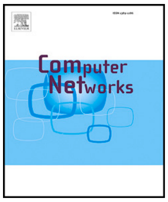
Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Design of fieldable cross-layer optimized network using embedded software defined radios: Survey and novel architecture with field trials

Jithin Jagannath [*], Anu Jagannath, Justin Henney, Tyler Gwin, Zackary Kane, Noor Biswas, Andrew Drozd

Marconi-Rosenblatt AI/ML Innovation Lab, ANDRO Computational Solutions, LLC, Rome NY, United States of America

ARTICLE INFO

Keywords:
Cross-layer Optimization
Software defined radio
Outdoor experiments
Survey
Field Trials

ABSTRACT

The proliferation of wireless devices and their ever increasing influence on our day-to-day life is very evident and seems irreplaceable. This exponential growth in demand, both in terms of the number of devices and Quality of Service (QoS) had spawned the concept of cross-layer optimization several years ago. The primary goal of the cross-layer approach was to liberate the strict boundary between the layers of the traditional Open Systems Interconnection (OSI) protocol stack. The objective was to enable information flow between layers which then can be leveraged to optimize the network's performance across the layers mitigating the challenges due to scarce resources while meeting QoS demands. The initial decade focused on establishing the theoretical feasibility of this revolutionary concept and gauging the effectiveness and limits of this idea. During the next phase, the advent of software defined radios (SDR) accelerated the growth of this domain due to its added flexibility. Even with the immense interest and progress in this area of research, there has been a gaping abyss between solutions designed in theory and ones deployed in practice. To establish this, we first present an elaborate survey of the cross-layer protocol stack literature with emphasis on their maturity scale. Next, we briefly discuss how a commercial off-the-shelf (COTS), low SWaP (Size, Weight, and Power) embedded SDR (e-SDR) was transformed into a standalone, fieldable transceiver. Thereafter, we provide the software design ethos that focuses on efficiency and flexibility such that the optimization objectives and cross-layer interactions can be reconfigured rapidly. To demonstrate our claims, we provide results from extensive outdoor over-the-air experiments in various settings with up to 10-node network topologies. The results from the field trials demonstrate high reliability, throughput, and dynamic routing capability. To the best of our knowledge, this is the first time in literature, a COTS e-SDR has been leveraged to successfully design a cross-layer optimized transceiver that is capable of forming an ad hoc network that provides high throughput and high reliability in a ruggedized, weatherized, and fieldable form factor.

1. Introduction

In recent years, we have seen explosive growth in the number of wireless devices that have become an inevitable component of our daily lives. This includes everything such as the connected devices in our smart homes, cellular network, the entire concept of Internet-of-Thing (IoT) networks controlling manufacturing, monitoring smart grids, space communications, underwater networks, tactical networks, among others. As we move from 5G (5th Generation) to 6G (6th Generation), the need to optimize the scarce resources is becoming evident and inevitable [1–7]. We have also seen a drive towards the software virtualization of these devices and networks to provide much-required flexibility and adaptability to the growing needs of each application. The inability to adapt the network operations depending on varying requirements and dynamic deployed scenarios due to black-box style wireless stack leads to inefficient resource usage and suboptimal performance. Dynamic network control and agile management (user scheduling, radio resources, mobility management) are the envisioned benefits of the next-generation wireless network paradigms such as software-defined networking (SDN), network function virtualization (NFV), and multi-access edge computing (MEC) [8]. Therefore, a reconfigurable radio stack that can be transformed based on diverse applications as well as to adapt to the dynamic wireless conditions is favored.

[*] Corresponding author.
E-mail addresses: jjagannath@androcs.com (J. Jagannath), ajagannath@androcs.com (A. Jagannath), jhenney@androcs.com (J. Henney), tgwin@androcs.com (T. Gwin), zkane@androcs.com (Z. Kane), nbiswas@androcs.com (N. Biswas), adrozd@androcs.com (A. Drozd).

https://doi.org/10.1016/j.comnet.2022.108917
Received 8 November 2021; Received in revised form 23 January 2022; Accepted 14 March 2022
Available online 25 March 2022

Traditionally, the strictly layered architecture proposed by the open systems interconnection (OSI) reference model has been the prevalent design for a majority if not all modern networking architectures. This is strict in the sense that they are designed to maintain only a limited interface between the neighboring layers [9]. Realizing the deficiencies in this layered architecture, the cross-layer optimized approach has been proposed over the past decade to maximize the utilization of scarce resources by "erasing" the strict boundaries between various layers of the protocol stack. In other words, any attempt to violate the OSI reference model is considered a cross-layer design [10]. While there are abundant solutions proposed in literature [9,11–14], the majority of it is limited to simulations that may have strong assumptions and/or do not consider all the hardware constraints and rigidness that may be encountered during a real-life deployment. During the next phase of advancement, the advent of software defined radios (SDR) provided the much-needed impetus to this concept providing the flexibility to implement novel cross-layer architectures. This enabled some of these efforts to be extended to hardware-based testbed evaluations. In most cases, these efforts still used one or more dedicated (non-embedded) host computers to implement the solutions which were then connected to SDRs.

1.1. Motivation, problem statement, and contribution

Motivation: Even with the advances discussed above, to the best of our knowledge, there does not exist a ruggedized and fieldable SDR with a comprehensive cross-layer optimization capable software module implemented on an embedded ARM processor. The main reason for this is the various hurdles that are associated with developing the solution from theory to effective hardware deployable software. In this article, we aim to demonstrate that this challenge can be overcome using a Commercial off-the-shelf (COTS) embedded SDR (e-SDR) and a novel efficient software architecture. Therefore, in this article, we present the first, completely stand-alone, ruggedized, and fieldable cross-layer optimized solution built using a low SWaP (Size, Weight, and Power) e-SDR.

Problem Statement: Driven by our motivation, we define a specific problem statement that is considered in this work. Design a fieldable e-SDR with an energy-aware cross-layer optimized protocol that aims to maximize network lifetime for enabling telemetry collection of tactical test and evaluation ranges.

The primary objective of this work is the transition of theoretical cross-layer concepts that have been shown to be effective in simulations to a fieldable hardware entity. Our implementation principle has been to facilitate rapid reconfiguration of network objectives with only a few lines of code thereby enabling a *truly* software-defined radio that evolves with the growing requirements. To accomplish this, in the extension to our prior work [15], we first discuss the hardware-level modifications required to customize the baseline e-SDR into a fieldable solution. Next, we outline how the software protocol stack was designed and implemented on a computationally constrained ARM processor. To demonstrate the feasibility of the implementation and the designed transceiver in terms of throughput and reliability, we performed extensive outdoor experiments with up to 10 nodes in the network. The proposed architecture and design principles can be leveraged to implement and mature several of the novel cross-layer optimized solutions to meet the evolving needs of both tactical and commercial communication systems. We hope and believe the unique cross-layer design methodologies and extensive outdoor evaluations (field trials) will serve as an impetus for the maturation of novel cross-layer solutions.

Contribution: The key contributions of the article are as follows;

- To place the contribution in perspective, we survey the domain of cross-layer optimization with an emphasis on the maturity of the solutions. Specifically, we discuss how the majority of the works in this domain is restricted to either simulations or preliminary in-lab hardware analysis.
- We provide a brief account of hardware components and their configuration that were used to customize the COTS e-SDR to ensure a standalone, portable solution that would provide extended range and is capable of running cross-layer optimized geographical ad hoc routing.
- Embedded software-defined cross-layer protocol stack implementation with a clearly laid out explanation showing all steps from the design concept to software implementation.
- Additionally, we provide details of the implementation of graphical user interface (GUI) as an example of how the software protocol stack also provisions end-user-defined applications.
- We have conducted an elaborate outdoor experimental evaluation that includes, peer-to-peer setting, line network, 5-node topology, and 10-node topology.
- The experiments demonstrated long-range, high throughput, and reliability in all the settings. The real-time adaptability and dynamic routing capability of the cross-layer optimized network have been demonstrated. Finally, we also demonstrated how the network can handle multiple sessions without degradation.

The rest of the paper is organized as follows, in Section 2, we survey the recent works in the cross-layer domain with an emphasis on the maturity of the proposed solutions. Next, in Section 3, we discuss the system design from the hardware and software point of view. We elaborate on the network topologies and various test objectives for the field trials that showcase the reliability, throughput, and dynamic routing capabilities of the fieldable e-SDR in Section 4. Finally, we conclude the article in Section 5.

2. Survey of cross-layer approaches

Cross-layer approaches have been explored as a novel solution to address a wide range of problems in wireless communication due to its perceived benefits in sustaining communications in a dynamic and constrained environment [9–14,16,17]. This includes tactical network [18–20], commercial terrestrial network [21–24], space networks [25,26], acoustic underwater networks [27–32], and even in the upcoming visible light communication networks [33–37]. These solutions are designed for various objectives such as optimizing throughput and/or latency [38–41], fairness [18,42,43], energy consumption [19,22,44–46], resource management [24,34], efficient multipath TCP [21]. The goal of this survey is not to cover the entirety of cross-layer approaches, but rather to demonstrate the gamut of the areas of application for cross-layer approaches to emphasize the impact and importance of these solutions. In contrast to other surveys, in this section, we specifically focus on reviewing the maturity of these state-of-the-art cross-layer solutions to *expose the absence of a deployable e-SDR based solution*. We divide the discussion into two broad categories, works that are limited to simulations and others that have provided some form of preliminary hardware testbed-based evaluation to validate their work.

2.1. Initial proof of concept in simulation

As in any novel research, simulations are the obvious first choice to establish feasibility and performance gains over existing approaches. To this end, the majority of the works that propose cross-layer optimization have been limited to simulations [18–22,38–40] due to the challenges, time, and effort it takes to evaluate them on hardware testbed. Most works in the cross-layer domain rely on MATLAB [20,38,45,47], NS2/3 [22,42,43,46], OMNET [19] or similar wireless network simulators. In most cases, these simulations are executed under various assumptions and/or are abstracted from the physical layer (PHY) of the protocol stack and are restricted to packet-level simulators. These assumptions and abstraction imply several intricacies of real-world deployment (operating channel conditions, computational resources, RF frontend capabilities, flexibility, and associated latency/overhead) are overlooked or set aside to be handled in the future which often thwarts the maturation level of cross-layer solutions.

Since there are a large number of solutions proposed in this domain and evaluated in simulations, we categorize them based on the objective of the cross-layer approach. At the same time, we would like to point out that these are not strict classifications as many cross-layer approaches take into account multiple metrics either in their objective function or constraints. Hence, a single work could arguably fall into multiple categories simultaneously. Therefore, in most cases, the categorization is based on at least one of its core objectives. The main purpose of the categorization is to make it easier for readers to assimilate all the content and show the widespread use of these techniques.

2.1.1. Maximizing throughput

From the very early days, throughput has been a central metric for any kind of communication device. The evolution of cellular networks, WiFi, or any other radio access technology is often driven by the desire to drastically improve the throughput of the network. Hence, it is not surprising to see a number of cross-layer optimized solutions have been developed for this exact purpose.

