network. Each round consists of three phases as illustrated in Figure 3.2. To (re-)elect a coordinator, the cluster votes on proposing nodes during the *consensus* phase, explained in Section 3.4.1, and exchanges the current traffic demand. During the *communication* phase, a contention slot can be used to notify the cluster coordinator of discovered clusters. If the coordinator decides to merge with another cluster, it then orders its cluster to disband as described in Section 3.4.2. Otherwise, it broadcasts a schedule which is subsequently executed to sequentially disseminate application data. In the final *discovery* phase, nodes use the rest of the round to listen for and advertise to other clusters in the vicinity. If another cluster is discovered, the coordinator will be notified in the next round.

Communication primitives. Demos builds on two complementing types of CT, a *one-to-all* and an *all-to-all* primitive. One-to-all floods, as introduced by Glossy [FZTS11], are employed for most slots of the protocol and consist of a packet that is initially broadcasted by a single node which is then concurrently forwarded by each receiver. An all-to-all primitive like Chaos [LFZ13] on the other hand is used during the consensus phase to efficiently propagate the vote and demand of each node through the network by continuously merging information during consecutive sub-slots. Combining these primitives enables Demos to aggregate information from an unknown number of nodes in parallel with an all-to-all exchange and then schedule one-to-all floods that reliably reach their target with minimal latency.

Deterministic randomization. Demos repeatedly seeds a random number generator (RNG) to share randomized state without having to distribute it explicitly. Sequences produced by this deterministic generator are used to (*i*) create a non-repeating sequence of designated leaders for the consensus phase, (*ii*) map the ID of a cluster coordinator $\in \mathcal{N}$ to a frequency channel on which the cluster communicates, and (*iii*) assign non-overlapping reception (RX) and transmission (TX) slots for discovery.

3.3.3 Consensus Phase

The consensus phase consists of E pairs of *proposal* (P) and *voting* (V) slots. The first pair is reserved for the previous leader that coordinated the cluster in the last round to provide stability. If the election is unsuccessful, as

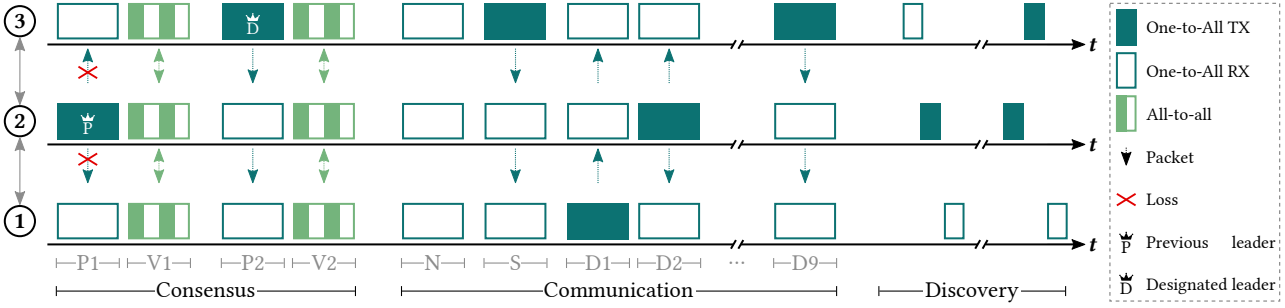

Figure 3.2: A Demos round consists of three phases: *Consensus*, *communication*, and *discovery*. During consensus, E pairs of proposal (P) and voting (V) slots are used for cluster coordination ($E = 2$ in this illustration). While the first pair is assigned to the leader of the previous round, later ones are designated so other nodes may get newly elected. The communication phase consists of a notification (N), a scheduling (S), and multiple data (D) slots. For discovery, nodes interact with other clusters.

explained in Section 3.4.1, the remaining pairs are designated to potential successors. A designated leader that is part of the cluster proposes if it has not yet voted in the current round and can be elected as the cluster coordinator upon success.

Proposal. Apart from the first P slot in which the previous leader can propose, the remaining $E - 1$ proposals are designated according to a randomized sequence containing the total set of nodes $\mathcal{N}$. The offset inside this sequence is based on the *round counter* r and the *election counter* $e \in [1, E]$ and synchronized across the cluster. The designated node then proposes to ask others for their vote in the current round.

Voting. A node that received a proposal casts its vote if it has not already done so in an earlier V slot of the same round. Together with its current traffic demand, votes from all nodes are then exchanged within the cluster and continuously merged. At the end, the designated leader locally determines the result of the election as detailed in Section 3.4.1.

3.3.4 Communication Phase

After having established consensus on the cluster coordinator and aggregated the current demand, the communication phase is executed. A *notification* (N) slot permits nodes to inform the coordinator about other discovered clusters. The cluster coordinator then decides whether a merge should be conducted, as explained in Section 3.4.2. The following *scheduling* (S) slot is used to either disseminate information on the cluster to be merged with or to distribute a schedule based on the current demand of the nodes. To avoid that missing demand information occasionally causes nodes to not be scheduled, a received demand is valid for up to P rounds. Lastly, application data is exchanged during multiple consecutive *data* (D) slots if a schedule has been received.

Notification. To extend pairwise to cluster-wide discovery, a node can flood a received beacon containing information on a discovered cluster so it reaches the coordinator. As access is contention-based, only one of potentially multiple simultaneously discovered clusters is successfully reported.

Scheduling. The S slot enables the coordinator to instruct the cluster on how to continue. If it has determined that the cluster should merge, it propagates the necessary information so the cluster can align itself to the communication and frequency channel of the new cluster. Otherwise, the

coordinator computes and distributes a schedule based on the previously gathered traffic demands, which includes a bitmap containing the current members of the cluster.

Data. Each D slot is assigned to a node that may initiate a one-to-all flood of application data. All others assist in propagating a received packet by retransmitting it concurrently.

3.3.5 Discovery Phase

The discovery of other clusters occurs during the remaining round time on a dedicated frequency channel that is shared among all clusters. To guarantee overlapping discovery phases between clusters, we presume that the phase duration T_d exceeds $T/2$. To permit the adaptation of T_d depending on the currently demanded number of D slots and round period T, Demos employs probabilistic discovery [MB01]. The discovery phase is split into slots of fixed length, during which nodes transmit a beacon (B), listen, or remain idle based on given probabilities. Building on this scheme, Demos leverages cluster coordination to increase its energy efficiency. As nodes of a cluster have overlapping communication ranges, we spread the discovery overhead equally across the cluster by assigning non-idle slots to nodes.

If another cluster is discovered, the cluster coordinator will be informed in the next N slot, as shown in Figure 3.3. Once discovery has terminated, the round ends and the frequency channel is updated depending on the information received from the coordinator during the S slot.

3.4 Demos in Detail

Next, we focus on the two key concepts that enable Demos to form autonomous clusters, cluster coordination and cluster discovery. Thereafter, we examine how the protocol leverages its flexibility and adapts to changing conditions.

3.4.1 Cluster Coordination

Without packet loss, the election of a coordinator is undisputed as all nodes of a cluster receive the first proposal and cast their votes accordingly. However, the minimum required number of votes in favor to safely call elections, called *quorum*, is non-trivial when considering imperfect

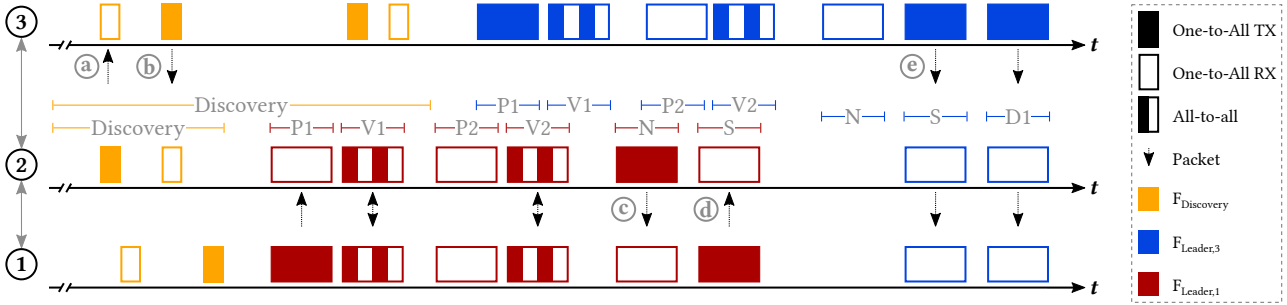

Figure 3.3: During discovery, a beacon from a leader with a lower ID will be ignored (*a*). To merge with a dominating cluster, the received beacon (*b*) will be forwarded to notify the current cluster coordinator (*c*). The leader then instructs nodes to disband and join the discovered cluster on the corresponding frequency channel (*d*), where they synchronize to the new coordinator (*e*).

communication links and variable cluster sizes. Based on a subset of votes from a total set of unknown size, the cluster must autonomously determine a unique leader. We develop two novel methods to define quorums that enable nodes to still find agreement on a coordinator under such circumstances.

Absolute quorum. The classical solution to guarantee a unique decision is to require agreement from a majority of nodes [ANDL17b, Lam01, OO14, PANL19]. Therefore, the *absolute quorum* Q_a can be defined as $Q_a := F > N_c/2$, where F is the number of votes in favor of the proposal and N_c is the cluster size.

However, N_c varies over time. To accurately keep track of the cluster size, each node locally estimates N_c and updates this value each round as follows. Nodes include their current estimate in the exchange during the `V` slot and store the maximum of all received estimates as N_{net}. After the consensus phase, they further count the number of votes they have encountered at least once as N_{votes} (independent of whether they are in favor or against). Updating N_c then occurs in two steps. First, a network filter sets $N_c \leftarrow \lfloor \alpha \cdot N_{net} + (1-\alpha) \cdot N_c \rfloor$, with $\alpha \in [0, 1]$. Thereafter, a temporal filter is applied to get $N_c \leftarrow \lfloor max(\beta \cdot N_{votes} + (1-\beta) \cdot N_c, N_{votes}) \rfloor$, with $\beta \in [0, 1]$. If a node has been elected as cluster coordinator, it distributes its own estimate of N_c with the schedule during the `S` slot so the entire cluster adopts the value of the leader.

This mechanism leverages centralized updates of N_c to equalize estimates when a cluster coordinator is available and otherwise enables nodes to adapt independently until a new leader is elected. The network filter first applies α to ensure that all nodes of a cluster use similar values and do not underestimate the cluster size, as this may lead to multiple simultaneous leaders. Choosing a high α prevents a strong decrease when packets were locally missed but have been received by others. Lowering α limits the influence of a single well-connected node if most nodes only receive a subset of votes so that the chance for successful elections remains high. The temporal filter integrates β to ensure that the estimate converges towards the true value over time. If more nodes than expected have been heard, the empirically validated number of nodes is adopted to prevent an underestimation. Otherwise, the new estimate is weighted with β for a gradual transition that avoids disturbances due to temporal fluctuations.

Relative quorum. The absolute quorum directly depends on an accurate estimate of N_c to safely call a vote. As the quorum is particularly relevant when a new leader needs to be elected after nodes split apart and N_c

changed drastically, waiting until the estimates converge to the true value may prevent the exchange of data for several rounds. To expedite elections in highly mobile scenarios, we introduce a *relative quorum* Q_r that only depends on the fraction of votes in favor of the proposal as gathered during the ongoing election.

To accomplish independence from N_c, we leverage the known high reliability of CT [ZMS20]. Suppose that for a connected cluster of N_c nodes, a one-to-all proposal reaches at least $N_o = \gamma_o \cdot N_c$ nodes, with $\gamma_o \in [0, 1]$. The designated leader then collects votes using the all-to-all exchange in the ∨ slot, with each of the N_o nodes that have received the proposal submitting a vote. At least $\gamma_a \cdot N_o$ votes reach the designated leader, with $\gamma_a \in [0, 1]$. In other words, the leader receives $V \in [\gamma_a \cdot \gamma_o \cdot N_c, N_c]$ votes and we can therefore bound $N_c \in [V, \frac{V}{\gamma_a \cdot \gamma_o}]$. As for the absolute quorum, a result is unique if $F > N_c/2$ is satisfied. Using the derived upper bound, a sufficient (but not necessary) condition is $F > \frac{V}{2 \cdot \gamma_a \cdot \gamma_o}$. We can hence find a quorum that is entirely based on the votes V of the current ∨ slot as

$$Q_r := \frac{F}{V} > \frac{1}{2 \cdot \gamma_a \cdot \gamma_o}$$

Extended absolute quorum. Knowledge of γ_o can also be exploited to improve Q_a. As at least N_o nodes receive the first proposal and vote in its favor, at most $N_c - N_o = (1 - \gamma_o) \cdot N_c$ votes remain uncast if a proposal was sent. It hence suffices to receive $F > (1 - \gamma o) \cdot N_c$ votes in favor to assert that no other leader was previously elected, enabling us to define

$$Q_{a+} := F > min(\frac{1}{2}, 1 - \gamma_o) \cdot N_c$$

Both absolute quorums permit nodes to skip consensus after voting but rely on N_c and may result in multiple leaders per cluster if the estimate does not reflect the actual cluster size. While the relative quorum guarantees safe elections if γ_a and γ_o are valid assumptions, it requires participation in all E elections per round and hence needs more energy. Demos supports the use of all three quorums, whose performance we will compare in real-world experiments in Section 3.6.4.

3.4.2 Cluster Discovery

With its cluster coordination, Demos enables dynamic cluster management that tolerates faults and permits network splits. However, to maximize its connectivity, the protocol also requires a mechanism to detect and

merge clusters. Demos builds on a well-known neighbor discovery scheme and extends it to *cluster-wide discovery* to increase its coverage. We thereby leverage knowledge of other nodes in the cluster to cooperate and distribute discovery across all of them.

Continuous discovery. In contrast to current WSN protocols that treat discovery as an isolated task during bootstrapping, Demos must retain the ability to find and merge clusters while continuously exchanging data in parallel. As traffic demand is dynamic and constantly influences the number of scheduled `D` slots as well as the round period, the round time remaining for discovery is restricted and requires a highly adaptive mechanism. The Birthday protocol [MB01] is a flexible scheme for pairwise discovery that achieves a low detection latency despite these constraints, as it does not require a fixed structure and permits nodes to adjust parameters independently of other clusters. By splitting the discovery phase into slots during which a node transmits beacons with probability P_t and listens with P_l, discovery is likely despite a low duty cycle $P_t + P_l$.

Cluster-wide discovery. Demos extends this basic concept with novel mechanisms to reduce energy consumption and discovery latency. First, we leverage that Demos coordinates an entire cluster and allot discovery equally across nodes instead of performing it redundantly in parallel. By prohibiting overlapping discovery slots within a cluster, we avoid the scalability issues of other discovery protocols due to colliding transmissions [KPSV17] and preclude inefficient simultaneous listening periods. Assigning discovery slots to nodes merely requires including the cluster members as a bitmap in the distributed schedule and randomizing their order locally using a deterministic RNG. Second, we exploit that Demos ultimately aims to discover clusters and not individual nodes, for which a single node receiving a beacon from a node in another cluster suffices. By notifying the entire cluster via the coordinator, all nodes can switch simultaneously without having to pause their data exchange.

Discovery interface. To merge with another cluster, a coordinator must be able to align the communication of its own cluster accordingly. The beacon distributed by a node during discovery contains the relative time since the round started, the round period T, the round counter r, the ID of the cluster coordinator, and N_c to enable this synchronization. Knowing the start of the last round and T, another cluster may directly reorient itself to the upcoming round and can infer the current offset in the sequence of designated leaders based on r. To remain valid, the time passed since the

reception of the beacon is added to the received time since the start of the round when the beacon is forwarded during the N and S slots.

Merge criteria. When the coordinator receives information on a discovered cluster, it must determine whether to merge. To avoid that two clusters attempt to simultaneously merge with each other, Demos relies on the ID of the cluster coordinator and merges a cluster led by a node with a lower ID into the cluster of a leader with a higher ID, as visualized in Figure 3.3. However, to prevent a single node with a high ID from temporarily disrupting the communication of a much larger cluster, N_c can also be included as a merge criterion, with a larger size dominating and the leader ID merely used to break ties.

3.4.3 Cluster Adaptivity

Building on top of robust cluster coordination and a network structure that continuously adjusts to the underlying topology, Demos can adapt essential protocol parameters to cater to the current application requirements. To maximize its flexibility and reduce complexity, the protocol does so by deliberately maintaining minimal persistent network state.

Traffic demand. To support high node mobility, Demos must expect frequent changes in the coordinator and the nodes belonging to a cluster. Therefore, classical solutions such as explicit registration when joining [BJT+20, RPL20] and timeouts to remove unavailable nodes [FZMT12] are not applicable. To continuously adapt the communication schedule according to current demand, Demos includes requests directly in the all-to-all communication of the V slot. As this exchange ensures that the demand of all nodes in the cluster is always up-to-date at the elected leader despite network splits and merges, it can immediately react to changes.

Round period. Many round-based protocols adopt a fixed round period T to circumvent scheduling [HMZ18, LFZ13] or temporally improve liveness by maintaining communication despite the unavailability of the coordinator [MZL+18, MZL+20]. However, such a limitation severely restricts the flexibility of a protocol, as it fixes parameters such as the frequency of cluster coordination, the number of data slots, and the likelihood of discovery. It further constrains the protocol in reacting to changing real-time scheduling requirements. Demos permits the cluster coordinator to adapt T and communicate its current value in the S slot. This flexibility is made possible by its robust cluster coordination, as Demos

tolerates missing such an update by quickly re-electing a leader after desynchronization. Combined with cluster discovery, the protocol rejoins nodes that are out-of-sync and thereby ensures continued connectivity while offering high adaptivity.

Frequency channel. Demos employs separate frequency channels per cluster based on the ID of the cluster coordinator to prevent packet collisions when clusters encounter each other. This mechanism further inherently avoids heavily contested channels due to interference from external sources, as it enables nodes to synchronize to leaders on reliable frequency channels. By voluntarily yielding as coordinator if the packet reception rate (PRR) drops below a given threshold, Demos supports temporal blacklisting of frequencies to boost its robustness.

3.4.4 Practical Example

To illustrate Demos' concepts, we investigate how a network splits and merges again in Figure 3.4. The previous leader proposes itself for re-election in the first `P` slot of each round to confirm the availability of the coordinator. However, a change in the network topology causes the network to temporarily split into two clusters. While leader 1 can still coordinate one cluster, the other consisting of nodes 3 and 4 must wait for a designated `P` slot to elect a new leader. Note that the stable re-election of leader 1 is made possible by our novel relative quorum, as the cluster size N_c could not be re-estimated yet and would have prevented node 1 from being elected based on an absolute quorum. With two autonomous clusters led by nodes 1 and 4, the cluster discovery mechanism enables them to merge once the communication link between nodes 2 and 3 recuperates. As the clusters operate on a separate frequency band, packet collisions are prevented during the discovery and merge process. The beacon containing the necessary information to synchronize is received by node 2 and forwarded to the leader using the `N` slot, whereafter node 1 decides to merge due to its lower ID and the network is once again reunited.

3.5 Formal Analysis of Election Latency

Next, we formally investigate the expected number of elections if a new coordinator is required. After a network splits, at least one new cluster coordinator must be elected. As Demos cycles through a non-repeating randomized sequence of the total set of nodes $\mathcal{N}$ to designate

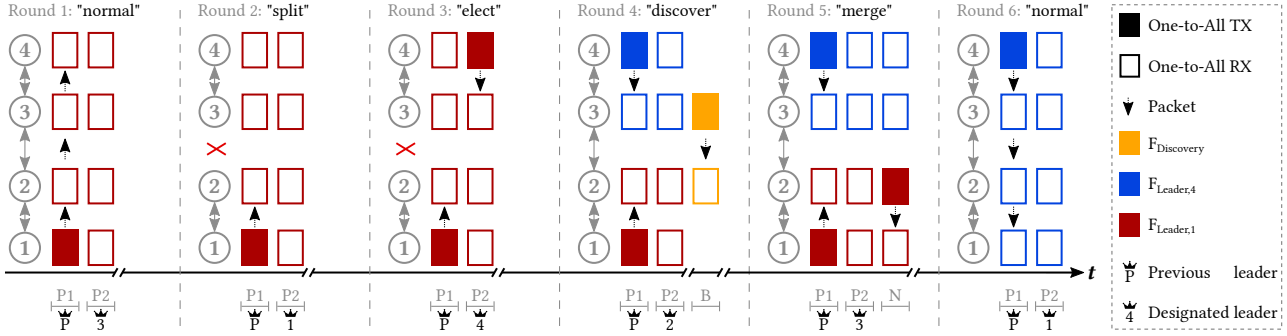

Figure 3.4: In round 1, a network of four nodes coordinated by node 1 successfully re-elects its previous leader. After a topology change breaks the communication link between nodes 2 and 3 in round 2, the network splits into two clusters. As soon as a P slot is designated to a node from the leaderless cluster in round 3, nodes 3 and 4 independently resume the data exchange. After the link is re-established in round 4, node 2 notifies node 1 about a received beacon (B) and the clusters merge in round 5.

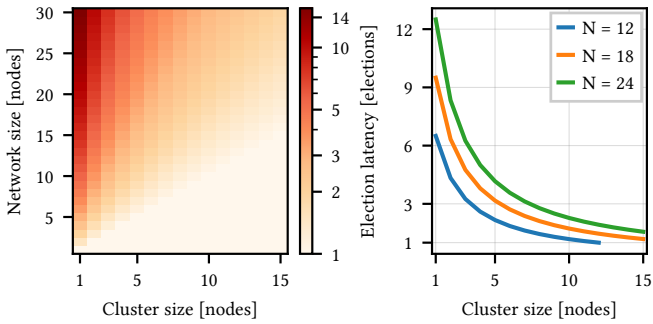

Figure 3.5: For all network sizes, Demos only requires few elections on average until a new designated leader proposes for clusters containing more than a small fraction of nodes.

leaders in the consensus phase, knowing the expected latency until a new cluster coordinator can be established is essential to estimate when communication may resume. Supposing a cluster size N_c and $N = |\mathcal{N}|$, the probability that a designated leader is not part of the cluster is $(N - N_c)/N$. We find $p_L = 1 - \frac{(N-N_c)\cdot(N-N_c-1)\cdot\ldots\cdot(N-N_c-L+1)}{N\cdot(N-1)\cdot\ldots\cdot(N-L+1)} = 1 - \frac{(N-N_c)!\,(N-L)!}{N!\,(N-N_c-L)!}$ that a leader is successfully elected within L tries. Similarly to p_L, we can derive the expected number of tries until a successful election occurs as

$$\mathrm{E}[L] = \sum_{L=1}^{N-N_c+1} L \cdot \frac{(N-N_c)!\,(N-L)!\,N_c}{N!\,(N-N_c-L+1)!} = \frac{N+1}{N_c+1}$$

For example, if a network of $N = 24$ splits in half, Demos only requires 1.92 elections on average to establish a new leader. As shown in Figure 3.5, even if only a small cluster of $N_c = 5$ separates, using $E = 3$ it likely already re-elects a new leader in the second round. We will see that these analytical results closely match our experiments presented in Section 3.6.4.

3.6 Evaluation

Demos is designed to provide high robustness and flexibility even under challenging conditions. To examine the protocol in action, we test (*i*) its ability to maintain cluster coordination despite node and link failures, (*ii*) the adaptivity to a fluctuating network topology that separates and combines clusters, (*iii*) the performance of the quorums introduced in

Section 3.4.1, and (*iv*) Demos' energy efficiency with its novel cluster coordination and discovery mechanisms.

3.6.1 Implementation

To support a wide coverage area using LoRa and also provide more efficient GFSK communication if a shorter range suffices, we utilize a Semtech SX1262 RF transceiver [Sem23] driven by an STMicroelectronics STM32L433CC microcontroller. Operating in the 868 MHz band, its broad TX power range from −9 to +22 dBm enables further adaptation to the targeted communication range. For our indoor tests, we use the GFSK modulation at 250 kbps and a TX power of +7 dBm.

Slot primitives. All-to-all communication during the `V` slot is based upon Chaos [LFZ13], with all nodes of a cluster probabilistically initiating the exchange of votes and demands. One-to-all floods, as employed in the `P`, `N`, `S`, and `D` slots, use Glossy [FZTS11] with consecutive transmissions for reliability [LDFST17]. For the discovery phase, one-to-all beacons are sent in Glossy slots with an increased RX duration. Multiple consecutive transmissions boost the discovery reliability and ensure overlaps with listening slots of other clusters in the discovery phase. Furthermore, beacon flooding quickly spreads information when multiple clusters converge simultaneously.

Randomization. Demos integrates two different sources for randomization, a built-in hardware unit that produces truly random output as well as a deterministic generator yielding pseudorandom numbers based on a given seed value. The former ensures that the usage of the Chaos sub-slots [LFZ13] differs between nodes and data can be exchanged, while the latter initially generates a fixed randomized sequence for designating leaders and the mapping of leader IDs to frequency channels. The designated leader is then selected in each election e using a sequence offset $(E-1)\cdot r+e$, where E denotes the number of elections per round and r is the round counter. Before each discovery phase, the deterministic RNG is re-seeded based on the ID of the cluster coordinator and r to locally compute the slot assignments for discovery.

Research artifacts. We provide the source code of Demos as open source [BDFK23a]. Additional test scripts grant extensive control of experiments on FlockLab [TDFS+20] by dynamically activating nodes at run time, offer the ability to easily sweep parameters, and visualize results to study their impact.

3.6.2 Experimental Setup

Test scenario. To test Demos under realistic conditions in confined spaces where non-line-of-sight conditions partition the network into clusters, we utilize the FlockLab testbed [TDFS+20]. 24 nodes are spread across the floor of an office building and provide fixed node locations. In addition, we employ a mobile node that is moved manually to dynamically change the network topology. We use four configurations to examine Demos' effectiveness in a variety of scenarios. 12 nodes with a mean hop distance of 1.31 form the smallest network, while an extended set of 18 nodes has an average of 1.65 hops, and all 24 nodes communicate via 1.84 hops on average. Figure 3.6 illustrates the setup where the mobile node connects two clusters of fixed observers.

Protocol parameters. For our tests, we use a round period $T = 3\,\text{s}$. By default, we conduct $E = 2$ elections during the consensus phase, with each `V` slot split into 36 contention sub-slots to exchange votes and demands. A received demand is valid for up to $P = 10$ rounds. Each node requests 1 data slot per round and sends 20 bytes using 3 transmissions per flood in one of 30 `D` slots. The remaining $T_d = 0.56 \cdot T$ is available for discovery. Due to Demos' fast discovery and merging mechanism, we do not expect multiple clusters to converge simultaneously and therefore set $P_t = P_l = 0.5$ to minimize the discovery latency [MB01]. The clusters share 14 non-overlapping frequency channels for communication, with an additional one reserved for discovery. We set $\alpha = 0.5$ and $\beta = 0.33$ and find safe lower bounds $\gamma_o = \gamma_a = 0.9$ based on an empirical PRR exceeding 97.6 % for one-to-all and 93.1 % for all-to-all primitives in the three static configurations.

Baseline. To investigate the impact of Demos' novel cluster coordination and discovery mechanisms, we compare it to LWB [FZMT12], a well-known protocol that builds on the same one-to-all floods to exchange data. While LWB relies on a preconfigured *host node* that coordinates the network, it is one of the only WSN protocols that include a failover policy if the coordinator is unavailable. In case of a host failure, nodes independently hop across frequency channels and try to reach the fixed coordinator on this channel. As in Demos' `S` slot, the host distributes a schedule for a subsequent succession of data slots that correspond to Demos' `D` slots. Demand is registered using a contention slot and assumed to be constant for multiple rounds. Exploiting this limitation, the host sends the next schedule already at the end of a round and retransmits it at the start of the next one to boost the reception probability.

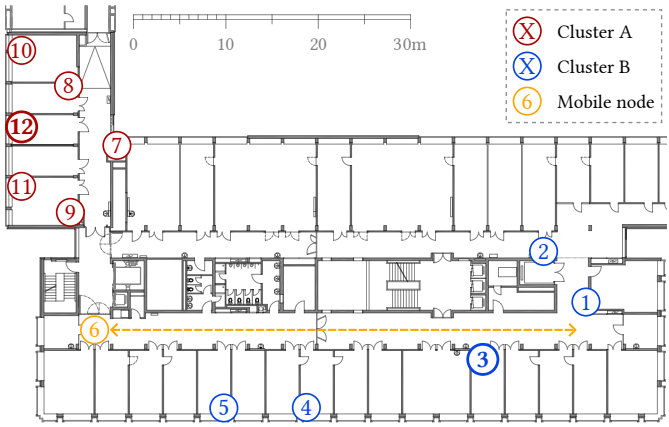

Figure 3.6: A mobile node traverses a corridor from left to right and then returns to its start position, thereby changing the network topology between two clusters of nodes. The two nodes marked in bold (nodes 12 and 3) represent the host ("H1") and failover host ("H2") for LWB.

3.6.3 Robustness to Network Topology Changes

To investigate how Demos reacts to changes in the network topology, we start with a static experiment and examine the protocols when the coordinator fails. In a second experiment, we instead include a mobile node and cause the network to first separate into two clusters and recombine again afterward. Throughout the experiments, we monitor the PRR, i.e., the percentage of received data packets compared to the maximal number of data packets that could have been received by all nodes in the network (i.e., sent packets times the number of receivers).

Scenario: Node failure. We employ 12 nodes distributed across the testbed as depicted in Figure 3.6. 11 of these nodes are static, whereas node 6 is mobile. While Demos uses the relative quorum Q_r, presented in Section 3.4.1, to elect its leader dynamically, LWB is configured to use node 12 as the host ("H1"), with node 3 serving as a failover ("H2") [FZMT12]. The timeout until nodes switch to the failover is set to 1 min.

In a first experiment, we keep node 6 static and let Demos elect a leader. After 60 s, we trigger a node failure by turning node 12 off and observe the behavior of Demos and LWB.

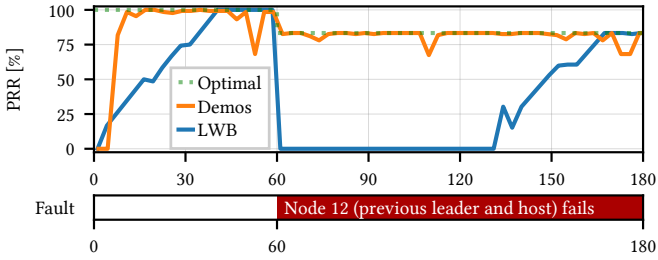

Figure 3.7: Failure of the host node halts LWB's communication until a failover takes over, while Demos instantly elects a new leader and approaches the optimal PRR at all times. Missed schedules cause minor PRR drops for both protocols.

Results: Node failure. Figure 3.7 shows the measured PRR of Demos and LWB as well as the optimal PRR over time. We find that all Demos nodes quickly discover each other and form a network, as Demos gathers and updates the current traffic demands instantly by aggregating them in the V slot directly preceding scheduling. At 60 s, we observe that the PRR dips as no more packets can be received by the faulty node 12. Even though node 12 was the leader as others merged with its cluster due to its high node ID, the remaining nodes directly elect a replacement when the previous leader is unavailable and seamlessly continue communication without a single missed round. LWB on the other hand adapts much slower to changing traffic demands, with maximally a single node sending its demand per round. The protocol further suffers from a catastrophic drop in PRR once the coordinator fails as no schedule is distributed anymore. After the nodes time out while listening for the host, the failover host slowly accumulates the demand again until LWB finally approaches the optimal PRR as well, 107 s after Demos.

Scenario: Link failure. In a second experiment, we investigate how the failure of an essential link between two clusters influences their data exchange. We let the nodes establish a network and start moving node 6 from left to right after 60 s. Once arrived at 105 s, node 6 remains stationary for a minute before returning to its start position which it reaches at 210 s. This recombination is particularly demanding as the clusters are uncoordinated and may interfere with each other.

Results: Link failure. In Figure 3.8, we observe that Demos quickly establishes a single network consisting of all 12 nodes as in the first experiment.

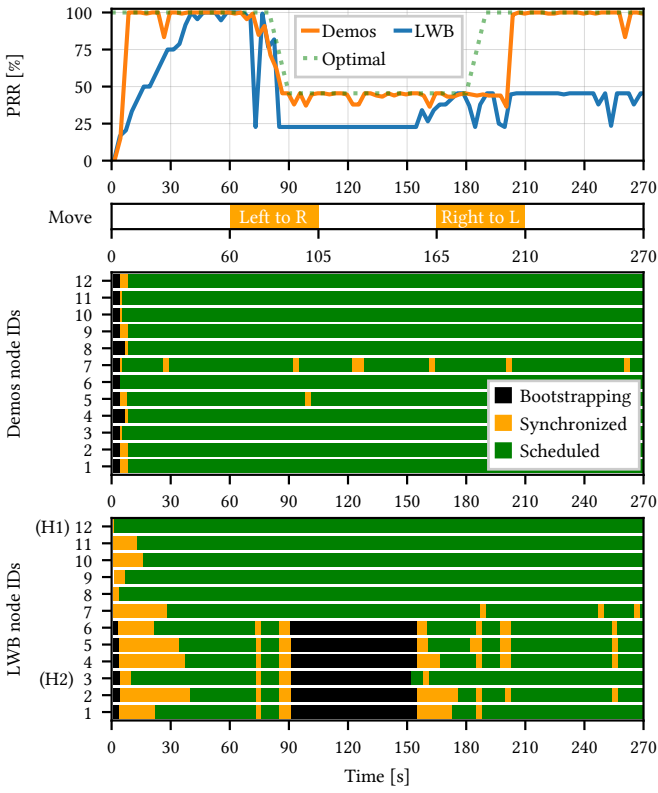

Figure 3.8: The move of node 6 causes the initial network topology to split into two clusters. Demos (middle) elects a cluster coordinator for each of them after separation and maintains optimal PRR by recombining the clusters when node 6 returns to its start position. LWB (bottom) only forms a second cluster after prolonged bootstrapping and cannot utilize the restored network topology to maximize the PRR as it lacks a discovery mechanism.

Once node 6 starts to move down the corridor at 60 s, its connectivity with cluster A (nodes 7–12) diminishes. As cluster B (nodes 1–6) relies on this link to communicate with cluster A, the reduced link quality results in a gradually decreasing PRR. At 88 s, we find that the link between the two clusters has completely failed and they can only communicate internally at maximally $\frac{2 \cdot 6 \cdot (6-1)}{12 \cdot (12-1)} = 45\,\%$ PRR. By directly electing a leader for cluster B, Demos matches this new optimal performance immediately

after the complete split. This success is only made possible by our novel quorums, as nodes in both clusters could not win a classical majority vote that requires more than 6 nodes. The plots in Figure 3.8 tracing the states of the protocol illustrate that Demos maintains its data exchange without interruption in contrast to LWB, as can be seen in the nodes of both Demos clusters continuously being scheduled. Once node 6 returns to its former position, the two clusters quickly discover each other. Leveraging cluster discovery, all nodes merge simultaneously, visible by the PRR surging back to the maximum at 206 s. LWB takes much longer to activate its failover policy after the clusters split apart. After a minute during which cluster B attempts to bootstrap and only cluster A can communicate at 22 % PRR, node 3 starts as a failover host at 157 s. Subsequently, cluster B gradually ramps up its data exchange on a separate frequency channel. As LWB lacks continuous cluster discovery like all WSN protocols apart from Demos, it is unable to merge clusters even after the original network topology has been restored and remains limited to less than half of the optimal PRR.

3.6.4 Dynamic Cluster Coordination

Next, we compare the three quorums Q_a, Q_{a+}, and Q_r introduced by Demos in terms of election latency, stability, and energy costs. In Section 3.5, we determined an expected election latency of $\mathrm{E}[L] = \frac{N+1}{N_c+1}$ elections until a designated leader may propose. Based on the subsequently cast votes, the quorum then stipulates whether the election was successful. Hence, $\mathrm{E}[L]$ is a lower bound for finding a new leader.

Scenario. To examine the differences between the quorums, we use $E = 3$ elections per round and test networks of 12, 18, and 24 nodes. After 30 s without disturbance, we cause two-thirds of the network, including its coordinator, to fail. During the next 30 s, we observe when elections occur and how stably the network performs under the new conditions. We run 60 tests per combination of quorum and network size and always mutate the sequence of designated leaders.