Cognitive radio networks (CRNs) are referred to as networks that are capable of sensing and dynamically accessing the spectrum of interest when it is idle and not being used by primary users. Primary users are entitled to a license to use the spectrum at the highest priority. The CRN can be seen as secondary users who are allowed to access idle parts of the spectrum without disrupting the primary user's activities. These are general definitions and could be more stringently defined from case to case in actual deployments. A distributed cross-layer approach to maximize the network throughput of CRN was proposed in [38]. The goal was to perform opportunistic spectrum access and dynamic routing algorithm and was referred to as ROSA (ROuting and Spectrum Allocation algorithm). The algorithm enabled individual nodes to perform joint routing, dynamic spectrum allocation, scheduling, and transmit power control. The proposed solution is shown to outperform approaches that considered only spectrum allocation or routing individually using a packet-level MATLAB simulator.

In the case of a CRN employing Code Division Multiple Access (CDMA) at the physical layer, a cross-layer algorithm to maximize throughput was proposed using a new spread-spectrum management paradigm using [39]. The algorithm referred to as ROCH (Routing and cOde division CHannelization) proposed jointly-designed routing and code division channelization similar to ROSA with the added dimension of codeword optimization. This work also relied on packet-level simulators for the initial proof-of-concept.

In [48], the authors propose a cross-layer approach that utilizes parameters from Medium Access Control (MAC) and PHY layer of the IEEE 802.11p protocol in Dedicated Short Range Communication (DSRC) based vehicular communication networks. To this end, they introduce cross-layer optimization modeling for real-time, on-road multimedia services and present a throughput-maximal framework that is designed for varying fading channel in mobile nodes with high velocities. The authors claim the performance in the simulation provided evidence for performance improvement which could be relevant even beyond vehicular networks.

In [49], the authors propose a cross-layer multi-path routing approach that utilizes non-correlated and node-disjoint paths able to concurrently transport multimedia content from sources to the destination by accounting for contextual information. To accomplish this the nodes utilize cross-layer interaction between the MAC and the network layer to tune their wake-up schedules and determine the disjoint paths. The work is limited to simulation but the authors have observed performance improvement in terms of throughput, latency ad energy usage.

2.1.2. Minimizing delay or latency

A key factor that impacts cellular networks and wireless mesh networks is the delay or latency experienced by the network. Latency is often a key Quality of Service (QoS) metric required by end customers relying on wireless services for the operations. Accordingly, cross-layer approaches are proposed to minimize end-to-end delay for a multihop wireless network in [50]. The authors propose two cross-layer schemes that they refer to as, *loosely coupled cross-layer scheme* and a *tightly coupled cross-layer scheme*. In the loosely coupled cross-layer scheme, routing is computed first and then the information of routing is used for link layer scheduling whereas, in the tightly coupled scheme, routing and link scheduling are jointly solved in one optimization model. They showcased the superiority of their cross-layer approaches over traditional non-cross-layer approaches through simulations.

One of the key objectives of 5G networks is to support ultra-reliable and low-latency communications (URLLC). Cross-layer approaches have been leveraged recently to enable URLLC in radio access networks by devising a resource allocation policy and a proactive packet dropping strategy [40]. In this work, the authors optimize the packet dropping policy, power allocation policy, and bandwidth allocation policy to minimize the transmit power under the Quality of Service (QoS) constraint. The feasibility of the proposed approach is validated using simulations.

In [51], the authors identify that the end-to-end delay performance has a complex dependence on the higher-order statistics of cross-layer algorithms, and hence optimization-based design methodologies that optimize the long term network utilization are not ideal for delay-aware design. To overcome this, the authors design a delay-aware joint flow control, routing, and scheduling algorithm for multihop wireless networks to maximize network utilization. The proposed cross-layer approach utilizes a regulated scheduling strategy based on a token-based service discipline, for shaping the per-hop delay distribution. Through simulation, the authors show optimal network utilization and better end-to-end delay performance for the proposed joint flow control, routing, and scheduling algorithm.

A cross-layer design is proposed in [47] using virtual queue structures to guarantee finite buffer size or worst-case delay performance. The algorithm solves a joint congestion control, routing, and scheduling problem in a multihop wireless network while ensuring average end-to-end delay constraints per-flow and minimum data rate requirements. Through MATLAB simulations they illustrate a tradeoff between the throughput and average end-to-end delay bound while satisfying the minimum data rate requirements for individual flows.

DRS (Distributed Deadline-Based Joint Routing and Spectrum Allocation) [20] was introduced to mitigate the last packet problem that has been shown to be a shortcoming of queue length-based backpressure scheduling algorithms such as ROSA. In this work, the authors proposed the use of a virtual queue such that the virtual queue length was designed to mitigate the effects of the last packet problem without substantial degradation of the throughput performance. The virtual queue length considered parameters such as length of the packet, remaining lifetime, and estimated time to the destination. The distributed implementation of DRS was shown to outperform ROSA in terms of effective throughput and reliability using a packet-level simulator. The reason we have this work categorized under latency is due to the definition of effective throughput used in this work. Only packets received at the destination before the deadline contributed to effective throughput which essentially is trying to ensure packet delay constraints. The simulations in MATLAB demonstrate that DRS outperforms ROSA both in terms of effective throughput and reliability.

2.1.3. Fairness

Two cross-layer algorithms, a dual-based algorithm, and a penalty-based algorithm are proposed in [18] to solve the rate control problem in a multihop random access network. Both algorithms can be implemented in a distributed manner, and work at the link layer to adjust link attempt probabilities and at the transport layer to adjust session rates. Their convergence and effectiveness of the proposed solution were established using simulations.

In [42], the authors investigate the issue of fairness between IEEE 802.11-compliant wireless local area networks (WLANs) stations using Transport Control Protocol (TCP). Specifically, focusing on send/receive TCP traffic in WiFi hot spots, they show WiFi hot spot provides more service to the wireless sending station compared to the receiving stations. This unfair service implies that the wireless sending stations dominates the use of network bandwidth restricting the access to the receiving stations. To mitigate this unfair service issue, the authors propose a cross-layer feedback mechanism in which the MAC layer at the access points measures the per-station channel utilization and system-wide channel utilization to calculate the channel access cost. Next, the TCP senders use the cost in order to assure per-station fairness and to maximize channel utilization simultaneously hence invoking a cross-layer interaction between the data link and transport layer. Their simulations in NS2 demonstrate that the proposed approach outperforms existing schemes with respect to fairness, delay, and channel utilization.

Stream Control Transmission Protocol (SCTP)-based Concurrent Multipath Transfer (CMT) was designed to improve the wireless video delivery performance with its parallel transmission and bandwidth aggregation features. In [43], the authors point out that the shortcoming in the existing CMT solutions deployed at the transport layer is due to uncertainties at the lower layer (for example caused by variations of the wireless channel). They also point out that CMT-based video transmission may unfairly use excessive bandwidth in comparison with the popular TCP-based flows. To overcome these issues, the authors propose a fairness-driven cross-Layer SCTP-based Concurrent Multipath Transfer approach (CMT-CL/FD). CMT-CL/FD monitors and analyzes path quality, which includes wireless channel measurements at the data-link layer and rate/bandwidth estimations at the transport layer. They also propose a window-based mechanism for flow control to obtain an acceptable tradeoff between delivery fairness and efficiency. They utilize simulations to demonstrate the effectiveness of the proposed approach over traditional solutions.

2.1.4. *Minimizing energy utilization*

In recent years, green communication protocols have gained substantial interest. The term "Green" in most cases refers to communication protocols that aim to reduce the energy consumption of wireless networks. One can imagine the implications and benefits of minimizing energy consumption in various scenarios in the context of operating cost, mitigate the need to re-deploy nodes that rely on battery, and its implication on climate change. Due to these reasons, cross-layer optimization has been leveraged as a tool for minimizing energy consumption or maximizing the network lifetime. Typically, network lifetime is defined as the total operation duration of the network before the first node depletes its energy resources.

Authors propose a joint cross-layer optimization scheme that considers modulation, power control, and routing to maximize the energy efficiency of a wireless network [44]. The constraints considered in their optimization problem include Bit Error Rate (BER) and the requested data rate. Each node in their multihop wireless network is capable of transmitting seven different modulations schemes. They use an event-driven, link-level simulator implemented using C++ to demonstrate improvement over Ad hoc On-demand Distance Vector (AODV) routing protocol.

As discussed previously, CRN aims to use an idle (unused) portion of the licensed spectrum to access the channel in a dynamic manner. This technology is often critical for IoT devices that may be energy-constrained. To overcome this challenge, the authors of [45] propose a cross-layer approach that jointly considered the modulation order at the physical layer and the backoff probability from the MAC-layer to minimizing the energy consumption in the CR-based green IoT networks. The constraints considered in this work are to ensure IoT delay guarantees, licensed primary radio channel availability, and PR user activities. The authors use MATLAB simulations to evaluate the performance of the proposed solution. The simulations show that the proposed cross-layer design is able to reduce energy consumption while achieving the delay requirements in the network compared to traditional approaches that operate independently at the physical layer or data link (MAC) layer.

One of the key challenges of Wireless Sensor Network (WSN) is ensuring energy efficiency while maintaining the required network performance metrics to support delay-sensitive applications. This issue is further exacerbated in networks where the sensors follow a sleep-wake duty cycle to conserve energy. To overcome some of these challenges, [52] propose a cross-layer multi-path routing approach designed to support duty-cycled WSN networks. Similar to [49], using the interaction between MAC and the network layer, this approach also establishes multi-node disjoint paths with complementary duty-cycling. The authors performed analytical study that proved the correctness of the proposed solutions. Simulations were able to demonstrate up to 65% cumulative energy saving when compared to the always-on multipath approach incurring 10% increase in response time delays.

In [53], authors propose cross-layer clustering based routing protocol for WSN to extend network lifetime. In this article, the author select cluster head based on energy consumption, delay, throughput, security, distance and overhead. The authors rely on MATLAB simulations to show how the proposed solution outperforms traditional approaches such as ant lion optimization approach and grouped gray wolf search optimization.

In order to improve the energy efficiency of the IEEE 802.15.4 based network, the authors use a cross-layer approach to control the transmission power and minimize the broadcast of control packets [46]. The authors use information from the application layer, network layer, data-link layer, and physical layer to minimize the channel occupancy time by reducing control packets (used for neighbor discovery). The proposed approach is evaluated using NS2 to achieve approximately 10% reduction in energy consumption while maintaining end-to-end delays and comparable packet delivery ratios.

Another work that focuses on using cross-layer optimization for energy consumption is presented in [22]. In this work, the authors demonstrate how cross-layer optimization can be implemented to extend the network lifetime. To this end, ISA 100.11 [54] and a Wireless highway addressable remote transducer (WiHART) [55] compatible sensor network is used in the context of petroleum refinery scenario. The authors demonstrate the utility of optimization that accounts for the fixed frame size constraint of current industrial wireless standards in smaller network scenarios. Their NS3 simulations depicted longer network lifetimes than the traditional approaches that use minimum hop routing for the industrial wireless network. At the same time, they do caution by stating that even frame-based optimization does experience a non-negligible failure rate when solving routes for large network sizes.

A mathematical model for cross-layer protocol optimization in SDR was proposed in [19]. The goal of the framework is to provide tactical SDR with the flexibility to adapt its objectives including minimizing energy consumption while maintaining reliable packet delivery and latency constraints. Though the proposed solution is designed for SDRs the solution is only implemented and evaluated in OMNET++. Their simulations used a number of mission-critical network scenarios to demonstrate enhanced performance where SDRs effectively adapt to the dynamic environment.

As we survey through the breadth and width of cross-layer solutions applied to achieve various objectives discussed above, we can clearly see how the majority of the solutions rely on simulations that do not completely model or capture various intricacies of a real-world wireless network. While this is a necessary and valuable step to design, evaluate, and refine novel solutions, it often also ends up being the finish line for several of these approaches due to the daunting challenges that are involved in the maturation of the solutions. There have been attempts to overcome this challenge and shortcomings by designing frameworks, and utilizing SDR-based testbed. In the next section, we discuss some of these attempts that have striven to bridge the gap.

2.2. Preliminary testbed evaluation

In several cases, the next logical step is to evaluate the feasibility and performance on a hardware-based testbed. The advent of SDR has significantly nurtured efforts in this direction. At the same time, there is a distinction in the maturity/utility of solutions that have been implemented using a host PC controlling the SDRs and ones efficiently ported to a standalone e-SDR. The key difference is in the rapidity of development on computationally capable hardware in the first case as opposed to carefully optimized (often C/C++ or VHDL/Verilog) implementation on embedded (resource-constrained) hardware. The host PC-based development approach saves time and resources to rule out impractical solutions before significant time is spend in optimizing the implementation for final matured deployment. Therefore, based on the resources, the risk associated with novel cross-layer solutions either approaches can be adopted to mature network control and management solutions. To this end, several frameworks have been proposed for SDR to enable cross-layer optimized control [56–58]. There has also been work that employs SDR hardware but rely on emulation platforms [59] to perform evaluations as opposed to over-the-air (OTA) experimentations. Such emulation-based approaches may provide more flexibility to perform larger number of experiments in various topologies but still cannot be substituted for OTA evaluations or field trials.

Several of the solutions discussed earlier have been successfully extended to preliminary hardware testbeds [41,60]. In most of these cases, SDRs like the universal software radio peripheral (USRP) are used in association with the host PC. The authors extend their work of the proposed DRS algorithm [20] by implementing it on a five-node USRP testbed [41]. To accomplish this, the authors develop a cross-layer framework they refer to as CrOss-layer Based testbed with Analysis Tool (COmBAT) [58] that is implemented in Python to run on host PC that controls the USRP SDRs. The authors compare the proposed solution with ROSA to demonstrate improvement in both effective throughput and reliability of the network.

Similarly, in [60] authors use four-node SDR testbed to demonstrate how cognitive channelization can be achieved by jointly optimizing the transmission power and the waveform channel of the secondary users. The experiments are conducted for both narrowband and wideband of primary users showing improvements in BER for both primary and secondary users.

A resource allocation approach for CRN was designed and implemented in [61] using two SDR-based testbeds of the ORCA federation [62]. The proposed distributed resource allocation employs a Markov Random Field (MRF) framework to be deployed on secondary nodes of the CRN. The implementation leveraged GNU Radio to implement functionalities such as spectrum sensing, collision detection, among others. The authors consider multi-channel CRNs and focus on the physical, data link layer (more specifically MAC), and network layer. The goal of any CRN is to ensure successful communication between secondary users in a dynamic environment without impacting the primary users. The secondary user SDRs are designed to calculate an energy function based on the current states of the neighbors. The objective of the secondary users is to minimize the local energy function which in turn minimized the interference. The authors claim that the secondary users asymptotically converge to global optimal solution by updating their energy function through local sampling. The authors successfully implement the proposed solution on two SDR testbeds, IRIS [63] and ORBIT [64]. The experiments ranged from using up to 2 primary users and 9 secondary users. They compare their solution with reinforcement learning-based channel allocation algorithms. Through their preliminary experimental evaluation, they were able to demonstrate complete transparency towards primary users and demonstrate performance in terms of collision percentage and the number of required transmission slots for the secondary users of the CRN. It is interesting to note that the achievable throughput has not been discussed in this work which in most cases are throttled by lower sustainable sampling rates on the testbed.

Beyond relying on SDRs, well-defined commercial wireless protocols like WiFi and LoRa (for physical layer) have also been used along with microcomputing platforms to design cross-layer approaches [65]. In this case, a cross-layer approach — distributed energy-efficient routing (SEEK) - was implemented on Raspberry Pi to maximize the network lifetime using LoRa as the PHY. Using their proposed geographical routing protocol, SEEK, the authors were able to demonstrate significant improvement in the network lifetime compared to greedy ad hoc routing approaches. In this form, it was highly restricted in throughput (due to LoRa) as a trade-off for a longer transmission range. The survey of cross-layer approaches and their level of maturity has been summarized in Table 1.

In all these examples, the achievable throughput is usually low due to the lower sampling rate or rely on the significant computing power of the host PC. Either of these factors renders such solutions highly restrictive in terms of utility, portability, fieldability, and often does not meet end-user requirements. In this article, we discuss how by customizing COTS low SWaP e-SDR with limited computing resources one can efficiently design and deploy the *first-known embedded software-defined cross-layer optimized network*.

In concluding this section, we have seen that the majority of the cross-layer optimized works are limited to simulations and have not been successfully validated on a hardware platform. This is a major hurdle and shortcoming of the current state of research and development. In the recent past, some of this has been mitigated by preliminary hardware-based evaluations but has often provided limited performance in terms of metrics like throughput (due to low sampling rate constrained by computations) and/or has been dependent on external computational platforms. Due to these reasons, even with the advances made in the field, there is no known cross-layer optimized e-SDR solution built using commercial-off-the-shelf (COTS) e-SDR that, (i) can be deployed as a standalone unit, (ii) makes cross-layer optimized distributed routing decisions, (iii) is ruggedized for outdoor deployment, and (iv) can provide reliable and high throughput (up to 11 Mbps) links over large distances (1 km for up to 5.5 Mbps). The difficulty of finding the right trade-off between software design choices and performance along with achieving standalone, fieldable hardware customization is a daunting task. This article provides such a solution for the first time showing that designing and deploying high Technology Readiness Level (TRL) cross-layer optimized solutions based on a COTS low SWaP e-SDR is feasible.

3. System design

In this section, we first discuss some core design challenges that may be encountered during similar endeavors. Thereafter, we describe the system design of the cross-layer optimized transceiver and how it has been executed.

3.1. Design challenges

3.1.1. Choice of COTS SDR hardware

In recent years, there has been a surge in the number of SDRs available in the market. While this has provided several tradeoff opportunities for system design decisions, yet, determining the appropriate choice is challenging due to the multi-dimensional tradeoff that exists with the decision making. At a high level, these choices can be divided into some core characteristics portability (size and weight), embedded computational capabilities, RF frontend parameters (instantaneous bandwidth, tunable frequency range, among others), and cost.

Table 1
Summary table of cross-layer solutions and their maturity.

Proposed work	Optimizing/Framework	Simulation	Preliminary testbed	Comments
Ding et al 2010 [38]	Throughput	✓		Algorithm designed to perform joint routing, dynamic spectrum allocation, scheduling, and transmit power control for cognitive radio networks.
Ding et al 2013 [39]	Throughput	✓		Algorithm designed to perform routing and code division channelization in a CDMA based cognitive radio networks.
Liu et al 2018 [48]	Throughput	✓		Throughput-maximal cross-layer framework is proposed for DSRC-based vehicular network.
Jemili et al 2020 [49]	Throughput	✓		Cross-layer multi-path routing approach that utilizes non-correlated and node-disjoint paths
Cheng et al 2013 [50]	Delay	✓		Two cross-layer approaches are proposed to minimize end-to-end delay for multihop wireless network and evaluated in simulations.
She et al 2018 [40]	Reliability Latency	✓		The policies for packet dropping, power allocation, and bandwidth allocation are optimized to minimize the transmit power under the QoS constraint.
Xlong et al 2011 [51]	Delay	✓		A delay-aware joint flow control, routing, and scheduling algorithm is introduced for multihop network to maximize network utilization.
Xue et al 2013 [47]	Delay	✓		Cross-layer approach that use virtual queue structures to guarantee finite buffer size or worst-case delay performance.
Jagannath et al 2016 [20]	Effective Throughput	✓		Incorporates virtual queue length into the objective function of the cross-layer algorithm to mitigate the last packet problem.
Wang et al 2006 [18]	Fairness	✓		A dual-based algorithm and a penalty-based algorithm designed and evaluated to address the fair rate control problem in a multihop random access network.
Park et al 2008 [42]	Fairness	✓		Authors propose a cross-layer feedback mechanism to improve fairness among sending and receiving WiFi stations. The solution is implemented and evaluated in NS2.
Xu et al 2015 [43]	Fairness	✓		Cross-layer fairness-driven approach to improve video delivery while remaining fair to the competing TCP flows. The solution is implemented and evaluated in NS2.
Herrmann et al 2018 [22]	Network Lifetime	✓		Authors design cross-layer algorithm for extending lifetime of industrial network for petroleum refinery. The solution was evaluated using NS3.
Salameh et al 2021 [45]	Energy consumption	✓		Algorithm designed for CRN to minimize energy consumption while adhering to delay constraints and minimize impact on primary users. Algorithm outperforms non-cross-layer approaches in MATLAB simulations.
Jemili et al 2021 [52]	Energy Consumption	✓		Cross-layer multi-path routing approach designed to support duty-cycled WSN networks for energy efficiency while reducing impact on response time delays.
Lahane et al 2021 [53]	Energy Consumption	✓		Cross-layer clustering based routing protocol that was shown to outperform traditional approaches like grouped gray wolf search optimization.
Al-Jemeli et al 2015 [46]	Energy Consumption	✓		Cross-layer network operation model proposed for energy efficiency in IEEE 802.15.4-based networks. The solution was evaluated using NS2.
Nosheed et al 2019 [19]	Energy efficiency	✓		Mathematical model designed for SDR and performance evaluated in OMNET++.
Herrmann et al 2018 [22]	Network Lifetime	✓		Authors design cross-layer algorithm for extending lifetime of industrial network for petroleum refinery. The solution was evaluated using NS3.
Dai et al 2018 [44]	Energy efficiency	✓		Authors aim to maximize energy efficiency by jointly optimizing modulation, transmit power, and route.
Shome et al 2015 [56]	Framework		✓	Framework for implementation and evaluation of cross-layer approaches.
Demirors et al 2015 [57]	Framework		✓	Framework for implementing and evaluating cross-layer approaches.
Jagannath et al 2016 [58]	Framework		✓	Framework for implementing and evaluating cross-layer approaches.
Soltani et al 2019 [59]	Not specified		✓	Designed to integrate dynamic spectrum access, backpressure algorithm, and network coding for multihop networking. Evaluation performed using emulations.
Jagannath et al 2018 [41]	Throughput		✓	Throughput and reliability performance evaluated on 5-node USRP N210 SDR testbed.
Sklivanitis et al 2015 [60]	SINR		✓	BER performance evaluated on 4-node USRP N210 SDR testbed.
Tsitseklis et al 2020 [61]	Avoid Collision		✓	Cross-layer framework designed for CRN and evaluated on 2-node SDR testbed named IRIS [63] and ORBIT [64].
SEEK 2019 [65]	Network Lifetime		✓	Algorithm considers link energy efficiency, backlog, forward progress, & residual energy. Solution is implemented on RPI & evaluated on 6-node testbed.