Results: Latency. As the classical, fixed quorum $N/2$ [Lam01, PANL19] can never be satisfied with $N_c = N/3$, we do not further consider it for this evaluation. Figure 3.9 depicts the average number of elections until a new leader could be established for the absolute and relative quorums. As expected, we find that Q_a requires multiple rounds until the estimated cluster size N_c is low enough for a node to gather a majority of votes. This latency slightly increases with the total network size, from

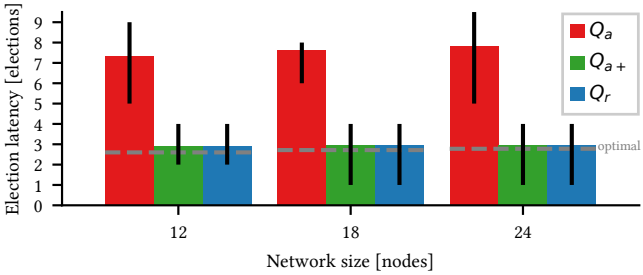

Figure 3.9: Using only the estimated cluster size, Q_a takes multiple rounds after two-thirds of the network become unavailable until the estimate of N_c is sufficiently low for a successful election. Leveraging the high reliability of concurrent transmissions, Q_{a+} and Q_r match the optimal election latency (gray). Black bars show the 25th and 75th percentile around the mean of the empirical distribution of the 60 test runs for each bar.

Quorum	Q_a			Q_{a+}			Q_r		
Network size	12	18	24	12	18	24	12	18	24
Pre-fault er.	0	0	1	0	1	0	0	0	0
Post-fault er.	13	6	4	0	1	1	0	0	0
Total errors	**24 (13.3 %)**			**3 (1.7 %)**			**0 (0.0 %)**		

Table 3.1: Instabilities in the first 30 s ("pre-fault") occur less often than fluctuations thereafter ("post-fault"). Q_r demonstrates excellent robustness without a single flawed election in any of its 180 test runs.

7.33 elections for 12 nodes to 7.83 for 24 nodes. On the other hand, Q_{a+} and Q_r can immediately find a new cluster coordinator once the previous leader becomes unavailable. We discover that both need only 2.90 elections on average for 12 nodes and 2.95 elections for a network of 24 nodes. These numbers further validate our formal analysis from Section 3.5 experimentally, from which we would expect at least 2.6 elections and 2.77 elections respectively.

Results: Stability. We encounter only a single run using Q_a (0.56 %) in which a previously elected leader is not successfully re-elected during the steady first 30 s as it could not gather enough votes to pass the quorum. However, due to the delay in estimating the new cluster size after the node failures, the first elected node cannot preserve its leadership and loses it again

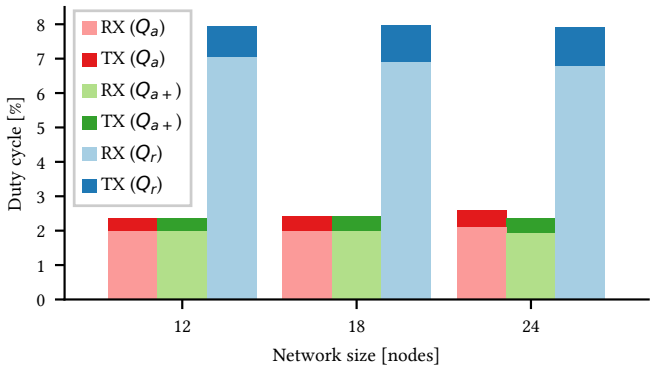

Figure 3.10: When using the absolute quorums Q_a and Q_{a+}, the rest of the consensus phase may be skipped after voting. In contrast, nodes employing Q_r need to participate in each election to ensure that the fraction of votes remains correct.

within a few rounds in 12.78 % of runs using Q_a. As seen in Table 3.1, this fluctuation predominantly affects the smallest network of 12 nodes where a single missing vote is often decisive and is quickly preceded by stable elections thereafter once N_c has converged. Q_{a+} is much more robust, with only 1.67 % of runs encountering two simultaneous leaders that both temporarily fulfill the quorum and immediately merge. Lastly, Q_r did not experience a single instability or unexpected loss of leadership in 180 runs.

Results: Costs. To analyze the energy costs of the cluster coordination mechanism, we combine the reception (RX) and transmission (TX) costs of the `P` and `V` slots in Figure 3.10. As expected, nodes using the absolute quorums Q_a and Q_{a+} can leverage that they only have to participate in a single election per round, after which they can skip the remaining two elections. Q_r on the other hand depends on the participation of all nodes, as receiving information on previously cast votes is necessary to prevent multiple leaders based on the fraction of votes in favor. This requirement results in an overhead that scales linearly with E, as seen in Q_r's costs which are three times higher compared to Q_a and Q_{a+}. Due to the excellent scalability of the all-to-all primitive, doubling the number of nodes from 12 to 24 merely increases TX costs by 25.8 % and Demos' total overhead remains almost identical.

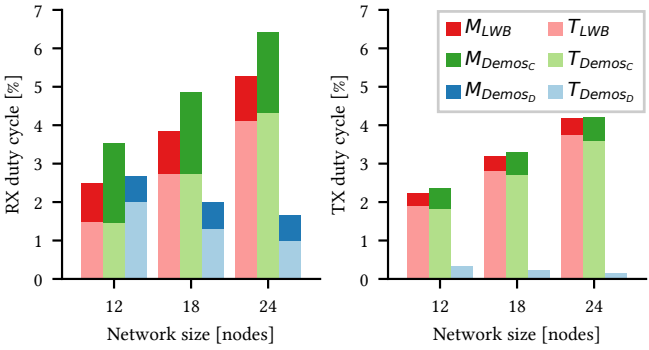

Figure 3.11: Comparing the management (M) and transport (T) duty cycles of LWB, the consensus and communication part of Demos ($Demos_C$), and the discovery scheme of Demos ($Demos_D$), we find that the energy overhead quickly diminishes for larger networks even for low data packet rates.

Verdict. We observe a trade-off between robust elections and energy efficiency. Q_a does not require assumptions on the communication reliability and reduces its energy consumption by only voting once per round in most cases, but takes longer to succeed after network changes and suffers from occasional instabilities while N_c is re-estimated. Q_r on the other hand is the most stable quorum and does not rely on an accurate estimation of the cluster size, but requires nodes to participate in each election even if they already cast their vote. We find that Q_{a+} strikes a good balance by matching the relative quorum in its low election latency while only requiring participation in one election per round.

3.6.5 Comparing the Costs of Robustness

While we have seen in Section 3.6.3 that Demos excels in reacting to a node failure through its cluster coordination mechanism and facilitates cluster merges through its cluster discovery scheme, these features introduce additional costs. Therefore, we investigate the energy overhead of Demos and its components and compare it to LWB [FZMT12] as a baseline.

Scenario. We test both protocols for networks of 12, 18, and 24 nodes. We run each configuration for 15 min and skip the first 30 s to only observe steady-state behavior. Demos employs Q_{a+} as a quorum based on the verdict in Section 3.6.4.

Results. In Figure 3.11, we separately look at the consensus and communication part of Demos ($Demos_C$) and its discovery scheme ($Demos_D$). We further differentiate between management traffic M (such as the `P`, `V`, and `S` slots for $Demos_C$ and the `N` slot for $Demos_D$) and the transport of data T (`D` slots for $Demos_C$ and the discovery phase for $Demos_D$).

As both LWB and Demos schedule equivalent traffic demands and use the same underlying one-to-all floods, the transport costs match. The management overhead of both protocols slightly rises with the network size as more information has to be shared and the average hop distance increases, prolonging listening until a packet reaches a node. However, we find that even though Demos updates the complete cluster information each round, this merely increases RX time by 92.3 % and TX time by 51.1 % on average compared to LWB. Note that the current data rate at 1 packet per round is minimal; in a real scenario, the demand to transport data is usually significantly higher [HYM16]. As the discovery slots are distributed across nodes using Demos' concept of cluster discovery, even the current maximal rate of continuously transmitting ($P_t = 0.5$) or listening ($P_l = 0.5$) is dominated by consensus and communication. If slower discovery is permitted or if discovery can be temporally discontinued in a static scenario, these costs diminish even further.

3.7 Related Work

Robust networking. While Demos offers an exceptional degree of resilience, many protocols can also adjust to changing conditions to varying extents. LWB [FZMT12] centrally schedules a series of Glossy floods [FZTS11], but adapts slowly and suffers from collapsing data exchange and network splits when the coordinator is unavailable. Chaos [LFZ13] and Mixer [HMZ18] only require a coordinator for synchronization but fix the amount of information that a node can share. Hybrid [SBDM20] and Harmony [MZL+20] leverage a combination of Glossy and Chaos slots to improve robustness. However, Hybrid only collects data at the coordinator, while Harmony can only enable fixed assigned slots and cannot cope with prolonged network splits. Instead, Demos supports the delivery of an arbitrary number of packets to any node and can merge clusters without restrictions. Furthermore, Demos dynamically elects a cluster coordinator and handles a node failure instantly without affecting the rest of the network.

To boost the reliability of floods, Dimmer [PL21] and OSF [BGS+22] flexibly adjust link-layer parameters depending on the current conditions

and SmarTiSCH [YNB+22] alters the timing and frequencies of transmissions to mitigate interference. However, these protocols handle orthogonal issues to Demos and do not address a failure of the coordinator or the discovery of other clusters to maximize data exchange.

Coordination in WSNs. Consensus protocols such as OTR [BLG15], A^2 [ANDL17b], and WirelessPaxos [PANL19] agree on a value such as a leader ID without leveraging it to bootstrap further communication. Other protocols such as STARC [RPL20] require an explicit handover from a previous leader, which is infeasible after leader failures. Therefore, protocols based on CT either hard-code the coordinator [HMZ18, ITMP18, LFZ13] or exclusively elect during bootstrapping [ANDL17a, FZMT12]. Timeout-based detection of leader absence is prone to failure and hence is usually configured to multiple minutes [FZMT12], and thereafter takes dozens of seconds to elect a substitute [ANDL17a]. At the other extreme, STeC [BDFG+21] elects an ad-hoc leader each round but prerequires external synchronization and BUTLER [MBTZ22] constantly synchronizes but cannot coordinate a decentralized network. Classical leader election algorithms for ad-hoc networks are also inapplicable, as they require the availability of most nodes of a known network [AKS15], guaranteed message delivery [SPZE20], at least dozens of network floods [AKNR13], or unrealistic restrictions on message propagation [CRW11]. Demos' cluster coordination avoids timeouts through continuous re-elections, handles failures with minimal delay, and supports arbitrary network splits.

Neighbor discovery. Both randomized [MB01, VAGT13] and deterministic [DC08, QLXL16, SYWL14] discovery protocols focus primarily on pairwise discovery [CB16]. As packet collisions increase with network size and impede discovery [KPSV17], self-adapting cycles [CW18] and the probabilistic skipping of transmissions [GWS+19] preserve a low discovery latency even for dense networks. Demos solves this issue by integrating nodes into clusters and allotting transmissions across its members to mitigate collisions. Group-based discovery shares discovery schedules to assist neighboring nodes [CSG+16, WSC+18] or deliberately de-synchronizes them to boost the group's discovery success [MQRW22]. Sharing information on discovered nodes [CK11] and a symbiosis of group management and discovery [PPL11] are concepts that are also found in Demos. Demos extends these concepts so that entire clusters of nodes can efficiently run cluster discovery while continuing to exchange data, and adds synchronized merging in combination with cluster coordination to consistently maximize network connectivity.

3.8 Summary

This chapter introduced a highly robust protocol to orchestrate autonomous clusters in low-power WSNs. By combining a novel cluster coordination mechanism based on constant elections with a continuous cluster discovery scheme that leverages cluster information to reduce discovery latency and energy costs, Demos supports highly mobile networks and persistently maximizes connectivity between all nodes. It does so without the additional complexity of a dual-radio architecture as presented in Chapter 2 and further improves the robustness after a failure of the leader. The latter is achieved through the introduction of three new quorums to determine a cluster coordinator, with which we show to effortlessly handle node failures and network splits. By instantly merging clusters, we demonstrate that Demos achieves the optimal PRR and gracefully reacts to changes in the network topology. With its high robustness, Demos serves the vision of dynamic, mobile clusters that operate autonomously and automatically recover from any node or link failure.

However, when trying to leverage this newly-found autonomy for WSN deployments that aim to provide long-term monitoring, we quickly find that current approaches come at a cost. As Chapter 2 and Chapter 3 rely on continuous discovery and periodic coordination, they constantly consume energy independently of whether an actual event of interest has occurred and whether data is available. In Chapter 4, we hence explore a fundamentally different approach to achieve autonomous formation that instead only bootstraps an ad-hoc network when it is truly needed and thereby entirely avoids the complexity of changing conditions and failures.

4

Exploiting Spatial and Temporal Correlation for Event-based Communication in WSNs

Low-power WSNs have demonstrated their potential for the detection of rare events such as rockfalls and wildfires, where rapid reporting as well as long-term energy-efficient operation is vital. However, current systems require periodic synchronization to maintain network coordination, heavily rely on node placement, or use costly long-range links to infrastructure. While the protocols introduced in Chapter 2 and Chapter 3 permit the autonomous formation of clusters, they require cluster discovery. This periodic communication negatively impacts their energy consumption in times of low environmental activity and correspondingly small demand to transmit sensed data. We will find in Chapter 5 that despite obviating discovery, maintaining an autonomous network generally depends on regular consensus and constantly drains batteries even when no application data is exchanged. In this chapter, we therefore explore whether we can entirely eliminate such periodic overhead for both cluster discovery and coordination if we directly leverage sensor signals to bootstrap ad-hoc clusters, thereby performing purely event-driven formation and avoiding any network-driven communication.

We present STeC, a novel wireless communication design that directly exploits the spatial and temporal correlation of signals from the sensed

This chapter is based on [BDFG+21], [BDFGG21], and [Gat20].

phenomenon to orchestrate event-based communication. We leverage the locality of a co-detection, where a physical event triggers multiple sensors quasi-simultaneously, to efficiently collect, characterize, and report sensor data. This eliminates the overhead of periodic network activity and centralized control, resulting in more energy-efficient communication with a lower, more consistent detection latency. By exploiting correlated sensor signals for autonomous networking, we propose a fundamentally new approach to avoid the elementary conflict between duty cycle and latency requirements immanent to synchronous protocols. Experiments using real-world traces show that STeC reduces the detection latency by up to 87 % compared to standard single-hop communication and outperforms traditional schedule-based methods by up to 58.4 × in energy efficiency.

4.1 Introduction

Motivation. WSNs are inherently suited for long-term observations with rare activity and have been employed for intruder [WDWS10] and sniper detection [SML+04], wildfire alerts [LWS06, YNL+12, GYR+20], and volcano monitoring [WAJR+05, WALJ+06]. In these scenarios, a physical event originates at a source and propagates through the environment until it reaches individual sensors with variable propagation delays and signal intensities, where it triggers a sensor event. This location dependency complicates the subsequent data analysis, as the distinction between a close physical event with small magnitude and a far-off but large physical event requires the relative interpretation of data inside the network. Therefore, including the correlation of spatial and temporal patterns is essential for data interpretation: The *co-detection* of multiple sensor events that are in proximity and occur within a bounded time frame permits conclusions on the intensity and location of the physical event [WBM+17]. These characteristics motivate event-based data collection that incorporates application-specific needs and adapts to events by filtering irrelevant data as shown in Figure 4.1. In this chapter, we leverage the physical characteristics of signals propagating to distributed sensors for (*i*) *establishing an ad-hoc network without prior coordination* and (*ii*) *efficient local processing and filtering*. While not all WSNs and observed phenomena show the above signal pattern, some do. We exemplify them with a case study of seismic event detection on an alpine rock glacier, in which we demonstrate the benefits of exploiting the spatial and temporal correlation of signals originating from a single source for synchronization and data aggregation.

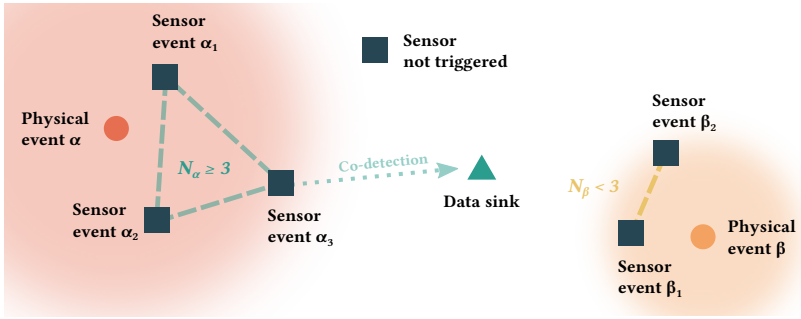

Figure 4.1: A physical event σ generates sensor events σ_m at triggered sensors m, which form an ad-hoc network of size N_σ for sensor events arriving within an interval ΔT. Physical events are locally filtered and only reported if they fulfill application-specific requirements. For event α, N_α reached a threshold of 3 nodes per co-detection and is reported to the data sink, while β is suppressed.

In the last two decades, WSNs have greatly benefited from more efficient sensing and communication hardware as well as advances in the protocol design for reliable multi-hop networks. However, the majority of these efforts have focused on continuous sampling and leverage extensive duty cycling in-between active phases to prolong system lifetime. While classical environmental monitoring applications such as bedrock [WBDF+19, GBG+21], structural [ABC+20], and agricultural [WC19] monitoring are well-suited for regular measurements, ephemeral and sporadic events such as sudden shifts in river sediments [DLPP20] or rockfalls [MFCP+19] demand more reactive protocols. This is of particular importance for rare events occurring after months or years of operation, for which classical solutions such as oversampling or event-triggered adaptation [SDFG+17] quickly deplete the available energy resources of battery-powered devices.

Challenges. A communication scheme targeted at rare event detection must solve four individual challenges. First, such protocols are primarily focused on optimizing detection probability and delay [HSR16] while maximizing lifetime. The latter requirement fundamentally contradicts traditional duty-cycled approaches, where an inherent trade-off between the reactivity and system lifetime is unavoidable. To achieve both simultaneously, we propose to refrain from any periodic activity and purely communicate event-driven. Second, as sensor events in our scenario occur quasi-

simultaneously due to the synchrony of event detections [HSR16, SDGJ08, WDWS10], previous asynchronous protocols have suffered from decreased performance due to packet collisions. In contrast, we leverage the temporal correlation of events to coordinate data collection and permit relative, local filtering of events. Third, complete network activation following a sensor trigger, either time-triggered [SDFG+17, IMPR16] or through wake-up circuitry [SBBT15, PIM17], is highly inefficient for events that predominantly concern a small subset of nodes within proximity of the physical event. By inherently restricting activity to nodes that have received a sensor trigger, we exclude nodes that are not involved in the co-detection of events, thereby making energy consumption exclusively dependent on a node's local activity. Fourth, as only a subset of sensor nodes is activated by an event, short-range links can be insufficient to connect to a distant data sink for reporting. We solve this limited reachability by leveraging the capabilities of modern radios to support both short- and long-range links within a single chip set.

Approach. We propose to incorporate event detection based on distributed sensors as an integral part of orchestrating WSN communication. If events are received by multiple sensors in close proximity w.r.t. space and time, nodes can leverage the sensor signal for synchronization. Following an event trigger, nodes activate their radios and form an ad-hoc network that enables the characterization and reporting of co-detections with minimum overhead. Communication only occurs as events appear and is bounded by their physical characteristics, effectively alleviating the elementary conflict between duty cycle and latency inherent to other reactive schemes. Through local processing and filtering, irrelevant events are suppressed by an application-specific filter, such as a threshold on the number of nodes or the signal intensity. Only events of interest are compressed and reported to the data sink. Our scheme ensures that at each step (synchronization, local aggregation, and reporting), energy is only spent on events deemed relevant, and that such critical events are detected with minimal delay.

With STeC, we make the following core contributions:

- We show that local aggregation of information in short-range ad-hoc networks enables us to filter the majority of events and improves energy efficiency, discussed in Section 4.3.1.
- We present a unique design for low-power communication protocols to reduce detection latency through a symbiosis of short- and long-range radio links in Section 4.3.2.

- We demonstrate that exploiting the correlation of physical events both in the spatial and temporal domain can benefit communication schemes greatly, as shown in Section 4.3.3.
- We develop methods to reliably discover and synchronize active sensor nodes in Section 4.4. A novel scheme is introduced to efficiently establish an ad-hoc network and leverage this knowledge to suppress futile data acquisition.

In Section 4.5, we present our open-source hard- and software implementation which we thereafter use for proof-of-concept testing. Based on real-world traces from a WSN deployment as well as synthetic events, we evaluate STeC in a variety of settings and validate its efficacy in a 7-month deployment for rock glacier monitoring in alpine environments, presented in detail in Section 4.6.

4.2 Related Work

Duty-cycled wireless networking. Synchronous low-power wireless protocols have proven their utility in countless applications for long-term monitoring with recurring traffic patterns [BvRW07, MELT08, IBS+10, WC19]. Initiated by Glossy [FZTS11], CT leverage topology-agnostic, low-latency, and highly reliable network floods to efficiently disseminate data to all nodes [ZMS20, FZMT12]. While this concept has shown to work well for one-to-many [SDFG+17] and convergecast networks [IMPR16, TVL+20], it still requires periodic network synchronization and scheduling, thereby inherently trading off communication latency and energy consumption. STeC uses CT to quickly establish ad-hoc networks, aggregate data with low latency, and spread energy costs, but eliminates the dominating scheduling overhead.

Event-based communication. Previous efforts to avoid this trade-off have employed wake-up radios and leverage their extraordinarily low idle listening costs for long-term event-based communication [SBBT15, SFBT19]. Similarly, a combination of wake-up radios and LoRa radios has been proposed to reduce the costs of long-range links [PMMB18], whereby local clusters remain synchronized to the data sink. While this concept has been applied to environmental monitoring and has been shown to significantly reduce communication overhead, the requirement for additional, highly specialized hardware as well as the frequent erroneous wake-ups and the severely restrictive range of 45 m [SMB+15, PMK+17] with

and 15 m [SBBT15, CCB+13] without line-of-sight limits its applicability. Schemes such as CTP-WUR [BPS16], which combine secondary wake-up radios with primary radios to extend this range, depend on up-to-date topology information and hence also require periodic activation of the entire network. Without the additional cost and complexity of dedicated hardware, protocols have been mostly constricted to reducing communication overhead in times of inactivity [SDFG+17, SSB10, MEB09]. Local collaboration to prevent the high energy costs of long-distance communication by removing a false positive event detection early on has been investigated for monitoring applications [HSR16]. The formation of a local network for each event has been shown to benefit area surveillance through a distributed evaluation algorithm [WDA+12]. This prompt suppression of superfluous data is beneficial for reliability and robustness and permits significant energy and latency savings. In particular, the partial information of aggregate queries is gaining interest for low-power wide-area networks (LP-WANs) [GYR+20]. Instead of introducing additional system complexity and costs, STeC's novel sensing-based coordination directly leverages the underlying physical events to synchronize local ad-hoc networks and suppress events that do not match given criteria before transmitting them on LP-WANs. This permits us to both eliminate scheduled communication overhead and reduce event detection latency.

Sensing for environmental monitoring. While continuous digital sensing has been shown to achieve hundreds of hours of lifetime [DGA+05], such devices have idle sensing costs in the order of multiple mW. To avoid this, analog trigger front-ends have been previously introduced to discover seismic events more efficiently using geophones [JKD+07, MFCP+19] and a variety of other sensors [MMF+07]. However, communication was disregarded in these cases and data was assumed to be collected opportunistically or in periodic batches. Previous work on structural monitoring using seismic events requires periodic low-power listening [PHC04] and leverages a transmitter with unrestricted power to wake up the network on demand [LHV+09]. In contrast, STeC's communication design is inherently coupled to its sensor triggers and reveals a mutual symbiosis of both reducing communication costs based on sensor input as well as limiting sensing costs if local network traffic identifies an irrelevant event.

4.3 Design Goals and Principles

While only co-occurring events are relevant in the considered application scenarios, traditional protocols either spend excessive energy on directly reporting all events to the data sink for processing or regularly collect data with a high detection latency. With STeC, we develop an event-driven protocol that aggregates events in ad-hoc networks and forwards the joint co-detection to a central entity. We thereby exploit the spatial and temporal correlation of sensor triggers originating from a single physical event to efficiently co-detect events. In this section, we give an overview of STeC and its six protocol stages and introduce a formal model of event co-detection, which we illustrate in a real-world case study of seismic events.

Setting. We target a set of M resource-constrained sensor nodes which can locally communicate over *short-range* links within the propagation radius of the physical event. Each sensor node m is equipped with a sensing system that can trigger further processing of samples as well as communication upon detection of a sensor event. We suppose an event-driven hard- and software architecture, i.e., the node is primarily in a low-power mode but is activated by the occurrence of an event to characterize it. In addition, physical events that fit application-specific patterns must be reported with minimal detection latency to a data sink kilometers away that is only accessible via a *long-range* link. This data sink is placed at a location where resources are unconstrained and can serve multiple sensor clusters in the vicinity. In summary, sensor nodes are sensor-triggered and support two kinds of wireless links, namely *short-range* between themselves and *long-range* to a data sink.

4.3.1 Local Event-based Aggregation

To efficiently detect co-occurring events, we leverage three key insights: First, a communication scheme that exploits correlated sensor-initiated triggers instead of timers can inherently confine its activity to locally involved sensors and does not require periodic activation of the entire network. Second, because of the restricted time window during which a physical event causes sensor events, we can limit communication to a brief interval directly after a sensor event and return to sleep thereafter. Third, co-detections of multiple sensors are rare and therefore incentivize a scheme that quickly detects lone wake-ups of a single sensor node. Early termination enables us to abort sensing and communication for irrelevant sensor events which do not fulfill filtering criteria. STeC integrates these

insights to achieve energy-efficient co-detection and low-latency event reporting. While we give a detailed description in Section 4.4, we briefly sketch the short- and long-range communication scheme as depicted in Figure 4.3 in the following paragraphs.

Nodes remain in their low-power mode and are exclusively activated if their sensor detects an event. Through analog trigger front-ends, this can be achieved with a power draw in the order of µW. After the physical event has occurred, its signal propagates through the media and arrives at sensor m with a propagation delay of δ_m. As nodes are triggered depending on their individual distance to the source, this results in a *staggered wake-up* throughout the network; while some nodes might already be synchronized, others receive the trigger with varying delays. We resolve this correlated but asynchronous activation through a technique inspired by Low-Power Probing [MELT08]. As soon as the sensor receives an event, the node enables its radio and starts listening. Simultaneously, a timer is set, which causes the node that first received the event to be activated earliest and flood a wake-up signal after a maximal propagation delay ΔT, thereby ensuring that all nodes within reach of the physical event have been triggered. Listening nodes directly synchronize to the wake-up and can immediately join the ongoing network flood without requiring further coordination. To quickly detect a lone sensor trigger and avoid futile protocol stages, a node sends multiple packets according to a unique pattern as described in Section 4.4.1 so active sensor nodes mutually *discover* each other.

As each physical event can generate a different ad-hoc network, it must always be bootstrapped from scratch. To do so, nodes perform *leader election* by exchanging packets based on their ID as shown in Section 4.4.2. The leader then announces a schedule containing a slot per node in the network to tightly synchronize the ad-hoc network and *aggregate data*. After activated nodes have exchanged their sensor events, an application-specific filter determines whether the co-detected physical event should be *reported* to a data sink.

4.3.2 Ensuring Connectivity to the Data Sink

As a data sink is typically situated in a location with Internet access and sufficient energy to support continuous listening, reachability from all remote sensor nodes via short-range links is not guaranteed. Furthermore, multi-hop connectivity cannot be assumed even if the data sink is in the vicinity of the sensor network as the ad-hoc network only consists

of a subset of sensor nodes. To resolve this issue, we make use of the ability of modern radios to transmit with modulations for both short- and long-range links using a single chip set [Sem23]. In case the co-detected event has not been filtered following data aggregation, a node assigned by the leader reports the compressed event information to the data sink, as described in Section 4.4.3. This occurs on the long-range link with single-hop connectivity. Because STeC primarily uses short-range links and as most events are locally filtered, congestion on the long-range link is minimized. An acknowledgment from the data sink ensures the successful reception of a critical event and allows for retransmissions if needed.

4.3.3 Exploiting Spatial and Temporal Correlation of Events

In order to exploit the spatial and temporal correlation of sensor events, protocol parameters need to be adjusted accordingly. In our application scenario, each one originates from a physical event σ occurring at time t_σ. A sensor event σ_m can be described by a tuple

$$\sigma_m = (m, f, t_m)$$

where m denotes the detecting sensor, f the associated features (such as duration and signal intensity) and t_m the time when the event was triggered at sensor m. As the sensors are within proximity of the physical event, we assume a maximal propagation delay $\delta_m = t_m - t_\sigma \leq \Delta T$ and hence $|\delta_i - \delta_j| \leq \Delta T$ for all $i, j \in M$, where M is the set of sensor nodes.

We suppose that a set of sensor events is relevant if they occur close in time and space, as they are highly likely to originate from the same physical event. In such a case, we talk about *co-occurring* events. Given a set of sensor events Σ, we distinguish between co-occurring events of degrees d, where d corresponds to the number of nodes detecting a single physical event. Informally speaking, a set of d sensor events are co-occurring if they happen within a time interval ΔT, after which a physical event has reached all sensors. More formally, the set of all sensor events $\sigma_m \in \Sigma$ can be partitioned into subsets Σ_d for $1 \leq d \leq M$ using Algorithm 1, where each element of Σ_d is a set of exactly d co-occurring events.

The purpose of the sensor network is to focus on and report such co-occurring events to a data sink, where a high degree d corresponds to a major event of critical importance. As our application requirement is to report co-occurring events of degrees larger or equal to the reporting threshold D, subsets Σ_d for all $d \geq D$ must be detected. STeC's goal is

Algorithm 1 Partition of a set of sensor events into sets of co-occurring events.

Input: Set of events $\sigma_m \in \Sigma$ with $\sigma_m = (m, f, t_m)$; number of sensors M ; maximal propagation delay ΔT.
Output: Sets of co-occurring events Σ_d of degree $1 \leq d \leq M$.

1: $\Sigma_d := \{\}$ for all $1 \leq d \leq M$;
2: $t_0 := -\infty$; $d := 1$; $C := \{\}$;
3: **while** $\Sigma \neq \{\}$ **do**
4: determine event σ_{next} with smallest t_{next} in Σ ;
5: $\Sigma := \Sigma \setminus \{\sigma_{next}\}$;
6: **if** $t_{next} \leq t_0 + \Delta T$ **then**
7: $C := C \cup \{\sigma_{next}\}$; $d := d + 1$;
8: **else**
9: $\Sigma_d := \Sigma_d \cup C$;
10: $C := \{\sigma_{next}\}$; $d := 1$; $t_0 := t_{next}$;
11: $\Sigma_d := \Sigma_d \cup C$;

to efficiently and reliably report all relevant subsets above the threshold to the data sink. To achieve this, we aim for an energy-efficient and low-latency *distributed* implementation of Algorithm 1 directly within the sensor network. Note that the distributed algorithm does not distinguish between multiple simultaneous physical events and a single large physical event. Since in both cases, multiple sensors could be triggered at the same time, the root cause must be analyzed during post-processing.

Co-detection. If a set $C_d \in \Sigma_d$ is successfully communicated to the data sink, co-detection has occurred. To obtain such a set of sensor events that are associated with a single physical event σ, we exploit two sensing properties. First, as σ propagates, it triggers sensors in increasing order of their distance to the source. It therefore usually only activates a connected multi-hop network of sensor nodes and inherently leverages the *spatial correlation* of sensors. In case of blocked signal paths without line-of-sight, this assumption might be constricted to physical events within the range of wireless communication. Second, it does so with a propagation delay δ_m depending on its propagation velocity and distance to sensor m. This allows us to upper-bound the maximal delay ΔT based on the spacing of sensors and utilize the *temporal correlation* of sensor events. STeC's location- and time-based communication permits us to inherently focus on co-detections and filter out irrelevant events.

d	1	2	3	4	5	6	7	8	9+
$\lvert\Sigma_d\rvert$	4044	253	79	34	12	8	3	7	4
$\lvert\Sigma_d\rvert/\lvert\Sigma\rvert$[%]	91.0	5.7	1.8	0.8	0.3	0.2	0.1	0.2	0.1

Table 4.1: Most physical events are only received by a single sensor. Co-occurring events with a high degree are rare, as they require a physical event with a large magnitude to trigger multiple sensors.

Case study: Seismic event detection. As a real-world example of co-detections, Table 4.1 contains seismic events that have been collected by a network of $M = 32$ seismic sensors on a rock glacier at Dirruhorn, Switzerland, over a period of 17 months through a traditional schedule-based solution [SDFG+17]. A maximal propagation delay $\Delta T = 100\,\text{ms}$ [FFBV19] and $|\Sigma| = \sum_{d=1}^{M} d \cdot |\Sigma_d| = 5154$ sensor events result in a total number of $\sum_{d=1}^{M} |\Sigma_d| = 4444$ detected physical events. Previous work [FFBV19, MFCP+19] has demonstrated that event co-detection is essential for the recognition of imminent catastrophic failure of the underlying rocks and hence critical for early warning systems. In this scenario, gravitational slope failure high above the valley floor must be reported with minimal detection latency to installations further below that can then restrict access to roads and rail lines or initiate the evacuation of populated areas.

Status quo. As shown in Figure 4.2, a co-detection with a high degree occurs sporadically and does not require the periodic activation of the complete network, as many sensors experience long intervals without receiving any sensor event. If traditional schedule-based data collection is employed, co-detections are only obtained retrospectively during the post-processing of all data, resulting in an inefficient over-reporting of events and significant delays due to the periodic data aggregation. This follows as current protocols are agnostic to the underlying event characteristics and do not exploit their spatial and temporal characteristics. As an example, with a reporting threshold $D = 6$, up to 99.5 % of all physical events could already be filtered through event-based local processing.

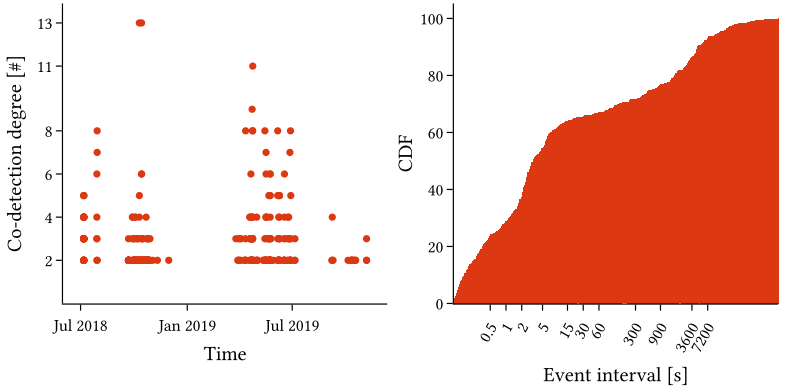

Figure 4.2: The co-detection degree d varies significantly over time for our case study, with up to 13 nodes being activated by a single physical event. We find that events frequently appear in clusters, with 50 % of physical events occurring with an event interval of less than 3.35 s after a previous one. 23 % of physical events are separated by more than 20 min and show long periods of inactivity.

4.4 STeC in Detail

After having presented the principles behind STeC in Section 4.3, the following section dives into the details of the scheme as shown in Figure 4.3. We first demonstrate how to obtain coarse-grained synchronization from asynchronous triggers and efficiently detect lone wake-ups. Thereafter, we present a time-efficient bootstrapping method for an ad-hoc network and illustrate how to reliably exchange sensor data and efficiently report it to a central entity.

4.4.1 Synchronization and Mutual Discovery

Staggered wake-up. STeC exploits sensor triggers for asynchronous wake-ups. The individual propagation delays δ_m result in a *staggered wake-up* with sensor nodes receiving triggers one after another. To sufficiently synchronize for network communication, we must first obtain a common time reference. As the maximal propagation delay ΔT depends both on the spacing of the network and the propagation velocity of the media, this value is deployment-specific. While some events such as smoke spread ($5\,m/s$ [ABK+19, KHY98]) or explosions ($343\,m/s$ [SML+04]) are relatively slow, even seismic waves propagating through rock at up to

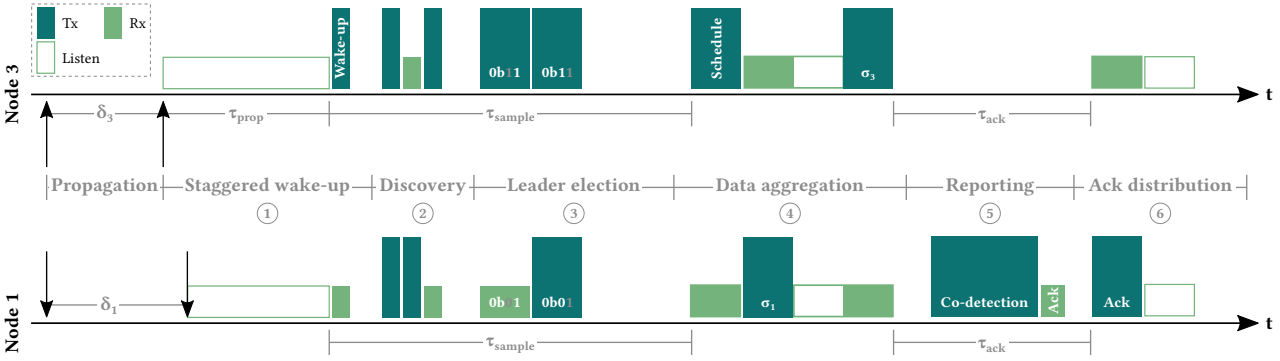

Figure 4.3: STeC consists of six protocol stages: After a physical event has occurred, a sensor event σ_m is triggered at each sensor m with propagation delay δ_m. **(1)** The node that receives a trigger with the shortest δ_m listens for the delay τ_{prop} before sending a wake-up signal for initial synchronization. **(2)** Nodes efficiently detect whether there are other neighbors nearby and **(3)** elect a local leader. **(4)** After having sampled the event for at least τ_{sample}, the sensor events are aggregated in the ad-hoc network. **(5)** If deemed relevant, the assigned reporter finally sends the co-detection to the data sink and **(6)** optionally distributes returned commands and timestamping information.