In our case, portability and embedded computational capability were the two prime candidates. This meant ruling out all the SDRs that may not have sufficient embedded computational resources. In this article, we refer to SDR with embedded processing capability (such as Field Programmable Gate Array (FPGA) and General Purpose Processors (GPP)) as e-SDR. Generally, the larger the SDR the higher the computational resources but may end up consuming higher power and leading to a larger design. Hence, at this stage, the designer has to determine the best tradeoff that works for a given design and make sure the constraints from this point onwards is acceptable to the overall software architecture.

3.1.2. Limited computational resources

Embedded radio programming is a non-trivial task and the primary factor that must be considered are the available computational resources. Accordingly, care must be taken to avoid redundant memory read and write operations to reduce energy consumption and latency of operation. From a protocol stack implementation perspective, the available accelerators are the CPU and FPGA housed in the Xilinx Zynq XC7Z010-2I SoC with 512 MB of DDR3L RAM and 128 MB of QSPI Flash memory on the radio platform. These impose constraints on the number and size of the packet queues as well as storage of configuration and other header information on the radio.

Rather than going for a top-down approach whereby the protocol stack is designed and simulated with evaluations followed by hardware implementation, we chose a bottom-up approach where these radio resource constraints are carefully considered right from the inception of the protocol stack. These are evident from the energy consumption factor considered in the routing, design of a segment size where the packets are grouped into segments for outbound transmission, and gathering Optimization Assisting Information (OAI) via piggybacking. This way the control packet exchange will not occur per packet rather for a segment of packets. These steps are adopted to reduce overhead which implicitly reduces latency and energy consumption. Hence, the hardware constraints, as well as the target application requirements (long-term unattended deployment), directed our design choices.

Most of the FPGA resources were utilized for the PHY layer which is a pure IEEE802.11b standard. The PHY was housed in the FPGA for reduced latency operation. Since the FPGA resources were nearly fully utilized for the 802.11b implementation, none of the remaining upper layers could be offloaded to FPGA for acceleration. Consequently, the entire stack except the PHY/L1 was implemented on the Dual-core ARM Cortex A9 CPU of the SoC. The software architecture utilizes both the user space as well as kernel space. To reduce repeated I/O operations and to parallelize the stack operations to a full extent, we leveraged daemons executing from the user space that hosted multi-threaded processes. A deep dive into the software implementation is elaborated in Section 3.3. The software architecture is designed and developed to execute agnostic to the underlying PHY/L1 layer, accordingly, we have followed a plug-and-play design ethos for the software stack. Our implementation demonstrates this by interfacing the software protocol stack executed fully from the ARM processor with the 802.11b FPGA core.

3.1.3. Gathering optimization assisting information

Network optimization can be generally divided into two categories, (i) centralized and (ii) distributed. Centralized approaches often provide globally optimal solutions but at the expense of large overhead incurred in accumulating global information. This also introduces delays and would suffer scalability issues for larger network sizes. In contrast, distributed approaches reduce the overhead and are in general scalable but may not converge to globally optimal solutions. In either case, there is a need to acquire information that is essential to execute cross-layer optimization.

For ad hoc networks that are highly dynamic and need to be scalable, distributed approaches are the desired choice. This is also the case for our solutions. Our approach (which will be discussed in detail in the next few sections) rely on gathering OAI from immediate neighbors. To reduce the overhead, this information can be appended to the control packets such as RTS, CTS, BEACON, etc. The frequency of BEACON packets can be a function of network parameters such as relative mobility of the nodes, rate of change of traffic, among others. The essential takeaway is the need to carefully design protocols to ensure efficient yet effective exchange of timely information for the cross-layer decision engine.

Table 2
Specifications of Z2.

Specs.	Values
Frequency	70 MHz–6 GHz
Sample Rate	Up to 61.44 MS/s
RAM	Up to 512 MB of DDR3L
Flash memory	128 MB
Temperature rating	−40 C to +85 deg C
Processor	Dual-core ARM Cortex A9
Size	30 × 51 × 5 mm
Weight	8 grams

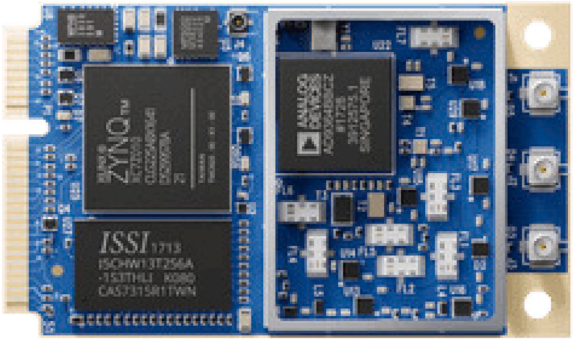


Fig. 1. Target baseline Z2 e-SDR.

3.2. Embedded software defined radio platform

One of the key objectives of the work was to develop a modular, programmable, portable, handheld, battery-powered, standalone solution which should operate in harsh conditions for several hours. This implied that the foundation of the design needs to be a low SWaP e-SDR. The proposed solution was implemented on a Epiq Solutions' Sidekiq Z2 e-SDR [66] (Fig. 1). It consists of an Analog Devices' AD9364 RFIC, Xilinx Zynq XC7Z010-2I system on chip (SoC), and the key device specifications are provided in Table 2.

Several hardware customizations were necessary to accomplish the objective of designing a fieldable transceiver using the COTS e-SDR. This includes adding a power amplifier, filters for the frequency of interest, power supply system that is capable of supplying power from the battery during standalone remote operation but could also operate from a direct current (DC) power source when available. To aid the implementation of various cross-layer routing techniques that use the location of nodes (such as geographical routing [65]), an embedded GPS receiver was also included in the final design. The block diagram of the final transceiver and the ruggedized prototype is shown in Fig. 2.

3.3. Cross-layer software architecture for embedded system

3.3.1. Design goals and overall architecture

The custom cross-layer protocol stack for the transceivers is implemented on the Zynq SoC of the Sidekiq Z2 platform. Specifically, the PHY is an IEEE802.11b implemented purely on the FPGA subsystem while the upper layers are implemented in the C/C++ language on the

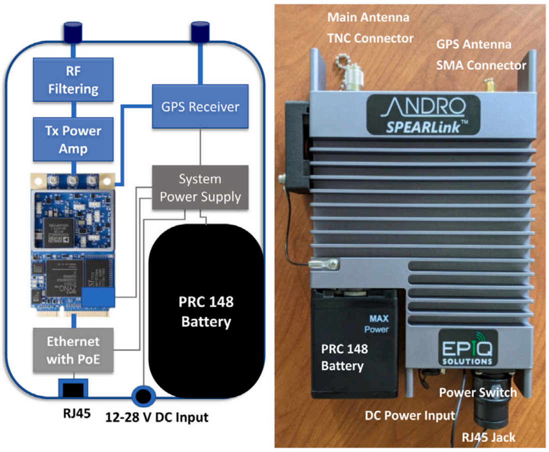


Fig. 2. Transceiver Design.

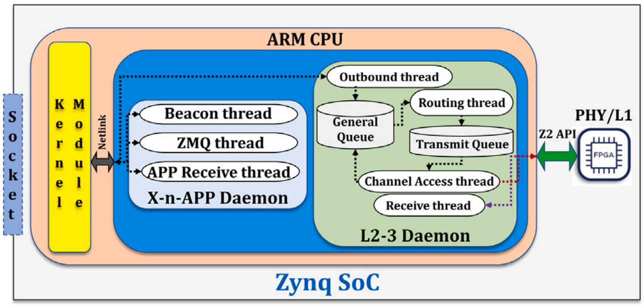


Fig. 3. Software architecture.

embedded Linux operating system of the Dual-core ARM Cortex A9 CPU of the SoC.

As mentioned previously, our design principle is to develop a plug-and-play software module of the protocol stack (except L1) to interface with any underlying PHY layer. The 802.11b FPGA core is a direct sequence spread spectrum (DSSS)-Complementary code keying (CCK) PHY with a 22MHz standard channel bandwidth which nearly fully utilized the logic cells and DSP blocks of the FPGA. The supported data rates with DSSS are 1 Mbps and 2 Mbps while CCK modulation supports 5.5 Mbps and 11 Mbps. The software stack is designed to be agnostic to the PHY layer and merely interfaces with the PHY FPGA core over a C-level SKIQ80211 application programming interface (API) provided by Epiq Solutions. The API provides all the necessary function calls to utilize the RF frontend.

We would like to reemphasize that the solution can be adapted to use most if not all PHY layer standards as long as the performance parameters (bit error rate, transmit power, data rates, etc.) are communicated to the upper layers over an appropriate C/C++ level interface. Since the focus of this work is the software implementation and evaluation of the upper layers (above PHY/L1), in the remainder of the section, we will detail the software architecture of the upper layers to facilitate long-range mesh networking.

Our primary goal in building the software architecture was to maintain *reconfigurability*. The reconfigurability was enforced by adopting a modular design framework with defined functions for each module while being resource-efficient. We define reconfigurability as the ability to modify the protocol characteristics such as routing objective, specifics of the cross-layer information exchanged, etc. Accommodating a software-defined cross-layer protocol stack in an ARM CPU is a non-trivial challenge. The software-defined architecture reaps benefits from the flexible software architecture which eases the future upgradability, customization (of factors like optimization objective among others), and maintenance of the transceiver software. Essentially, designing the entire radio stack on the SoC presents a daunting challenge owing to the memory, computational, and latency constraints. An unorganized framework could add overhead from unnecessary resource utilization consequently increasing the system latency.

The software architecture is broadly categorized into user and kernel space with daemons running in the user-space as in Fig. 3. We resort to daemons each of which hosts its own threads to attain a parallelized architecture that does not perform redundant and unnecessary memory accesses. In other words, we can say the user-space hosts multi-threaded processes which interface with the kernel space. However, there are certain design challenges with moving key functions to the kernel space. Kernel programming requires the most trusted operations as any bug or corruption may cause severe system crashes. Nevertheless, kernel space enjoys the benefits of low latency memory access operations. Furthermore, we also leverage the transport layer and IP headers supported by the Linux kernel. The software architecture in Fig. 3 shows a *socket* which is a generalized custom socket architecture for interfacing with external devices. The socket architecture is generalized to be able to reprogram and enable multiple interfacing options with a wide range of socket protocols such as ZMQ, UDP, etc.

The stack employs two daemons, namely; L2-3 and Cross-layer & Application (X-n-APP) that interact with the kernel module via Netlink sockets. Notice that the entire stack only has two daemons since it is more efficient to use threads over processes as they share the process' resources. On the other hand, having more processes would require more computational and memory resources. The kernel module handles the transport layer, packet encapsulation, and partial IP header population prior to passing over to the L2-3 daemon. The L2-3 daemon is the cross-layer L1, L2 (MAC), and L3 (Network) module which performs CSMA/CA-based medium access and cross-layer routing. Another level of cross-layer interactions occurs when the L2-3 daemon acquires L1 information such as the link reliability, data rate, etc., via the Sidekiq Z2's SKIQ80211 API. Hence, the term cross-layer as it involves interaction between L1, L2, and L3 to perform the optimized decision making.

3.3.2. Cross-layer routing protocol overview

The desired use case application of this work was to deploy an independent energy-aware ad hoc network in large remote areas where traditional communication infrastructure for networks such as 5G, WiFi, or LTE are not readily available. A prime example of such an application are large test and evaluation (T&E) ranges that are usually set up in remote places, collecting data from networks of remote installations, supporting rescue operations after a disaster where traditional communication infrastructure is unavailable, among others. In many of these cases the core requirements of a desired routing protocol are as follows,

- The solution must be easy to deploy in ad hoc manner such that nodes can move in and out of the network without effecting the overall network performance.
- The solutions should be able to operate in a distributed manner such that it can scale without incurring large overhead.
- Operates in an energy efficient manner to maximize network lifetime since it is challenging to change batteries regularly in a large remotely situated network.
- Solutions should be able to adapt to changing network traffic patterns and avoid congested paths by load balancing.

Keeping the above requirements in mind, the *diStributed Energy Efficient bacKpressure (SEEK)* routing algorithm [23,65] is leveraged in this work which utilizes the geographic information of nodes, differential queue backlog, residual battery energy, and link reliability to compute the optimal next hop. In this section, we will present a formal derivation of our utility function ($\mathcal{U}_{ij}$ with respect to wireless link $\mathfrak{L}_{(i,j)}$) between node i and node j and formulate the network optimization problem.

The utility function considers the following parameters associated with potential next-hop; (i) proximity to destination, (ii) differential queue backlog, (iii) residual battery energy, (iv) energy efficiency of the link, and (v) the corresponding link throughput. This information is gathered from traditional control packets like the Beacon packets. The Beacon packets will contain updated OAI. The energy efficiency of a given link can be expressed as follows [67],

$$\eta_{ij} = \frac{\mathrm{R}_b^{ij}\left(1-\mathrm{e}_b^{ij}\right)}{\mathrm{P}_{ij}} = \frac{\mathrm{G}_{ij}}{\mathrm{P}_{ij}} \tag{1}$$

where η_{ij} gives the measure of the number of bits successfully transmitted over $\mathfrak{L}_{(i,j)}$ per Joule of transmission energy. Here, G_{ij} is the corresponding goodput measure and e_b^{ij} is the bit error rate with respect to $\mathfrak{L}_{(i,j)}$, T_{ij} is the transmission strategy which includes choice of transmission data rate R_b^{ij} and transmit power P_{ij}.

Another key factor that needs to be considered in routing is the differential queue backlog ($\Delta Q_{ij} = \mathrm{q}_i - \mathrm{q}_j$) with respect to the source node (i) and next-hop (j) [20,68,69]. The queue backlog at the destination node is considered to be zero. Considering the queue backlog is necessary to mitigate congestion in the network and traditional back-pressure algorithms have been shown to be throughput optimal [68]. Since achieving maximum throughput is not the sole objective of SEEK algorithm, the differential backlog is just one parameter in our utility function. The effective progress made by a packet can be represented as $d_{is} - d_{js}$. Choosing nodes that provide larger progress implies fewer hops to the sink node which in turn could lead to smaller energy consumption. Finally, to ensure uniform depletion of energy per node, we need to consider the E_r^j (residual energy) of potential next hops [70]. Therefore, we define our utility function as follows,

$$\mathcal{U}_{ij} = \eta_{ij} \overbrace{\left(\frac{\max\left[\Delta Q_{ij},0\right]}{\mathrm{q}_i}\right)}^{\text{Differential Backlog}} \underbrace{\left(\frac{d_{is}-d_{js}}{d_{is}}\right)}_{\text{Effective Forward Progress}} \overbrace{\left(\frac{\mathrm{E}_r^j}{\mathrm{E}_0^j}\right)}^{\text{Relative Residual Energy}}, \forall j \in \mathbb{NB}_i \tag{2}$$

η_{ij} aims to improve the energy efficiency of the network and can be replaced by reliability when using constant power and modulation. It is also interesting to note that the maximum value of $\mathcal{U}_{ij} = \eta_{ij}$ when each of the three normalized terms is 1. This implies that each of the other terms penalizes the utility function based on the instantaneous value. For example, a small differential backlog ($\mathrm{q}_i - \mathrm{q}_j < \mathrm{q}_i$) will dampen the value of $\mathcal{U}_{ij}$. Both $d_{is} - d_{js}$ and E_r^j will have similar effects on $\mathcal{U}_{ij}$.

The objective of the network is to maximize the summation of $\mathcal{U}_{ij}$ for all possible links $\mathfrak{L}_{(i,j)}$ in order to maximize the overall energy efficiency of the network. This, in turn, will ensure reliable communication while maximizing the network lifetime (which is defined as the time when the first node in the network depletes its energy leading to a network hole). The optimization problem is subject to residual battery energy, queue backlog, bit error rate, and capacity constraints. This is formulated as Problem $\mathcal{P}_1$ shown below,

$$\mathcal{P}_1: \text{Given}: \mathcal{G}(\mathbb{N}_{net},\mathbb{E}),\ \mathbb{G},\ \mathbb{E}_r,\ \mathbb{Q}$$

$$\text{Find}: \mathbb{NH}^*, \mathbb{T}^*$$

$$\text{Maximize}: \sum_{i\in\mathbb{N}_{net}} \sum_{j\in\mathbb{NB}_i} \mathcal{U}_{ij} \tag{3}$$

subject to :

$$\mathrm{R}_b^{ij} \le C_{ij}, \quad \forall i \in \mathbb{N}_{net},\ \forall j \in \mathbb{NB}_i \tag{4}$$

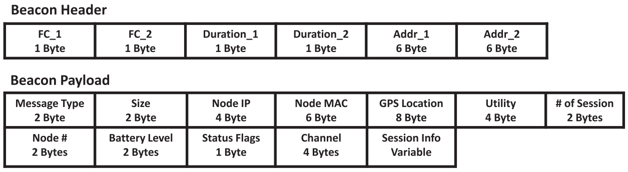
$$\mathrm{e}_b^{ij} > \mathrm{e}_{b*}^{ij}, \quad \forall i \in \mathbb{N}_{net},\ \forall j \in \mathbb{NB}_i \tag{5}$$

$$\mathrm{E}_r^i > 0, \quad \forall i \in \mathbb{N}_{net} \tag{6}$$

$$\mathrm{q}_i \ge 0, \quad \forall i \in \mathbb{N}_{net} \tag{7}$$

where the objective is to find the set of next-hop and transmission strategy for all nodes in the network which can be represented as $\mathbb{NH}^* = [\mathrm{NH}_i^*]$ and $\mathbb{T}^* = [\mathrm{T}_{ij}^*]$ respectively, $\forall i \in \mathbb{N}_{net}$ $j \in \mathbb{NB}_i$. In the above optimization problem $\mathcal{P}_1$, $\mathbb{G} = [\mathrm{G}_{ij}]$, $\mathbb{E}_r = [\mathrm{E}_r^i]$ and $\mathbb{Q} = [\mathrm{q}_i]$, $\forall i \in \mathbb{N}_{net}$, $\forall j \in \mathbb{NB}_i$ denote the set of goodput measure, residual battery energy and queue backlogs respectively. The constraint (4) restricts the total amount of data rate in link (i,j) to be lower than or equal to the physical link capacity. Constraint (5) imposeS that any transmission should guarantee the required BER. Finally, constraints (6) and (7) ensure the residual energy and queue backlog of each node will not have negative values. It can be seen that for solving the above optimization problem, nodes would require global knowledge of the network. This would require the centralized controller to maintain largeR network tables storing node parameters which will eventually flood with data as the network grows in size. Since the centralized optimization method is not a scalable solution, this motivates the need for a scalable distributed solution. We adopt SEEK which will operate in a distributed fashion and enable each node to find the next-hop based on the local information available to them. Each node with a packet to transmit chooses an optimal next-hop and transmission parameters such that it maximizes its own local utility function. This can be considered as a divide-and-conquer approach to solving the optimization problem in a distributed manner. Accordingly, every source node (i) will aim to maximize the utility function $\mathcal{U}_{ij}$ and select the optimal next-hop and transmission strategy as follows,

$$[j^*, \mathrm{T}_{ij}^*] = \arg\max_j\ \mathcal{U}_{ij}, \forall j \in \mathbb{NB}_i \tag{8}$$



Beacon Header

FC_1 1 Byte	FC_2 1 Byte	Duration_1 1 Byte	Duration_2 1 Byte	Addr_1 6 Byte	Addr_2 6 Byte

Beacon Payload

Message Type 2 Byte	Size 2 Byte	Node IP 4 Byte	Node MAC 6 Byte	GPS Location 8 Byte	Utility 4 Byte	# of Session 2 Bytes
Node # 2 Bytes	Battery Level 2 Bytes	Status Flags 1 Byte	Channel 4 Bytes	Session Info Variable		

Fig. 4. Beacon packet structure.

Each node will maintain a neighbor table with node parameters of its neighbors and will update the table as needed based on information from the beacon packets. The structure of the beacon packet is depicted in Fig. 4. with self-explanatory field names. The effectiveness of SEEK has been demonstrated in [65]. The emphasis of this work and contribution is not SEEK but the design and feasibility of maturing similar solutions for tactical applications. *Due to the modular implementation, just by changing a few lines of code that defines the utility function, one can reconfigure the stack to execute a new optimization objective.* This flexibility in adapting the routing metric is demonstrated in the code snippet in Listing 1.

```
double calculateUtility(neighborThroughputInfo
    & sourceNode,neighborThroughputInfo&
    destinationNode)
{
    // Compute reliability term
    double reliability = reliabilityMean();

    // Compute forward progress term
    double distanceToTarget=haversine_distance
    (sourceNode.latitude,sourceNode.longitude,
    destinationNode.latitude,destinationNode.
    longitude);
    double distanceToTargetFromNextHop =
    haversine_distance(latitude,longitude,
    destinationNode.latitude,destinationNode.
    longitude);
    double forwardProgress =((distanceToTarget
    -distanceToTargetFromNextHop)/
    distanceToTarget);

    // Compute buffer backlog term
12     double sourceBacklog=(double)sourceNode.
    backlog;
13     if(sourceBacklog==0)
14         sourceBacklog=1;
15
16     double nextHopBacklog = (double)backlog;
17     if(isInfoPotentiallyStale())
18         nextHopBacklog=0;
19
20     double minimumValue=0;
21     double diffBacklog=0;
22     if(nextHopBacklog>0)
23         minimumValue=1/nextHopBacklog;
24     double BacklogTerm=(sourceBacklog-
    nextHopBacklog)/sourceBacklog;
25     if(nextHopBacklog<sourceBacklog)
26         diffBacklog=BacklogTerm;
27     else
28         diffBacklog=std::max(BacklogTerm,
    minimumValue);
29
30     // Compute residual battery energy
31     double nextHopBattery = (double)battery;
32     double residualBattery = (nextHopBattery)
    /100;
33
34     // Compute utility term --- Routing metric
35     double routeUtility=reliability*
    forwardProgress*diffBacklog*
    residualBattery;
36
37     return routeUtility;}
```

Listing 1: Routing Utility Computation Snippet — Battery powered scenario

Here, line 35 shows the routing utility metric computation. This metric can be flexibly changed to include other route optimization metrics. For example, if all the nodes are powered by direct DC power supply and not on battery, then the routing metric can be modified to address this scenario as in Listing 2.

```
// Compute utility term --- Routing metric
double routeUtility=reliability*
    forwardProgress*diffBacklog;
```

Listing 2: Routing Utility Metric Modification in scenario when all nodes have DC power supply

Finally, for the benefit of the readers, we also provide Table 3 summarizing all the notations/symbols used in the discussion above.