4000 m/s [ES07, GHL+20] still result in maximal propagation delays up to 100 ms [FFBV19]. For wireless communication, which requires millisecond-level accuracy to efficiently exchange packets, directly synchronizing on sensor triggers is insufficient.

Therefore, STeC uses an additional radio wake-up signal to obtain timing accuracy in the order of single milliseconds independent of the event type. When a node receives a sensor trigger, it starts listening on its radio. In parallel, a timer is set to initiate the transmission of a wake-up signal after a delay $\tau_{prop} = \Delta T$. This ensures that even if a sensor is co-located with the physical event source, all nodes within event reach have already been activated before a wake-up signal is sent. If a node receives a wake-up, it cancels its local timer and synchronizes to the packet. The maximal time uncertainty within the ad-hoc network at this step is solely determined by the radio-specific switching time t_{switch} from listening to preamble transmission, with $t_{switch} = 1.15$ ms for our implementation. This follows as multiple nodes can be close together and start their own signal transmission just before being able to receive the one of others. Notice that as only the detection of a preamble is needed, the wake-up signal can still be recognized even if packets are sent concurrently. After this stage, active nodes have reduced the timing uncertainty due to the asynchronous wake-up to single milliseconds. However, as multiple nodes can detect an event quasi-simultaneously, the wake-up signal itself does not yet provide a unique time reference for reliable synchronous data aggregation and leaves sending nodes uncertain whether others are also active.

Discovery. Because 91 % of physical events only trigger a single sensor, as shown in Table 4.1, most data exchanges can be preemptively terminated if a lone wake-up is detected. Therefore, we seek to quickly assert the presence of other nodes, as subsequent protocol stages are otherwise executed in vain. Additionally, a single wake-up transmission only permits time synchronization for single-hop networks. However, multi-hop communication must be supported due to challenging environments with non-line-of-sight links as sensors are close to the ground and are distributed over a large, diverse area.

To solve both issues, we propose a novel discovery mechanism called *DiscoSync*, shown in Figure 4.4. Instead of transmitting only a single wake-up signal, each activated node executes an individual pattern of signal transmissions (Tx) and receptions (Rx). Because nodes synchronize their pattern based on received signals from others, the wake-up signal floods the network across multiple hops. As the Tx/Rx pattern is randomized

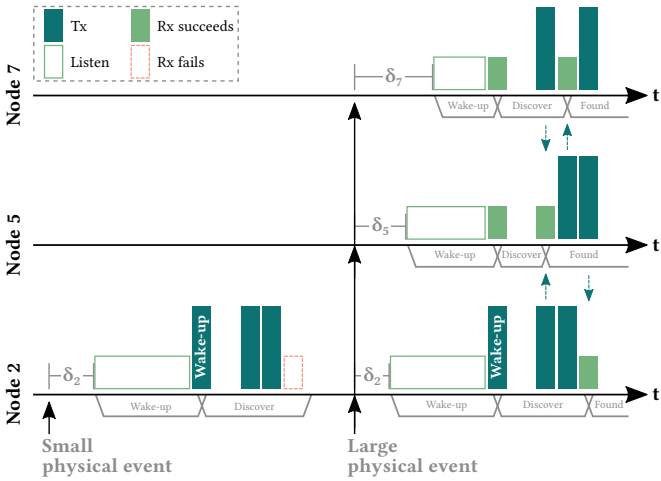

Figure 4.4: DiscoSync uses a balanced number of Tx and Rx slots to maximize the probability of mutual detection, enabling nodes to efficiently discover that a co-detection is occurring. If no reception succeeds in any slot, a lone wake-up has been asserted.

based on the node ID and differs for each node, a node has a high chance of overlapping Rx slots with a Tx slot of activated neighbors and detecting their presence. Through combinatorial analysis [BDFGG21], we find that 9 slots reduce the probability that a node in our case study with $M = 32$ nodes incorrectly assesses a lone wake-up to less than 0.1 %, with the failure probability decreasing even further for larger networks. In case more false negatives are tolerable, 5 slots already provide a reliability of 99.0 % and significantly increase the energy efficiency of STeC. Transmission occurs in the middle of a slot after t_{switch}, ensuring that slots sufficiently overlap despite the limited timing uncertainty of the wake-up signal. After *DiscoSync* has finished, sensor nodes have verified the existence of other activated nodes and synchronized to their transmissions with µs-accuracy, a prerequisite for subsequent network communication.

4.4.2 Ad-hoc Network Communication

With all nodes synchronized, we can start the exchange of sensor events. To efficiently disseminate data in a multi-hop network, we leverage concurrent transmissions (CT), initially introduced by Glossy [FZTS11].

CT use the principle of constructive interference, where the transmission of multiple equivalent packets can improve reception quality [ZMS20], as well as the capture effect [LF76], which allows receiving radios to switch to the strongest signal. STeC exploits CT's purely time-based mechanism to communicate instantaneously in an ad-hoc network. Unlike routing, CT do neither gather nor require neighbor relationships apart from an upper bound on the diameter of the overall network to propagate a packet to all sensor nodes. The fact that all nodes receive the complete set of data is later on leveraged to distribute the high energy costs of reporting over long-range links and increase system lifetime by avoiding overload on a single node.

Leader election. The use of CT requires both scheduling due to its TDMA nature as well as tight time synchronization so nodes can utilize their floods efficiently. A naive synchronization approach based directly on the wake-up signal would be fragile, as nodes might synchronize to different references, thereby preventing reliable data aggregation. To deterministically provide a unique time reference, the activated nodes in STeC first elect the node with the highest ID as the leader of the ad-hoc network, as shown in Figure 4.5. We efficiently achieve this through a binary search [CGR02, CD19, CKP12] by initiating synchronous floods of the multi-hop network based on the unique node IDs. This enables us to agree on a single reference in a multi-hop network with deterministic latency, in contrast to most leader election algorithms which either only work locally or terminate with a significant and often variable delay. To swiftly determine the highest active ID, $\lceil log(M) \rceil$ slots are utilized, where M is the number of nodes in the network. In the first slot, all nodes whose most significant bit is set initiate a flood. Other nodes whose current bit is not set relay the flood but withdraw themselves from ever starting future slots, as they know that they do not themselves possess the highest ID. With the relevant bit for initiating a flood shifting each slot towards the least significant bit, nodes start to gradually drop out until only the highest ID remains active.

Data aggregation. With a known leader, the ad-hoc network based on CT is now established. However, the data aggregation stage in Figure 4.3 does not follow instantly after leader election has finished. As the sensor itself requires time to sample sufficient data for the ensuing feature extraction, the schedule distribution starts with a delay τ_{sample} after the initial wake-up signal. This delay depends on the demands on the application-specific features f, which can for example include a minimum sampling time.

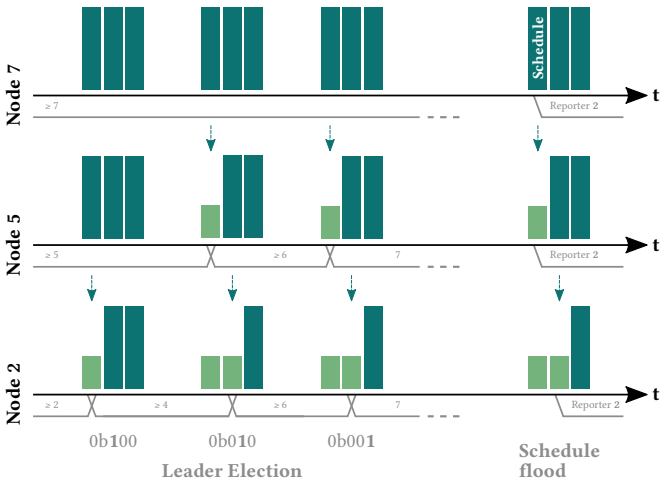

Figure 4.5: As nodes 5 and 7 have their most significant bit set, they simultaneously start floods in the first slot. As node 2 is aware of a higher ID, it does not initiate a flood in the second slot. However, it is assigned as the reporter by node 7 during the schedule distribution.

In the schedule distributed by the elected leader, each potential node in the network is statically assigned a data slot, resulting in M slots. An activated sensor node then floods its sensor event σ_m to all other nodes in the ad-hoc network during its corresponding slot. While this schedule might initially seem inefficient, co-detection only occurs sporadically and the actual number of activated nodes is unknown. Furthermore, a contention-based scheme could result in non-deterministic detection latencies which quickly soar for the most critical cases where many nodes co-detect a major event.

Suppressing irrelevant events. While we have previously used the correlation of sensor events, STeC also applies local knowledge of all current sensor events to finally determine whether the event should be filtered or reported to a central entity. This enables nodes to refrain from using energy-intensive long-range links until a relevant physical event that passes the filter arises.

Filtering occurs based on application-specific requirements and can include features such as the co-detection degree d, the difference in propagation delays δ_m, as well as signal-specific characteristics such as

event duration, number of consecutive triggers as well as extremal and averaged data points. As it has been shown that the co-detection degree is a viable surrogate for slope instability in our case study scenario of seismic events [FFBV19], STeC by default reports all events of degree d larger or equal to the reporting threshold D.

Exploiting communication for sample termination. The quick assertion of lone wake-ups and co-detections below the reporting threshold further allows us to reduce *sensing* costs. In the majority of cases where a physical event only triggers a single sensor, our implementation of DiscoSync determines within 138 ms that no further node is activated. If a co-detection occurs, STeC can aggregate data in a network of $M = 32$ nodes within 1.18 s in our implementation and afterward directly terminate sampling if the physical event is irrelevant. Combining these principles, we show in Section 4.6.2 that sensing costs can be reduced by 76 % compared to traditional systems which continue sampling futile data.

4.4.3 Bridging the Gap to the Data Sink

To save energy and reduce detection latency, short-range links are used for all protocol stages except for the final *reporting*. Due to the large distance between sensor nodes and the data sink, a more resource-demanding long-range link is required to directly communicate over a single hop and is therefore only used for relevant co-detections. This long-range link is similarly dependent on the occurrence of sensor events and always initiated from the sensor network, as only the data sink with its unconstrained energy resources can afford to be continuously listening for incoming traffic. The following paragraphs describe how we spread reporting costs across an unknown set of nodes and support bi-directional communication despite one-sided contact initiation by a single node.

Reporting. In traditional cluster-based networks, a cluster lead gathers the data from all associated sensor nodes and forwards their aggregate to the data sink [ATN14]. However, this significantly increases the energy consumption of the lead and results in a reduced lifetime. By exploiting that CT inherently disseminate the complete set of data to all nodes, we can delegate this communication overhead to any node in the newly-formed ad-hoc network. By randomly assigning a node per report which changes for each ad-hoc network, the energy overhead is distributed and the system lifetime can be significantly extended. While this works straightforward in persistent networks where the set of active nodes is known, the ad-hoc

network differs with each physical event and therefore does not permit assignment to a node known in advance. On the other hand, if the leader transmits the assignment after data aggregation, crucial events could be lost if this single packet does not successfully reach the reporter.

STeC solves this issue by having the node which is *closest* to an ID contained in the schedule report the physical event. To do so, the leader randomly chooses an ID out of the set of known node IDs and already includes it in the schedule even before nodes have communicated their events. This ensures that the reporter is uniquely defined within the network and only shares data if it is aware of its assignment, having received the schedule. Thereafter, each active node computes its difference to the assigned reporter ID, whereby a cyclic computation is used that wraps around from the maximal to the minimal ID. This ensures that each node has the same probability to be elected if the reporter ID is chosen uniformly at random. A node then compares its own difference to all other active nodes and determines whether it is closest. This scheme guarantees that at least one node will identify as the reporter even if not all packets are successfully received at each node, as the closest ID will never defer from reporting and nodes only participate if they successfully received the schedule containing the assignment.

Reaching sensor nodes. While long-range communication is only initiated by the sensor network in STeC, WSNs often also require bi-directional communication from the data sink to deliver information to the sensor network. The ability to adjust protocol parameters is indispensable for real-world WSN deployments [BISV08], as optimal settings are hard to estimate a-priori and may change over time. Duty-cycled networks can schedule periodic communication to accommodate these needs, but our purely event-driven scheme requires a more reactive approach. We utilize that acknowledgments are essential for LP-WANs to ensure that the reported data has been successfully received by the data sink, as the reporter must be able to detect the lack of a response from the data sink and retransmit the packet. STeC leverages this to send commands to sensor nodes by directly embedding them into the acknowledgment packet. In addition, the data sink also includes the current global time so nodes can synchronize their local clocks with each reported event.

Because only one node communicates with the data sink to reduce the energy consumption, the remaining ad-hoc network must be separately informed. In the last protocol stage, the *acknowledgment is distributed* as shown in Figure 4.3. All activated nodes wake up with a delay τ_{ack} after

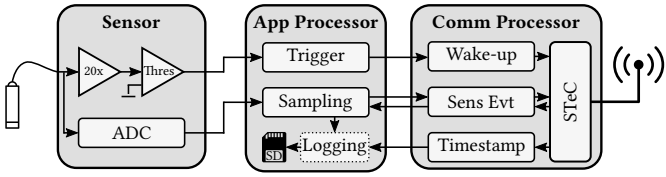

Figure 4.6: The sensor node consists of a sensor (left), an application processor (middle), and a communication processor (right).

the data aggregation has terminated to receive the response flooded by the reporter. A contention slot permits nodes to exchange final information before the ad-hoc network disbands and each sensor node returns to its low-power mode.

4.5 Implementation

To test STeC in realistic environments for a case study of natural hazard warning, we employ a wireless sensor platform consisting of a seismic sensor with integrated trigger and data acquisition front-end, a processor for sensor data and event processing as well as a dedicated processor to control radio communication as depicted in Figure 4.6.

4.5.1 Prototype Hardware

The seismic sensor [MFCP+19] consists of an ION SM-6 14 Hz Omni-tilt Geophone [ION21] which is connected both to MAX9019 comparators for dual-sided triggering as well as a MAX11214 analog-to-digital converter (ADC) to convert the analog signal to digital samples at up to 1000 Hz. An STMicroelectronics STM32L496VG MCU containing an ARM Cortex-M4F as well as 1 MB of Flash and 320 KB RAM is used as an application processor to interface with the trigger and perform the signal characterization. While the complete signal is logged onto an SD card, the extracted features are used to generate sensor event packets for a communication processor. To modularize and reduce system complexity, we follow the dual-processor paradigm [SZDF+15, BTF+19] and implement STeC separately on an STMicroelectronics STM32L433CC with 256 KB of Flash and 64 KB RAM. This MCU controls the heart of the wireless sensor, a Semtech SX1262 868 MHz LoRa transceiver [Sem23]. Supporting both GFSK modulation for short-range links ($<$ 1 km) as well as LoRa for long-range links and a

high Tx power of up to +22 dBm, this radio provides the opportunity for both efficient local communication as well as long-range connectivity to a data sink. The complete sensor node fits into a water- and shockproof aluminium enclosure measuring 160 × 100 × 81 mm and weighing 1552 g with a 13 Ah battery.

Research artifacts. We provide all our hard- and software products for STeC as open-source, including a custom discrete event simulator written in Python and all event traces used in the evaluation as introduced in Section 4.6.1. Access to our real-time network monitoring tools as well as all gathered sensor and communication data is publicly available at https://gitlab.ethz.ch/tec/public/stec [BDFGG21].

4.5.2 Making STeC Work in Practice

Ensuring precise timing information. To report gathered sensor events to the data sink, data from all nodes is combined into a single packet. However, timestamps based on the local time reference of nodes are subject to individual clock drifts of up to 20 ppm and hence 1.7 s/d and strongly depend on environmental conditions such as temperature [EFD+18] as well as manufacturing tolerances [SCF+08]. In contrast to duty-cycled networks, which can synchronize clocks periodically, STeC remains inactive for prolonged periods of time if events are rare. Therefore, it is inconclusive to compare local trigger timestamps as initially measured by each node. To avoid this, sensor nodes compute the time *difference* between their sensor trigger and the start of the data aggregation, as this is the first network-wide synchronized point in time. This extension of a concept introduced by the ETA algorithm [KDL+06] allows us to obtain relative propagation delays with sub-ms accuracy even if nodes have not been activated for days and have experienced significant clock drift since their last event. To still obtain absolute timestamps as a final data product, timestamps are then adjusted at the data sink, which has the ability to synchronize itself to global time through its Internet connection.

Supporting different propagation delays. While systems detecting events with a high propagation velocity such as light can benefit from a short staggered wake-up stage, some event types might incur longer propagation delays. To also support physical phenomenons with different characteristics such as smoke spread or motion, we can leverage the available radio capabilities which provide carrier activity detection (CAD) instead of constant listening during the staggered wake-up stage for up to

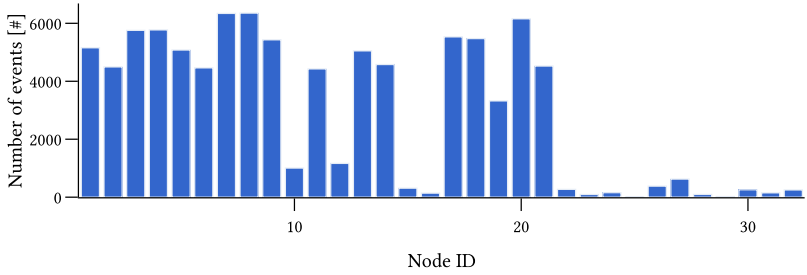

Figure 4.7: While some nodes gather up to 6351 sensor events over a period of 17 months on a rock glacier in alpine environments, others are primarily inactive and experience less than 100 triggers.

τ_{prop}. This specialized radio mode permits receivers to periodically poll whether a transmission is ongoing and avoid idle listening in-between, thereby leveraging the long preambles of modulations supporting long-range communication. We have seen in experiments that this technique would allow us to reduce listening costs by 75.2 %.

Requesting additional slots through contention. As sensor nodes only have a single opportunity to disseminate their event data, they are unable to communicate additional data after the data aggregation has started. As the latency of this stage increases with the network size and the ensuing reporting might take several seconds depending on the chosen modulation, nodes are temporally incapable of reacting to new sensor events. While we show in Section 4.6 that this limitation only results in a minor detection loss of 8.8 % of physical events that should be reported, STeC caters to this problem through its optional contention slot. By allowing nodes that have received sensor events after their scheduled data slot to inform others, the ad-hoc network can be kept alive and additional data aggregation can be scheduled. While this mechanism does not involve re-synchronization based on the newly gathered sensor triggers, we thereby exploit the fact that subsequent physical events of event bursts often only activate a similar subset of nodes as is already part of the ad-hoc network. This contention slot essentially permits nodes to periodically schedule slots during bursts of activity. In addition, it can be used to retransmit and re-evaluate if a node detects that its packet was lost during the data aggregation stage.

4.6 Evaluation

To demonstrate STeC's efficacy in detecting and reporting rare events, we first perform an extensive analysis of the communication scheme based on real-world traces from our deployment. Through simulations, we compare our method to both a standard single-hop protocol as well as a representative of state-of-the-art schedule-based multi-hop protocols for event-based WSNs w.r.t. (*i*) the *energy per event* and (*ii*) *detection latency*. The simulation results are validated using synthetic traces on a wireless testbed in order to expose different detailed aspects of our implementation. Subsequently, we present insights from our deployment case study where seismic events are detected in an alpine natural hazard detection application over the course of 7 months. Since validation against ground truth in our case study deployment is difficult, we have implemented a shorter 15-week test deployment in an urban setting where ground truth data has been captured in parallel.

4.6.1 Experimental Setup

Trace-based simulation. In order to analyze and compare protocols that fit our application scenario of rare event detection, we use a custom trace-based discrete event simulation [BDFGG21]. This setup enables us to evaluate months of protocol execution with various configurations. To correctly reproduce communication characteristics in simulation, we gather detailed energy and timing measurements of our hard- and software implementation (described in Section 4.5) using the FlockLab testbed [TDFS+20]. While our simulation supports low-power mode costs, we exclude them for the rest of this evaluation as they equally affect all protocols. At 1.7 µA, the low-power current draw of our implementation is comparable to the 1.4 µA that STeC uses on average with default settings. Our input trace consists of 80 675 physical events with a total of 92 803 sensor events over a period of 511 days from a deployment of $M = 32$ seismic sensors in the alpine setting presented in Section 4.3.3. This data set contains both days with and without rain to represent a realistic environment, as raindrops can trigger uncorrelated seismic sensor events. As shown in Figure 4.7, the number of events varies significantly between nodes and incentivizes a communication scheme that only requires the participation of active sensors to save energy.

Base lines. ALOHA [Abr70] represents an event-based protocol that does not rely on the formation of a local network and serves as a simple

reference for direct one-hop communication with a data sink. As it cannot filter events locally, nodes report each sensor event directly over the long-range link. To avoid packet collisions, we grant it the ability to perfectly detect ongoing transmissions and delay reporting until the channel is clear. To showcase how schedule-based schemes can use a local network for data aggregation, we choose eLWB [SDFG+17]. This protocol is designed for event-based communication via a persistent network and relies on periodic schedules for nodes to request slots and exchange data. Similar to STeC, eLWB relies on CT to flood packets but requires the entire network to be activated with a default round period of 15 s, whose influence we investigate later on, to remain synchronized and ensure adequate detection latencies. To simulate long-range reporting, we extend eLWB with STeC's reporter assignment scheme. Note that we compare higher-layer protocols and do not include wake-up radio schemes to avoid restrictions on range and hardware.

STeC parameters. As for the other protocols, we simulate STeC based on a real-world deployment of $M = 32$ sensor nodes and extract its characteristics from our implementation. Short-range links employ GFSK at 250 kbps, while long-range links use LoRa with spreading factor 11 and a bandwidth of 125 kHz, with both links transmitting at +14 dBm for reliable communication despite non-line-of-sight. We use a delay $\tau_{prop} = 100$ ms according to our scenario in Section 4.3.3 and 9 DiscoSync slots to detect other nodes with a probability exceeding 99.9 % and synchronize within 138 ms. A delay $\tau_{sample} = 100$ ms ensures that key signal features such as the peak intensity can be obtained. Thereafter, mirroring our implementation, we flood 65-byte sensor events using CT with three transmissions for robustness and compress each to 12 bytes for reporting after nodes finished data aggregation within 1.18 s.

4.6.2 Simulation-based Comparison

The main metrics of comparison between the protocols and their variants are the average energy spent for the communication of each observed sensor event and the detection latency defined as the time span between the first trigger in a set of co-occurring events and the time when the co-detection is reported to the data sink.

Default behavior. To see how the protocols fare, we simulate them in Figure 4.8 using default parameters as described above and a reporting threshold $D = 2$, i.e., each physical event detected by two or more sensors

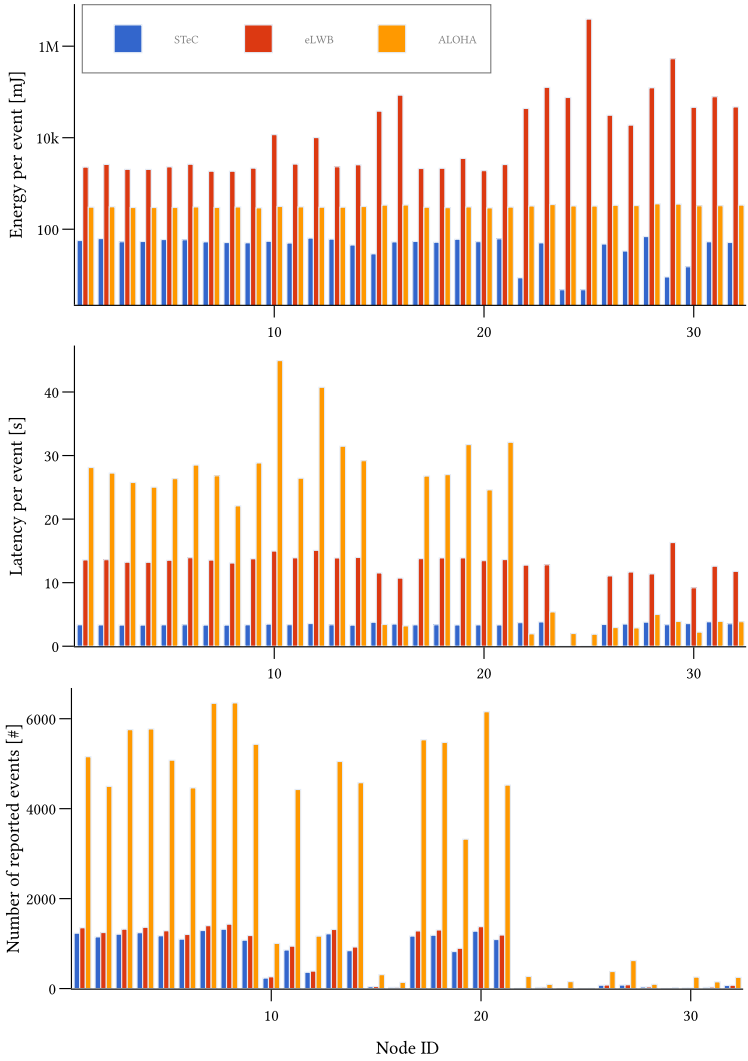

Figure 4.8: While eLWB is an energy-efficient synchronous protocol, the rare occurrence of seismic events results in many unused rounds and a high energy consumption (notice the logarithmic scale). ALOHA cannot locally filter and must report each event, which results in high detection latencies during bursts of activity. Node 24 and 25 are not part of any co-detection and correctly filter their events locally with STeC and eLWB, but cannot for ALOHA.

should be reported. We find that STeC enables nodes to spend 52 mJ for the communication of a sensed event, which is significantly less than the median communication costs of 308 mJ with ALOHA or 3087 mJ with eLWB by a factor of 5.83 × and 58.4 ×, respectively. The network-wide energy consumption throughout the deployment amounts to 5.04 kJ for STeC, compared to 28.10 kJ for ALOHA and 375.11 kJ for eLWB. In addition to event-driven local processing, STeC achieves its high energy efficiency by reducing costs by 37.9 % through using only 4.76 mJ on lone wake-ups with DiscoSync and terminating early. For the detection of reported events, eLWB takes a median 13.49 s through synchronous communication. While purely event-driven, ALOHA frequently suffers from strong contention and waits for a median 25.39 s until it can successfully report. STeC combines event-based communication with the efficiency of synchronous exchanges to achieve a low and highly deterministic median detection latency of 3.38 s. Through local data aggregation, STeC can filter 79.1 % of events and avoid reporting them over the resource-demanding long-range link.

Parameter tuning. The reporting threshold D is the primary tuning knob of STeC, as it permits adjusting the severity of events which should be reported independently of event characteristics (affecting τ_{prop}) or data requirements (affecting τ_{sample}). With a high threshold, only a major event reaching many nodes due to its large magnitude is reported and others can be filtered. Therefore, we compare the protocol behavior depending on D in Figure 4.9. As eLWB spends an average 4016 mJ per sensed event for periodic scheduling overhead during the 511 days of the simulated deployment, the difference in reporting costs due to filtering is minimal. Simulations with increased round periods show that even with communication occurring every 5 min, this overhead of 365 mJ still exceeds ALOHA's energy costs. ALOHA is unable to leverage a raised reporting threshold, as processing occurs after reporting due to its lack of local communication. On the other hand, STeC can exploit thresholds and filter co-detections $d < D$, reducing communication costs from an average 54 mJ for $D = 2$ to 17 mJ for $D = 4$. However, as each report contains more data, the long-range link delay increases the average detection latency from 3.36 s to 4.34 s.

Next, we investigate how the reporting threshold influences the number of reported events in Figure 4.10. We find that STeC correctly reports 91.2 % of all co-detections with the default settings of $D = 2$ and spreading factor 11. In case only larger events are of interest, $D = 7$

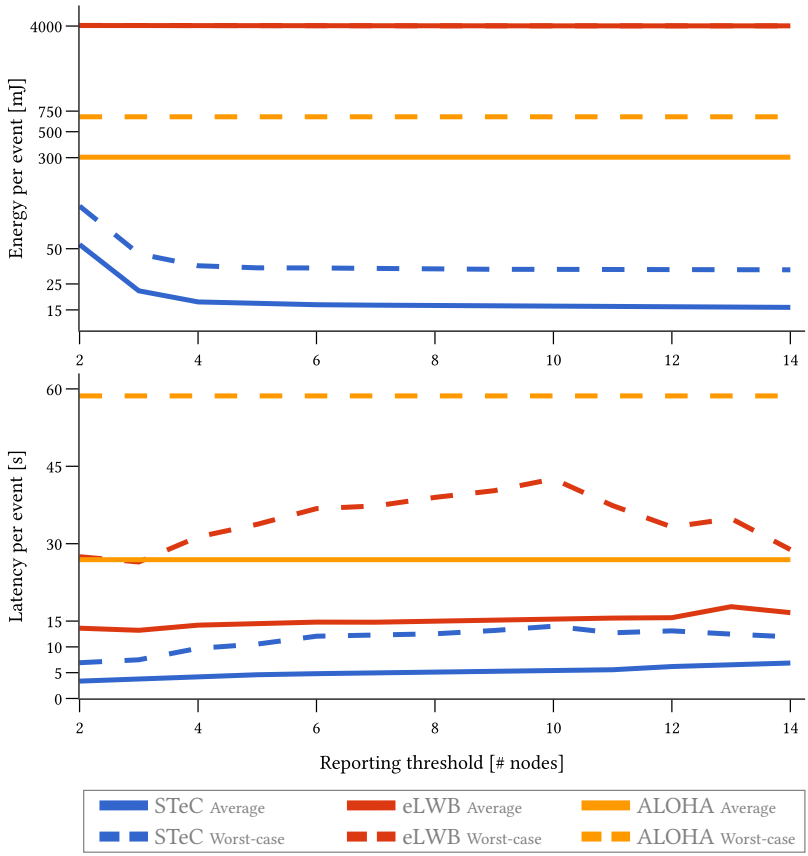

Figure 4.9: The reporting threshold D primarily influences energy consumption for $D \leq 4$, as the long-range communication thereafter does not significantly impact the overall energy consumption anymore. However, the worst-case nodes in the network can still suffer from increased detection latencies with higher D.

increases reliability to 97.3 %. The considerable amount of missed co-detections is due to the long reporting duration with a spreading factor of 11, during which the radio cannot be used for synchronization to the next sensor trigger if event intervals are small; apart from collisions, the simulated links themselves are assumed to be perfect. By requesting additional slots as described in Section 4.5.2, these missed sensor events could still be reported if desired. The use of a lower spreading factor 7

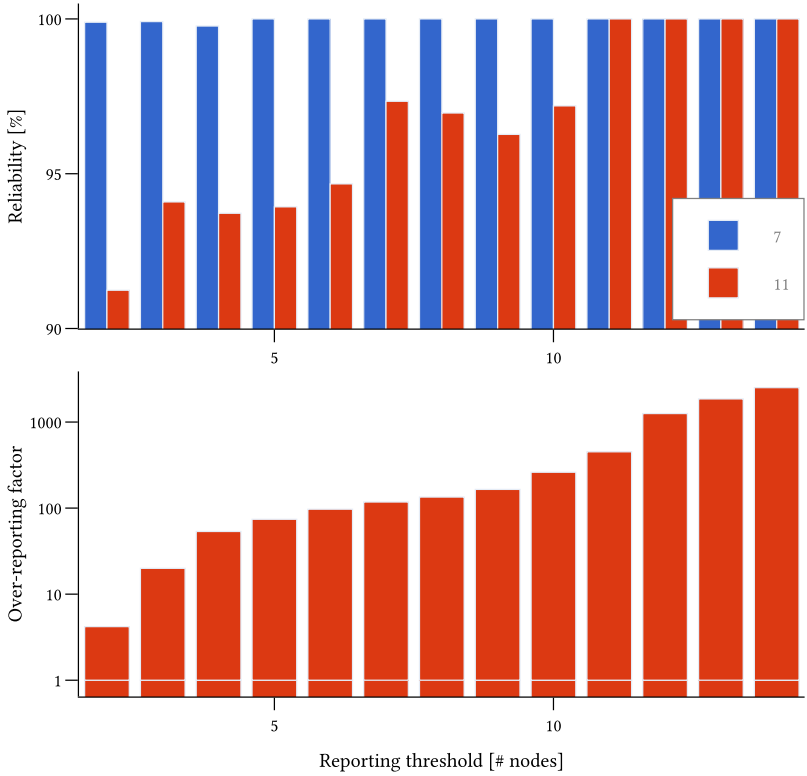

Figure 4.10: As expected, STeC manages to reliably report the most important events, as they are sufficiently rare for the network to finish reporting all previous events (top). On the other hand, ALOHA drastically over-reports as it cannot filter any events locally (bottom).

further mitigates this issue for deployments with frequent event bursts, boosting reliability to 99.89 % for $D = 2$ and 100 % for $D > 4$. However, this requires the data sink to be positioned closer to the sensor cluster to ensure that it is still within reach of the lower spreading factor. In our deployment, discussed in Section 4.6.5, spreading factors below 11 were inadequate for highly reliable communication to all nodes. On the other hand, ALOHA depletes its battery by sending all events and thereby reports more than 116 × the relevant packets for $D > 6$.

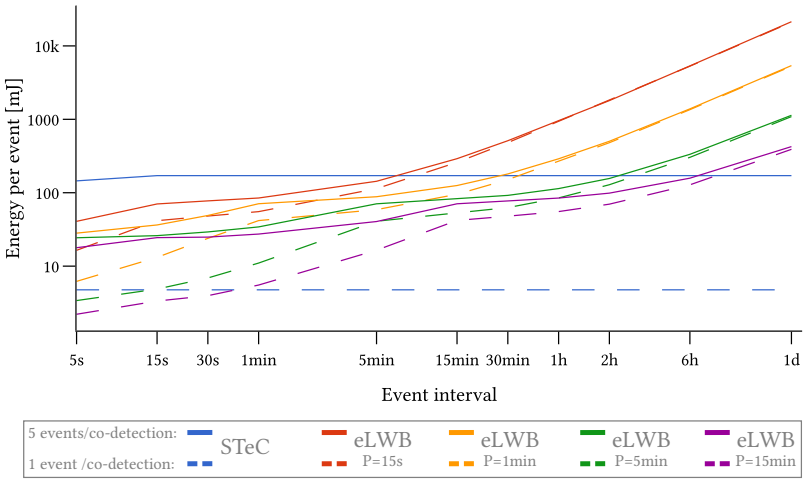

Figure 4.11: Even with consistent co-detections of 5 nodes, STeC outperforms all schedule-based variants if events occur rarer than every 7.5 h. For lone wake-ups, this already occurs after 46 s.

Event-driven communication exploits rarity. To observe how the sparsity of events affects energy consumption, we search the event interval beyond which STeC outperforms schedule-based protocols. For this, we compare it against eLWB with periods of 15 s (default), 1 min, 5 min, and 15 min in Figure 4.11 with a constant reporting threshold of $D = 2$. We run two synthetic traces of 500 physical events each with either only 1 sensor event per physical event or a high co-detection degree at 5 sensor events per physical event (occurring fewer than 1 % in our case study). Each physical event is separated by the event interval ±25 % to avoid artifacts due to matching schedule periods. Sensor events have a uniformly distributed propagation delay $\delta_m = 0 - 100$ ms. While eLWB variants with high scheduling frequencies quickly lose their efficiency, the energy consumption of STeC is independent of the event interval. We find that even compared to an impractically long period of 15 min, STeC outperforms schedule-based protocols in energy efficiency while reporting events with a drastically shorter latency. Due to DiscoSync, STeC particularly excels if, as for our case study shown in Table 4.1, most physical events are only observed by a single sensor and co-detections are rare.

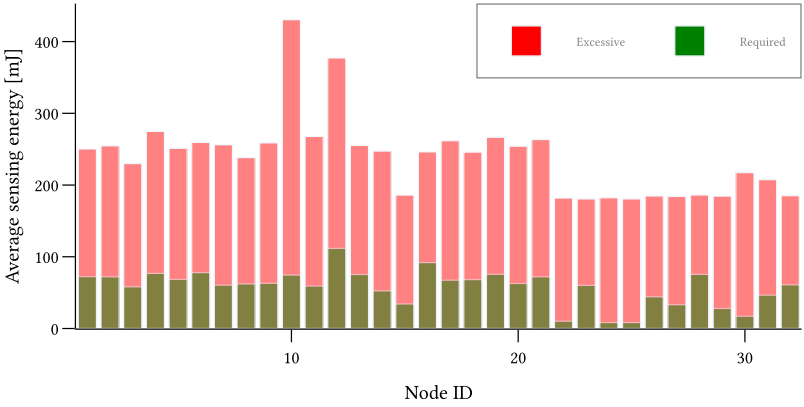

Figure 4.12: The termination of irrelevant events allows sensors to eliminate excessive sampling and reduce sensing costs by an average of 76.4 %, with some sensors saving as much as 95.6 %.

Sample termination. In contrast to other protocols, STeC can quickly determine whether an event is relevant and terminate sampling if the number of active nodes is below the reporting threshold. As shown in Figure 4.12, this enables us to reduce sampling costs by an average of 76.4 %. Because sensing costs are significantly higher than the communication costs of our event-based protocol for our implementation, this results in an effective median lifetime improvement of 3.25 ×.

4.6.3 Testbed Validation

To validate our simulation results, we use synthetic traces, which we run both through our simulator as well as on the FlockLab testbed [TDFS+20]. We simulate 50 physical events with a spreading factor of 11 and an ad-hoc network of $2-9$ nodes, where each node in the network receives one sensor trigger per physical event. Additionally, we set $\tau_{sample} = 1000$ ms to represent the impact of longer sampling. As demonstrated in Section 4.6.2, the event interval has no influence on STeC and is hence set to 20 s for all events to reduce testing time. To emulate sensor events, we use the GPIO actuation feature of FlockLab, which permits us to individually trigger sensor nodes at sub-ms granularity while running the same software as our deployment implementation used in Section 4.6.4 and Section 4.6.5.

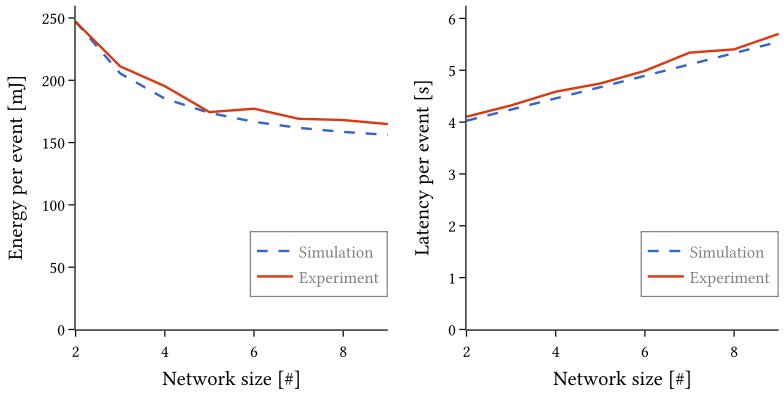

Figure 4.13: Comparisons between our simulation and testbed experiments demonstrate that the simulator correctly predicts both energy and latency metrics. However, occasional packet loss leads to a slightly increased energy consumption. STeC's reporter assignment successfully spreads the costs of long-range links across nodes in larger networks.

Network size. We find in Figure 4.13 that our simulator accurately predicts experimental results of both energy per event and detection latency for various network sizes. As expected based on our simulations, we observe that an increased network size can be effectively used by our reporter assignment scheme, as presented in Section 4.4.3, to distribute reporting across a larger set of nodes. This enables us to decrease average energy costs, a dynamic that the testbed runs confirm. Notice that because each synthetic event needs to be reported, protocol execution cannot be terminated early, resulting in a higher average energy consumption compared to the real-world traces. Out of 2200 sensor events, STeC successfully reports 99.22 % to the data sink in our experiments. Due to the indoor location of the testbed with non-line-of-sight, sensor nodes occasionally experience packet loss and are forced to retransmit, resulting in a slightly higher energy consumption than anticipated.

Event accuracy. As the triggers are known, we further investigate the accuracy of reported event timestamps. We find that STeC obtains absolute timestamps with a median accuracy of 185 µs and a maximum of 493 µs for the 2200 sensor events. For example, this precision would enable us to determine distances to a seismic event source with sub-meter accuracy and potentially triangulate it.

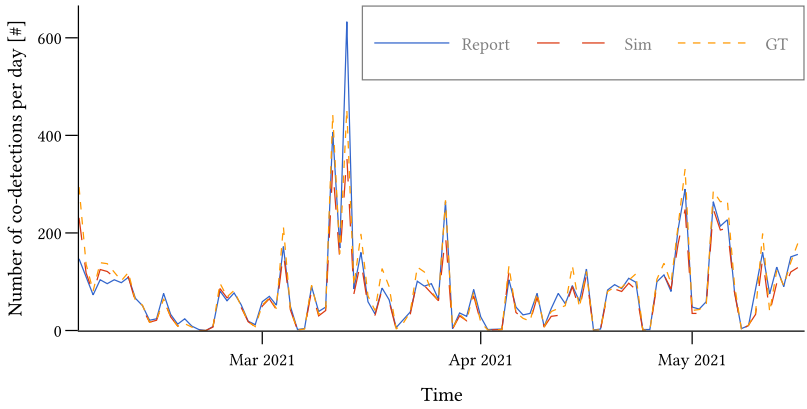

Figure 4.14: In our test deployment, nodes correctly report most co-detections (`Report`) compared to ground truth based on logged events (`GT`), which our simulation (`Sim`) reliably predicts.

4.6.4 Test Deployment

In a first phase, we deploy 9 nodes as described in Section 4.5.1 at a local urban location which permits us to retrieve ground truth data by accessing the logs on the SD card of the nodes. We spread 6 of them across the rooftop of an office building, while three are situated within range on a lower floor to be unaffected by precipitation. Due to ongoing construction work, the sensor nodes regularly experience seismic triggers. This allows us to verify that STeC can correctly detect and aggregate physical events in realistic settings. For comparison, ground truth is computed with Algorithm 1 on the gathered logged sensor events. Figure 4.14 shows that the sensors receive up to 43 825 triggers per node over a 15-week period and a total of 19 334 co-detections. We observe that STeC correctly represents the dynamics compared to the ground truth and reliably reports major events to the data sink. Using the logged sensor events as an input trace for our simulator, we find that STeC successfully reports 75.2 % of events despite a noisy environment, including rain on 40 % of days, which causes frequent uncorrelated sensor triggers. During rain, STeC can misinterpret triggers by raindrops at multiple sensors for the co-detection of a single physical event arriving at an event interval that is too short for events to be constantly reported. As only 7.62 % of events must be reported to the data sink, STeC performs well at an average energy cost of 21 mJ. In comparison, ALOHA would spend an average 313 mJ to report all events.

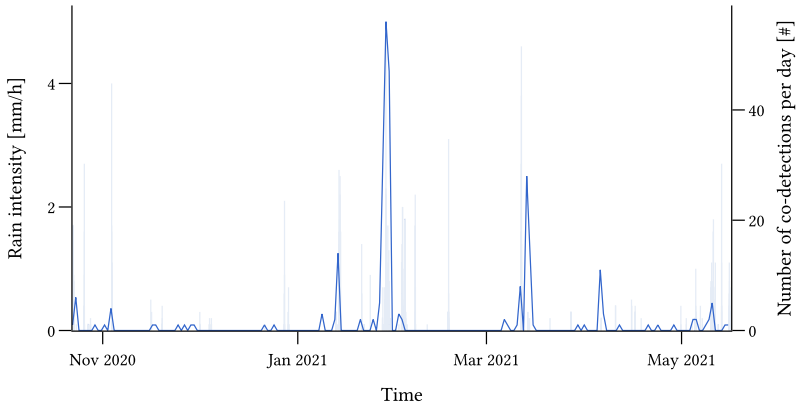

Figure 4.15: Increased detection rates are frequently correlated with precipitation events. The increased availability of liquid water strongly affects the downslope kinematics of active rock glaciers [FFBV19].

4.6.5 Real-world Deployment

To test STeC under real-world conditions, we install 9 sensors for 7 months in mountainous terrain at 2500 m a.s.l. as shown in Figure 4.16. The sensors are situated in a cluster at the tongue of a rock glacier where material is continuously breaking off due to increased slope movement. Therefore, the data sink is located on the other side of the valley multiple kilometers away, where the necessary infrastructure can be supported. During our deployment, we gather 566 sensor events resulting in 275 co-detections as depicted in Figure 4.15, with 8 physical events being detected by 3 up to 6 nodes. As the main ground motion dynamics are happening in the summer months and not in winter, the low detection numbers meet expectations. However, in the last deployment weeks, STeC has detected increasing event activity, e.g., following heavy rain on March 11, which resulted in 31 co-detections reported on March 13 and 14 on March 14.

4.7 Summary

With STeC, we develop a novel communication scheme that exploits the spatial and temporal correlation of sensed signals to effectively construct and control an ad-hoc communication network. This is the first time that a network is structured from the ground up by events

Figure 4.16: Distributed sensor nodes detect trigger events when seismic signals emanating from friction within the rock and sediment mass moving downslope exceed predefined threshold levels. Sensors are located across an area of 50 × 20 m on a rock glacier with movements of up to 5 m/a [FFBV19]. A close-up of a sensor node shows the mounting at ground level to tightly couple the sensor with the surface.

as they appear. As such, this approach alleviates the elementary conflict of synchronous systems between optimizing either for duty cycle or latency and demonstrates that autonomous formation does not mandate the integration of neighbor discovery as done in Chapter 2 and Chapter 3. STeC achieves fast and energy-efficient bootstrapping of a connected network of nodes, minimizing overhead and idle listening times. We show that this technique permits a drastic reduction in energy costs as well as detection latency and is particularly effective for applications where co-detections are rare. Furthermore, we demonstrate that additional optimization techniques, e.g., local data aggregation and filtering of events as well as long-range reporting to a data sink, can be integrated without affecting the fundamental event-driven wake-up based on sensor events.

Instead of enabling autonomous deployments by eliminating infrastructure as in Chapter 2, this chapter demonstrates that it can also be leveraged to *boost* autonomy if we exploit long-range links while keeping energy costs low. However, although a data sink is unavoidable for monitoring purposes, it is a single point of failure (SPOF) whose malfunction causes the entire system to cease operation. While we have increasingly reduced such fundamental dependencies, from eliminating infrastructure in Chapter 2 over the robust orchestration of clusters in Chapter 3 to entirely ad-hoc cluster formation in this chapter, each chapter so far still integrated nodes that served a special role. The final Chapter 5 explores whether we can eradicate such distinctions entirely and fully decentralize wireless networking.

5

Concurrent Coordination for Fault-tolerant Networking

For traditional WSN applications, a centralized base station is necessary to collect data and serve as a gateway to forward sensor data to the back end for further processing. We have seen in Chapter 4 that decentralization based on the exploitation of signal correlations in time and space improves the autonomy of network formation and only requires dependence on a particular device if it is unavoidable as a link to the outer world. However, autonomous resilience is prohibited as long as such connectivity to a single base station is needed, as it inherently constitutes a single point of failure (SPOF). While Chapter 3 demonstrated that robust leader election can alleviate many of the problems of centralized coordination if a set of nodes only requires cluster-internal information exchange, Demos is still unable to provide persistent and safe communication when facing node and link failures. For many scenarios where nodes operate in hostile environments and must guarantee a reliable communication service, existing network protocols do not meet the dependability requirements as the failure of a single node or link can completely disrupt communication and take significant time and energy to recover.

This chapter presents Hydra, a low-power wireless protocol that guarantees robust communication despite arbitrary node and link failures. Unlike most existing deterministic protocols, including the ones presented in Chapter 2 and Chapter 3, Hydra steers clear of centralized coordination

This chapter is based on [BDFK+23c], [BDFK+23b], and [Kuo21].

to avoid a single point of failure. Instead, all nodes are equivalent in terms of protocol logic and configuration, performing coordination tasks such as synchronization and scheduling concurrently. This concept of *concurrent coordination* relies on a novel distributed consensus algorithm that yields provably unique decisions with low delay and energy overhead. In addition, nodes leverage pre-established information to maintain reliable data exchange through concurrent transmissions even when interference temporarily prevents coordination updates or input from failed nodes is missing. This ability to guarantee persistent communication despite ongoing disturbances significantly extends the autonomous resilience compared to our protocol presented in Chapter 3, where the data exchange is interrupted during leader election and can only resume once consensus has been re-established. In addition to a theoretical analysis that provides formal proofs, we evaluate Hydra in a multi-hop network of 23 nodes. Our experiments demonstrate that Hydra withstands random node failures without increasing coordination overhead and that it re-establishes efficient and reliable data exchange within seconds after a major disruption.

5.1 Introduction

The last few years have seen substantial innovations that have enabled WSNs with unprecedented dependability. For instance, novel protocols enable distributed battery-powered devices to reach an agreement [ANDL17b], deliver messages in the desired order [FZMT13], and exchange data within hard real-time deadlines to meet the requirements of cyber-physical and Industrial Internet of Things applications [MBJ+19]. To achieve these capabilities reliably and with broad applicability, the concept of concurrent transmissions (CT) [FZTS11, ZMS20] has been instrumental in becoming highly resilient to external interference and network topology changes.

Problem. While such wireless systems are promising for enabling novel applications in pivotal scenarios, they cannot tolerate the failure of critical devices or key communication links. However, such failures are common as WSNs are frequently deployed in hostile or inaccessible scenarios where extreme weather [BISV08], high ambient humidity [ABC+20, WCPC18], the presence of living beings [BCN+21, MFCP+19], disasters [CXC+19], and non-line-of-sight conditions [DCR+22, JP21] render the availability of devices and wireless links fragile and highly variable.

Current systems typically adopt a centralized design and suffer from a single point of failure: If the *network coordinator* that manages synchronization and scheduling fails or gets disconnected, the entire rest of the network also breaks down as reliable communication is no longer possible. This vulnerability stands in stark contrast to the required dependability of wireless communication in many applications where even minor system outages may involve significant financial costs and lead to long-term repercussions [ZCDH12, WCPC18]. Typical scenarios include autonomous drone swarms [LSD+20], early-warning systems in harsh conditions [BDFG+21, WBDF+19], and networks operating under strong interference [BSS+22] or high dependability requirements [MBJ+19]. Designating co-located devices as a failover for the primary network coordinator cannot solve the fundamental problem, as this approach is known to provide limited fault tolerance if failures are correlated, the common case in practice [WC19]. Modern WSNs should have the liberty to operate independently without requiring pre-assigned backups or expensive, redundant hardening.

Contribution. To address this problem, we introduce Hydra, the first fully distributed protocol that avoids centralized coordination to enable fault-tolerant low-power wireless networking. The fundamental paradigm underlying Hydra is that every node in the network is equivalent in terms of protocol logic and configuration, thus avoiding any entity that maintains a unique state or that serves a special role. The protocol adapts to the application's traffic demands while tolerating arbitrary message losses, temporary or permanent node failures, and sudden topology changes. Hydra uses distributed processing in tandem with communication primitives based on CT (*i*) to continuously track the set of nodes that is currently part of the network, (*ii*) to ensure that all nodes that concurrently compute and distribute a new schedule do so based on the same information, and (*iii*) to accurately time-synchronize the network to achieve high reliability and efficiency.

Concurrent coordination empowers Hydra to take full advantage of physical redundancy to yield outstanding fault tolerance and dependability for WSNs. Real-world experiments validate that Hydra's energy overhead almost matches that of LWB [FZMT12], a comparable centralized design. Hydra demonstrates robust, efficient data exchange while swiftly adapting to changing traffic demands despite arbitrary node failures and severe communication disruptions. In summary, this chapter makes the following major contributions:

- We present Hydra, a communication protocol for low-power WSNs that excels in its fault tolerance through a novel distributed consensus and synchronization mechanism.
- We provide an algorithmic specification of Hydra and prove its properties: (*i*) freedom of harmful packet collisions, thus ensuring safe operation, (*ii*) adaptivity to changing traffic demands and dynamic network topologies, and (*iii*) persistent data exchange even under node and link failures.
- We implement Hydra on a microcontroller driving a Semtech SX1262 RF transceiver and provide the code as open source together with tools for reproducible fault injection [BDFK+23b].
- We evaluate Hydra on the FlockLab testbed [TDFS+20]. Our experiments not only confirm its safety, adaptivity, and liveness but also demonstrate its fault tolerance at a negligible runtime overhead even if almost half of the nodes in the network fail.

After defining the problem space in Section 5.2, we provide an overview of Hydra in Section 5.3, a detailed protocol description in Section 5.4, and a formal analysis in Section 5.5. We implement Hydra to demonstrate its efficacy in realistic testbed experiments in Section 5.6, where we show that Hydra achieves fault tolerance despite a high degree of induced link errors and prolonged, adversarial node failures. Lastly, Section 5.7 discusses design trade-offs and limitations of Hydra.

5.2 Problem and Challenges

We first introduce our failure model and derive the objectives and corresponding protocol requirements to achieve fault-tolerant networking before analyzing the challenges in Hydra's design space.

5.2.1 Objectives and Protocol Requirements

Failure model. To ensure general applicability, we assume that communication can fail in any phase of the protocol. This includes the possibility that a node can neither receive from nor send to any other node over prolonged periods. Packet loss may occur at any time and we do not make any assumption on the loss distribution or correlation between losses. We consider non-Byzantine failures, i.e., nodes adhere to their specifications and a packet either arrives correctly or is not received at all. The latter

follows from corrupted packets being filtered using checksums such as cyclic redundancy checks (CRCs) and discarded. We further assume that a node may temporarily or permanently crash at any time.

Objectives. Motivated by cyber-physical and Industrial Internet of Things applications, we aim to design a protocol that provides reliable low-power wireless networking under the above-mentioned failure model. In particular, unlike state-of-the-art solutions, the protocol aims to avoid any single point of failure by design.

While tolerating faults, the protocol must also achieve the following objectives: (*i*) Because of the constrained nature of WSNs, efficient use of limited resources such as energy and bandwidth is vital. (*ii*) These resources should be flexibly allocated so that the network can react to changing environments and traffic demands. (*iii*) The protocol must be able to deliver packets to any node in the network despite dynamically changing, multi-hop network topologies. Data exchange needs to be highly reliable and predictable, and therefore packet collisions must be avoided.

Protocol requirements. Based on these objectives, we identify three requirements for a fault-tolerant low-power wireless protocol that must be satisfied under the failure model described above:

- *Safety:* Guaranteeing correct operation so that any two nodes never use the same time slot to exchange data, thereby avoiding packet collisions that impair reliability.
- *Adaptivity:* Enabling each non-faulty node to react to changes in traffic demands and environmental conditions, including the addition of nodes and the release of obsolete resources.
- *Liveness:* Ensuring that consistent data exchange is maintained between all non-faulty nodes that are physically able to communicate with each other (i.e., remaining network links suffice to transfer information).

5.2.2 Challenges and Trade-offs

All three requirements must be simultaneously satisfied, yet they are mutually conflicting. There are four fundamental challenges in meeting the objectives and protocol requirements outlined above.

First, one central network coordinator is a single point of failure. Unavailability of this node leads to unstable control loops endangering

safety [MBJ+19], inefficient resource usage due to a loss of adaptivity [WC19], or disrupted data flows violating liveness [FZMT12]. Using backup nodes that take over when the primary network coordinator fails is difficult as incompatible schedules during handover violate safety and liveness. Moreover, the fault tolerance of this approach is limited as correlated failures likely affect both the primary network coordinator and its backups [WC19], further complicating the handover procedure and endangering safety.

Second, instead of using a single leader, it is possible to increase fault tolerance by letting multiple nodes perform network coordination concurrently. However, the complexity of this approach increases as each of the participating nodes must base its decisions on the same inputs at the same time, thereby requiring consensus as soon as more than a single coordinator is involved. When information is missing, liveness and adaptivity are at risk.

Third, consensus on these protocol inputs is critical for safety but is particularly difficult to find when nodes become unreachable due to temporary or permanent faults. In this case, the set of nodes that have to agree must be flexible to maintain adaptivity.

Fourth, we either face reduced liveness if only nodes that obtained the latest inputs can communicate, or we violate safety if nodes with outdated decisions cause packet collisions. As an alternative to coordinated communication, both adaptivity and liveness are preserved with unsynchronized data exchange schemes, but packet collisions may violate safety.

In the design of Hydra, we have carefully explored this trade-off space, further discussed in Section 5.7.1. As we will show, Hydra involves all nodes in concurrent coordination of the network to maximize robustness and leverages the contention-free data floods to maintain redundant synchronization without requiring a fixed time reference. Furthermore, we choose to unconditionally provide safety for reliable, predictable performance and will hence be forced to occasionally delay adaptivity or restrict liveness.

5.3 Protocol Design

We introduce Hydra, a fault-tolerant low-power wireless network protocol. In Hydra, each node in the network is equivalent in terms of protocol logic and runtime configuration. As a result, network coordination tasks

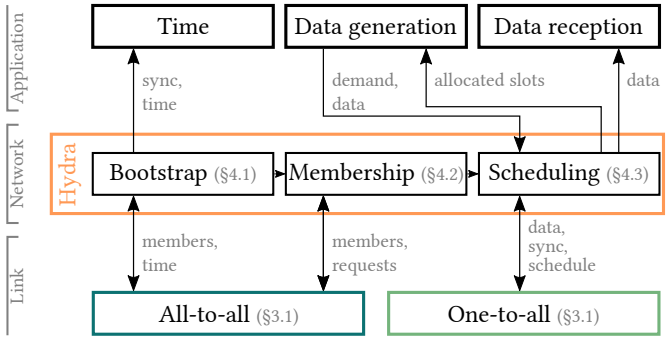

Figure 5.1: Hydra works on the network layer and interfaces with link-layer primitives to make fault tolerance available to the application. The three individual components of Hydra are explained in further detail in the respective subsections denoted in grey.

such as time synchronization and scheduling are distributed across all nodes and run concurrently. A formal analysis in Section 5.5 shows that Hydra provably satisfies safety, adaptivity, and liveness under arbitrary node and link failures. We first describe the protocol interfaces and provide an overview of Hydra and the mechanisms to achieve concurrent coordination.

5.3.1 Overview

Existing designs, presented in Section 5.8, have a single point of failure that limits their fault tolerance. While the crash of a centralized coordinator for scheduling and synchronization directly impairs the networking capabilities [FZMT12, HL20], even partially decentralized solutions [DANLW15, KKK19, KKK21] rely on topology information like the routing tree of RPL-based protocols [WTB+12] for coordination that roots in a single device. However, modern communication primitives such as CT [ZMS20] permit reliable and topology-agnostic communication on the link layer if nodes are sufficiently synchronized. Based on this insight, Hydra adds a network layer that sits between the application and the link layer (see Figure 5.1) and guarantees fundamental properties through concurrent coordination. As detailed in Section 5.4, it includes novel concepts for bootstrapping the network, finding distributed consensus on network membership, and scheduling persistent communication.

Application layer. Hydra provides accurate network-wide time synchronization as well as a bi-directional communication service. The application submits its traffic demands to Hydra, which then takes care of facilitating corresponding communication resources.

Link layer. Hydra can be based on many communication primitives, as we will show in Section 5.5 that its safety is formally proven without relying on the primitives' reliability. However, CT have become the basis for countless robust and efficient multi-hop wireless protocols [FZTS11, HMZ18, ZMS20]. Network flooding permits predictable communication at high throughput and low latency, with its broadcast nature making it ideally suited to exploit physical redundancy in a system. Due to topology-agnostic message passing based on concurrent forwarding across hops, these primitives seamlessly handle network changes. In Hydra, we employ a combination of two classes of such communication primitives.

All-to-all primitives such as Chaos [LFZ13] and Mixer [HMZ18] exchange per-node information for network coordination. Nodes independently send and receive in consecutive contention slots and continually adapt packets so that information efficiently propagates through the network. However, they require tight synchronization as a prerequisite to receive packets based on the capture effect [LF76].

One-to-all floods like Glossy [FZTS11] reliably disseminate data packets. The initiator of the flood requires an exclusive time slot through scheduling for contention-free communication and serves as a unique time reference for (re-)synchronization.

By combining the ability of these all-to-all primitives to exchange scheduling information based solely on a shared time basis with the capability of fast network flooding to tightly synchronize a multi-hop network, Hydra creates a symbiosis on the link layer that empowers nodes to fully decentralize network coordination.

5.3.2 Hydra in a Nutshell

Hydra includes several dedicated mechanisms to fulfill its three protocol requirements. Next, we introduce the protocol structure and outline the algorithm presented in detail in Section 5.4.

Structure. Communication in Hydra occurs in rounds. As shown in Figure 5.2, each round consists of three phases: To ensure safety, a contention-free *data dissemination* (DD) phase permits nodes to reliably exchange data. To determine an adaptive schedule that allocates such exclusive time slots,

nodes exchange their demand during the *schedule negotiation* (SN) phase. Lastly, the newly computed schedule is shared in the *schedule distribution* (SD) phase so nodes can independently communicate without permanently requiring each other's input and preserve liveness.

Hydra executes such rounds of period T for a network $\mathcal{N}$ of N nodes i with $\mathcal{N} = \{i \in \mathbb{N} \mid 1 \leq i \leq N\}$. An *epoch* consists of F rounds and defines the rate at which the schedule may change.

Data dissemination. During the DD phase, each node that knows a schedule initiates a one-to-all flood if it is the owner of the current slot and otherwise assists in propagating a packet by concurrently retransmitting it after reception. Application data is exclusively disseminated in this phase, with each initiator serving as a time reference to re-synchronize without a centralized entity.

Schedule negotiation. Nodes may leave at any time due to volatile links or node failures and may (re-)join. Hydra permits a flexible set of nodes to form a network and determines a schedule that reflects the dynamically changing bandwidth demands of nodes.

To ensure that concurrent coordination finds consensus on the set of participating nodes and their bandwidth demands, network membership is determined dynamically during the SN phase. To this end, nodes maintain a local notion of whom they consider part of the current network as membership flags and exchange these flags in the SN phase. By awaiting and merging information from all expected nodes, a unique decision among all nodes of the current network can be guaranteed, as formally proven in Section 5.5.

To compute the next schedule, each node determines its request (e.g. the number of slots it desires) according to its current demand and known schedule. During the SN phase, nodes aggregate schedule requests and memberships flags from other nodes using all-to-all messages, as explained in detail in Section 5.4.2, and update the minimum and maximum schedule versions they have encountered. At the end, they have the ability to determine the next schedule if (*i*) they have obtained the complete set of information, meaning that they have received requests from all nodes that are part of the current network, i.e., whose membership flags are set after merging, and (*ii*) all these nodes have the same schedule version. The next schedule can then be computed using a deterministic algorithm based on the current schedule and this complete set of information.

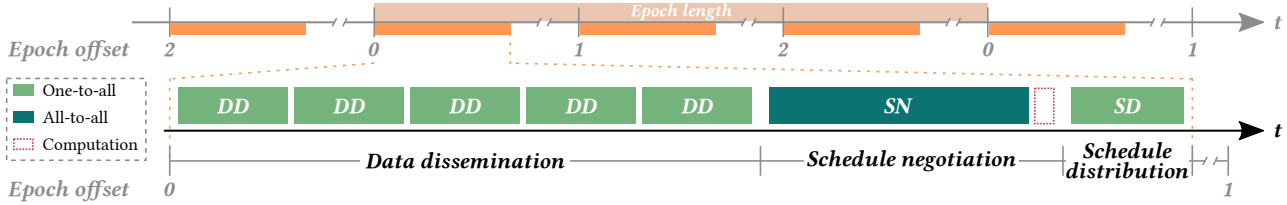

Figure 5.2: Each round in Hydra consists of three phases. During *data dissemination* (DD), nodes exchange application data according to a local schedule and synchronize with each other. During *schedule negotiation* (SN), requests are aggregated to compute the next schedule, which will be flooded during *schedule distribution* (SD) to ensure reliable progress. Throughout an epoch consisting of multiple rounds (e.g. three in this illustration), information regarding network membership and demand is accumulated to reach consensus and enable distributed scheduling.

Schedule distribution. During the SD phase, nodes that have computed the next schedule use a single, concurrent one-to-all flood at the end of an epoch to distribute it to others that have been unable to do so themselves or newly joined. In case some nodes have missed a previous update and still use an older schedule, the most recent schedule will be retransmitted by updated nodes. A node knowing the next schedule, having either computed it or received it from others, will start using it after the end of the epoch.

5.3.3 Concepts to Ensure Fault Tolerance

To ensure that Hydra does not suffer from a single point of failure and simultaneously provides safety, adaptivity, and liveness, we develop three specialized components represented in Figure 5.1.

Ensuring safety: Distributed synchronization. To make sure that data exchange does not interfere, nodes must be reliably synchronized. For this purpose, the initiators of slots in the DD phase provide a unique time reference. Initial synchronization is achieved through a bootstrapping algorithm, introduced in Section 5.4.1, on a separate channel to prevent interference with an existing network.

Safety of the DD phase further requires that all used schedules are compatible, i.e., they do not assign slots to two different nodes. This is achieved by guaranteeing that (*i*) at most two successive schedules are present, (*ii*) successive schedules are compatible, and (*iii*) only a single, connected network operates at any point in time to avoid inter-network interference, as we will show in Section 5.5.

Ensuring adaptivity: Flexible network membership. If schedule computation depends on input from all nodes, node failures block progress [ANDL17b]. To permit nodes to join and leave the network anytime while safely adapting scheduling, Hydra requires consensus on a flexible set of nodes that form the members of the current network. By exchanging the local notions of reachable nodes in the SN phase, elaborated in Section 5.4.2, we provably guarantee that a schedule is only computed if the input from all nodes in the current network is obtained and that the result is unique.

In the case of link failures resulting in a missed schedule, future schedule computations fail due to a schedule version mismatch and progress would be permanently prohibited. To preserve adaptivity, nodes knowing the most recent schedule must detect this case and retransmit it in the SD phase, giving other members the chance to catch up to the rest of the network so progress may resume.

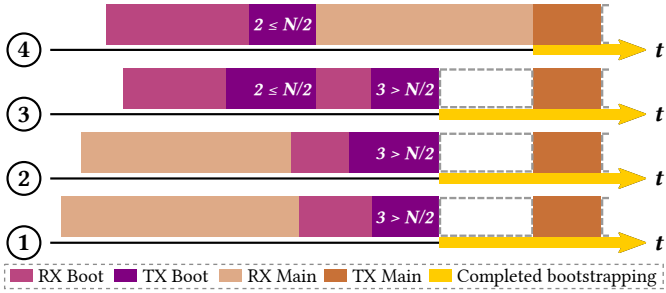

Figure 5.3: Bootstrapping nodes start independently and synchronize if a majority of the $N = 4$ nodes in this example exchange their information on the boot channel. As seen for node $i = 4$, a node can also join an existing network when listening on the main channel.

Ensuring liveness: Local schedules. Synchronously changing schedules incurs the risk that nodes must halt communication when coordination is disturbed and consensus is unclear. To preserve liveness during such transitions between schedules, the conditions of Hydra's scheduling algorithm introduced in Section 5.4.3 ensure that two successive schedules cannot interfere. Therefore, nodes can asynchronously update without restricting anyone's ability to keep exchanging application data. To continuously maintain this compatibility between used schedules, the schedule versions ensure that a new schedule is only computed if all nodes have obtained the previous version. Therefore, a known schedule guarantees safe data exchange even if coordination is temporarily prohibited.

5.4 Hydra in Detail

Next, we take a closer look at Hydra's algorithms and first show how a Hydra network is initially formed. Then, the concept of distributed consensus based on flexible network membership is presented. Lastly, we discuss how careful scheduling provides liveness and safety.

Notation. $\perp$ denotes a variable containing no information. The operator $\circ$ combines sets whose elements may have the value $\perp$ as follows: $\perp \circ \perp = \perp$ and $a \circ \perp = \perp \circ a = a \circ a = a$ for any value $a \neq \perp$. Combining $a \neq b$ with $a, b \neq \perp$ is undefined. The operator $\vee$ is the element-wise OR with $0 \vee 0 = 0$ and $1 \vee 0 = 0 \vee 1 = 1 \vee 1 = 1$.

5.4.1 Bootstrapping

After a reset or lost connectivity with a majority of nodes, a node must discover other nodes to synchronize or assemble a new network. Hydra's bootstrapping algorithm described in Algorithm 2 requires no pre-existing synchronization and terminates with the node being synchronized to a majority of nodes. It further sets the initial conditions for Algorithm 3 as introduced below, i.e., the current membership flags $M[j]$, the latest schedule version v, schedule S^v and epoch offset f, and values $\forall j \in [1, N] : C_e[j] \leftarrow 0, \forall j \in [1, N] : I_e[j] \leftarrow 0$, *updated* $\leftarrow 0$, and *retransmit* $\leftarrow 0$.

To not endanger the safety of a pre-existing network, the execution of the bootstrapping algorithm as shown in Algorithm 2 occurs on a separate boot channel c_{boot}. To synchronize, each node randomly chooses whether it should listen on the main or the boot channel for a non-deterministic duration, depicted in Figure 5.3. On the main channel c_{main}, it synchronizes to an existing network in line 4 if it receives a packet, for which it uses the function *EPOCH_OFFSET* to derive the required network parameters, or reboots otherwise if no existing network is discovered in line 14. On the boot channel c_{boot}, a node floods a sync packet in line 16 if it has not received a packet to which other listening nodes align in line 4. Nodes then aggregate information and sum the number of encounters. If a majority of nodes has exchanged information as affirmed in line 28, the newly founded network synchronizes based on the transmission time of a confirmation packet using the function *TX_TIME* and simultaneously switches to Algorithm 3 in line 43 after $\Delta\tau \geq F \cdot T$. Note that bootstrapping is opportunistic and does not fulfill the requirements in Section 5.2.1. However, as it can be arbitrarily restarted without affecting the performance of Hydra's normal operation, it guarantees synchronization after termination and enables liveness.

5.4.2 Consensus on Network Membership

Hydra guarantees safety under any failure by design. To achieve such robustness despite nodes joining, leaving, or even failing, reliably deciding on the set of nodes whose *requests* r_j^v are considered for a new schedule is paramount. While quick convergence is desirable for adaptivity, the mechanism's key property is that its result is provably unique if consensus between a set of nodes is found.

Consensus. To make unique decisions for a flexible set of nodes, we introduce the *membership flags* M, which reflect a node's local notion

Algorithm 2 Protocol behavior of node i during bootstrapping

YD (synchronization distribution) phase
  $c \leftarrow PICK(\{c_{main}, c_{boot}\})$; $t_{listen} \leftarrow PICK([0, K)) \cdot T_{slot}$;
  if (packet q received within t_{listen}) **then**
    synchronize to packet ;
    if $(c = c_{main})$ **then**
      $v \leftarrow 0$; $f \leftarrow EPOCH\_OFFSET(q)$;
      $synced \leftarrow 1$; $updated \leftarrow 0$; $unchanged \leftarrow 0$; $retransmit \leftarrow 0$;
      for all $k \in [1, K]$ **do**
        $S^v[k] \leftarrow \perp$;
      for all $j \in [1, N]$ **do**
        $C_e[j] \leftarrow 0$; $I_e[j] \leftarrow 0$; $M[j] \leftarrow 0$;
      switch to Algorithm 3 ;
  else if $(c = c_{main})$ **then**
    restart Algorithm 2 ;
  else
    send sync packet ;

YN (synchronization negotiation) phase
  Initialization
    $synced \leftarrow 0$;
    for all $j \in [1, N]$ **do**
      $M[j] \leftarrow \begin{cases} 1 & \text{if } j = i \\ 0 & \text{otherwise} \end{cases}$;
  Sending
    send packet $p = (M)$;
  Receiving
    if (packet $q = (M_q)$ received) **then**
      $M \leftarrow M \vee M_q$;
  Final
    if $(\sum_{j \in [1,N]} M[j] > N/2)$ **then**
      $synced \leftarrow 1$;

YD (synchronization distribution) phase
  if $(synced = 1)$ **then**
    send sync packet p ; $\tau \leftarrow TX\_TIME(p)$;
  else if (sync packet q received) **then**
    $synced \leftarrow 1$; $\tau \leftarrow TX\_TIME(q)$;

End of round
  if $(synced = 1)$ **then**
    wait until $\tau + \Delta\tau$;
    $v \leftarrow 1$; $f \leftarrow 0$; $updated \leftarrow 0$; $unchanged \leftarrow 0$; $retransmit \leftarrow 0$;
    for all $k \in [1, K]$ **do**
      $S^v[k] \leftarrow \perp$;
    for all $j \in [1, N]$ **do**
      $C_e[j] \leftarrow 0$; $I_e[j] \leftarrow 0$;
    switch to Algorithm 3 ;

of whom it considers part of the network. The flags remain constant throughout an epoch, i.e., $M[j] = 1$ if the node expects node j to be reachable as part of the current network and $M[j] = 0$ otherwise. Note that nodes may differ in their view of the current network and may have unequal membership flags.