3.3.3. L2-3 daemon

The outbound DATA packets that are handed over to the L2-3 daemon from the kernel module are queued in the *General Queue* awaiting the best route assignment. This is accomplished by the *Outbound* thread which continuously listens for incoming packets from the kernel module. The *SEEK* thread continuously performs route computation and assignment for the outbound packets in the *General Queue*. Following route assignment, the packets will await their transmission opportunity in the *Transmit Queue*. A third thread - *Channel Access* - performs CSMA/CA awaiting a clear-to-send (CTS) from its intended next-hop/destination to dequeue the segment (set of DATA packets) from the *Transmit Queue*. The segment for which the CTS was received will be forwarded to the FPGA via the L1 API for *over-the-air* transmission. It must be noted that if the intended next-hop for a segment is unresponsive, the segment will be returned back to the *General Queue* for rerouting. Additionally, a *Receive* thread continuously monitors for incoming packets from the FPGA, processes, and responds (such as with CTS or ACK) accordingly based on received packet type. A supplementary auxiliary thread also tracks DATA timeouts.

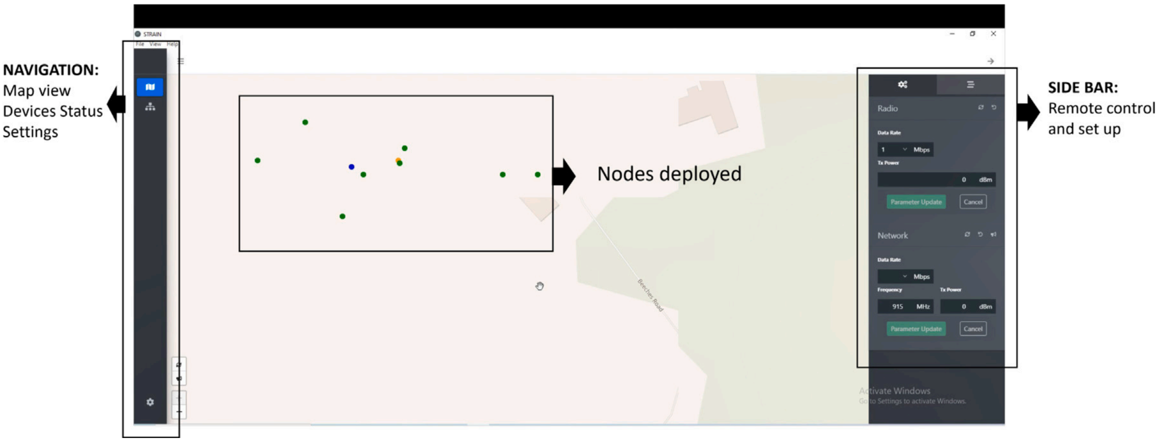
Table 3
Summary of notations.

Symbols	What it represents
$\mathfrak{L}(i,j)$	Wireless link between node i and node j
$\mathcal{U}_{ij}$	Cross-layer utility function representing the efficacy of wireless link $\mathfrak{L}(i,j)$
η_{ij}	Number of bits successfully transmitted over $\mathcal{L}(i,j)$ per Joule of transmission energy
G_{ij}	Goodput of wireless link $\mathfrak{L}(i,j)$
e_b^{ij}	Bit error rate of wireless link $\mathfrak{L}(i,j)$
e_{b*}^{ij}	Minimum acceptable bit error rate
R_b^{ij}	Data rate chosen for $\mathfrak{L}(i,j)$
P_{ij}	Transmit power chosen for $\mathfrak{L}(i,j)$
T_{ij}	Transmission strategy comprising of data rate and transmit power
q_i	Backlog of node i
ΔQ_{ij}	Differential Backlog
d_{is}	Distance between node i and destination s
E_r^j	Residual energy of node j
E_0^j	Total initial energy of node j
$\mathbb{NB}_i$	Set of neighbors of node i
$\mathcal{P}_1$	Optimization problem 1
$\mathcal{G}(N,E)$	Graph with nodes N and edges E
$\mathbb{N}_{net}$	Set of all nodes in the wireless network
$\mathbb{E}$	Set of all edges between nodes in the network
$\mathbb{E}_r$	Set of residual energies of all nodes in the network
$\mathbb{G}$	Set of all goodput measures
$\mathbb{Q}$	Set of all node backlogs
$\mathbb{NH}^*$	Set of all optimal next hop for the given time step
$\mathbb{T}^*$	Set of all optimal transmission parameters for the given time step
$\mathbb{NH}_i$	Set of all neighbors of node i
C_{ij}	Maximum channel capacity

3.3.4. X-n-APP deamon

The X-n-APP daemon handles two tasks; (i) preparing the beacon packet with the necessary information to be shared with immediate neighbors for distributed optimization, and (ii) handling packets for any application such as a graphical user interface (GUI) in this case. The beacon packets as well as the GUI requires location information to reflect the most recent coordinates on the GUI. We choose an efficient approach and access the GPS module from only one location - X-n-APP daemon — in the stack to avoid GPS pinging from multiple software locations. This retrieved GPS location information is subsequently populated in the *Beacon* packet. The GPS location extraction as well as beacon packet construction is carried out by the *Beacon* thread. The *Beacon* thread periodically sends these beacon packets to the lower layers for outbound transmission where the remaining fields such as residual battery and current buffer backlog are updated. Here, we note that, unlike the DATA, the beacon packet is treated as a control packet and is not queued in the general or transmit queues of the L2-3 daemon rather it is directly sent to the L1 FPGA for transmission.

The X-n-APP daemon also functions as a generalized application daemon for direct interfacing with an application such as a GUI. Both *ZMQ* and *APP receive* threads are designed to support such applications and hence can be customized based on the application at hand. Since the use case of this particular transceiver is remote deployment in test and evaluation ranges, a GUI application is developed to interface with the transceiver at a central command and control location. We emphasize that the remote monitoring and configuration (parameters like data rates, frequency, among others.) capability enabled by the GUI will ease the operator load by alleviating the need to physically travel



Radio

Data Rate: 1 Mbps

Tx Power: 0 dBm

Parameter Update   Cancel

Network

Data Rate: Mbps

Frequency: 915 MHz

Tx Power: 0 dBm

Parameter Update   Cancel

Fig. 5. Graphical User Interface (GUI) Application.

to the deployed locations. The functioning and design of the GUI has been described in detail in Section 3.4.

We further reemphasize that the software-defined stack is designed to be reconfigurable to modify the cross-layer routing as the requirements evolve in the future or depending on the desired network application. It is noteworthy that the utility function discussed above is representative of one such example of the algorithm where the network application desires energy-aware routing. This specific choice was application-specific.*The broad impact of this work is the reconfigurable nature of the software-defined stack such that it keeps evolving to meet the future requirements rapidly.*

3.4. Graphical User Interface Application

In this article, we provide an example of how an application such as a GUI can be configured for the network using the proposed software architecture. As described in the previous section, remote network (re-)configuration and monitoring are achieved with the GUI shown in Fig. 5 coupled with a central node (referred to as gateway node). The gateway node is the only node that directly communicates with the GUI (which runs on a host PC) via Ethernet. The GUI issues command packets to the gateway node which are then transmitted to the other nodes over the air. Multiple packet types are designed to issue specific commands. These packets are divided by their purpose and target (i.e. entire network or just a particular node). Essentially, each of the new packet types is associated with a command and can have a unicast or broadcast variant depending on the intended destination(s) of the packet.

The GUI uses a build system of Babel and Webpack. This ensures the GUI has a small footprint and is both faster to develop on and faster to run as an end-user. The GUI is split into two processes: *Main* and *Renderer*. The *Main* process handles the communication to the network (through the gateway node). The *Renderer* process handles user input and displaying output information that is used to interact with the GUI. When the user wants to issue a command to the network, they are required to enter the value they would like to change and click the button labeled *Update Parameter*. Once the *Update Parameter* button is clicked, the *Renderer* process of the GUI will create a control packet and send it to the *Main* process of the GUI. Upon receiving the control packet, the *Main* process will send it to the gateway node on a ZMQ Dealer/Dealer socket. The GUI is constantly listening to the Dealer socket for any packet sent from the network.

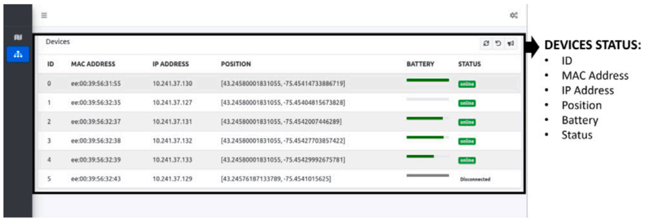
After receiving the control packet, the gateway node will extract the command and target fields of the packet and use them to determine what type of packet – broadcast or unicast – needs to be sent. For example, the command could be *Get Status* and the target could be all the nodes in the network which will cause the gateway node to respond by issuing the *Get Status* command for all nodes using a broadcast packet.

A broadcast packet is broadcasted for all nodes in range of the gateway node and then rebroadcast by each node that receives it for the first time. Upon receiving a broadcast packet, a node will process the packet, strip its header, and send the packet payload up to user-space for packet handling. A unicast packet is sent to a target node on the route determined by the SEEK routing daemon using the target IP field from the GUI's command packet. The reset command is utilized with two different packet types, *Reset Node* and *Reset Network*. The *Reset Network* packet is a broadcast packet and is used to force each node on the network to restart the protocol stack. Restarting involving safe closing and exiting of all active daemons and starting the software stack again. The *Reset Node* packet is a unicast packet that forces a specific target node to restart.

There are two packets that utilize the *Get Status* command, Request Network Status and Request Node Status. A Request Network Status packet is a broadcast packet that is sent upon the gateway receiving a *Get Status* command from the GUI that targets all nodes on the STRAIN network. The Request Node Status packet is a unicast packet that is sent upon the gateway node receiving a *Get Status* command for a target node. Upon receiving one of these packets from the gateway, the receiving node will send a *Response to Gateway* packet as an acknowledgment to the gateway.

A *Response to Gateway* packet will include OAI about the receiving node including the node's IP address and GPS location. Once the responses reach the gateway node, they are sent back to the GUI via the dealer/dealer ZMQ socket. The GUI will use this information to display the node's status. If after a set duration, a node's response does not reach the GUI, the status of the node will be set to *Disconnected* in the GUI. An example of the list of nodes in the network displaying its status can be seen below in Fig. 6. The battery bar being gray, indicates DC power in addition to battery power. No battery bar indicates only DC power, and a green bar is only battery power. The list of devices also includes action buttons that are *Get Network Status*, *Reset Network*, and *Announce Gateway* respectively. These are displayed in the upper right of the table.