The key to reaching consensus can be found in how the membership flags are used during the SN phase, which is illustrated in Figure 5.4. Instead of only gathering *schedule requests* R' from the nodes whose flags are set in M, each node also merges the information on network membership that it receives from others into the temporary membership flags M'. Information is only merged (line 18 in Algorithm 3) if both transmitter and receiver of a packet consider each other members of their network (line 14), as discussed below in more detail. This merging mechanism prevents the computation of a new schedule until a node has obtained all the required information (membership flags M and requests r_j^v) from all nodes it considers part of its network as well as their merged notion of the network (line 20). We call such a set where the corresponding request is known for each temporary membership flag, i.e., $\forall j \in [1, N] : M'[j] = 1 \Rightarrow R'[j] \neq \perp$, a *complete set of information*.

Merging criteria. A node only accepts information from another node if both consider one another part of the same network. Checking whether nodes expect each other is done by verifying that a node i is part of the temporary membership flags of j and vice versa (line 14). This acceptance check is crucial to prove the uniqueness of the complete set of information, presented in Section 5.5, and especially relevant for asymmetric links, as discussed in Section 5.6.5.

Updates. To update the membership flags and maintain adaptivity, we introduce the *connectivity counters* C_r and C_e which persist for a round and an epoch, respectively. Whenever a request r_j^v from node j is received, $C_r[j]$ is set (line 13). At the end of the SN phase, $C_e[j]$ is incremented if a request from node j has been received (line 32). Finally, at the end of an epoch, a node j may stay in the expected set of nodes in M if its request has been received at least C_{stay} times or it may join if $C_e[j] \geq C_{join}$ (line 13). Otherwise, it is removed from the local membership flags and not expected anymore for consensus. Therefore, C_e serves as a means to update the current network structure and offers flexibility to the system designer. Usually, one might set $C_{stay} \leq C_{join}$ so a node that already joined the network is likely to stay and scheduling is not disturbed. Decoupling these two thresholds permits setting a high entry level to the network (by

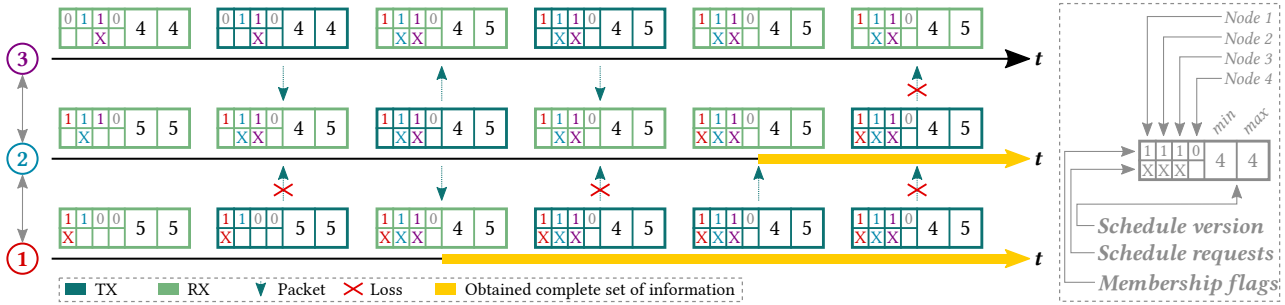

Figure 5.4: During the SN phase, membership flags and schedule requests are exchanged to obtain the complete set of information. In this example, node 4 failed and is not considered part of the network by nodes 1 - 3, which can reach consensus without requiring its input. At the end, nodes 1 and 2 have obtained the complete set of information, whereas node 3 is missing the request from node 1. While node 3 does not initially expect node 1, it correctly determines a lack of information after merging node 2's packet in the third slot. As node 3's schedule version 4 is outdated, nodes 1 and 2 cannot compute the next schedule and will retransmit their schedule 5 in the following SD phase.

increasing C_{join}) to enforce reliable communication links while preserving network stability.

5.4.3 Schedule Computation

To ensure liveness, each node stores its current *schedule* S^v, where $S^v[k] \in \{\perp\} \cup [1, N]$ denotes the allocation of a slot $k \in [1, K]$ in the DD phase to a node. However, always transmitting according to a known schedule might endanger safety unless dedicated mechanisms prevent the collision of packets from Hydra nodes.

Versioning. The *schedule version* v uniquely identifies a schedule and denotes a node's current level of information. A request r_i^v for the next schedule must be unique for every schedule version v, i.e., it can only be adjusted when the schedule is updated. We use schedule versioning to control the pace of adaptivity and ensure that the entire network has been able to obtain the same scheduling information before applying further changes to it based on new requests. Nodes independently acquire requests until they have a complete set of information (line 20), but can only compute a new schedule if all requests are based on the same schedule version (line 21). All nodes then apply a deterministic scheduling algorithm with the current schedule S^v and the complete set of information, i.e., the requests $R'[j] = r_j^v$ from all nodes with $M'[j] = 1$, as input for the scheduling function *SCHEDULE* (line 26). If a node detects that an old schedule version is still in use based on the minimum schedule version in the network, the most recent schedule will be retransmitted (line 30) until all nodes have been able to catch up. Thereafter, adaptation may resume.

Compatibility. To ensure that successive schedules S^v and S^{v+1} are safe, i.e., they do not assign the same DD slot to different nodes, the following conditions must be satisfied. Suppose that we are given a schedule S^v based on previous requests and that new schedule requests R' lead to a different number of assigned slots. To update the schedule, slots of S^v that are not required anymore to reach the newly determined number of slots are released. If a node receives more slots in the next schedule, free slots in S^v are assigned to it. This means that a slot assigned in S^v can only be used by the same node in S^{v+1} or be released, and a free slot in S^v can be assigned to any node in S^{v+1} or remain free. Crucially, safety is guaranteed even if both schedules are used in parallel. Furthermore, this scheme can be used for any scheduling algorithm as transitions to an arbitrary slot assignment are possible in two consecutive steps, which we will prove in Section 5.5.3.

Algorithm 3 Protocol behavior of node i after bootstrapping

DD (data dissemination) phase
  Every node i with $v > 0$ participates with its current schedule S^v ;

SN (schedule negotiation) phase
  Initialization
    $v^{min} \leftarrow v$; $v^{max} \leftarrow v$;
    for all $j \in [1, N]$ **do**
      $C_r[j] \leftarrow 0$; $M'[j] \leftarrow M[j]$; $R'[j] \leftarrow \begin{cases} r_i^v & \text{if } j = i \\ \perp & \text{otherwise} \end{cases}$;
  Sending
    send packet $p = (i, v^{min}, v^{max}, M', R')$;
  Receiving
    if (packet $q = (j_q, v_q^{min}, v_q^{max}, M_q, R_q)$ received) **then**
      for all $j \in [1, N] : R_q[j] \neq \perp$ **do**
        $C_r[j] \leftarrow 1$;
      if $((M'[j_q] = 1) \wedge (M_q[i] = 1))$ **then**
        for all $j \in [1, N] : R_q[j] \neq \perp$ **do**
          $I_e[j] \leftarrow 1$;
        $v^{min} \leftarrow \min\{v^{min}, v_q^{min}\}$; $v^{max} \leftarrow \max\{v^{max}, v_q^{max}\}$;
        $R' \leftarrow R' \circ R_q$; $M' \leftarrow M' \vee M_q$;
  Final
    if $((\forall j \in [1, N] : M'[j] = 1 \Rightarrow R'[j] \neq \perp) \wedge$
       $(|\{j \in [1, N] : M'[j] = 1\}| > N/2))$ **then**
      if $(v^{min} = v^{max})$ **then**
        if $(v > 0)$ **then**
          if $(S^v = SCHEDULE(S^v, R'))$ **then**
            $unchanged \leftarrow 1$;
          else
            $S \leftarrow SCHEDULE(S^v, R')$; $updated \leftarrow 1$;
        else
          $synced \leftarrow 0$; switch to Algorithm 2 ;
      else if $(v = v^{max})$ **then**
        $retransmit \leftarrow 1$;
    for all $j \in [1, N]$ **do**
      $C_e[j] \leftarrow C_e[j] + C_r[j]$;

SD (schedule distribution) phase
  if $((f = F - 1) \wedge (updated = 1))$ **then**
    $v \leftarrow v + 1$; $S^v \leftarrow S$; send packet (v, S^v) ;
  else if $(retransmit = 1)$ **then**
    send packet (v, S^v) ; $retransmit \leftarrow 0$;
  else if $(unchanged = 1)$ **then**
    wait until the end of the phase ;
  else if (packet $q = (v_q, S_q)$ received) **then**
    $v \leftarrow v_q$; $S^v \leftarrow S_q$;
    for all $j \in [1, N]$ **do**
      $I_e[j] \leftarrow 1$;

Algorithm 4 Protocol behavior of node i at the end of the round

```
1:  End of round
2:      f ← f + 1 ;
3:      if (f = F) then
4:          updated ← 0 ; unchanged ← 0 ; f ← 0 ;
            ▷ Check if schedule must expire as not part of majority ◁
5:          if (∑_{j∈[1,N]} I_e[j] ≤ N/2) then
6:              v ← 0 ; E ← E + 1 ;
                ▷ Check if node has lost connection to the network ◁
7:              if (E ≥ E_max) then
8:                  synced ← 0 ; switch to Algorithm 2 ;
9:          else
10:             E ← 0 ;
11:         for all j ∈ [1, N] do
                ▷ Check if nodes newly joined or stayed ◁
12:             if (((M[j] = 0) ∧ (C_e[j] ≥ C_join)) ∨
                    ((M[j] = 1) ∧ (C_e[j] ≥ C_stay)) ∨
                    (j = i))                                then
13:                 M[j] ← 1 ;
14:             else
15:                 M[j] ← 0 ;
16:             C_e[j] ← 0 ; I_e[j] ← 0 ;
```

Epochs. Nodes start using a new schedule at the beginning of an epoch at *epoch offset* $f = 0$. During an epoch of *epoch length* F, Hydra gathers information on a node's connectivity to update its local notion of the network for the next epoch. If node j's request r_j^v was received as collected in $C_e[j]$, it is included in the network by setting the membership flag $M[j]$ (line 13). Increasing F stabilizes network membership through a longer observation period, but delays the adaptation of the network membership and the schedule.

Expiration. Nodes that are not part of the network anymore must be prevented from using outdated schedules, as they would break the compatibility between schedules that is only guaranteed for two successive ones. Therefore, such schedules must expire.

Similar to the connectivity counters, we introduce the *interaction counter* I_e. $I_e[j] \leftarrow 1$ is set if either information from a node j in the network was received (line 16), or if a schedule was obtained (line 43). At the end of an epoch at $f = F$ (line 3), a node only keeps its schedule if it is mutually connected to a majority of nodes or if it has received the schedule in this epoch (line 5). This mechanism guarantees that a possibly outdated schedule expires (line 6) unless it either has been received in the

current epoch, i.e. it is up-to-date, or if the node has ensured that it is still part of the network. Notice that a schedule can be kept even if no node has obtained the complete set of information. To verify that a node remains part of the current network, information is aggregated throughout the epoch. This duration can be tuned using F to ensure liveness despite a high failure rate. If this verification succeeds, the node's schedule cannot be outdated and can still be used without violating safety, as we will show in Section 5.5.2.

5.5 Formal Analysis of Hydra

Consensus protocols are known to be notoriously complicated to understand and prove [Lam01, OO14]. However, due to the variety of states in distributed systems, it is essential that their properties are not only experimentally shown but are based on theoretical guarantees covering all conditions. In the following, we hence present formal proofs for safety, liveness, and adaptivity.

FLP and CAP. The FLP [FLP85] and CAP [Bre00] theorems are well-known impossibility statements and hence might cause concern regarding the ability to provide proofs for Hydra. However, the FLP theorem applies to asynchronous systems where arbitrary message delay is possible and cannot be distinguished from faulty nodes, while Hydra uses timeouts to detect and tolerate node failures. Similarly, the CAP theorem is not violated because Hydra limits partitions as it only permits a single majority network to operate at any time.

5.5.1 Safety

For Hydra's distributed consensus, we rely on the fact that for arbitrary initial membership flags M_k of a node k, i.e., independent of the set of nodes assumed to be reachable by k, the schedule requests R'_k used to compute the next schedule (line 26) are identical for nodes with a complete set of information. In short, nodes with a complete set of information compute the same schedule.

Theorem 5.1 (Consensus on current network)**.** *After the* SN *phase and independently of the initial membership flags M_k of any node $k \in [1, N]$, any two nodes $i, j \in [1, N]$ with a complete set of information either satisfy $R'_i = R'_j$ (identical requests) or $R'_i[k] \neq \perp \Rightarrow R'_j[k] = \perp$ for all $k \in [1, N]$ (requests from disjoint sets of nodes).*

Proof. To prove the statement, we use a graph interpretation of the protocol where each node corresponds to a node $i \in V$ in a directed graph $G = (V, E)$. The initial membership flags M_i lead to the edges E of G: If $M_i[j] = 1$, then $(i, j) \in E$. Note that $(i, i) \in E$. We associate two sets with each node $i \in V$:

- $m_i \subseteq V$, where initially we have $m_i := \{j : (i, j) \in E\}$. This set corresponds to M'_i, i.e., we have $j \in m_i \Leftrightarrow M'_i[j] = 1$.
- $r_i \subseteq V$, where initially we have $r_i := \{i\}$. This set corresponds to R'_i, i.e., we have $j \in r_i \Leftrightarrow R'_i[j] \neq \bot$.

We perform arbitrary updates on these sets according to the protocol. In particular, node i may merge information with a node j if $j \in m_i \wedge i \in m_j$ (line 14). If so, the sets of node i are updated to $m_i \leftarrow m_i \cup m_j$ and $r_i \leftarrow r_i \cup r_j$. A node has a complete set of information if $m_i = r_i$. Using this graph representation, we now prove an equivalent statement independently of G and the number and order of merges: *Any two complete nodes $i, j \in V$ either satisfy $r_i = r_j$ or $r_i \cap r_j = \emptyset$.* To show this result, we apply three invariants:

- If $k \in r_i$, then $l \in m_i$ for all l with $(k, l) \in E$, i.e., if a node k is in the set r_i, then not only is this node in the set m_i but also all its immediate successor nodes. This also leads to $r_i \subseteq m_i$. The proof of this property can be done using induction with the initial sets m_i and r_i as a base case. Supposing that the property holds for any nodes i and j, and, without loss of generality, that information of node j is merged with the information at node i. Then we can show that it holds for the new sets of node i after merging, namely $m'_i := m_i \cup m_j$ and $r'_i := r_i \cup r_j$. If $k \in r'_i$, then either $k \in r_i$ or $k \in r_j$. If $k \in r_i$, then all immediate successors are in m_i and therefore also in m'_i. If $k \in r_j$, then all immediate successors are in m_j and therefore also in m'_i.
- If $k \in m_i$, then there exists a directed path in G from i to k. Initially, this property holds as $k \in m_i \Leftrightarrow (i, k) \in E$. Supposing that the property holds for all m_j and that j merges its information with i, we then show that the property also holds for the new set $m'_i := m_i \cup m_j$. If $k \in m'_i$, then either $k \in m_i$ or $k \in m_j$. In the first case, the property holds due to the assumption. In the second case, note that $j \in m_i$ due to the merge criteria, and therefore, there exists a path $i \to j$ and a path $j \to k$ which leads to a path $i \to k$ after merging.
- If $k, l \in r_i$, then there exists a directed path in G from k to l and vice versa. Supposing that the property holds for all r_j and that j merges

its information with i, we then show that the property also holds for the new set $r_i' := r_i \cup r_j$. If $k, l \in r_i$ or $k, l \in r_j$, then the property holds due to the assumption. Now, without loss of generality, consider that $k \in r_i$ and $l \in r_j$. Note that due to the merge criteria, we have $i \in m_j$ and $j \in m_i$ and therefore, we know from the statement above that paths $i \longrightarrow j$ and $j \longrightarrow i$ exist. Moreover, $i, k \in r_i$ and $j, l \in r_j$. Therefore, we can conclude that there are paths $k \longrightarrow i \longrightarrow j \longrightarrow l$ and $l \longrightarrow j \longrightarrow i \longrightarrow k$, which proves the property.

Based on the above invariants, we can conclude that if a node has the complete set of information ($m_i = r_i$), the nodes in m_i are strongly connected (each node $k \in m_i$ can reach any other node $l \in m_i$), and for any node in m_i, all its immediate successors are also in m_i. As a result, m_i of a complete node i forms an strongly connected component (SCC) of G. As all immediate successor nodes are already in m_i and therefore no node can be added, it is a *maximal* SCC. As such SCCs form a partition of all nodes [Tar72], each node $i \in V$ can be in exactly one SCC, which proves the property. □

Corollary 5.1.1 (Uniqueness of schedule computation)**.** *Any two nodes $i, j \in [1, N]$ that have obtained the necessary information to compute a new schedule will compute the same schedule with the same schedule version.*

Proof. We know from Theorem 5.1 that such nodes must have complete sets of information that are either identical or disjunct. Because only a single disjunct set can contain a majority of nodes required for schedule computation (line 20), at most one of them can compute a new schedule if their requests are disjunct. Only nodes that share the complete set of information of the majority strongly connected component (MSCC) as defined in Theorem 5.1 have the preconditions for schedule computation, with all such nodes in the current network having the same set of schedule requests R' and the same schedule version (line 21). As the scheduling algorithm is deterministic, they compute the same schedule S with the new schedule version $v' := v + 1$, which guarantees the uniqueness of a newly computed schedule. □

Based on the above result, the following two theorems show the safety of Hydra, i.e., no two nodes send different packets at the same time in the SD phase (Theorem 5.2) or DD phase (Theorem 5.3).

Theorem 5.2 (Safety of schedule distribution)**.** *In the* SD *phase, no two nodes $i, j \in [1, N]$ transmit packets with different contents (v, S^v) (lines 35 and 37 of Algorithm 3).*

Proof. Only nodes that have received the identical complete set of information satisfy either *updated* = 1 or *retransmit* = 1 (see proof of Theorem 5.1) and send packets in the SD phase. If the condition in line 21 is satisfied, all nodes in the current network have the same schedule version and the same set of schedule requests R'. As the scheduling algorithm is deterministic, they compute the same schedule S with the new schedule version and distribute it in line 35. If the condition in line 29 is satisfied, there exists exactly one maximum schedule version v^{max} for the set of nodes in the current network that obtained the complete set of information. This follows as the element-wise max operator is idempotent and commutative, and therefore all nodes in the MSCC that obtain the complete set of information compute the same v^{max}. From these nodes, only the subset that knows the corresponding schedule will transmit it. Notice that as reaching line 29 is mutually exclusive with fulfilling the condition in line 21 and as all nodes in the network operate on the same set of information, only one of these two cases can occur within the network, which proves the statement. □

Theorem 5.3 (Safety of data dissemination)**.** *In the* DD *phase, no two nodes* $i, j \in [1, N]$ *transmit different packets in the same* DD *slot (line 2 of Algorithm 3).*

Proof. According to the scheduling conditions described in Section 5.4.3, two successive schedule versions can be used by any nodes $i, j \in [1, N]$ without violating safety. Note that nodes with $v = 0$ do not participate in the DD phase and are safe (line 2). Therefore, we need to show that at the beginning of the DD phase, nodes have either successive schedule versions or satisfy $v = 0$. We will prove this theorem by induction, where the schedule with version $v^* = 2$, the first one involving requests from other nodes, as well as the empty schedule with version $v^* - 1 = 1$ originating from the bootstrapping protocol (line 38 in Algorithm 2) serve as a base case.

Suppose that two schedule versions v^* and $v^* - 1$ are in use at the start of a round. Let us first look at nodes within the current network, i.e., the SCC of G in the proof of Theorem 5.1. Due to multiple used schedule versions, the next schedule cannot be computed in line 26 and hence cannot be distributed in line 35. Only after all nodes with $v^* - 1$ have received version v^* in line 41 (and $v^* - 1$ is not used anymore), a new schedule $v^* + 1$ can be computed as an induction step. At any time, there are at most two schedules in use which are successive.

A node that is not part of the current network may still know the outdated schedule version $v^* - 1$ if it did not receive any of the newer

schedules distributed in the SD phases. The continued use of $v^* - 1$ is prevented by forcing $v \leftarrow 0$ in line 6 of Algorithm 4 as neither has a schedule been received in any SD phase of the current epoch nor is the node part of a majority network, i.e., the MSCC of G with more than $N/2$ nodes. The latter is derived in line 5 from variables $I_e[j]$ which are only set in line 16 of Algorithm 3 if the node has a bidirectional path to a node j in G. As the node is not part of the current network anymore, it can only be part of a sub-network containing a minority of nodes and hence cannot acquire sufficient interactions to prevent such a schedule expiration. □

As a result, nodes can independently and asynchronously update their schedules and remain assured that their transmissions will not collide with data packets from nodes using another schedule. However, it is not obvious that matching schedule versions also indicate that the schedules themselves (i.e., the allocation of slots) are equivalent. In particular, because only a minority of nodes might be able to compute the next schedule and may then get disconnected, a situation may arise that forces the remaining majority to find consensus anew and compute a schedule of the same version (but with a different allocation of slots) based on only a subset of the previously participating nodes. In Theorem 5.4, we demonstrate that such unambiguity is indeed given at any point in time.

Theorem 5.4 (Unambiguity of schedule version)**.** *At any point in time, a schedule with version v is unique, i.e., if two nodes $i, j \in [1, N]$ simultaneously know a schedule with version v, then $\forall k \in [1, K] : S_i^v[k] = S_j^v[k]$.*

Proof. We prove this theorem by induction. As a base case, we use the first schedule S^1 with the unique initialization of the slot assignments $\forall k \in [1, K] : S^1[k] = \perp$ (line 40 of Algorithm 2).

As an induction step, we assume that a unique schedule S^{v-1} has been generated based on a complete set of information (M'^{v-1}, R'^{v-1}), where $|\{j \in [1, N] : M'^{v-1}[j] = 1\}| > N/2$. As M'^{v-1} consists of a majority of nodes, we know that a subset of nodes from M'^{v-1} must be part of any complete set of information (M'^v, R'^v) in order to generate schedule S^v (as two majorities of nodes cannot be disjoint and input from a majority of nodes is required in line 20). During each round, we know from Corollary 5.1.1 that a maximum of one complete set of information can be generated. As the membership flags remain fixed during an epoch (with lines 13 and 15 of Algorithm 4 only being executed at the end of an epoch), this property also holds for the entire epoch. Therefore, we can conclude that maximally one unique schedule version v can be generated per epoch.

However, it remains to be seen whether the version can persist after a change of membership flags at the end of the epoch and will become the dominant schedule version. We know that a schedule is only prevented from expiration if a node receives information from a majority of nodes that are within its MSCC at least once during the epoch (line 5). If sufficient nodes are using the previous schedule S^{v-1} in a future epoch so that a new complete set of information $(M'^{v'}, R'^{v'})$ can be formed, the previously computed schedule S^v is not in use by a majority of nodes (otherwise, no majority of nodes using version $v-1$ could have been found to generate version v'). Therefore, nodes using S^v must belong to a different SCC and could only set $I_e[j] \leftarrow 1$ (line 16) for nodes $j \in \Delta M = \{i \in [1,N] : M'^{v'}[i] = 0\}$, because others cannot pass the check in line 14 as they belong to the SCC using $M'^{v'}$. As we know that $|\{j \in [1,N] : M'^{v'}[j] = 1\}| > N/2$, $|\Delta M| < N/2$. Therefore, all nodes with schedule S^v must invalidate their schedule in line 6 at the latest at the end of the same epoch in which $S^{v'}$ has been generated. Hence, we can conclude that at any point in time, a schedule of version v is unique. □

Lastly, we also need to consider the interplay of bootstrapping nodes with an already existing network. While such nodes are separated based on the use of different frequency channels c_{boot} and c_{main}, switching between algorithms must be done with care, as we show in Theorem 5.5.

Theorem 5.5 (Safe conduct of unsynchronized nodes)**.** *A network operating on the main channel c_{main} does not experience interference during the* DD *phase due to unsynchronized nodes.*

Proof. A node that lost contact with its network $\mathcal{N}$ will have its schedule expire at the end of the next epoch (line 6) and will not participate in line 2 during the DD phase thereafter. It can still try to regain connectivity and will communicate with other nodes during the SN phase for up to E_{max} epochs without influencing the DD phase, after which it will be forced to revert to bootstrapping (line 8). Note that E_{max} must be chosen appropriately so that the maximal clock drift within this time may not cause interference.

As Algorithm 2 is using the boot channel c_{boot} for all transmissions, no interference with the main channel c_{main} can result from bootstrapping nodes. A node can only return to the main channel in one of two cases. In line 12, it can switch to the main channel in case it re-synchronizes to the network $\mathcal{N}$ on the main channel in line 4 and hence cannot interfere with the DD phase as it just synchronized and uses $v = 0$ (line 6). In line 43, a node can switch to the main channel as part of a majority of nodes that

forms a new network $\mathcal{N}'$. As the previous network $\mathcal{N}$ can only remain operational as long as at least a majority of nodes regularly interact and execute line 16 to prevent schedule expiration in line 6, this indicates that only a minority remains on the main channel. For a node to switch in line 8, it must have lost connectivity for at least $E_{max} > 0$ epochs during which no majority from $\mathcal{N}$ was interacting. As a single epoch of missing interaction is sufficient for all these nodes to have their schedules expire, all remaining nodes of $\mathcal{N}$ must have executed line 6 at least once and therefore have $v = 0$. Therefore, the previous network $\mathcal{N}$ does not exchange packets anymore during the DD phase as well as the SD phase. The latter conclusion follows as a new schedule cannot be computed due to the condition in line 20 not being satisfied. With $\Delta\tau \geq F \cdot T$, the remaining minority of $\mathcal{N}$ will have had to execute line 8 and have switched to the boot channel before the new majority $\mathcal{N}'$ finishes waiting in line 37, resulting in no more transmissions during an SN phase on the main channel. Therefore, when a new network $\mathcal{N}'$ starts communicating on the main channel, all other nodes must have switched to the boot channel and the network can operate without interference. □

5.5.2 Liveness

The following Theorem 5.6 shows that nodes may continue participating using the current schedule and can communicate data packets in the DD phase of every round even under broad failure conditions.

Theorem 5.6 (Liveness of data dissemination)**.** *Every node participates in the* DD *phase unless both of the following conditions are satisfied: (i) It did not receive a schedule in any* SD *phase of the previous epoch and (ii) during the entire previous epoch, it only stored the requests from a minority of nodes.*

Proof. A node does not participate in the DD phase if $v = 0$, i.e., if line 6 was executed. This only happens if a minority of nodes $j \in [1, N]$ satisfy $I_e[j] = 1$ (line 5). If the node has received a schedule, then $I_e[j] \leftarrow 1$ for all nodes (line 43) and the condition is not satisfied. If the node stores the request of a node j, it sets $I_e[j] \leftarrow 1$ (line 16). Therefore, if the node did store the requests from more than $N/2$ different nodes at least once during the epoch, the condition to set $v \leftarrow 0$ is not satisfied either. □

Note that a schedule is typically computed and distributed by many nodes due to the high level of redundancy. Therefore, a node will only fail to receive a schedule in any round of an epoch under a significant amount of very specific and simultaneous node or link failures. And even in such

a case, a node may still participate in the data exchange if it has received and stored requests from a majority of nodes at least once during the entire previous epoch. Through the use of the epoch length F, this aggregation period can be arbitrarily extended, which diminishes the chance of a loss of liveness due to short-term failures or disturbances.

5.5.3 Adaptivity

Finally, we must make sure that Hydra always remains operational and cannot result in a deadlock that may permanently prevent nodes from adapting to new conditions and requests. In particular, care needs to be taken with corner cases such as all nodes having their network expire at the same time (i.e., setting $v \leftarrow 0$) as this still prevents nodes from reverting to bootstrapping but also renders schedule computation impossible (which is handled in line 28 in Algorithm 3).

Theorem 5.7 (Perpetuality)**.** *Nodes using Hydra are always able to eventually adjust to a change in operating conditions such as demand or environmental influences.*

Proof. While failures are occurring, adaptivity might be persistently delayed due to nodes constantly joining and leaving. However, Hydra makes sure to avoid reaching a state where the nodes cannot adapt even though no more disturbances are encountered.

With Algorithm 2, any majority of nodes can eventually re-establish a network in line 43 once they could synchronize and exchange messages. Furthermore, a node that is able to communicate with a majority of nodes will eventually encounter their communication and join in line 12. As it can arbitrarily often execute line 14 and restart the algorithm, it will never get stuck during bootstrapping.

With Algorithm 3 and Algorithm 4, an established network is also guaranteed to eventually find consensus and progress if communication is possible. If a node or link has failed, the network membership will be updated in line 15 at all remaining nodes and consensus can be found again. If a majority cannot be established anymore due to insufficient nodes, nodes can return to bootstrapping in line 8 and attempt to recombine with other sub-networks. If a node is missing the newest schedule, it can receive it in line 41 as it will be retransmitted by nodes that have indicated that they know the maximum schedule version in the network. Once the preconditions for schedule computation in line 20 (a complete set of information from a majority of nodes) and line 21 (equal schedule versions) are met, the network can adapt its schedule. □

In Theorem 5.8, we show that Hydra does not restrict the choice of deterministic real-time scheduling policies despite its scheduling requirements. Hence, adapting the schedule remains effectively unrestricted by the protocol for the system designer.

Theorem 5.8 (Support for scheduling algorithms)**.** *All deterministic real-time scheduling policies can be supported, i.e., the slot assignment S^v can be transformed into any slot assignment $S^{v'}$ in at most two steps.*

Proof. We distinguish three cases of slot transitions from one schedule to another. For $\{k \in [1,K] : S^v[k] = S^{v'}[k]\}$ (i.e., the slots whose assignment does not change from S^v to $S^{v'}$), no changes need to be made. For $\{k \in [1,K] : S^v[k] = \bot\}$, $S^{v'}[k]$ can be set to any desired value within one step as the assignment is not restricted in this case. For $\{k \in [1,K] : S^{v'}[k] = \bot\}$, assigned slots from $S^v[k]$ can be released within one step.

For the remaining set $\Delta S = \{k \in [1,K] : (S^v[k] \neq S^{v'}[k]) \wedge (S^v[k] \neq \bot) \wedge (S^{v'}[k] \neq \bot)\}$, we create an intermediate schedule $\tilde{v}$ as a first step where

$$S^{\tilde{v}}[k] = \begin{cases} \bot & \text{if } k \in \Delta S \\ S^v[k] & \text{else} \end{cases}$$

In a second step, we can use the same argument for the case of $S^{\tilde{v}}[k] = \bot$ to conclude that we can now also set $S^{v'}[k]$ to any desired value $\forall k \in \Delta S$. This shows that a maximum of two schedules is required to transform one slot assignment into any other desired slot assignment. As the algorithm is deterministic by definition and the inputs to the schedule have been shown to be equivalent in Theorem 5.1, the decentralized computation of any deterministic real-time scheduling policy in line 26 is supported. □

5.6 Evaluation

To validate Hydra's fault tolerance to node and link failures and demonstrate its ability to provide a reliable communication service, we test the protocol for different failure types and network sizes. We thereby investigate how the network (*i*) maintains safe data exchange, (*ii*) adapts to changing conditions, and (*iii*) keeps communication alive between non-faulty nodes. By comparing against LWB [FZMT12], a state-of-the-art wireless protocol using centralized scheduling, we highlight that concurrent coordination eradicates any single point of failure without noticeably increasing protocol overhead. Throughout our experiments, we

never encountered any safety violation in thousands of Hydra rounds while scheduling under various conditions and adverse failures, experimentally verifying that our formal proof of safety also holds in practice.

5.6.1 Implementation

We implement Hydra on an STMicroelectronics STM32L433CC microcontroller driving a Semtech SX1262 RF transceiver [Sem23]. The package is available as a target [BTF+19] on the FlockLab testbed [TDFS+20], operates in the 868 MHz band, and provides a wide range of TX power from −9 to +22 dBm, making it highly versatile for changing environmental conditions. To minimize the impact of external influences, we use the GFSK modulation at 250 kbps. One-to-all floods are implemented using Glossy [FZTS11] with multiple consecutive transmissions [LDFST17], while the all-to-all exchange is based on Chaos [LFZ13] with randomized initial TX/RX decisions to commence without a network coordinator which usually triggers the exchange.

Research artifacts. The source code is publicly available as open source [BDFK+23b]. In addition, we provide tools to thoroughly and reproducibly test Hydra in different failure scenarios using a highly-flexible input method leveraging GPIO actuation. Analysis scripts automatically validate the correct execution of the communication primitives and protocol phases, including safety and synchronization checks. Our companion document [BDFK+23b] further provides a more in-depth description of the algorithms employed during and after bootstrapping.

Packet structure. As the data exchange in the DD phase consists of normal Glossy floods, Hydra requires no adjustments to the data packets. The packets in the SN phase contain $N/8$ B for the membership flags and N requests of 3 bits each to add or remove up to 3 slots per node and schedule version. Including 2 B for the extremal values of the schedule version, 14 B are sufficient to obtain the complete set of information of a network with $N = 24$ nodes. However, we encountered occasional bit flips which were not caught by the standard 16-bit hardware CRC appended by the radio. As the distributed consensus mechanism relies on the correctness of the packet content and does not consider Byzantine failures, we included an additional 32-bit CRC in the payload that successfully eliminates corrupted packets. Lastly, the schedule in the SD phase is compressed to $log(N)$ bits per DD slot and fits into 50 B for 80 slots and $N = 24$.

SN *propagation policy.* As Hydra does not rely on the successful completion of the SN phase for liveness and safety, the propagation policy according to which a node either transmits or receives in a Chaos slot is flexible. We opt to remain close to the standard Chaos policy [LFZ13] to achieve reliable performance but adapt the initiation to avoid a single point of failure. Through randomized TX/RX decisions with a fixed TX probability of 25 % until the first successful reception, Hydra begins without relying on a dedicated node. This concurrent start is only made possible by the tight time synchronization obtained through the preceding Glossy floods. Thereafter, we rely on more selective transmissions either after a randomized timeout of 3 to 5 slots or after a packet with new information has been received so that its content can quickly propagate to neighbors which are likely also lacking this information. Nodes that obtain the complete set of information aggressively transmit 5 consecutive times and thereafter fall back to randomized TX/RX decisions if they detect that others are still missing information. Policies considering the local node density, such as presented by Mixer [HMZ18], could further improve performance at the cost of delayed adaptivity as they require gathering reliable neighborhood information over multiple rounds.

Finite schedule versions. Our packet structure uses 1 B to store a schedule version which should monotonically increase with each new schedule computation but is limited by the variable size. To solve this issue, we leverage the fact that we know that the difference between schedule versions in a network is bounded to 1, as only successive schedules can co-exist. This allows us to apply modulo arithmetic and preserves the correctness of comparisons if the version exceeds the variable size.

Time synchronization. Without a unique time reference whose long-term availability is guaranteed, clock drift compensation becomes challenging as a node keeps synchronizing with a varying set of nodes. Nevertheless, continuous clock correction is important to maintain reliable communication even if synchronization is temporarily prevented. Therefore, we collect the clock tick shifts that a node applies per round when it re-synchronizes to others and compute a local correction factor. However, it may be that groups of nodes observe a perpetually increasing or decreasing clock drift. To maintain a stable clock speed, we exchange the local correction factors in the SN phase and aggregate the extremal values. This network information is used to estimate an average correction factor as the mean of the minimum and maximum and normalize the local factor that nodes apply to stabilize the clock drift.

In addition, the synchronization preceding the SN phase can be improved if the last few DD slots are assigned and used. This follows as these slots are used by all nodes in the network to re-synchronize and provide a highly reliable and precise synchronization for the contention-based communication following directly thereafter. Therefore, our implementation of the scheduling algorithm prefers to assign slots towards the end of the round and fills the schedule starting at the end. Note that this does not restrict scheduling as it is entirely optional and does not influence any other features of the protocol, but that it can be included as an orthogonal consideration when designing a scheduling algorithm for Hydra.

Direct slot transfer. In order to permit a transfer of schedule slot assignments within a single schedule update, we can optionally introduce a scheduling policy where nodes can entirely dictate the number of released slots through their requests. This means that if a node's request is set to $\delta < 0$, the node will already stop using its first δ slots before it transmits the request. During the schedule computation, we can then first free all released slots and thereafter directly reallocate these (already unused) slots. As a node knows that it can dictate the number of released slots and already stops using them before the new schedule is computed, successive schedules still do not cause interference even if the releasing node might not be able to update its schedule and will continue using the previous version. Notice that this does not require any assumptions on whether additional slots are granted and is purely based on a node releasing its own slots earlier than required by the schedule.

5.6.2 Experimental Setup

Test scenario. To observe Hydra in practice including hardware failures and varying link characteristics, we utilize the FlockLab testbed [TDFS+20] with 23 nodes evenly spread across the 65×30 m floor of an office building. We test networks of three sizes to investigate Hydra's scalability: a small network consisting of 5 nodes, a medium network of 12 nodes, and a larger network containing all 23 usable indoor nodes of the testbed. A TX power of +7 dBm results in an average of 1.24 hops for the small network, 1.57 hops for the medium network, and 1.70 hops for the larger network. During normal operation, we use a main frequency channel at 869.8875 MHz with an average one-to-all packet reception rate (PRR) of 99.72 %, while bootstrapping occurs at 868.4375 MHz.

Fault injection. To reproducibly inject failures, we use the GPIO actuation feature of FlockLab [TDFS+20] to precisely affect the protocol execution of individual nodes. Actuating the reset pin allows us to cause full hardware failure. Depending on the configuration, a second GPIO pin changes link characteristics by either enforcing that nodes drop packets with a given probability or completely disconnecting them through a shift to an isolated frequency band. This mechanism enables us to re-play a real scenario multiple times to investigate it in more detail, promotes the re-production of our results, and permits a fair comparison between competing protocols.

Protocol parameters. As a default configuration, we use a round period of $T = 3$ s and an epoch length $F = 3$. Each node requests 3 slots per DD phase which are fairly allocated to one of 80 DD slots, transmitting data packets of 20 B in each flood using three transmissions. 36 contention slots during the SN phase permit nodes to exchange membership flags and requests. Nodes leave the network after $E_{max} = 2$ epochs with $v = 0$ and revert to bootstrapping. A bootstrapping node listens for traffic of an existing network with a probability of 20 % and otherwise attempts to establish a new network.

Protocol comparison. Since our analysis of the state of the art in Section 5.8 shows that no existing design satisfies all three requirements of safety, adaptivity, and liveness while tolerating node and link failures, we compare the concept of concurrent coordination to its closest relative based on a traditional centralized design.

We benchmark Hydra against LWB [FZMT12], a state-of-the-art protocol using CT, to represent a protocol with a single point of failure due to its centralized scheduling. Similar to Hydra's DD phase, LWB consists of a sequence of data slots where packets are flooded using Glossy [FZTS11]. A contention slot allows a node to transmit its demand to a host node as the network coordinator which centrally executes the scheduling algorithm and distributes the resulting schedule using two additional slots. In stark contrast to Hydra, nodes are only permitted to transmit if they have received the schedule directly preceding the round. To avoid that a single schedule miss prevents a node from participating, the schedule is transmitted both at the end of the previous round and at the beginning of the round for which it is valid. We use the same parameters for all slots as for Hydra's DD phase.

5.6.3 Hydra's Fault Tolerance in Action

Node failure. We first investigate how Hydra copes with node failures. During the entire experiment shown in Figure 5.5, we monitor the network PRR of fault-free nodes to examine how well the protocol adapts and whether it preserves liveness. The PRR is defined as the percentage of the total number of data packets received compared to the total number of data packets the nodes expect to receive according to their schedule. If no schedule is known, such as at the start, the PRR is defined to be 0 %.

Initially, we bootstrap a network of 23 nodes and do not inject failures for the first 90 s. We find that Hydra enables nodes to quickly form a network and transmit packets according to their demands within 13 s on average by merging requests into a single packet, while LWB requires three times as long to send individual requests to the centralized scheduler with a mean scheduling delay of 39 s. Hydra nodes that first searched for pre-existing networks during bootstrapping and did not participate from the start (such as nodes 2 and 3) quickly join the formed network after discovery. Hydra's PRR remains high once all nodes joined and validates its liveness, while LWB suffers occasional drops from schedule misses.

At 90 s, we inject a first node failure at node 1, which causes a drop in PRR for both protocols. Hydra quickly adapts by excluding node 1 after one epoch and determining a new schedule, thereby regaining efficiency as all assigned schedule slots are used. LWB on the other hand relies on a centralized timeout mechanism and delays the exclusion of node 1 until the network coordinator could confirm over multiple rounds that the node is permanently missing.

At 120 s, we inject a second node failure at node 2, which serves as the host node for LWB where the centralized scheduler is situated. Lacking a schedule, all LWB nodes must immediately cease transmissions to preserve safety as an old schedule might cause interference. As LWB's fixed failover leader only takes over after 2 min [FZMT12], communication collapses and nodes are forced to idle listen for a new leader. Hydra on the other hand has previously re-established efficient communication at 100 % PRR and now only experiences another minor drop. Even three nodes failing almost simultaneously at 150 s do not endanger Hydra's liveness.

At 180 s, we reactivate nodes 3 - 5, which quickly join the existing Hydra network but must wait for the network coordinator to reappear for LWB. As soon as node 2 is restarted at 190 s, LWB recommences and gradually includes one node per round due to the contention-based relaying of requests to the centralized scheduler.

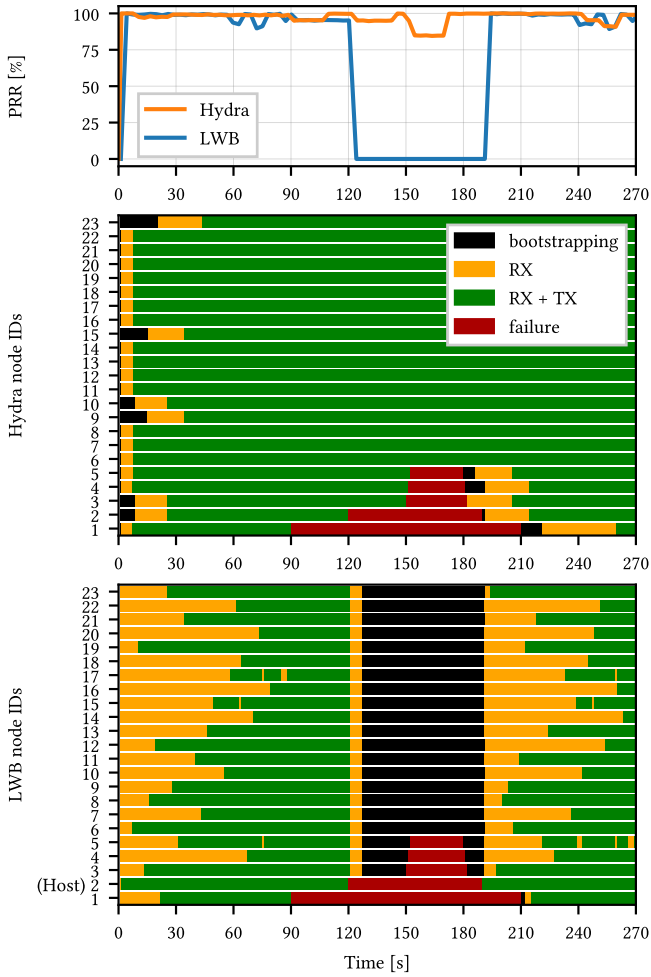

Figure 5.5: Hydra (middle) quickly bootstraps a network and starts exchanging scheduling information, with synchronized nodes listening according to the schedule ("RX") until their own request is granted. The network swiftly reaches distributed consensus, whereafter nodes efficiently transmit and receive data packets ("RX+TX") despite node failures. LWB (bottom) cannot maintain communication if the centralized scheduler at the host node 2 is unreachable.

We find that Hydra successfully adapts to node failures and is able to quickly re-establish efficient communication independently of which nodes are affected. Liveness is ensured even if distributed consensus is temporarily delayed as nodes leave and join, and a coordinated transition between schedules ensures that safety is preserved. On the other hand, LWB experiences long delays until nodes could communicate their requests to the centralized scheduler. While LWB provides both adaptivity and safety, it cannot maintain liveness as it suffers from a single point of failure at the host node which orchestrates communication. This centralization of network coordination results in a catastrophic collapse of the entire network when the host node is not reachable.

Link failure. Next, we investigate Hydra's behavior when the network splits, a common scenario for devices moving in swarms [LSD+20]. As network separation results in partitions that might interfere and endanger safety, we artificially form two sets of 11 and 12 nodes. Figure 5.6 shows how a network of 23 nodes is initially established. After 45 s, we prevent all communication between the two sets and observe a brief drop in the PRR of the majority set of 12 nodes as half of the expected packets are missed.

After these nodes obtain the new complete set of information, visible in the top plot just before 60 s by the return to 100 % of nodes being complete, they quickly determine a new schedule of version 5 as seen in the middle plot around 60 s. The schedule of the minority set of nodes already expires beforehand and their schedule version returns to 0. Finally, the minority again enters bootstrapping.

At 90 s, we reactivate communication between the two sets of nodes and observe that the minority set quickly rejoins the majority set. Around 105 s, the network determines a new schedule of version 6 which already includes a combination of the set of nodes.

We draw two major conclusions. First, we find that even if an arbitrary set of almost half the network is unreachable, the remaining nodes still continue to coordinate and are not impeded in disseminating their data due to the failure of others. This demonstrates Hydra's high degree of fault tolerance. Second, the expiration of the schedule for the minority set of nodes, seen by the drop of the schedule version to 0 in the middle plot, is crucial for safety. If the minority kept using it, it could immediately start interfering with a new schedule determined by the majority of nodes when both sets of nodes are re-combined at 90 s. Only by asserting that no one but the majority of nodes is scheduled can the compatibility of used schedules be ensured and safety unconditionally guaranteed.

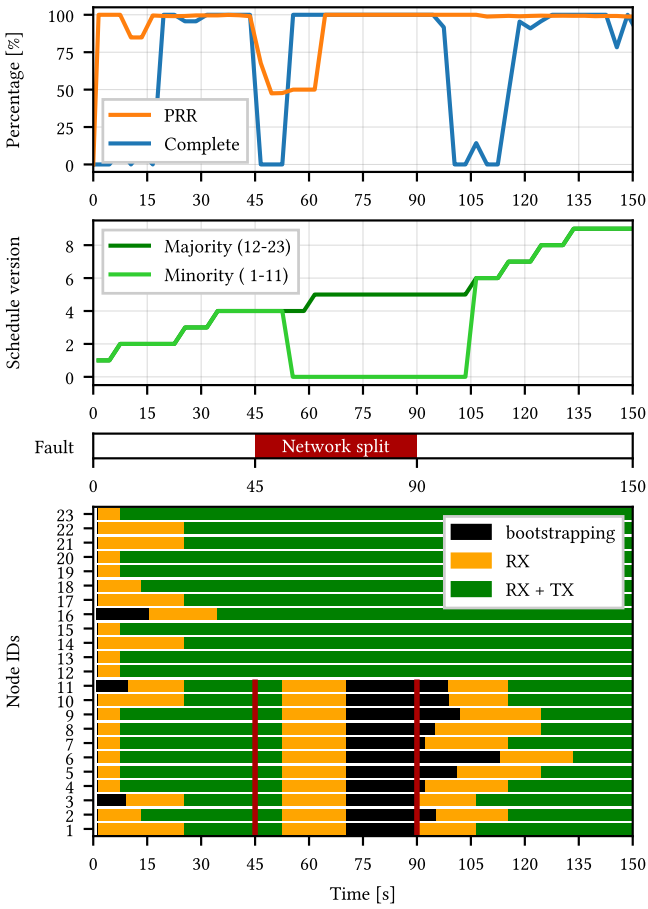

Figure 5.6: Splitting a network into a majority and a minority demonstrates that Hydra correctly forces the schedule of nodes being part of the minority to expire. To maintain safety, such nodes can only resume communication after re-connecting to the majority, whereafter a new complete set of information is swiftly obtained.

5.6.4 Distributed Scheduling Overhead

We have observed in Section 5.6.3 that distributed scheduling can greatly outperform centralized scheduling in case of node failures. However, concurrent coordination achieves its fault tolerance by relying on redundancy

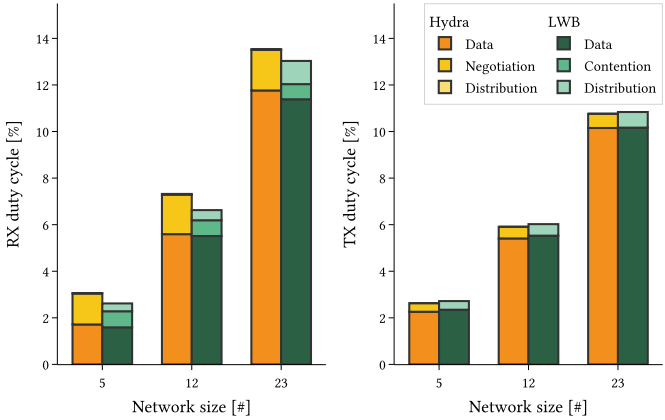

Figure 5.7: While data exchange is almost identical for both protocols as they rely on the same communication primitive, the difference in scheduling overhead is also negligible for larger networks.

and therefore strives for every node in the network to obtain the required scheduling information. To investigate whether this increases runtime overhead, we execute both Hydra and LWB for 900 rounds and compare the average RX and TX duty cycles of their different phases for multiple network sizes in Figure 5.7. As mentioned in Section 5.6.2, LWB includes a data dissemination phase just like Hydra's DD phase. In addition, LWB uses a contention slot for nodes to relay requests to the centralized scheduler. This functionality corresponds to the SN phase of Hydra's distributed scheduler. Lastly, LWB's network coordinator transmits the new schedule once at the end of a round and retransmits it at the beginning of the next round, while multiple Hydra nodes initiate a single schedule flood in the SD phase.

As data dissemination is equivalent for both Hydra and LWB with three packets flooded per node, the TX duty cycle does not differ for all network sizes. We find that Hydra's RX duty cycle for data exchange is slightly increased as Hydra starts to listen early in the first data slot of a round to compensate for potential clock drifts. Astonishingly, the scheduling overhead of both protocols is also almost identical, with the RX duty cycle rising by only 0.12 % and the TX duty cycle dropping by −0.02 % for a network of 23 nodes.

With the scheduling costs being almost equivalent for larger networks and smaller networks showing increases by less than a quarter despite

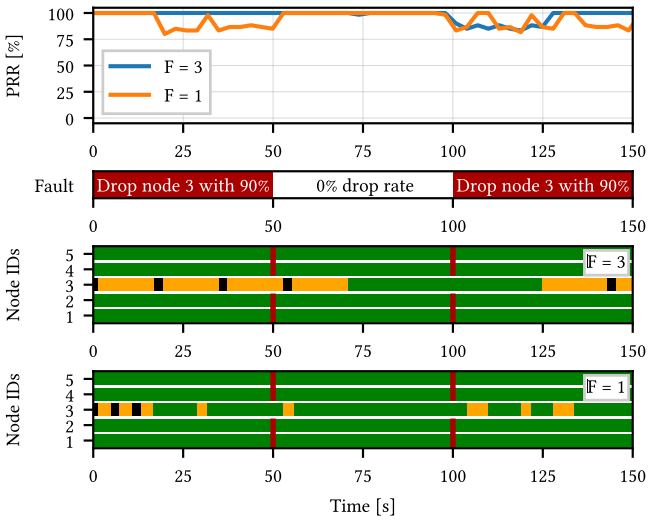

Figure 5.8: The epoch length F offers control of the speed of adaptivity and quality of included links. For both networks, 90 % of the packets from node 3 are dropped for the first and last 50 s. In contrast to $F = 1$ (bottom), node 3 is only part of the network with $F = 3$ (middle) when it communicates reliably, which boosts PRR.

over-provisioning for quick adaptivity (see Figure 5.9), we conclude that the overhead of distributed scheduling is negligible. While this result might initially be unexpected as the same data needs to be at all nodes instead of only one, it quickly becomes apparent that concurrent coordination also enables a reduction of the communication costs. First, as most nodes complete the SN phase successfully and no schedule needs to be distributed if requests remain constant, the SD overhead diminishes to only 0.02 % RX duty cycle. Second, the constant merging of received information in the SN phase leads to the exchange of scheduling information only requiring relatively few transmissions. Third, because the SN phase consists of short contention slots, Hydra does not need to idle listen for long Glossy slots provisioned for multiple hops, while LWB's RX duty cycle increases with a larger network size as floods take longer to reach nodes further away.

5.6.5 Investigating Hydra in Detail

Epoch length. Next, we examine the epoch length F as a tool to increase network stability in conjunction with C_{join} and C_{stay}, used in line 12 of Algorithm 4. By default, $C_{join} = C_{stay} = 1$ so nodes are included in M if their information is received at least once in an epoch. We now increase this threshold to $C_{join} = C_{stay} = F$ and compare the behavior of networks with $F = 1$ and $F = 3$ upon link failures in Figure 5.8. To simulate packet loss, we force a network of nodes 1, 2, 4, and 5 to drop received packets from node 3 with a probability of 90 % for the first 50 s. We find that a network with $F = 3$ successfully prevents such a node with unreliable links from joining and therefore preserves a high PRR. A network with $F = 1$ accepts node 3 even if it is only heard once, causing a PRR drop due to the included links with high loss.

At 50 s, we stop the artificial drop of packets and find that node 3 can quickly join the network with $F = 3$. At 100 s, we re-introduce link failures and observe that Hydra with $F = 3$ adjusts by excluding node 3 after it has been confirmed that its link quality has decreased. Thereafter, the PRR for the remaining network again reaches 100 %. With $F = 1$, node 3 is permitted to stay to preserve liveness but incurs a reduced PRR. We conclude that the epoch length is a valuable knob to increase network stability and can enable efficient networks by enforcing reliable links.

Asymmetric links. A critical case that may endanger safety is the occurrence of asymmetric communication links, as a node might continue to consider itself part of the majority without them being aware of the node. To prevent such imbalances, the merging criteria presented in Section 5.4.2 ensure that only nodes that mutually include each other in their membership flags can prevent schedule expiration. The effect of the merging criteria can be seen in Figure 5.8 for $F = 3$ at 125 s, where the exclusion of a node from the majority forces its schedule to expire and preserves safety.

Time synchronization. In contrast to traditional Chaos which is initiated by a network coordinator, Hydra needs another means of time synchronization. We find that we can successfully use the Glossy floods during the `DD` phase to synchronize transmissions of all 23 nodes during the contention-based transmissions of the `SN` phase to an average of 0.4 µs. With a maximum offset of 1.43 µs, this is more than sufficient for the capture effect requiring 160 µs [LFZ13].

Time to completion. Lastly, we test how quickly distributed consensus is found in Figure 5.9. We run 300 rounds for different network sizes and

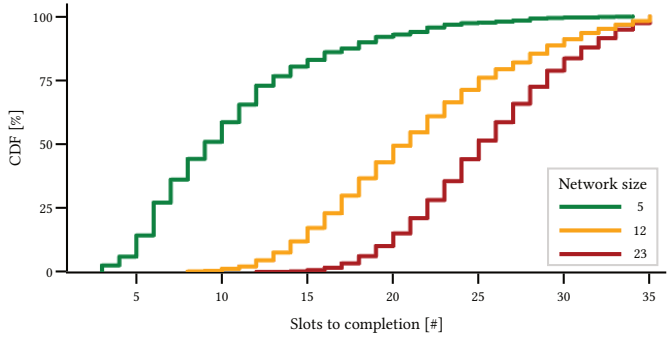

Figure 5.9: The average number of slots during the SN phase to obtain the complete set of information depends on the network size, with small networks requiring less overhead as they complete quickly.

observe in which of the 36 contention slots of the SN phase the complete set of information is obtained. A network of 5 nodes succeeds for 99 % of nodes in gathering the complete set with an average of 10 slots. For 12 nodes, 91 % of nodes are complete at an average of 21 slots, and 23 nodes complete 88 % of the time in a mean of 26 slots. As a single successful node is sufficient to determine the next schedule and distribute it to the rest of the nodes in the SD phase, using fewer contention slots could substantially reduce the scheduling overhead, particularly for smaller networks.

5.7 Discussion

5.7.1 Exploring the Trade-off Space

Trading off adaptivity and liveness. Hydra's three protocol requirements are conflicting. As a design choice, we unconditionally guarantee safety, because avoiding packet collisions is an indispensable precondition for reliable packet delivery and hence dependable communication. If we were to relax safety, then achieving adaptivity and liveness would be easily possible using best-effort strategies, e.g., by letting nodes send data purely according to current demand.

However, there is a trade-off between adaptivity and liveness. If the schedule were fully static, safety and liveness can be guaranteed as the schedule is always up-to-date. At the other extreme, adapting the network membership every round could result in a disproportionate loss of liveness

for nodes briefly losing connectivity as schedules expire to preserve safety. With the epoch length F, Hydra offers control of this trade-off. While Hydra's design uses a fixed length, dynamically reducing this parameter at runtime based on current channel state information and past performance may boost adaptivity when environmental conditions remain stable.

Trading off complexity and robustness. While centralized network coordination constitutes a straightforward solution, it lacks robustness as demonstrated by the collapse in network communication in Figure 5.5 and must therefore be decentralized to provide fault tolerance. As soon as more than a single node is involved, the protocol complexity of consensus needs to be introduced independently of the number of participating coordinators. Therefore, while a hybrid approach with only a subset of nodes coordinating concurrently may seem appealing, reducing the fraction of involved nodes does not simplify the underlying problem. Furthermore, as the employed communication primitives based on CT result in all nodes indiscriminately obtaining all information, only the negligible local overhead of the schedule computation could be avoided. As nodes that do not participate in coordination must listen during the SD phase, we expect the resulting idle listening overhead to significantly surpass any potential computational savings.

5.7.2 Limitations and Extensions

Supporting flexible network sizes. With Hydra, we choose to prioritize safety over liveness. However, Hydra's consensus mechanism can also be employed for a fault-tolerant network where liveness should be maintained at the cost of potential packet collisions. This is of particular importance if independent clusters should be supported, which Hydra deliberately prevents by enforcing that only a majority of nodes may persist. By removing the check on the number of included nodes (line 20 in Algorithm 3) and the schedule expiration (line 6 in Algorithm 4), such an extension is straightforward and would remedy a limitation on the minimum and maximum network sizes. However, an extra discovery mechanism to merge clusters, such as the one introduced in Chapter 3, is required to avoid a continuous decay into smaller sets of nodes.

Scaling to large networks. The use of all-to-all primitives such as Chaos [LFZ13] demands the allocation of fixed variables per potential node in each packet so that information can be merged. While this only amounts to 4 bits per node in our implementation for the membership flag

and the request, the energy consumption scales proportionally with N. In addition to the packet size, we also see in Figure 5.9 that the length of the SN phase needs to be increased with N as more information has to be exchanged to obtain the complete set of information. While we find that an all-to-all primitive enables a network of $N = 23$ to bootstrap $2.9\times$ faster than a traditional design based on a single contention slot (Figure 5.5) and is therefore favorable if requests are changing frequently, a large and stable network may benefit from only exchanging information on demand.

5.8 Related Work

Consensus in WSNs. LWB [FZMT12] ensures safety by limiting nodes to only transmitting when a schedule from a network coordinator has been received. It cyclically shifts through leaders upon failure, further dropping liveness in between. Virtus [FZMT13] provides atomic multicast with packet delivery ordering in addition to group membership but builds on LWB. Chaos [LFZ13] and Mixer [HMZ18] do not require explicit scheduling to exchange data among all nodes but depend on a static network coordinator to initiate the transfer and thus cannot ensure liveness. A^2 [ANDL17b] and Wireless Paxos [PANL19] are consensus protocols built on top of Chaos to take decisions on values proposed by nodes in the network, but rely on a static coordinator and hence a single point of failure. Furthermore, Hydra's specialized scope reaches asynchronous consensus without the overhead of distinct phases common in consensus protocols [Lam01, OO14]. BUTLER [MBTZ22] enables the use of existing protocols based on CT without a unique time source once coarse synchronization could be established. wChain [XLZ+21] focuses on fault-tolerant blockchain operations in wireless networks, but may not sustain leader failure. BCA [CL21] evaluates sensor data based on majority-consensus voting, but does not tolerate link failures.

Decentralized scheduling. Time Slotted Channel Hopping (TSCH) [DBA16] supports reliable multi-sink applications but requires fragile routing trees and a central network coordinator, which limits liveness and presents a single point of failure. Most TSCH schedulers, such as MASTER [HL20], are centralized and send all requests to a single entity, which then computes and distributes a schedule. Distributed schedulers, such as MSF [CVV+21] and DeTAS [AVP+15], leverage information provided by RPL [WTB+12], which requires a stable routing tree and hence increases latency upon link

failures by up to 25 min [LEG20], thereby forfeiting liveness.

By contrast, Orchestra [DANLW15] uses local schedules to perform synchronous communication through pseudo-random resource usage. However, it still builds upon RPL and relies on its tree structure for coordination and synchronization. ALICE [KKK19] and A^3 [KKK21] are autonomous scheduling schemes that adapt to arbitrary traffic patterns, but cannot avoid a single point of failure due to their reliance on RPL. TREE [LEG20] tries to learn demands without RPL, and AUTOBAHN [HL21] bridges the gap between TSCH and CT. However, both require centralized coordination and therefore cannot tolerate arbitrary node failures.

Fighting interference. Significant effort has increased network robustness under external interference [YNB+22, PL21, MZL+18, MZL+20, SBDM20, ITMP18, IIPK19, MSM19, ZGH+21]. However, these protocols remain constrained by their requirement to reach a network coordinator for either scheduling or synchronization. We consider such efforts to optimize link robustness orthogonal to Hydra, which adds an extra layer of fault tolerance and guarantees fundamental properties despite failures.

5.9 Summary

In this chapter, we introduce the first fault-tolerant WSN protocol that provides safety, adaptivity, and liveness for multi-hop networks. Unlike the protocols presented in the previous chapters, Hydra can fully distribute coordination and execute it concurrently at all nodes. We have seen that in contrast to Chapter 3, autonomous resilience is preserved even under arbitrary node failure as the exchange of data will not be interrupted until a new leader could be established. Hydra provides the unique property that it can *guarantee the safety of communication* and simultaneously preserve liveness even while experiencing disturbances.

At Hydra's core is a distributed consensus algorithm which is formally proven to result in unique decisions and does not increase overhead compared to centralized schemes. We demonstrate that combined with a versioning-based scheduling mechanism, collisions with packets from other nodes running Hydra can be eradicated by design. By inducing node and link failures, we experimentally validate that *concurrent coordination* preserves efficient data exchange while remaining adaptive to changing conditions. We show that centralized approaches such as presented in

Chapter 2 and Chapter 3 are unable to provide an equally reliable communication service, even though they build on the same robust communication primitives. However, the protocol can only guarantee its properties as it enforces a single active network and therefore deliberately restricts the ability of nodes to form independent clusters; in Chapter 4, we have shown that robust decentralization is also possible with an arbitrary number of parallel clusters if one leverages local sensor signals.

While Hydra does not focus on mitigating failures, it addresses their impact on a higher layer by providing guarantees despite them. Hydra offers a unique ability to create fault-tolerant networks where each node is equivalent in protocol logic and configuration, paving the way for a highly dependable IoT.

6

Conclusions and Outlook

In the past decades, wireless sensor networks (WSNs) have undergone a remarkable evolution, with capabilities at each system level from sensing at the node over communication within a network to the data processing in the back end increasing tremendously. With sensors becoming increasingly smaller and cheaper, we encounter platforms with a multitude of sensor modalities and hardware that may dynamically adapt to the situation at hand [ADY+19]. Local computational power has exploded, with available random access memory (RAM) progressively becoming sufficient for sophisticated embedded machine learning [GHS+22, PAL22] and dedicated accelerators being directly integrated into the sensor to automatically pre-process sampled data [STM23]. Low-power wireless networks benefit from continuous progress in more capable and efficient radio technologies such as mmWave radars [DSTX23] and backscattering [PKD18] as well as communication schemes that permit robust and quick data exchanges [ZMS20, STVP23]. Lastly, the pervasiveness of the Internet means not only that people are increasingly expected to be constantly available, but also that each of their devices should be permanently online as part of the Internet of Things (IoT) so we can stream data, monitor deployments, and control dynamic processes in real time.

In this thesis, we instead propose that many systems may benefit from additional capabilities if we increase their autonomy and allow independent low-power wireless networks. Concretely, we have investigated (*i*) the autonomy from infrastructure, (*ii*) the autonomy from pre-configurations, and (*iii*) the autonomy from manual interventions.

Firstly, the dependence on infrastructure to serve as data sinks and reference points restricts systems to covered areas and thereby strongly limits their mobility. However, many systems rely on the ability to operate in previously unknown and potentially unpredictable locations and do not necessitate constant connectivity to the Internet as their functionality is based on the exchange within the wireless network itself. While some innovative technologies with a longer range such as LoRa may significantly reduce the constraints of infrastructure-based operation, they come at the cost of increased energy consumption and require novel concepts to integrate them into a low-power network.

Secondly, many protocols require fundamental parameters such as a designated leader, the round period for synchronous communication, or the total number of nodes to be fixed ahead of a deployment. Such pre-configurations severely limit their reactivity to changes in the environmental conditions and application demands and thereby prevent the system from adapting and automatically improving its performance.

Thirdly, a lack of fault tolerance and adaptivity may necessitate manual interventions after a leader failed and cannot coordinate a network anymore or when a cluster of nodes got temporally separated from the rest of the network and lacks the orchestration to preserve communication. In order to reduce maintenance costs, scale deployments, and increase reliability, a system should be able to independently cope with node and link failures and ensure that it can continue to operate to the best of its abilities by adjusting to the new operating conditions.

6.1 Contributions

In Section 1.1, we presented our aim to boost the autonomy of low-power wireless networks by improving the two aspects of autonomous formation and autonomous resilience. *Autonomous formation* has been covered in Chapter 2, Chapter 3, and Chapter 4. We have presented approaches to use both combinations of radios and modulations to form autonomous networks by coupling networking and sensing, and further showed that a symbiosis of cluster coordination and cluster discovery supports highly dynamic networks. *Autonomous resilience* has been the focus of Chapter 3 and Chapter 5. By exchanging information on connected nodes and their demands, we have demonstrated that consensus enables the nodes to autonomously orchestrate their network and even provably guarantee safe, adaptive, and persistent communication despite node and link failures.

Infrastructure-Free Tracking through Mobile Sensor Networks

In Chapter 2, we present a novel dual-radio architecture for infrastructure-free interaction tracking with SociTrack. We achieve this unprecedented autonomy from infrastructure for UWB-based systems through a combination of complementary radio technologies, which permits us to form dynamic clusters using continuous discovery and capture interaction data in situ. The system's high mobility is supported by an independent clustering and network management scheme that adapts sensing to changes in the network topology. Reliable multi-hop networking then ensures that all pairwise interactions between connected nodes are efficiently gathered with high measurement fidelity.

The presented approach to enable infrastructure-free sensor deployments based on SociTrack has been successfully adopted in its target scenario to capture caregiver-infant interactions. Three research groups are actively participating in the development of successive generations of the platform by improving its ease of use through iterative testing in user studies. Furthermore, initial domain-specific findings have been published [SPH+22], with groups in North America and Europe interested in adopting the system for research in social sciences. Similarly, other researchers in low-power wireless networking have followed suit and also developed a wireless sensing system based on our proposed heterogeneous concept [ILM+21].

Robust Orchestration for Autonomous Networking

In Chapter 3, we demonstrate how to achieve robust network orchestration despite arbitrary node and link failures with Demos. The protocol bases its autonomy from manual interventions on a symbiosis of cluster discovery and cluster coordination, allowing it to quickly adapt to changes in the network topology without halting the exchange of data. We show that by introducing novel election quorums to independently find consensus on cluster leadership, we can further boost autonomy from pre-configurations. Demos forms autonomous clusters of connected nodes and instantly reacts after a leader fails or the network splits, thereby leveraging its dynamic coordination mechanism to ensure that the cluster information remains up-to-date. Through a cluster-wide discovery scheme that quickly merges clusters once they encounter each other, the protocol supports high node mobility and ensures maximal network coverage. With its adaptivity, Demos is the first low-power wireless protocol that provides robust network orchestration for autonomous multi-hop networks.

Exploiting Spatial and Temporal Correlation for Event-based Communication in WSNs

In Chapter 4, we propose to directly incorporate sensor signals for networking with STeC. By forming ad-hoc clusters using the co-detection of an event at multiple nodes quasi-simultaneously, the protocol exploits the spatial and temporal correlation of such sensed events to bootstrap a network. Through purely event-based communication without any fixed, periodic overhead and the local election of a leader from a dynamic set of nodes, we boost autonomy from pre-configurations. Furthermore, we find that locally filtering data and only utilizing a long-range link for events of confirmed relevance enables us to integrate an energy-intensive technology such as LoRa and still facilitate long-term, battery-powered deployments. As this combination of radio modulations permits us to employ a single data sink with a large coverage for communication with various independent clusters of sensor nodes, we thereby boost the autonomy from infrastructure. With STeC, we are the first to structure a network from the ground up by leveraging correlated sensor events. This autonomous formation represents a fundamentally new approach to alleviate the elementary conflict of synchronous systems between duty cycling and reactivity.

Concurrent Coordination for Fault-tolerant Networking

In Chapter 5, we introduce the concept of concurrent coordination to increase the fault tolerance of low-power wireless networks with Hydra. Unlike most deterministic network protocols, all nodes in Hydra are equivalent in terms of protocol logic as well as configuration. Leveraging a novel distributed consensus algorithm that yields provably unique decisions combined with decentralized synchronization, the protocol avoids any entity that maintains a unique state or serves a special role. Together with the protocol's ability to adapt to the application's traffic demand while tolerating arbitrary message loss, Hydra hence provides high autonomy from pre-configurations. By integrating a versioning-based scheduling scheme, it ensures safe and persistent communication despite frequent node and link failures. Concurrent coordination empowers the protocol to take full advantage of physical redundancy and makes it excel in its autonomy from manual interventions through formally proven guarantees. This concept makes Hydra the first fully distributed protocol that avoids centralization by design to enable fault-tolerant low-power wireless networking.

6.2 Future Directions

In particular due to the high required interdependence between protocol components, the development of an autonomous low-power wireless network protocol is demanding and must factor a diverse set of considerations into the design process. While this thesis presented progress in achieving autonomous formation and autonomous resilience, we identify additional opportunities to further build upon our work in the following paragraphs.

Autonomy from infrastructure

While dependence on infrastructure does constrain a system's deployability, such installations can also be exploited to complement and extend the system's capabilities. For example, SociTrack may benefit from static anchors at known locations to translate relative pairwise distance measurements to the absolute position of an object. For this reason, the design in Section 2.5 includes an exclusive responder that can leverage the plentiful resources of infrastructure-based nodes to boost the performance of more restricted nodes. Similarly, while STeC may directly notify an alarm system of a relevant co-detection over a long-range link, it still requires such a component to permanently listen and thereby draw on resources that can only be provided by deploying infrastructure. While connectivity to the Internet via a data sink is unnecessary for the co-detection itself, it is advantageous to keep decision-makers informed of ongoing activity and monitor a situation over time. Therefore, infrastructure can be of great benefit to an autonomous system and may be considered in the design process as an option to significantly increase its effectiveness.

Future approaches to infrastructure usage without restricting a system's mobility may include the opportunistic integration of omnipresent technology such as GNSS or cellular information to enrich the location information of interactions. Furthermore, most WSN deployments will require connectivity to a back end in practice for monitoring and maintenance purposes in production systems. Recent developments such as global LoRa connectivity using cube satellites [SEA+20] may enable a transceiver such as the one used by STeC to relay information via space and effectively alleviate coverage restrictions in remote outdoor deployments. Complementarily, growing collaborative networks [The23] may similarly reduce such coverage problems in more rural settings.