Fig. 7 shows the *Message List* tab of the sidebar with the *Success* section toggled. It is used to view the status of packets sent from the GUI. The *Sent* section displays packets sent from the GUI. The *Pending* section displays packets that are yet to reach the gateway node. The *Success* section is where packets that have received a response appear. *All* indicates the packet sent was a broadcast packet intended for every node in the network. An*Announce Gateway* for *All* will be listed under success when the gateway successfully receives a response from any node in the network. An IP address, such as "10.241.37.131", indicates the packet was unicast. A *Get Status* for "10.241.37.131" will be listed

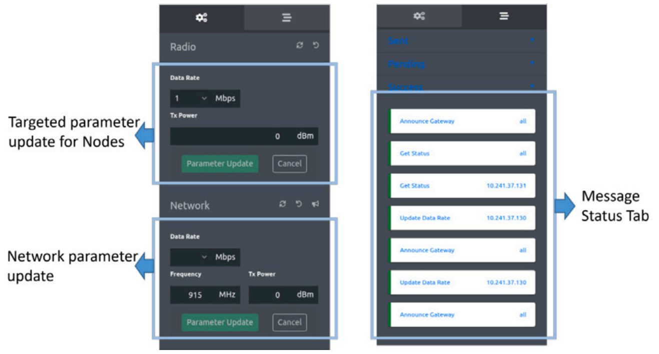


Devices

ID	MAC ADDRESS	IP ADDRESS	POSITION	BATTERY	STATUS
0	ee:00:39:56:31:55	10.241.37.130	[43.24580001831055, -75.45414733886719]		online
1	ee:00:39:56:32:35	10.241.37.127	[43.24580001831055, -75.45404815673828]		online
2	ee:00:39:56:32:37	10.241.37.131	[43.24580001831055, -75.4542007446289]		online
3	ee:00:39:56:32:38	10.241.37.132	[43.24580001831055, -75.45427703857422]		online
4	ee:00:39:56:32:39	10.241.37.133	[43.24580001831055, -75.45429992675781]		online
5	ee:00:39:56:32:43	10.241.37.129	[43.24576187133789, -75.4541015625]		Disconnected

DEVICES STATUS:
- ID
- MAC Address
- IP Address
- Position
- Battery
- Status

Fig. 6. GUI's device Status Page.



Radio

Data Rate: 1 Mbps

Tx Power: 0 dBm

Parameter Update | Cancel

Network

Data Rate: Mbps

Frequency: 915 MHz

Tx Power: 0 dBm

Parameter Update | Cancel

Sent

Pending

Success

Announce Gateway	all
Get Status	all
Get Status	10.241.37.131
Update Data Rate	10.241.37.130
Announce Gateway	all
Update Data Rate	10.241.37.130
Announce Gateway	all

Targeted parameter update for Nodes

Network parameter update

Message Status Tab

Fig. 7. Sidebar of the GUI.

Table 4
Summary of GUI Commands.

Application command	Description
Get Node Status	Gets the status of the selected node
Get Network Status	Gets the status of the entire networks
Announce Gateway	Announces the gateway node's information to all nodes connected on the network
Set Data Rate: Apply to Node	Sets the data rate of the selected node.
Set Frequency	Sets the frequency of the network
Set Transmit Attenuation	Sets the transmit power of a target node or the entire network
Reset Node	Forces a selected node to restart
Resent Node	Forces the entire network to restart

Table 5
Parameters Of Outdoor evaluation.

Parameters	Values
Data rates	1, 2, 5.5, 11 Mbps
Transmit Power	1–3.5 W (based on attenuation)
Center Frequency	430 MHz
Bandwidth	22 MHz
Payload Size	1000 Bytes
Segment Size	32 Packets

under *Success* once an acknowledgment from the node at IP address "10.241.37.131" is received at the Gateway. Fig. 7 also shows the *Config* tab of the sidebar in the GUI and is used to execute commands on a selected node. Fig. 5 shows the map view in the GUI which reflects the network topology. The gateway node is displayed as a blue dot to make it evident to the user.

Since the gateway node is the only node that can directly communicate with the GUI, the gateway must recognize every new node added to the network. Upon startup, each new node broadcasts a *New Node Announcement* packet to neighboring nodes requesting the gateway node IP. This request is broadcast until a neighboring node responds with the requested IP in a *New Node Announcement Response* packet. As a result, the new node and the neighboring nodes update the node list and neighbor lists. Once the new node has the gateway IP, it repeatedly sends a *Hello Gateway* packet until the new node is acknowledged by the gateway. Once the gateway receives this *Hello Gateway* packet, the gateway informs the GUI of the new node and sends a *Gateway New Node Acknowledgment* completing the addition of a new node to the network.

A *Set Network Parameter* packet is broadcast to all nodes in range of the gateway node. The three commands currently available with *Set Network Parameter* are *Update Data Rate*, *Update Tx Power*, and *Update Frequency* commands which are sent throughout the network starting from the gateway and being rebroadcast from the nodes until every node has received the command. The command *Update Data Rate* has four values associated with that can be sent through the network. It changes current data rate of every node in the network to the supported values. In this work, it was set to one of the four values, which are *1 Mbps*, *2 Mbps*, *5.5 Mbps*, and *11 Mbps*. The command *Update Tx Power* sets the attenuation for every node in the network based on the frontend supported values. Similarly, the *Update Frequency* command will set the frequency of the nodes to the specified value. A Set Node Parameter packet is unicast from the gateway node to a targeted node in the network. The commands currently available with Set Node Parameter are *Update Data Rate* and *Update Tx Power*. The *Update Data Rate* and the *Update Tx Power* command have the same possible values as above for Set Network Parameter.

An *Acknowledge Gateway* packet is transmitted by non-gateway nodes after the nodes receive a *Set Network Parameter* or a *Set Node Parameter* packet. The *Acknowledge Gateway* packet is a unicast packet from the non-gateway node to the destination (gateway) node. Its purpose is to acknowledge the gateway node after making the requested change with *Set Network Parameter* or *Set Node Parameter* packet. It is only created and sent after the change occurs. Summary of remote control and monitoring capabilities are provided in Table 4

4. Outdoor experiments and results

As part of the feasibility analysis, we performed extensive outdoor experiments to evaluate the performance of the implemented software-defined protocol stack in a realistic environment. Since the proposed solution is implemented on an e-SDR, it can be extended to any frequency of interest supported by the target hardware. To accomplish the experiments, we received a temporary experimental license from Federal Communications Commission (FCC) to utilize the 430 MHz frequency within a geographical area. The novelty and utility of the SEEK algorithm itself have been established in previous work [65,71]. Hence here the focus is to compare various settings and the impact on overall performance (throughput and reliability) of implementing cross-layer optimized algorithms on GPP supported with FPGA-based PHY layer for outdoor deployment. To the best of our knowledge, this is the first time such a comprehensive evaluation has been undertaken using deployment-ready cross-layer optimized e-SDRs. The default parameters are listed in the Table 5 unless otherwise specified.

4.1. Transmission range evaluation

The target deployment scenario for the device is remote test and evaluation sites, hence, the design goal for transmission range was to achieve up to 1 km. The setup of the transmitter and the key testing locations from our range testing experiments is shown in Fig. 8. Since the authorization to transmit was only for a limited geographical area, we redesigned the MAC protocol to ensure the receiving node does not transmit any control packet and just acts as a receiver for the purpose of range testing. This ensured that the receiver that needs to be outside the geographical locations does not transmit and just acts as a passive receiver to evaluate the reliability of the link from the transmitter. The rest of the software protocol stack was the same as the architecture discussed in Section 3. For each reported value in the Table 6, the

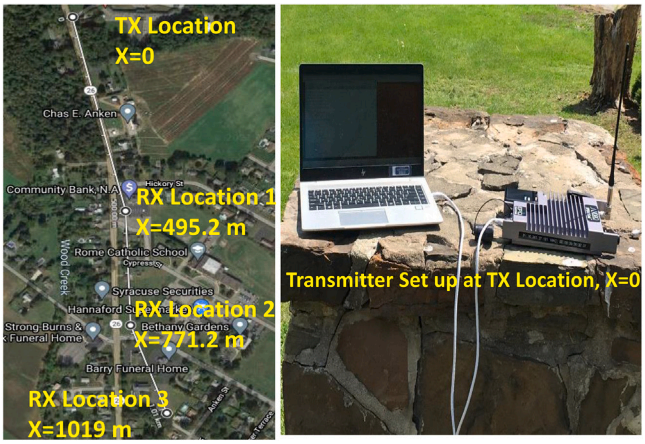


Fig. 8. Range testing setup and map-view of the locations.

Table 6
Link reliability for varying transmission ranges.

Data rate	Distance	Reliability
2 Mbps	495.2 m 771.2 m 1019 m	99.9% 99.77% 98.08%
5.5 Mbps	495.2 m 771.2 m 1019 m	99.62% 96.03% 97.16%
11 Mbps	495.2 m 771.2 m 1019 m	85.28% 31.56% 13.9%

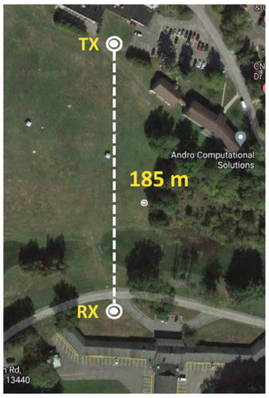


Fig. 9. Peer-to-Peer Topology.

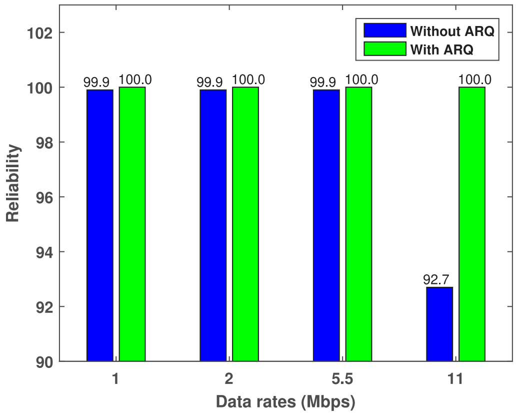


Fig. 10. Reliability vs data rates.