Autonomy from pre-configurations

In this thesis, we primarily focused on avoiding the hard-coded leader configurations that can be found in the vast majority of deterministic network protocols. Furthermore, the presented protocols were designed to adapt scheduled communication based on the current traffic demand in order to preserve efficient long-term performance in contrast to protocols that use fixed schedules. However, the demand for adaptivity occasionally directly conflicts with other desired properties. For example, Hydra fixes both the maximal number of nodes in the network as well as the epoch length, as discussed in Section 5.7. It does so to preserve the safety property which fundamentally relies on all nodes in the network using the same parameters. However, safety may be of secondary concern in some deployments compared to improved flexibility or energy efficiency; while we sketch how such options could be instantiated, additional research on how to systematically permit key system parameters to be adapted network-wide could significantly improve the adaptivity of protocols.

In particular, more dynamic protocols may boost energy efficiency through a single, crucial decision. As already noted more than a decade ago, the most consequential decision for a low-power system is to determine whether to stay awake or return to sleep [DDHC+10]. For example, Chapter 3 required a fixed number of elections which cannot be skipped for the most reliable relative quorum unless it can be confirmed that no node still intends to propose. Using novel techniques such as Flick [STVP23], nodes may first conduct a binary cluster-wide operation to check whether anyone is still interested in additional elections and otherwise save time and energy by skipping more involved communication.

Autonomy from manual interventions

Protocols often need to rely on assumptions or hard requirements on the deployment scenario. For example, while Hydra provides unprecedented guarantees, it still requires a majority of nodes to participate in its distributed consensus mechanism and may halt operation if more than half of the nodes are unavailable. Such a situation can be encountered following permanent hardware failures that may be correlated throughout the network [WC19] or because of complete battery depletion at nodes that experience more frequent activity.

However, a combination of mechanisms on the hardware level as well as on the protocol level could help to mitigate such disruptions. Reconfigurable hardware [ADY+19] may enable networks to flexibly adjust

capabilities at a certain node and transfer responsibilities after another one failed, and the significant advances in energy harvesting systems help to extend the lifetime of autonomous deployments. In particular, harvesting systems that can reliably provide energy using a backup system [JAD19] may be used to reduce the occurrence of such issues in the best case, but at the minimum ensure that the network can be notified of imminent node failure so that assumptions (in this case on the total number of available nodes) can be adjusted. Such mechanisms could contribute to identifying cases when assumptions are on the verge of being violated; in combination with a clear assessment of what consequences such a violation could cause and potential intermediate steps to alleviate them, the fault tolerance of systems could become more pronounced. With Demos and Hydra, we have proposed protocols that enable multiple parallel clusters using assumption-based quorums and provide hard, provable guarantees if given requirements are met. We hope that our focus on formally verifying properties under known conditions inspires other works to follow suit and provide the means for a more dependable IoT.

Bibliography

[AB70] M. Ainsworth and S. Bell. Attachment, exploration, and separation: Illustrated by the behavior of one-year-olds in a strange situation. *Child Development*, 1970. DOI: 10.7312/stei93738-006.

[ABC+20] M. Afanasov, N. A. Bhatti, D. Campagna, G. Caslini, F. M. Centonze, K. Dolui, A. Maioli, E. Barone, M. H. Alizai, J. H. Siddiqui, and L. Mottola. Battery-Less Zero-Maintenance Embedded Sensing at the Mithræum of Circus Maximus. In *18th ACM Conference on Embedded Networked Sensor Systems*, SenSys '20, page 368–381, New York, NY, USA, 2020. Association for Computing Machinery. DOI: 10.1145/3384419.3430722.

[ABK+19] C.-S. Ahn, B.-H. Bang, M.-W. Kim, S. James, A. Yarin, and S. Yoon. Theoretical, numerical, and experimental investigation of smoke dynamics in high-rise buildings. *International Journal of Heat and Mass Transfer*, 135:604–613, 2019. DOI: 10.1016/j.ijheatmasstransfer.2018.12.093.

[Abr70] N. Abramson. The ALOHA system: Another Alternative for Computer Communications. In *Fall Joint Computer Conference*, AFIPS '70, page 281–285, New York, NY, USA, 1970. Association for Computing Machinery. DOI: 10.1145/1478462.1478502.

[ADY+19] E. Aras, S. Delbruel, F. Yang, W. Joosen, and D. Hughes. A Low-Power Hardware Platform for Smart Environment as a Call for More Flexibility and Re-Usability. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '19, pages 194–205, USA, 2019. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3324320.3324344.

[AKNR13] J. Augustine, T. Kulkarni, P. Nakhe, and P. Robinson. Robust Leader Election in a Fast-Changing World. In *9th International Workshop on Foundations of Mobile Computing*, volume 132 of *FOMC '13*, pages 38–49. Open Publishing Association, October 2013. DOI: 10.4204/EPTCS.132.4.

[AKS15] J. Augustine, T. Kulkarni, and S. Sivasubramaniam. Leader Election in Sparse Dynamic Networks with Churn. In *IEEE International Parallel and Distributed Processing Symposium*, IPDPS '15, pages 347–356. IEEE, 2015. DOI: 10.1109/IPDPS.2015.80.

[ANDL17a] B. Al Nahas, S. Duquennoy, and O. Landsiedel. Network Bootstrapping and Leader Election Utilizing the Capture Effect in Low-Power Wireless Networks. In *15th ACM Conference on Embedded Network Sensor Systems*, SenSys '17, New York, NY, USA, 2017. Association for Computing Machinery. DOI: 10.1145/3131672.3137002.

[ANDL17b] B. Al Nahas, S. Duquennoy, and O. Landsiedel. Network-Wide Consensus Utilizing the Capture Effect in Low-Power Wireless Networks. In *15th ACM Conference on Embedded Network Sensor Systems*, SenSys '17, New York, NY, USA, 2017. Association for Computing Machinery. DOI: 10.1145/3131672.3131685.

[ANDL19] B. Al Nahas, S. Duquennoy, and O. Landsiedel. Concurrent Transmissions for Multi-Hop Bluetooth 5. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '19, pages 130–141, 2019. URL: https://dl.acm.org/doi/abs/10.5555/3324320.3324336.

[API+11] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland. Social fMRI: Investigating and shaping social mechanisms in the real world. *Pervasive and Mobile Computing*, 7(6):643–659, 2011. DOI: 10.1016/j.pmcj.2011.09.004.

[App20] Apple. Privacy-Preserving Contact Tracing. apple.com/covid19/contacttracing, August 2020.

[ATN14] M. M. Afsar and M. Tayarani-N. Clustering in sensor networks: A literature survey. *Journal of Network and Computer Applications*, 46:198–226, 2014. DOI: 10.1016/j.jnca.2014.09.005.

[Aus20] Australian Health Protection Principal Committee. Coronavirus measures, March 2020.

[AVP+15] N. Accettura, E. Vogli, M. R. Palattella, L. A. Grieco, G. Boggia, and M. Dohler. Decentralized Traffic Aware Scheduling in 6TiSCH Networks: Design and Experimental Evaluation. *IEEE Internet of Things Journal*, 2(6):455–470, 2015. DOI: 10.1109/JIOT.2015.2476915.

[AZU18] F. Ahmed, N. Zviedrite, and A. Uzicanin. Effectiveness of workplace social distancing measures in reducing influenza transmission: a systematic review. *BMC public health*, 18(1):518, 2018. DOI: 10.1186/s12889-018-5446-1.

[BCKP20] A. Biri, B. Campbell, B. Kempke, and P. Pannuto. SociTrack: Source code. github.com/lab11/socitrack, 2020.

[BCN+21] A. Bonde, J. Codling, K. Naruethep, Y. Dong, W. Siripaktanakon, S. Ariyadech, A. Sangpetch, O. Sangpetch, S. Pan, H. Y. Noh, and P. Zhang. PigNet: Failure-Tolerant Pig Activity Monitoring System Using Structural Vibration. In *20th International Conference on Information Processing in Sensor Networks*, IPSN '21, pages 328–340, New York, NY, USA, 2021. Association for Computing Machinery. DOI: 10.1145/3412382.3458902.

[BDFG+21] A. Biri, R. Da Forno, T. Gsell, T. Gatschet, J. Beutel, and L. Thiele. STeC: Exploiting Spatial and Temporal Correlation for Event-Based Communication in WSNs. In *19th ACM Conference on Embedded Networked Sensor Systems*, SenSys '21, pages 274–287, New York, NY, USA, 2021. Association for Computing Machinery. DOI: 10.1145/3485730.3485951.

[BDFGG21] A. Biri, R. Da Forno, T. Gatschet, and T. Gsell. STeC: Source code. gitlab.ethz.ch/tec/public/stec, 2021.

[BDFK23a] A. Biri, R. Da Forno, and T. Kuonen. Demos: Source code. gitlab.ethz.ch/tec/public/demos, 2023.

[BDFK+23b] A. Biri, R. Da Forno, T. Kuonen, F. Mager, M. Zimmerling, and L. Thiele. Hydra: Companion document and source code. gitlab.ethz.ch/tec/public/hydra, 2023.

[BDFK+23c] A. Biri, R. Da Forno, T. Kuonen, F. Mager, M. Zimmerling, and L. Thiele. Hydra: Concurrent Coordination for Fault-tolerant Networking. In *22nd International Conference on Information Processing in Sensor Networks*, IPSN '23, New York, NY, USA, 2023. Association for Computing Machinery. DOI: 10.1145/3583120.3587047.

[Bea04] C. Beaulieu. Intercultural Study of Personal Space: A Case Study. *Journal of applied social psychology*, 34(4):794–805, 2004. DOI: 10.1111/j.1559-1816.2004.tb02571.x.

[BEGNMC19] V. Barral, C. Escudero, J. García-Naya, and R. Maneiro-Catoira. NLOS identification and mitigation using low-cost UWB devices. *Sensors*, 19(16):3464, 2019. DOI: 10.3390/s19163464.

[BGS+22] 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 *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 180–185, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/abs/10.5555/3578948.3578965.

[Bir18] A. Biri. TotTernary: A wearable platform for social interaction tracking. Master's thesis, ETH Zürich, 2018.

[BISV08] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli. The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments. In *6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 43–56, New York, NY, USA, 2008. Association for Computing Machinery. DOI: 10.1145/1460412.1460418.

[BJT+20] A. Biri, N. Jackson, L. Thiele, P. Pannuto, and P. Dutta. SociTrack: Infrastructure-Free Interaction Tracking through Mobile Sensor Networks. In *26th Annual International Conference on Mobile Computing and Networking*, MobiCom '20, New York, NY, USA, 2020. Association for Computing Machinery. DOI: 10.1145/3372224.3419190.

[BLG15] A. Benchi, P. Launay, and F. Guidec. Solving Consensus in Opportunistic Networks. In *16th International Conference on Distributed Computing and Networking*, ICDCN '15, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2684464.2684479.

[BM13] N. Bruno and M. Muzzolini. Proxemics Revisited: Similar Effects of Arms Length on Men's and Women's Personal Distances. *Universal Journal of Psychology*, 1(2):46–52, 2013. DOI: 10.13189/ujp.2013.010204.

[BMW+16] S. Bennetts, F. Mensah, E. Westrupp, N. Hackworth, and S. Reilly. The Agreement between Parent-Reported and Directly Measured Child Language and Parenting Behaviors. *Frontiers in Psychology*, 7:1710, 2016. DOI: 10.3389/fpsyg.2016.01710.

[BPD19] A. Biri, P. Pannuto, and P. Dutta. TotTernary - a Wearable Platform for Social Interaction Tracking: Demo Abstract. In *18th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '19, page 346–347, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302506.3312486.

[BPS16] S. Basagni, C. Petrioli, and D. Spenza. CTP-WUR: The collection tree protocol in wake-up radio WSNs for critical applications. In *International Conference on Computing, Networking and Communications*, ICCNC '16, pages 1–6. IEEE, 2016. DOI: 10.1109/ICCNC.2016.7440687.

[Bre00] E. Brewer. Towards robust distributed systems. In *19th ACM Symposium on Principles of Distributed Computing*, PODC '00, pages 343477–343502, New York, NY, USA, July 2000. Association for Computing Machinery. URL: http://people.eecs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf.

[BSLP13] W. Bischoff, K. Swett, I. Leng, and T. Peters. Exposure to influenza virus aerosols during routine patient care. *The Journal of Infectious Diseases*, 207(7):1037–1046, 2013. DOI: 10.1093/infdis/jis773.

[BSS+22] H. Brunner, M. Stocker, M. Schuh, M. Schuß, C. A. Boano, and K. Römer. Understanding and Mitigating the Impact of Wi-Fi 6E Interference on Ultra-Wideband Communications and Ranging. In *21th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '22, pages 92–104, New York, NY, USA, 2022. Association for Computing Machinery. DOI: 10.1109/IPSN54338.2022.00015.

[BTF+19] J. Beutel, R. Trüb, R. D. Forno, M. Wegmann, T. Gsell, R. Jacob, M. Keller, F. Sutton, and L. Thiele. The Dual Processor Platform architecture: Demo abstract. In *18th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '19, pages 335–336, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302506.3312481.

[BvRW07] N. Burri, P. von Rickenbach, and R. Wattenhofer. Dozer: Ultra-Low Power Data Gathering in Sensor Networks. In *6th IEEE International Conference on Information Processing in Sensor Networks*, IPSN '07, page 450–459, New York, NY, USA, 2007. Association for Computing Machinery. DOI: 10.1145/1236360.1236417.

[BZT23] A. Biri, M. Zimmerling, and L. Thiele. Demos: Robust Orchestration for Autonomous Networking. In *Under submission for International Conference on Embedded Wireless Systems and Networks*, EWSN '23, 2023.

[CAD+20] D. Chu, E. Akl, S. Duda, K. Solo, S. Yaacoub, H. Schünemann, et al. Physical distancing, face masks, and eye protection to prevent person-to-person transmission of SARS-CoV-2 and COVID-19: A systematic review and meta-analysis. *The Lancet*, 2020. DOI: 10.1016/S0140-6736(20)31142-9.

[CB16] L. Chen and K. Bian. Neighbor discovery in mobile sensing applications: A comprehensive survey. *Ad Hoc Networks*, 48:38–52, 2016. DOI: 10.1016/j.adhoc.2016.05.005.

[CCB+13] L. Chen, S. Cool, H. Ba, W. Heinzelman, I. Demirkol, U. Muncuk, K. Chowdhury, and S. Basagni. Range extension of passive wake-up radio systems through energy harvesting. In *IEEE International Conference on Communications*, ICC '13, pages 1549–1554. IEEE, 2013. DOI: 10.1109/ICC.2013.6654734.

[CD19] A. Czumaj and P. Davies. Communicating with beeps. *Journal of Parallel and Distributed Computing*, 130:98–109, 2019. DOI: 10.1016/j.jpdc.2019.03.020.

[Cen20] Centers for Disease Control USA. Interim US Guidance for Risk Assessment and Public Health Management of Persons with Potential Coronavirus Disease 2019 (COVID-19) Exposures, March 2020.

[CGR02] M. Chrobak, L. Gasieniec, and W. Rytter. Fast broadcasting and gossiping in radio networks. *Journal of Algorithms*, 43(2):177–189, 2002. DOI: 10.1016/S0196-6774(02)00004-4.

[CGR+09] D. Christakis, J. Gilkerson, J. Richards, F. Zimmerman, M. Garrison, D. Xu, S. Gray, and U. Yapanel. Audible television and decreased adult words, infant vocalizations, and conversational turns: A population-based study. *Archives of Pediatrics & Adolescent Medicine*, 163(6):554–558, 2009. DOI: 10.1001/archpediatrics.2009.61.

[CK11] R. Cohen and B. Kapchits. Continuous Neighbor Discovery in Asynchronous Sensor Networks. *IEEE/ACM Transactions on Networking*, 19(1):69–79, 2011. DOI: 10.1109/TNET.2010.2053943.

[CKP12] B. Chlebus, D. Kowalski, and A. Pelc. Electing a Leader in Multihop Radio Networks. In *International Conference On Principles Of Distributed Systems*, OPODIS '12, pages 106–120, Berlin, Heidelberg, 2012. Springer. DOI: 10.1007/978-3-642-35476-2_8.

[CL21] J. Chang and F. Liu. A Byzantine Sensing Network Based on Majority-Consensus Data Aggregation Mechanism. *Sensors*, 21(1), 2021. DOI: 10.3390/s21010248.

[CMY+02] A. Chen, R. Muntz, S. Yuen, I. Locher, S. Park, and M. Srivastava. A support infrastructure for the smart kindergarten. *IEEE Pervasive Computing*, 1(2):49–57, 2002. DOI: 10.1109/MPRV.2002.1012337.

[CRW11] H. C. Chung, P. Robinson, and J. L. Welch. Optimal Regional Consecutive Leader Election in Mobile Ad-Hoc Networks. In *7th International Workshop on Foundations of Mobile Computing*, FOMC '11, pages 52–61, New York, NY, USA, 2011. Association for Computing Machinery. DOI: 10.1145/1998476.1998485.

[CSG+16] L. Chen, Y. Shu, Y. Gu, S. Guo, T. He, F. Zhang, and J. Chen. Group-Based Neighbor Discovery in Low-Duty-Cycle Mobile Sensor Networks. *IEEE Transactions on Mobile Computing*, 15(8):1996–2009, 2016. DOI: 10.1109/TMC.2015.2476471.

[CSM13] A. Chughtai, H. Seale, and C. MacIntyre. Availability, consistency and evidence-base of policies and guidelines on the use of mask and respirator to protect hospital health care workers: A global analysis. *BMC research notes*, 6(1):216, 2013. DOI: 10.1186/1756-0500-6-216.

[CVdBB+10] C. Cattuto, W. Van den Broeck, A. Barrat, V. Colizza, J.-F. Pinton, and A. Vespignani. Dynamics of person-to-person interactions from

distributed RFID sensor networks. *PLOS ONE*, 5(7), 2010. DOI: 10.1371/journal.pone.0011596.

[CVV+21] T. Chang, M. Vučinić, X. Vilajosana, S. Duquennoy, and D. R. Dujovne. 6TiSCH Minimal Scheduling Function (MSF). URL: dl.acm.org/doi/10.17487/RFC9033, May 2021.

[CW18] H. Cai and T. Wolf. Self-Adapting Quorum-Based Neighbor Discovery in Wireless Sensor Networks. In *IEEE Conference on Computer Communications*, INFOCOM '18, pages 324–332. IEEE, 2018. DOI: 10.1109/INFOCOM.2018.8486268.

[CXC+19] L. Chen, J. Xiong, X. Chen, S. I. Lee, K. Chen, D. Han, D. Fang, Z. Tang, and Z. Wang. WideSee: Towards Wide-Area Contactless Wireless Sensing. In *17th ACM Conference on Embedded Networked Sensor Systems*, SenSys '19, pages 258–270, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3356250.3360031.

[CZ10] H. Cheng and W. Zhuang. Bluetooth-enabled in-home patient monitoring system: Early detection of Alzheimer's disease. *IEEE Wireless Communications*, 17(1):74–79, 2010. DOI: 10.1109/MWC.2010.5416353.

[DANLW15] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne. Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH. In *13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, pages 337–350, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2809695.2809714.

[dB19] K. de Barbaro. Automated sensing of daily activity: A new lens into development. *Developmental psychobiology*, 61(3):444–464, 2019. DOI: 10.1002/dev.21831.

[DBA16] D. De Guglielmo, S. Brienza, and G. Anastasi. IEEE 802.15.4e: A survey. *Computer Communications*, 88:1–24, 2016. DOI: 10.1016/j.comcom.2016.05.004.

[DC08] P. Dutta and D. Culler. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 71–84, New York, NY, USA, 2008. Association for Computing Machinery. DOI: 10.1145/1460412.1460420.

[DCR+22] M. Daepp, A. Cabral, V. Ranganathan, V. Iyer, S. Counts, P. Johns, A. Roseway, C. Catlett, G. Jancke, D. Gehring, C. Needham, C. von Veh, T. Tran, L. Story, G. D'Amone, and B. Nguyen. Eclipse: An End-to-End Platform for Low-Cost, Hyperlocal Environmental Sensing in Cities. In *21th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '22, pages 28–40,

New York, NY, USA, 2022. Association for Computing Machinery. DOI: 10.1109/IPSN54338.2022.00010.

[DDHC+10] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless. In *8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, page 1–14, New York, NY, USA, 2010. Association for Computing Machinery. DOI: 10.1145/1869983.1869985.

[Dec18] DecaWave Limited. DW1000 Radio IC. decawave.com/product/dw1000-radio-ic, 2018.

[DFPA+17] C. Di Franco, A. Prorok, N. Atanasov, B. Kempke, P. Dutta, V. Kumar, and G. Pappas. Calibration-free network localization using non-line-of-sight ultra-wideband measurements. In *16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '17, pages 235–246, New York, NY, USA, 2017. Association for Computing Machinery. DOI: 10.1145/3055031.3055091.

[DGA+05] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *4th International Symposium on Information Processing in Sensor Networks*, IPSN '05, pages 497–502. IEEE, 2005. DOI: 10.1109/IPSN.2005.1440983.

[DLPP20] M. Dietze, J. Losee, L. Polvi, and D. Palm. A seismic monitoring approach to detect and quantify river sediment mobilization by steelhead redd-building activity. *Earth Surface Processes and Landforms*, 45(12):2840–2849, 2020. DOI: 10.1002/esp.4933.

[DSTX23] Y. Dai, X. Shuai, R. Tan, and G. Xing. Interpersonal Distance Tracking with MmWave Radar and IMUs. In *22nd International Conference on Information Processing in Sensor Networks*, IPSN '23, page 123–135, New York, NY, USA, 2023. Association for Computing Machinery. DOI: 10.1145/3583120.3586958.

[DWVT14] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert. 6TiSCH: deterministic IP-enabled industrial internet (of things). *IEEE Communications Magazine*, 52(12):36–41, 2014. DOI: 10.1109/MCOM.2014.6979984.

[EFD+18] A. Elsts, X. Fafoutis, S. Duquennoy, G. Oikonomou, R. Piechocki, and I. Craddock. Temperature-Resilient Time Synchronization for the Internet of Things. *IEEE Transactions on Industrial Informatics*, 14(5):2241–2250, 2018. DOI: 10.1109/TII.2017.2778746.

[ES07] D. Ellis and J. Singer. Acoustic Waves in Porous Rocks and Boreholes. In *Well Logging for Earth Scientists*, pages 499–530. Springer, 2007. DOI: 10.1007/978-1-4020-4602-5_18.

[Eur20] European Centre for Disease Prevention and Control. Infection prevention and control for COVID-19 in healthcare settings, March 2020.

[EW07] G. Evans and R. Wener. Crowding and Personal Space Invasion on the Train: Please Don't Make Me Sit in the Middle. *Journal of Environmental Psychology*, 27(1):90–94, 2007. DOI: 10.1016/j.jenvp.2006.10.002.

[fDPC19] E. C. for Disease Prevention and Control. The use of evidence in decision making during public health emergencies, July 2019.

[Fed20a] Federal Ministry of the Interior Building and Community. Corona FAQ, May 2020.

[Fed20b] Federal Office of Public Health. New corona virus: Precautionary measures, August 2020.

[FFBV19] J. Faillettaz, M. Funk, J. Beutel, and A. Vieli. Towards early warning of gravitational slope failure with co-detection of microseismic activity: the case of an active rock glacier. *Natural Hazards and Earth System Sciences*, 19(7):1399–1413, 2019. DOI: 10.5194/nhess-19-1399-2019.

[FLP85] M. Fischer, N. Lynch, and M. Paterson. Impossibility of Distributed Consensus with One Faulty Process. *Journal of the ACM*, 32(2):374–382, April 1985. DOI: 10.1145/3149.214121.

[FZMT12] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Low-Power Wireless Bus. In *10th ACM Conference on Embedded Network Sensor Systems*, SenSys '12, pages 1–14, New York, NY, USA, 2012. Association for Computing Machinery. DOI: 10.1145/2426656.2426658.

[FZMT13] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Virtual Synchrony Guarantees for Cyber-physical Systems. In *IEEE 32nd International Symposium on Reliable Distributed Systems*, SRDS '13, pages 20–30. IEEE, 2013. DOI: 10.1109/SRDS.2013.11.

[FZTS11] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with Glossy. In *10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '11, pages 73–84, New York, NY, USA, 2011. Association for Computing Machinery. URL: https://ieeexplore.ieee.org/abstract/document/5779066.

[Gar00] F. Gardner. Methodological issues in the direct observation of parent–child interaction: Do observational findings reflect the natural behavior of participants? *Clinical child and family psychology review*, 3(3):185–198, 2000. DOI: 10.1023/A:1009503409699.

[Gat20] T. Gatschet. Event-triggered multi-hop communication for wireless sensor networks. Master's thesis, ETH Zürich, 2020. DOI: 10.3929/ethz-b-000513703.

[GBG+21] A. Guillemot, L. Baillet, S. Garambois, X. Bodin, A. Helmstetter, R. Mayoraz, and E. Larose. Modal sensitivity of rock glaciers to elastic changes from spectral seismic noise monitoring and modeling. *The Cryosphere*, 15(2):501–529, 2021. DOI: 10.5194/tc-15-501-2021.

[GET22] A. Gräfe, J. Eickhoff, and S. Trimpe. Event-triggered and distributed model predictive control for guaranteed collision avoidance in UAV swarms. In *9th IFAC Conference on Networked Systems*, NECSYS '22, pages 79–84. IFAC-PapersOnLine, 2022. DOI: 10.1016/j.ifacol.2022.07.239.

[GHL+20] A. Guillemot, A. Helmstetter, E. Larose, L. Baillet, S. Garambois, R. Mayoraz, and R. Delaloye. Seismic monitoring in the Gugla rock glacier (Switzerland): ambient noise correlation, microseismicity and modelling. *Geophysical Journal International*, 221(3):1719–1735, March 2020. DOI: 10.1093/gji/ggaa097.

[GHS+22] T. Goyal, P. Huang, F. Sutton, B. Maag, and P. Sommer. SMiLe: Automated End-to-End Sensing and Machine Learning Co-Design. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 12–23, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.5555/3578948.3578950.

[GLG+21] Z. Gao, A. Li, Y. Gao, Y. Wang, and Y. Chen. Hermes: Decentralized Dynamic Spectrum Access System for Massive Devices Deployment in 5G. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '21, pages 13–24, USA, 2021. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3451271.3451273.

[Goo20] Google. Exposure Notification: Helping fight COVID-19. google.com/covid19/exposurenotifications, August 2020.

[GRS+17] K. Gururaj, A. Rajendra, Y. Song, C. Law, and G. Cai. Real-time identification of NLOS range measurements for enhanced UWB localization. In *International Conference on Indoor Positioning and Indoor Navigation*, IPIN '17, pages 1–7. IEEE, 2017. DOI: 10.1109/IPIN.2017.8115877.

[GSR+19] B. Großwindhager, M. Stocker, M. Rath, C. A. Boano, and K. Römer. SnapLoc: an ultra-fast UWB-based indoor localization system for an unlimited number of tags. In *18th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '19, pages 61–72, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302506.3310389.

[GWS+19] Z. Gu, Y. Wang, W. Shi, Z. Tian, K. Ren, and F. Lau. A Practical Neighbor Discovery Framework for Wireless Sensor Networks. *Sensors*, 19(8), 2019. DOI: 10.3390/s19081887.

[GYR+20] A. Gadre, F. Yi, A. Rowe, B. Iannucci, and S. Kumar. Quick (and Dirty) Aggregate Queries on Low-Power WANs. In *19th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '20, pages 277–288, New York, NY, USA, 2020. Association for Computing Machinery. DOI: 10.1109/IPSN48710.2020.00031.

[HKPD14] W. Huang, Y.-S. Kuo, P. Pannuto, and P. Dutta. Opo: A wearable sensor for capturing high-fidelity face-to-face interactions. In *12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 61–75, New York, NY, USA, 2014. Association for Computing Machinery. DOI: 10.1145/2668332.2668338.

[HL20] O. Harms and O. Landsiedel. MASTER: Long-Term Stable Routing and Scheduling in Low-Power Wireless Networks. In *16th International Conference on Distributed Computing in Sensor Systems*, DCOSS '20, pages 86–94. IEEE, 2020. DOI: 10.1109/DCOSS49796.2020.00025.

[HL21] O. Harms and O. Landsiedel. Opportunistic Routing and Synchronous Transmissions Meet TSCH. In *46th Conference on Local Computer Networks*, LCN '22, pages 107–114. IEEE, 2021. DOI: 10.1109/LCN52139.2021.9524952.

[HMZ18] C. Herrmann, F. Mager, and M. Zimmerling. Mixer: Efficient Many-to-All Broadcast in Dynamic Wireless Mesh Networks. In *16th ACM Conference on Embedded Networked Sensor Systems*, SenSys '18, pages 145–158, New York, NY, USA, 2018. Association for Computing Machinery. DOI: 10.1145/3274783.3274849.

[HSR16] D. Harrison, W. Seah, and R. Rayudu. Rare Event Detection and Propagation in Wireless Sensor Networks. *ACM Computing Surveys*, 48(4), March 2016. DOI: 10.1145/2885508.

[HYM16] S. Hayat, E. Yanmaz, and R. Muzaffar. Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint. *IEEE Communications Surveys and Tutorials*, 18(4):2624–2661, 2016. DOI: 10.1109/COMST.2016.2560343.

[IBS+10] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange. SensorScope: Application-Specific Sensor Network for Environmental Monitoring. *ACM Transactions on Sensor Networks*, 6(2), March 2010. DOI: 10.1145/1689239.1689247.

[iFi20] iFixit. iPhone 11 Pro Max Teardown (Step 19). ifixit.com/Teardown/iPhone+11+Pro+Max+Teardown, Accessed: 22-03-2020.

[IIKN19] B. Islam, T. Islam, J. Kaur, and S. Nirjon. LoRaIn: Making a Case for LoRa in Indoor Localization. In *IEEE International Conference on Pervasive Computing and Communications Workshops*, PerCom '19, pages 423–426, March 2019. DOI: 10.1109/PERCOMW.2019.8730767.

[IIPK19] T. Istomin, O. Iova, G. P. Picco, and C. Kiraly. Route or Flood? Reliable and Efficient Support for Downward Traffic in RPL. *ACM Transactions on Sensor Networks*, 16(1), October 2019. DOI: 10.1145/3355997.

[ILM+21] T. Istomin, E. Leoni, D. Molteni, A. L. Murphy, G. P. Picco, and M. Griva. Janus: Dual-Radio Accurate and Energy-Efficient Proximity Detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(4), December 2021. DOI: 10.1145/3494978.

[IMPR16] T. Istomin, A. L. Murphy, G. P. Picco, and U. Raza. Data Prediction + Synchronous Transmissions = Ultra-Low Power Wireless Sensor Networks. In *14th ACM Conference on Embedded Network Sensor Systems*, SenSys '16, page 83–95, New York, NY, USA, 2016. Association for Computing Machinery. DOI: 10.1145/2994551.2994558.

[ION21] ION Geophysical Corporation. SM-6 - Omni-directioal geophones. iongeo.com/technologies/hardware/seismic-equipment/precision-geophones/, Accessed: 17-05-2021.

[IoT23] IoT Analytics. State of IoT 2023. iot-analytics.com/number-connected-iot-devices/, 2023. Accessed: 24-05-2023.

[ITMP18] T. Istomin, M. Trobinger, A. L. Murphy, and G. P. Picco. Interference-Resilient Ultra-Low Power Aperiodic Data Collection. In *17th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '18, pages 84–95. IEEE, 2018. DOI: 10.1109/IPSN.2018.00015.

[JAD19] N. Jackson, J. Adkins, and P. Dutta. Capacity over Capacitance for Reliable Energy Harvesting Sensors. In *18th International Conference on Information Processing in Sensor Networks*, IPSN '19, page 193–204, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302506.3310400.

[JFT+19] R. Jacob, R. D. Forno, R. Trüb, A. Biri, and L. Thiele. Dataset: Wireless Link Quality Estimation on FlockLab - and Beyond. In *2nd Workshop on Data Acquisition To Analysis*, DATA'19, page 57–60, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3359427.3361907.

[JKD+07] S. Jevtic, M. Kotowsky, R. Dick, P. Dinda, and C. Dowding. Lucid Dreaming: Reliable Analog Event Detection for Energy-Constrained Applications. In *6th IEEE International Conference on Information*

Processing in Sensor Networks, IPSN '07, page 350–359, New York, NY, USA, 2007. Association for Computing Machinery. DOI: 10.1145/1236360.1236405.

[JLMP17] C. Julien, C. Liu, A. L. Murphy, and G. P. Picco. BLEnd: Practical Continuous Neighbor Discovery for Bluetooth Low Energy. In *16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '17, pages 105–116. IEEE, 2017. URL: https://ieeexplore.ieee.org/abstract/document/7944783.

[JP21] D. Jagtap and P. Pannuto. Repurposing Cathodic Protection Systems as Reliable, in-Situ, Ambient Batteries for Sensor Networks. In *20th International Conference on Information Processing in Sensor Networks*, IPSN '21, pages 357–368, New York, NY, USA, 2021. Association for Computing Machinery. DOI: 10.1145/3412382.3458277.

[KDL+06] B. Kusy, P. Dutta, P. Levis, M. Maroti, A. Ledeczi, and D. Culler. Elapsed time on arrival: a simple and versatile primitive for canonical time synchronisation services. *International Journal of Ad Hoc and Ubiquitous Computing*, 1(4):239–251, 2006. DOI: 10.1504/IJAHUC.2006.010505.

[KGKR14] S. Kumar, S. Gil, D. Katabi, and D. Rus. Accurate Indoor Localization with Zero Start-up Cost. In *20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, page 483–494, New York, NY, USA, 2014. Association for Computing Machinery. DOI: 10.1145/2639108.2639142.

[KHY98] M. B. Kim, Y. S. Han, and M. O. Yoon. Laser-assisted visualization and measurement of corridor smoke spread. *Fire Safety Journal*, 31(3):239–251, 1998. DOI: 10.1016/S0379-7112(98)00009-5.

[KKK19] S. Kim, H.-S. Kim, and C. Kim. ALICE: Autonomous Link-Based Cell Scheduling for TSCH. In *18th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '19, pages 121–132, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302506.3310394.

[KKK21] S. Kim, H.-S. Kim, and C.-k. Kim. A3: Adaptive Autonomous Allocation of TSCH Slots. In *20th International Conference on Information Processing in Sensor Networks*, IPSN '21, pages 299–314, New York, NY, USA, 2021. Association for Computing Machinery. DOI: 10.1145/3412382.3458273.

[KLS+10] M. Kazandjieva, J. W. Lee, M. Salathé, M. Feldman, J. Jones, and P. Levis. Experiences in measuring a human contact network for epidemiology research. In *6th Workshop on Hot Topics in Embedded Networked Sensors*, HotEMNETS '10, pages 1–5, New York, NY, USA, 2010. Association for Computing Machinery. DOI: 10.1145/1978642.1978651.

[KMO+12] T. Kim, E. McFee, D. Olguin, B. Waber, and A. Pentland. Sociometric badges: Using sensor technology to capture new forms of collaboration. *Journal of Organizational Behavior*, 33:412–427, April 2012. DOI: 10.1002/job.1776.

[KPCD16] B. Kempke, P. Pannuto, B. Campbell, and P. Dutta. SurePoint: Exploiting ultra wideband flooding and diversity to provide robust, scalable, high-fidelity indoor localization. In *14th ACM Conference on Embedded Network Sensor Systems*, SenSys '16, pages 137–149, New York, NY, USA, 2016. Association for Computing Machinery. DOI: 10.1145/2994551.2994570.

[KPD15] B. Kempke, P. Pannuto, and P. Dutta. PolyPoint: Guiding indoor quadrotors with ultra-wideband localization. In *2nd International Workshop on Hot Topics in Wireless*, HotWireless '15, pages 16–20, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2799650.2799651.

[KPSV17] J. A. Khan, R. Pujol, R. Stanica, and F. Valois. On the Energy Efficiency and Performance of Neighbor Discovery Schemes for Low Duty Cycle IoT Devices. In *14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, PE-WASUN '17, pages 63–70, New York, NY, USA, 2017. Association for Computing Machinery. DOI: 10.1145/3134829.3134835.

[Kuo21] T. Kuonen. Primus inter pares no more - Increasing the fault tolerance of wireless sensor networks. Master's thesis, ETH Zürich, 2021. DOI: 10.3929/ethz-b-000599913.

[Lam01] L. Lamport. Paxos Made Simple. *ACM SIGACT News*, 32(4):51–58, December 2001. URL: https://www.microsoft.com/en-us/research/publication/paxos-made-simple/.

[Lan09] S. Lanzisera. *RF Ranging for Location Awareness*. PhD thesis, University of California at Berkeley, May 2009.

[LAY19] Q. Lin, Z. An, and L. Yang. Rebooting Ultrasonic Positioning Systems for Ultrasound-Incapable Smart Devices. In *25th Annual International Conference on Mobile Computing and Networking*, MobiCom '19, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3300061.3300139.

[LBL+21] M. A. Lopez, M. Baddeley, W. T. Lunardi, A. Pandey, and J.-P. Giacalone. Towards Secure Wireless Mesh Networks for UAV Swarm Connectivity: Current Threats, Research, and Opportunities. In *17th International Conference on Distributed Computing in Sensor Systems*, DCOSS '21, pages 319–326. IEEE, 2021. DOI: 10.1109/DCOSS52077.2021.00059.

[LDFST17] R. Lim, R. Da Forno, F. Sutton, and L. Thiele. Competition: Robust Flooding using Back-to-Back Synchronous Transmissions with Channel-Hopping. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '17, pages 270–271, USA, 2017. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3108009.3108076.

[LEG20] T. V. D. Lee, G. Exarchakos, and S. H. D. Groot. Distributed Reliable and Energy-Efficient Scheduling for LR-WPANs. *ACM Transactions on Sensor Networks*, 16(4), August 2020. DOI: 10.1145/3399805.

[LF76] K. Leentvaar and J. Flint. The capture effect in FM receivers. *IEEE Transactions on Communications*, 24(5):531–539, May 1976. DOI: 10.1109/TCOM.1976.1093327.

[LFZ13] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and Efficient All-to-All Data Sharing and in-Network Processing at Scale. In *11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, New York, NY, USA, 2013. Association for Computing Machinery. DOI: 10.1145/2517351.2517358.

[LHV+09] A. Ledeczi, T. Hay, P. Volgyesi, D. R. Hay, A. Nadas, and S. Jayaraman. Wireless Acoustic Emission Sensor Network for Structural Monitoring. *IEEE Sensors Journal*, 9(11):1370–1377, 2009. DOI: 10.1109/JSEN.2009.2019315.

[LRS+15] P. Lazik, N. Rajagopal, O. Shih, B. Sinopoli, and A. Rowe. ALPS: A Bluetooth and ultrasound platform for mapping and localization. In *13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, pages 73–84, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2809695.2809727.

[LSD+20] D. Lakew, U. Sa'ad, N.-N. Dao, W. Na, and S. Cho. Routing in Flying Ad Hoc Networks: A Comprehensive Survey. *IEEE Communications Surveys and Tutorials*, 22(2):1071–1120, 2020. DOI: 10.1109/COMST.2020.2982452.

[LWS06] Y. Li, Z. Wang, and Y. Song. Wireless Sensor Network Design for Wildfire Monitoring. In *6th World Congress on Intelligent Control and Automation*, volume 1 of *WCICA '06*, pages 109–113. IEEE, 2006. DOI: 10.1109/WCICA.2006.1712372.

[MB01] M. McGlynn and S. Borbash. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '01, pages 137–145, New York, NY, USA, 2001. Association for Computing Machinery. DOI: 10.1145/501431.501435.

[MBH+22] F. Mager, D. Baumann, C. Herrmann, S. Trimpe, and M. Zimmerling. Scaling beyond Bandwidth Limitations: Wireless Control with Stability Guarantees under Overload. *ACM Transactions on Cyber-Physical Systems*, 6(3), September 2022. DOI: 10.1145/3502299.

[MBJ+19] F. Mager, D. Baumann, R. Jacob, L. Thiele, S. Trimpe, and M. Zimmerling. Feedback Control Goes Wireless: Guaranteed Stability over Low-Power Multi-Hop Networks. In *10th ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '19, pages 97–108, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3302509.3311046.

[MBTZ22] F. Mager, A. Biri, L. Thiele, and M. Zimmerling. BUTLER: Increasing the Availability of Low-Power Wireless Communication Protocols. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 108–119, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/abs/10.5555/3578948.3578958.

[MBYP09] C. McCall, J. Blascovich, A. Young, and S. Persky. Proxemic Behaviors as Predictors of Aggression Towards Black (but not White) Males in an Immersive Virtual Environment. *Social Influence*, 4(2):138–154, 2009. DOI: 10.1080/15534510802517418.

[MC20] L. Morawska and J. Cao. Airborne transmission of SARS-CoV-2: The world should face the reality. *Environment International*, page 105730, 2020. DOI: 10.1016/j.envint.2020.105730.

[MEB09] Z. Merhi, M. Elgamel, and M. Bayoumi. EB-MAC: An event based medium access control for wireless sensor networks. In *IEEE International Conference on Pervasive Computing and Communications*, PERCOM '09, pages 1–6. IEEE, 2009. DOI: 10.1109/PERCOM.2009.4912863.

[MELT08] R. Musaloiu-E., C.-J. Liang, and A. Terzis. Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks. In *7th IEEE International Conference on Information Processing in Sensor Networks*, IPSN '08, pages 421–432. IEEE, 2008. DOI: 10.1109/IPSN.2008.10.

[MFB15] R. Mastrandrea, J. Fournet, and A. Barrat. Contact Patterns in a High School: A Comparison between Data Collected Using Wearable Sensors, Contact Diaries and Friendship Surveys. *PLoS ONE*, 10, September 2015. DOI: 10.1371/journal.pone.0136497.

[MFCP+19] M. Meyer, T. Farei-Campagna, A. Pasztor, R. Da Forno, T. Gsell, J. Faillettaz, A. Vieli, S. Weber, J. Beutel, and L. Thiele. Event-triggered Natural Hazard Monitoring with Convolutional Neural Networks on the Edge. In *18th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '19, pages 73–84. IEEE, 2019. DOI: 10.1145/3302506.3310390.

[MFL+14] K. Min, A. Forys, A. Luong, E. Lee, J. Davies, and T. Schmid. WRENSys: Large-scale, rapid deployable mobile sensing system. In *39th Conference on Local Computer Networks Workshops*, LCN '14, pages 557–565. IEEE, 2014. DOI: 10.1109/LCNW.2014.6927703.

[MGWW10] S. Marano, W. Gifford, H. Wymeersch, and M. Win. NLOS identification and mitigation for localization based on UWB experimental data. *IEEE Journal on Selected Areas in Communications*, 28(7):1026–1035, 2010. DOI: 10.1109/JSAC.2010.100907.

[Min20] Ministry of Home Affairs. MHA Guidelines, June 2020.

[MMF+07] M. Malinowski, M. Moskwa, M. Feldmeier, M. Laibowitz, and J. Paradiso. CargoNet: A Low-Cost Micropower Sensor Node Exploiting Quasi-Passive Wakeup for Adaptive Asychronous Monitoring of Exceptional Events. In *5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, page 145–159, New York, NY, USA, 2007. Association for Computing Machinery. DOI: 10.1145/1322263.1322278.

[MNMS17] A. Montanari, S. Nawaz, C. Mascolo, and K. Sailer. A Study of Bluetooth Low Energy performance for human proximity detection in the workplace. In *IEEE International Conference on Pervasive Computing and Communications*, PerCom '17, pages 90–99. IEEE, 2017. DOI: 10.1109/PERCOM.2017.7917855.

[MPGG+17] A. Migliano, A. Page, J. Gómez-Gardeñes, G. Salali, S. Viguier, M. Dyble, J. Thompson, N. Chaudhary, D. Smith, J. Strods, R. Mace, M. G. Thomas, V. Latora, and L. Vinicius. Characterization of hunter-gatherer networks and implications for cumulative culture. *Nature Human Behaviour*, 1(2):0043, 2017. DOI: 10.1038/s41562-016-0043.

[MQRW22] R. Morillo, Y. Qin, A. Russell, and B. Wang. More the Merrier: Neighbor Discovery on Duty-Cycled Mobile Devices in Group Settings. *IEEE Transactions on Wireless Communications*, 21(7):4754–4768, 2022. DOI: 10.1109/TWC.2021.3133099.

[MSA17] Y. Ma, N. Selby, and F. Adib. Minding the Billions: Ultra-Wideband Localization for Deployed RFID Tags. In *23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, page 248–260, New York, NY, USA, 2017. Association for Computing Machinery. DOI: 10.1145/3117811.3117833.

[MSM19] V. Modekurthy, A. Saifullah, and S. Madria. DistributedHART: A Distributed Real-Time Scheduling System for WirelessHART Networks. In *IEEE Real-Time and Embedded Technology and Applications Symposium*, RTAS '19, pages 216–227. IEEE, 2019. DOI: 10.1109/RTAS.2019.00026.

[MZL+18] X. Ma, P. Zhang, X. Li, W. Tang, J. Wei, and O. Theel. DeCoT: A Dependable Concurrent Transmission-Based Protocol for Wireless Sensor Networks. *IEEE Access*, 6:73130–73146, 2018. DOI: 10.1109/ACCESS.2018.2877692.

[MZL+20] 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 Conference on Computer Communications*, INFOCOM '20, pages 1024–1033. IEEE, 2020. DOI: 10.1109/INFOCOM41043.2020.9155423.

[Nat20] National Institute for Public Health and Environment. The approach to tackling coronavirus in the Netherlands, March 2020.

[Nor20] Nordic Semiconductors. nRF52840 - Ultra Low Power Wireless Solutions from Nordic Semiconductors. nordicsemi.com/eng/Products/nRF52840, Accessed: 25-03-2020.

[OGT+18] L. Ozella, F. Gesualdo, M. Tizzoni, C. Rizzo, E. Pandolfi, I. Campagna, A. Tozzi, and C. Cattuto. Close encounters between infants and household members measured through wearable proximity sensors. *PLOS ONE*, 13, June 2018. DOI: 10.1371/journal.pone.0198733.

[OO14] D. Ongaro and J. Ousterhout. In search of an understandable consensus algorithm. In *USENIX Annual Technical Conference*, USENIX ATC '14, pages 305–319, USA, June 2014. USENIX Association. URL: https://dl.acm.org/doi/10.5555/2643634.2643666.

[Ope20] Opensignal. Coverage Maps. opensignal.com/networks, Accessed: 09-03-2020.

[OWK+09] D. Olguín, B. Waber, T. Kim, A. Mohan, K. Ara, and A. Pentland. Sensible organizations: Technology and methodology for automatically measuring organizational behavior. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):43–55, 2009. DOI: 10.1109/TSMCB.2008.2006638.

[PAL22] C. Profentzas, M. Almgren, and O. Landsiedel. MiniLearn: On-Device Learning for Low-Power IoT Devices. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 1–11, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.5555/3578948.3578949.

[PANL19] V. Poirot, B. Al Nahas, and O. Landsiedel. Paxos Made Wireless: Consensus in the Air. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '19, pages 1–12, USA, 2019. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3324320.3324322.

[PHC04] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *2nd International Conference on*

Embedded Networked Sensor Systems, SenSys '04, page 95–107, New York, NY, USA, 2004. Association for Computing Machinery. DOI: 10.1145/1031495.1031508.

[Phi20] Philippe's government. Coronavirus COVID-19, June 2020.

[PIM17] R. Piyare, T. Istomin, and A. L. Murphy. WaCo: A Wake-Up Radio COOJA Extension for Simulating Ultra Low Power Radios. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '17, page 48–53, USA, 2017. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3108009.3108016.

[PKD18] P. Pannuto, B. Kempke, and P. Dutta. Slocalization: Sub-uW Ultra Wideband Backscatter Localization. In *17th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '18, pages 242–253, New York, NY, USA, April 2018. Association for Computing Machinery. DOI: 10.1109/IPSN.2018.00052.

[PL21] V. Poirot and O. Landsiedel. Dimmer: Self-Adaptive Network-Wide Flooding with Reinforcement Learning. In *IEEE 41st International Conference on Distributed Computing Systems*, ICDCS '21, pages 293–303. IEEE, 2021. DOI: 10.1109/ICDCS51616.2021.00036.

[PMK+17] R. Piyare, A. L. Murphy, C. Kiraly, P. Tosato, and D. Brunelli. Ultra Low Power Wake-Up Radios: A Hardware and Networking Survey. *IEEE Communications Surveys and Tutorials*, 19(4):2117–2157, 2017. DOI: 10.1109/COMST.2017.2728092.

[PMMB18] R. Piyare, A. L. Murphy, M. Magno, and L. Benini. On-Demand LoRa: Asynchronous TDMA for Energy Efficient and Low Latency Communication in IoT. *Sensors*, 18(11):3718, 2018. DOI: 10.3390/s18113718.

[PPL11] A. Purohit, B. Priyantha, and J. Liu. WiFlock: Collaborative group discovery and maintenance in mobile sensor networks. In *10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '11, pages 37–48. IEEE, 2011. URL: https://ieeexplore.ieee.org/abstract/document/5779063.

[PSZ+07] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan. BeepBeep: A high accuracy acoustic ranging system using COTS mobile devices. In *5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, pages 1–14, New York, NY, USA, 2007. Association for Computing Machinery. DOI: 10.1145/1322263.1322265.

[Pub20a] Public Health Agency of Canada. Physical distancing, June 2020.

[Pub20b] Public Health England. Guidance on social distancing for everyone in the UK, March 2020.

[QLXL16] Y. Qiu, S. Li, X. Xu, and Z. Li. Talk more listen less: Energy-efficient neighbor discovery in wireless sensor networks. In *35th IEEE International Conference on Computer Communications*, INFOCOM '16, pages 1–9. IEEE, 2016. DOI: 10.1109/INFOCOM.2016.7524336.

[RDC18] B. Rogoff, A. Dahl, and M. Callanan. The importance of understanding children's lived experience. *Developmental Review*, 50:5–15, 2018. DOI: 10.1016/j.dr.2018.05.006.

[RL22] P. Rathje and O. Landsiedel. TraceBand: Privacy-Preserving Contact Tracing on Low-Power Wristbands. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 150–155, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.5555/3578948.3578962.

[RLS+07] E. Rea, J. Lafleche, S. Stalker, B. Guarda, H. Shapiro, I. Johnson, S. J. Bondy, R. Upshur, M. Russell, and M. Eliasziw. Duration and distance of exposure are important predictors of transmission among community contacts of Ontario SARS cases. *Epidemiology & Infection*, 135(6):914–921, 2007. DOI: 10.1017/S0950268806007771.

[RLS08] W. u. Rehman, E. d. Lara, and S. Saroiu. CILoS: A CDMA Indoor Localization System. In *10th International Conference on Ubiquitous Computing*, UbiComp '08, New York, NY, USA, January 2008. Association for Computing Machinery. DOI: 10.1145/1409635.1409650.

[RPL20] P. Rathje, V. Poirot, and O. Landsiedel. STARC: Low-power Decentralized Coordination Primitive for Vehicular Ad-hoc Networks. In *IEEE/IFIP Network Operations and Management Symposium*, NOMS '20, pages 1–6. IEEE, 2020. DOI: 10.1109/NOMS47738.2020.9110347.

[SBBT15] F. Sutton, B. Buchli, J. Beutel, and L. Thiele. Zippy: On-Demand Network Flooding. In *13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, page 45–58, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2809695.2809705.

[SBDM20] A. Spina, M. Breza, N. Dulay, and J. McCann. XPC: Fast and Reliable Synchronous Transmission Protocols for 2-Phase Commit and 3-Phase Commit. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '20, pages 73–84, USA, 2020. Junction Publishing. URL: https://dl.acm.org/doi/10.5555/3400306.3400316.

[SCF+08] T. Schmid, Z. Charbiwala, J. Friedman, Y. Cho, and M. Srivastava. Exploiting Manufacturing Variations for Compensating Environment-Induced Clock Drift in Time Synchronization. In *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '08, page 97–108, New

York, NY, USA, 2008. Association for Computing Machinery. DOI: 10.1145/1375457.1375469.

[SDB16] J. Schloemann, H. Dhillon, and R. Buehrer. Toward a Tractable Analysis of Localization Fundamentals in Cellular Networks. *IEEE Transactions on Wireless Communications*, 15(3):1768–1782, March 2016. DOI: 10.1109/TWC.2015.2496273.

[SDFG+17] F. Sutton, R. Da Forno, D. Gschwend, T. Gsell, R. Lim, J. Beutel, and L. Thiele. The Design of a Responsive and Energy-Efficient Event-Triggered Wireless Sensing System. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '17, page 144–155, USA, 2017. Junction Publishing. URL: https://dl.acm.org/doi/abs/10.5555/3108009.3108028.

[SDGJ08] Y. Sun, S. Du, O. Gurewitz, and D. Johnson. DW-MAC: A Low Latency, Energy Efficient Demand-Wakeup MAC Protocol for Wireless Sensor Networks. In *9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '08, page 53–62, New York, NY, USA, 2008. Association for Computing Machinery. DOI: 10.1145/1374618.1374627.

[SEA+20] N. Saeed, A. Elzanaty, H. Almorad, H. Dahrouj, T. Al-Naffouri, and M.-S. Alouini. CubeSat Communications: Recent Advances and Future Challenges. *IEEE Communications Surveys & Tutorials*, 22(3):1839–1862, 2020. DOI: 10.1109/COMST.2020.2990499.

[Sem23] Semtech. Semtech SX1262 - LoRa Connect Long Range Low Power LoRa Transceiver. semtech.com/products/wireless-rf/lora-core/sx1262, Accessed: 10-02-2023.

[SFBT19] F. Sutton, R. D. Forno, J. Beutel, and L. Thiele. BLITZ: Low Latency and Energy-Efficient Communication for Event-Triggered Wireless Sensing Systems. *ACM Transactions on Sensor Networks*, 15(2):1–38, 2019. DOI: 10.1145/3309702.

[SH18] G. Schroeer and B. Haefner. Predictive NLOS Detection for UWB Indoor Positioning Systems Based on the CIR. In *15th Workshop on Positioning, Navigation and Communications*, WPNC '18, pages 1–5. IEEE, 2018. DOI: 10.1109/WPNC.2018.8555804.

[SHH11] T. Sathyan, D. Humphrey, and M. Hedley. WASP: A system and algorithms for accurate radio localization using low-cost hardware. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(2):211–222, 2011. DOI: 10.1109/TSMCC.2010.2051027.

[SLB+17] M. Starnini, B. Lepri, A. Baronchelli, A. Barrat, C. Cattuto, and R. Pastor-Satorras. Robust modeling of human contact networks across different scales and proximity-sensing techniques. In

International Conference on Social Informatics, SocInfo '17, pages 536–551. Springer, 2017. DOI: 10.1007/978-3-319-67217-5_32.

[SMB+15] D. Spenza, M. Magno, S. Basagni, L. Benini, M. Paoli, and C. Petrioli. Beyond duty cycling: Wake-up radio with selective awakenings for long-lived wireless sensing systems. In *IEEE Conference on Computer Communications*, INFOCOM '15, pages 522–530. IEEE, 2015. DOI: 10.1109/INFOCOM.2015.7218419.

[SML+04] G. Simon, M. Maróti, A. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor Network-Based Countersniper System. In *2nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, page 1–12, New York, NY, USA, 2004. Association for Computing Machinery. DOI: 10.1145/1031495.1031497.

[SPDG+20] L. Setti, F. Passarini, G. De Gennaro, P. Barbieri, M. G. Perrone, M. Borelli, J. Palmisani, A. Di Gilio, P. Piscitelli, and A. Miani. Airborne transmission route of COVID-19: why 2 meters/6 feet of inter-personal distance could not be enough. *International Journal of Environmental Research and Public Health*, 2020. DOI: 10.3390/ijerph17082932.

[SPH+22] V. Salo, P. Pannuto, W. Hedgecock, A. Biri, D. Russo, H. Piersiak, and K. Humphreys. Measuring naturalistic proximity as a window into caregiver–child interaction patterns. *Behavior Research Methods*, 54:1580–1594, 2022. DOI: 10.3758/s13428-021-01681-8.

[SPZE20] B. Sidik, R. Puzis, P. Zilberman, and Y. Elovici. PALE: Time Bounded Practical Agile Leader Election. *IEEE Transactions on Parallel and Distributed Systems*, 31(2):470–485, 2020. DOI: 10.1109/TPDS.2019.2933620.

[SSB10] L. Sitanayah, C. Sreenan, and K. Brown. ER-MAC: A Hybrid MAC Protocol for Emergency Response Wireless Sensor Networks. In *4th International Conference on Sensor Technologies and Applications*, SENSORCOMM '10, pages 244–249. IEEE, 2010. DOI: 10.1109/SENSORCOMM.2010.45.

[STM23] STMicroelectronics. MEMS Sensors Ecosystem for Machine Learning. st.com/content/st_com/en/ecosystems/MEMS-Sensors-Ecosystem-for-Machine-Learning, 2023. Accessed: 29-05-2023.

[STVP23] E. Soprana, M. Trobinger, D. Vecchia, and G. P. Picco. Network On or Off? Instant Global Binary Decisions over UWB with Flick. In *22nd International Conference on Information Processing in Sensor Networks*, IPSN '23, page 261–273, New York, NY, USA, 2023. Association for Computing Machinery. DOI: 10.1145/3583120.3586967.

[SVB+11] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, V. Colizza, L. Isella, C. Régis, J.-F. Pinton, N. Khanafer, W. Van den Broeck, and P. Vanhems. Simulation of an SEIR infectious disease model on the dynamic contact network of conference attendees. *BMC medicine*, 9(1):87, 2011. DOI: 10.1186/1741-7015-9-87.

[SYWL14] W. Sun, Z. Yang, K. Wang, and Y. Liu. Hello: A generic flexible protocol for neighbor discovery. In *IEEE Conference on Computer Communications*, INFOCOM '14, pages 540–548. IEEE, 2014. DOI: 10.1109/INFOCOM.2014.6847978.

[SZDF+15] F. Sutton, M. Zimmerling, R. Da Forno, R. Lim, T. Gsell, G. Giannopoulou, F. Ferrari, J. Beutel, and L. Thiele. Bolt: A Stateful Processor Interconnect. In *13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, page 267–280, New York, NY, USA, 2015. Association for Computing Machinery. DOI: 10.1145/2809695.2809706.

[Tar72] R. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972. DOI: 10.1137/0201010.

[TDFB+23] R. Trüb, R. Da Forno, A. Biri, J. Beutel, and L. Thiele. LSR: Energy-Efficient Multi-Modulation Communication for Inhomogeneous Wireless IoT Networks. *ACM Transactions on Internet of Things*, 4(2), April 2023. DOI: 10.1145/3579366.

[TDFS+20] R. Trüb, R. Da Forno, L. Sigrist, L. Mühlebach, A. Biri, J. Beutel, and L. Thiele. FlockLab 2: Multi-Modal Testing and Validation for Wireless IoT. In *3rd Workshop on Benchmarking Cyber-Physical Systems and Internet of Things*, CPS-IoTBench '20. OpenReview.net, 2020. DOI: 10.3929/ethz-b-000442038.

[The23] The Things Industries. The Things Network - a global collaborative Internet of Things ecosystem that creates networks, devices and solutions using LoRaWAN. thethingsnetwork.org/, 2023. Accessed: 29-05-2023.

[TLKL+17] C. Tamis-LeMonda, Y. Kuchirko, R. Luo, K. Escobar, and M. Bornstein. Power in methods: Language to infants in structured and naturalistic contexts. *Developmental science*, 20(6):e12456, 2017. DOI: 10.1111/desc.12456.

[TSZ+07] F. Troesch, C. Steiner, T. Zasowski, T. Burger, and A. Wittneben. Hardware aware optimization of an ultra low power UWB communication system. In *IEEE International Conference on Ultra-Wideband*, pages 174–179. IEEE, 2007. DOI: 10.1109/ICUWB.2007.4380936.

[TVL+20] M. Trobinger, D. Vecchia, D. Lobba, T. Istomin, and G. P. Picco. One Flood to Route Them All: Ultra-Fast Convergecast of

Concurrent Flows over UWB. In *18th Conference on Embedded Networked Sensor Systems*, SenSys '20, page 179–191, New York, NY, USA, 2020. Association for Computing Machinery. DOI: 10.1145/3384419.3430715.

[VAGT13] S. Vasudevan, M. Adler, D. Goeckel, and D. Towsley. Efficient Algorithms for Neighbor Discovery in Wireless Networks. *IEEE/ACM Transactions on Networking*, 21(1):69–83, 2013. DOI: 10.1109/TNET.2012.2189892.

[VKK16] D. Vasisht, S. Kumar, and D. Katabi. Decimeter-Level Localization with a Single WiFi Access Point. In *13th USENIX Conference on Networked Systems Design and Implementation*, NSDI '16, page 165–178, USA, 2016. USENIX Association. URL: https://dl.acm.org/doi/abs/10.5555/2930611.2930623.

[VZA+18] D. Vasisht, G. Zhang, O. Abari, H.-M. Lu, J. Flanz, and D. Katabi. In-body backscatter communication and localization. In *Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '18, pages 132–146, New York, NY, USA, 2018. Association for Computing Machinery. DOI: 10.1145/3230543.3230565.

[WAJR+05] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *2nd European Workshop on Wireless Sensor Networks*, EWSN '05, pages 108–120. IEEE, 2005. DOI: 10.1109/EWSN.2005.1462003.

[WALJ+06] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and Yield in a Volcano Monitoring Sensor Network. In *7th Symposium on Operating Systems Design and Implementation*, OSDI '06, page 381–396, USA, 2006. USENIX Association. URL: https://dl.acm.org/doi/abs/10.5555/1298455.1298491.

[WAO+11] B. Waber, S. Aral, D. Olguin, L. Wu, E. Brynjolfsson, and A. Pentland. Sociometric badges: A new tool for IS research. *SSRN*, 2011. DOI: 10.2139/ssrn.1789103.

[WBDF+19] S. Weber, J. Beutel, R. Da Forno, A. Geiger, S. Gruber, T. Gsell, A. Hasler, M. Keller, R. Lim, P. Limpach, M. Meyer, I. Talzi, L. Thiele, C. Tschudin, A. Vieli, D. Vonder Mühll, and M. Yücel. A decade of detailed observations (2008–2018) in steep bedrock permafrost at the Matterhorn Hörnligrat (Zermatt, CH). *Earth System Science Data*, 11(3):1203–1237, 2019. DOI: 10.5194/essd-11-1203-2019.

[WBM+17] F. Walter, A. Burtin, B. McArdell, N. Hovius, B. Weder, and J. Turowski. Testing seismic amplitude source location for fast debris-flow detection at Illgraben, Switzerland. *Natural Hazards and Earth System Sciences*, 17(6):939–955, 2017. DOI: 10.5194/nhess-17-939-2017.

[WC19] D. Winkler and A. Cerpa. WISDOM: Watering Intelligently at Scale with Distributed Optimization and Modeling. In *17th Conference on Embedded Networked Sensor Systems*, SenSys '19, pages 219–231, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3356250.3360023.

[WCPC18] D. Winkler, M. Carreira-Perpinan, and A. Cerpa. Plug-and-Play Irrigation Control at Scale. In *17th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '18, pages 1–12, New York, NY, USA, 2018. Association for Computing Machinery. DOI: 10.1109/IPSN.2018.00008.

[WDA+12] G. Wittenburg, N. Dziengel, S. Adler, Z. Kasmi, M. Ziegert, and J. Schiller. Cooperative event detection in wireless sensor networks. *IEEE Communications Magazine*, 50(12):124–131, 2012. DOI: 10.1109/MCOM.2012.6384461.

[WDTB+05] M. Walters, K. Dautenhahn, R. Te Boekhorst, K. L. Koay, C. Kaouri, S. Woods, C. Nehaniv, D. Lee, and I. Werry. The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment. In *IEEE International Workshop on Robot and Human Interactive Communication*, ROMAN'05, pages 347–352. IEEE, 2005. DOI: 10.1109/ROMAN.2005.1513803.

[WDWS10] G. Wittenburg, N. Dziengel, C. Wartenburger, and J. Schiller. A System for Distributed Event Detection in Wireless Sensor Networks. In *9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '10, page 94–104, New York, NY, USA, 2010. Association for Computing Machinery. DOI: 10.1145/1791212.1791225.

[WH11] Y. Wang and I. Hussein. Multiple vehicle Bayesian-based domain search with intermittent information sharing. In *American Control Conference*, ACC '11, pages 1280–1285. IEEE, 2011. DOI: 10.1109/ACC.2011.5990845.

[Wor20] World Health Organisation (WHO). Coronavirus disease (COVID-19) advice for the public, March 2020.

[WSC+18] L. Wei, W. Sun, H. Chen, P. Yuan, F. Yin, Q. Luo, Y. Chen, and L. Chen. A Fast Neighbor Discovery Algorithm in WSNs. *Sensors*, 18(10), 2018. DOI: 10.3390/s18103319.

[WTB+12] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, and R. Alexander. RPL: IPv6 routing protocol for low-power and lossy networks. URL: doi.org/10.17487/RFC6550, March 2012.

[WTJ09] S. Waharte, N. Trigoni, and S. Julier. Coordinated Search with a Swarm of UAVs. In *6th IEEE Annual Communications*

Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, pages 1–3. IEEE, 2009. DOI: 10.1109/SAHCNW.2009.5172925.

[XLZ+21] M. Xu, C. Liu, Y. Zou, F. Zhao, J. Yu, and X. Cheng. wChain: A Fast Fault-Tolerant Blockchain Protocol for Multihop Wireless Networks. *IEEE Transactions on Wireless Communications*, 20(10):6915–6926, October 2021. DOI: 10.1109/TWC.2021.3078639.

[XXLJ19] Y. Xie, J. Xiong, M. Li, and K. Jamieson. MD-Track: Leveraging Multi-Dimensionality for Passive Indoor Wi-Fi Tracking. In *25th Annual International Conference on Mobile Computing and Networking*, MobiCom '19, New York, NY, USA, 2019. Association for Computing Machinery. DOI: 10.1145/3300061.3300133.

[YNB+22] Z. Yu, X. Na, C. A. Boano, Y. He, X. Guo, and M. Jin. SmarTiSCH: An interference-aware engine for IEEE 802.15. 4e-based networks. In *21st ACM/IEEE Conference on Information Processing in Sensor Networks*, IPSN '22, pages 350–362. IEEE, 2022. DOI: 10.1109/IPSN54338.2022.00035.

[YNL+12] I. Yoon, D. K. Noh, D. Lee, R. Teguh, T. Honma, and H. Shin. Reliable Wildfire Monitoring with Sparsely Deployed Wireless Sensor Networks. In *IEEE 26th International Conference on Advanced Information Networking and Applications*, AINA '12, pages 460–466. IEEE, 2012. DOI: 10.1109/AINA.2012.107.

[ZCDH12] P. Zand, S. Chatterjea, K. Das, and P. Havinga. Wireless Industrial Monitoring and Control Networks: The Journey So Far and the Road Ahead. *Journal of Sensor and Actuator Networks*, 1(2):123–152, 2012. DOI: 10.3390/jsan1020123.

[Zea18] C. Zeanah. *Handbook of Infant Mental Health*. Guilford Publications, 2018.

[ZGH+21] T. Zhang, T. Gong, S. Han, Q. Deng, and X. S. Hu. Fully Distributed Packet Scheduling Framework for Handling Disturbances in Lossy Real-Time Wireless Networks. *IEEE Transactions on Mobile Computing*, 20(2):502–518, 2021. DOI: 10.1109/TMC.2019.2950913.

[ZJGD22] T. Zachariah, N. Jackson, B. Ghena, and P. Dutta. ReliaBLE: Towards Reliable Communication via Bluetooth Low Energy Advertisement Networks. In *International Conference on Embedded Wireless Systems and Networks*, EWSN '22, pages 96–107, New York, NY, USA, 2022. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.5555/3578948.3578957.

[ZLW18] Z. Zeng, S. Liu, and L. Wang. NLOS Identification for UWB Based on Channel Impulse Response. In *12th International Conference on*

Signal Processing and Communication Systems, ICSPCS '18, pages 1–6. IEEE, 2018. DOI: 10.1109/ICSPCS.2018.8631718.

[ZMS20] M. Zimmerling, L. Mottola, and S. Santini. Synchronous Transmissions in Low-Power Wireless: A Survey of Communication Protocols and Network Services. *ACM Computing Surveys*, 53(6), December 2020. DOI: 10.1145/3410159.

List of Publications

The following list includes publications that form the basis of this thesis. The corresponding chapters are indicated in parentheses.

A. Biri, P. Pannuto, P. Dutta. **TotTernary - a wearable platform for social interaction tracking: demo abstract.** *18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '19).* Montreal, QC, Canada, 2019. DOI: 10.1145/3302506.3312486. (Chapter 2)

A. Biri, N. Jackson, L. Thiele, P. Pannuto, P. Dutta. **SociTrack: infrastructure-free interaction tracking through mobile sensor networks.** *26th Annual International Conference on Mobile Computing and Networking (MobiCom '20).* Online, 2020. DOI: 10.1145/3372224.3419190. (Chapter 2)

A. Biri, R. Da Forno, T. Gsell, T. Gatschet, J. Beutel, L. Thiele. **STeC: Exploiting Spatial and Temporal Correlation for Event-based Communication in WSNs.** *19th ACM Conference on Embedded Networked Sensor Systems (SenSys '21).* Coimbra, Portugal, 2021. DOI: 10.1145/3485730.3485951. (Chapter 4)

A. Biri, R. Da Forno, T. Kuonen, F. Mager, M. Zimmerling, L. Thiele. **Hydra: Concurrent Coordination for Fault-tolerant Networking.** *22nd International Conference on Information Processing in Sensor Networks (IPSN '23).* San Antonio, TX, USA, 2023. DOI: 10.1145/3583120.3587047. (Chapter 5)

A. Biri, M. Zimmerling, L. Thiele. **Demos: Robust Orchestration for Autonomous Networking.** *Under submission for International Conference on Embedded Wireless Systems and Networks (EWSN '23).* Rende, Italy, 2023. (Chapter 3)

The following list includes publications that are not part of this thesis.

R. Jacob, R. Da Forno, R. Trüb, A. Biri, L. Thiele. **Dataset: Wireless Link Quality Estimation on FlockLab - and Beyond.** *2nd Workshop on Data: Acquisition to Analysis (DATA '19).* New York City, NY, USA, 2019. DOI: 10.1145/3359427.3361907.

R. Jacob, A. Schaper, A. Biri, R. Da Forno, L. Thiele. **Synchronous transmissions on Bluetooth 5 and IEEE 802.15.4 – A replication study.** *3rd Workshop on Benchmarking Cyber-Physical Systems and Internet of Things (CPS-IoTBench '20).* Online, 2020. DOI: 10.3929/ethz-b-000442044.

R. Trüb, R. Da Forno, L. Sigrist, L. Mühlebach, A. Biri, J. Beutel, L. Thiele. **FlockLab 2: Multi-Modal Testing and Validation for Wireless IoT.** *3rd Workshop on Benchmarking Cyber-Physical Systems and Internet of Things (CPS-IoTBench '20).* Online, 2020. DOI: 10.3929/ethz-b-000442038.

L. Heim, A. Biri, Z. Qu, L. Thiele. **Measuring what Really Matters: Optimizing Neural Networks for TinyML.** *ArXiv.* Online, 2021. DOI: 10.48550/arXiv.2104.10645.

R. Trüb, R. Da Forno, L. Daschinger, A. Biri, J. Beutel, L. Thiele. **Non-Intrusive Distributed Tracing of Wireless IoT Devices with the FlockLab 2 Testbed.** *ACM Transactions on Internet of Things (ACM TIOT).* ACM, 2021. DOI: 10.1145/3480248.

V. Salo, P. Pannuto, W. Hedgecock, A. Biri, D. Russo, H. Piersiak, K. Humpreys. **Measuring naturalistic proximity as a window into caregiver–child interaction patterns.** *Behavior Research Methods.* Springer, 2022. DOI: 10.3758/s13428-021-01681-8.

A. Cicoira, S. Weber, A. Biri, B. Buchli, R. Delaloye, R. Da Forno, I. Gaertner-Roer, S. Gruber, T. Gsell, A. Hasler, R. Lim, P. Limpach, R. Mayoraz, M. Meyer, J. Noetzli, M. Phillips, E. Pointner, H. Raetzo, C. Scapozza, T. Strozzi, L. Thiele, A. Vieli, D. Vonder Mühll, V. Wirz, J. Beutel. **In situ observations of the Swiss periglacial environment using GNSS instruments.** *Earth System Science Data (ESSD).* Copernicus, 2022. DOI: 10.5194/essd-14-5061-2022.

F. Mager, A. Biri, L. Thiele, M. Zimmerling. **BUTLER: Increasing the Availability of Low-Power Wireless Communication Protocols.** *International Conference on Embedded Wireless Systems and Networks (EWSN '22).* Linz, Austria, 2022. https://dl.acm.org/doi/abs/10.5555/3578948.3578958.

R. Trüb, R. Da Forno, A. Biri, J. Beutel, L. Thiele. **LSR: Energy-Efficient Multi-Modulation Communication for Inhomogeneous Wireless IoT Networks.** *ACM Transactions on Internet of Things (ACM TIOT).* ACM, 2023. DOI: 10.1145/3579366.