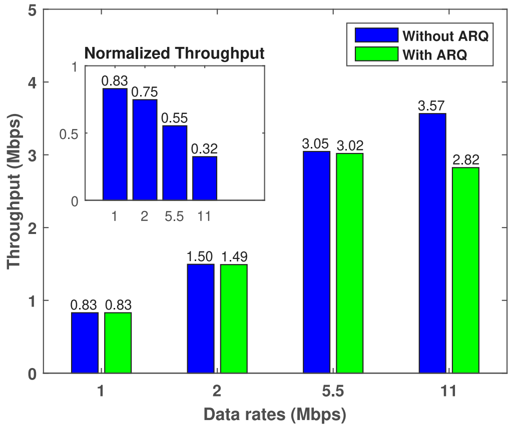


Fig. 11. Throughput vs data rates.

results were averaged over 10,000 packets transmitted in each run. *We define **reliability** as the percentage of packets received with respect to packets sent.*

Table 6 shows that the performance was consistently good for both 2 Mbps and 5.5 Mbps up to 1019 m but the performance of 11 Mbps degraded sooner. Due to the FCC license restrictions and urban environment, the next feasible point was at 1942 m from the transmitter at which point no packets were received. The performance degradation of 11 Mbps was attributed to sensitivity to lower signal-to-noise-ratio and multipath propagation effects. This was further substantiated since the performance did improve when we tested a smaller payload size. For example, reducing the payload size to 100 Bytes at a distance of 1019 m increased the reliability from 13.9% to 30%. Further, we also analyzed the Received Signal Strength Indicator (RSSI) values which provided us insight that the performance of the physical layer was deteriorating at RSSI of ∼ 15 dB or higher than that observed during wired in-laboratory experiments. This finding revealed there is room for improvement in sensitivity (∼ 15 dB) in future iterations of the FPGA PHY. Overall, the experimentation was successful as we hit our target distance of 1 km even at 5.5 Mbps.

4.2. Peer-to-peer experiments

As described earlier, the decision to implement the PHY on FPGA and the rest of the protocol stack was to achieve the best combination of high data rates and the required flexibility to implement cross-layer routing protocols. Yet, it is important to study the delays and associated overhead it takes to make the necessary calls between the GPP and FPGA to support packet transmission and how it impacts the data rates. To accomplish this, we perform outdoor peer-to-peer testing for varying data rates and payload sizes to analyze the impact. The outdoor locations of the transmitter and receiver are depicted in Fig. 9.

First, in Fig. 10, we study the reliability at the different data rates with and without automatic repeat request (ARQ) enabled at the MAC layer. It can be clearly seen that the transceiver has high reliability close to 100% for 1, 2, and 5.5 Mbps even without ARQ enabled. At 11 Mbps the reliability decreases to 92% without ARQ and returns to 100% with ARQ by trading off throughput. Fig. 11 depicts how the peer-to-peer throughput varies at different PHY data rates 1, 2, 5.5, 11 Mbps. *The **throughput** in this case only considered the payload of the packets and does not include the headers or the control packets*. The throughput considered here is also referred to as goodput in some literature. The figure also depicts the ***normalized throughput** which is defined as the ratio between throughput and data rate*. The design choice provides solid performance at 1 and 2 Mbps especially considering it depicts the goodput achieved.

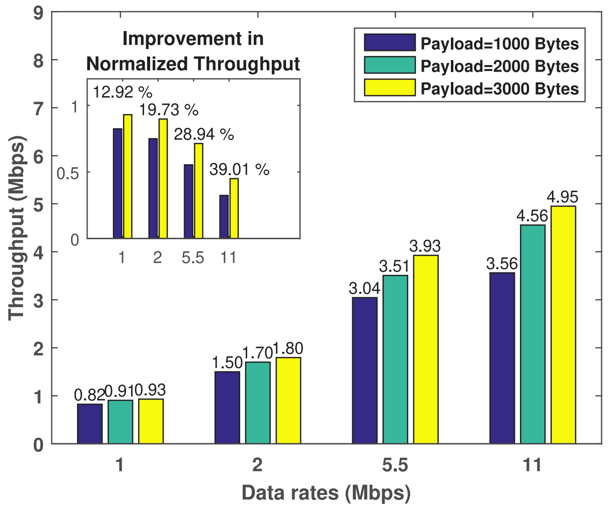


Fig. 12. Throughput for varying payload.

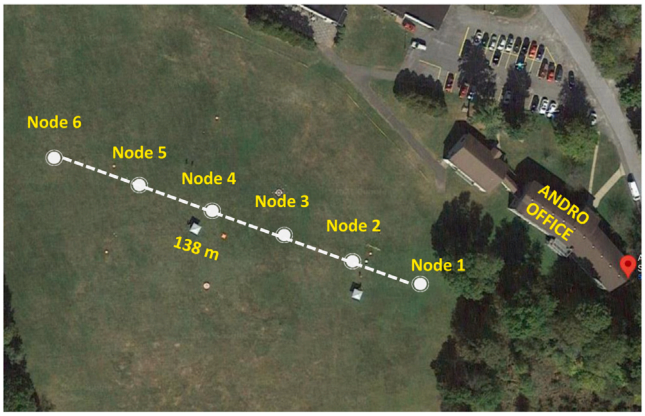


Fig. 13. 6-node line network topology.

The impact of the overhead becomes visible as the data rate increases to 5.5 and 11 Mbps. This hypothesis is further substantiated in our experiments that show an increase in normalized throughput with the increase in the payload as shown in Fig. 12. The normalized throughput increased by up to 39% for 11 Mbps when the payload was increased from 1000 Bytes to 3000 Bytes.

4.3. 6-Node line network

In the next set of experiments, we examined the impact of the number of hops on normalized throughput and the reliability of the network. The line network topology is shown in Fig. 13. In this topology only the nodes adjacent to each other are neighbors. Hence five hops are required from node 1 to node 6. The experiments were conducted with and without ARQ. To ensure a line network in the limited geographical area authorized for experiments by the FCC, we had to set neighbor tables in software. This also implied that all nodes will cause interference to each other which also serves as a good test for ad hoc channel access capabilities/performance. The results followed exactly the hypothesized pattern in the case of normalized throughput. In the single-hop a normalized throughput of 0.83 and all the other measures for hops 2, 3, 4, 5 followed close to 1/2, 1/3, 1/4, and 1/5 of the normalized throughput of the single-hop case as shown in Fig. 14. This implied that the nodes were utilizing (sharing) the spectrum efficiently without too much loss due to collision. As shown in Fig. 15, the reliability also seemed to confer with this finding as in most cases reliability was over 95%. The lowest reliability was still 93.9% for the 5-hop case even when the ARQ was disabled. All the reliability was back to 100% when ARQ was enabled. There was only a marginal drop in normalized throughput when ARQ was enabled which was mainly attributed to the link being very reliable even without ARQ leading to only a few retransmissions even when ARQ was enabled.

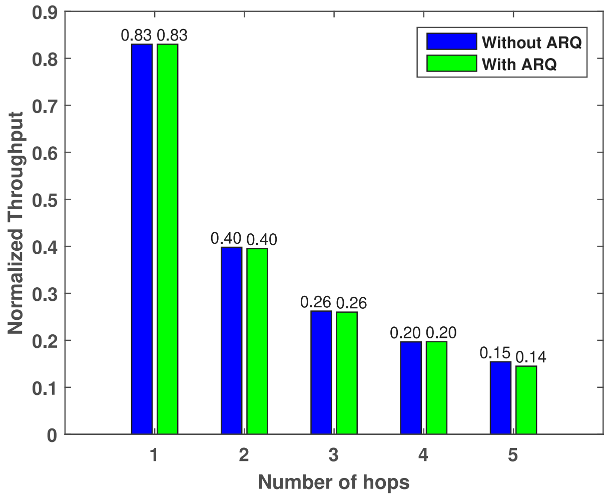


Fig. 14. Line network throughput analysis.

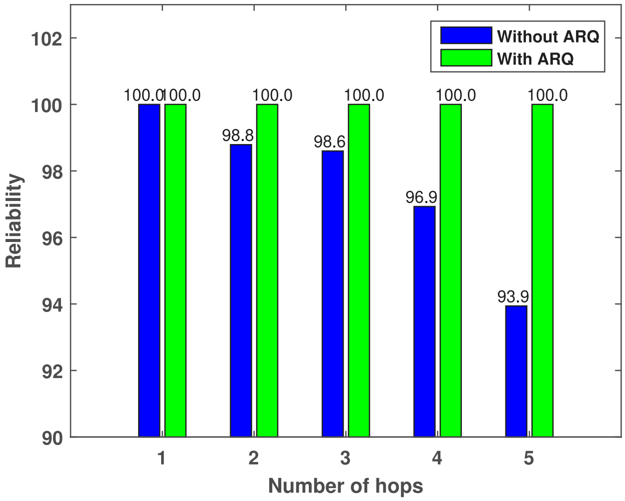


Fig. 15. Line Network Reliability Analysis.

4.4. 5-Node network experiment — dynamic routing

For timely evaluation, we designed a specific experiment such that we can determine the effectiveness of the implementation of SEEK algorithm to dynamic changes in the network. To accomplish this, we consider a topology shown in Fig. 16. The source *S1* continuously transmits packets destined for the gateway node *GW*. *S1* chooses the appropriate relay among *R1*, *R2*, and *R3* based on SEEK algorithm. We plot the average packets received from each relay node every 10 s interval. The values are averaged using a moving window of 60 s duration to get smoother curves.

In the first part of Fig. 18 (blue shade) depicts a scenario where the relay nodes (i.e. nodes with possible forward progress for *S1*) have approximately uniform parameters. As expected, in this scenario, the traffic is evenly distributed between the three relays. Next, we significantly increase the backlog of the *R1* to emulate a congestion

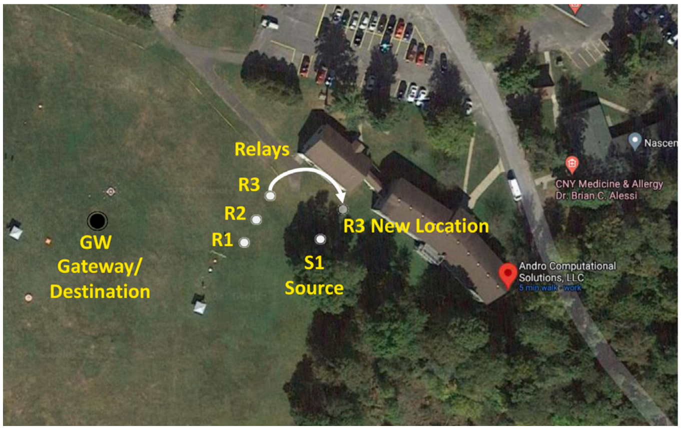


Fig. 16. 5-node Outdoor Topology.

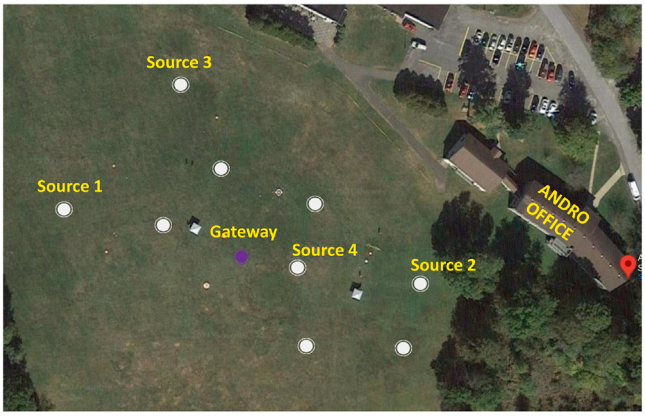


Fig. 17. 10-node Outdoor Topology.

scenario at a relay node. As it can be seen (yellow shade), *S1* learns to avoid *R1* and routes the packets through the other two options, *R2* and *R3*. At the beginning of the experiment, *R2* was set up with a battery with low residual energy but was connected to a DC power supply. When a node is connected to a DC power supply, it is equivalent to having the battery completely charged with full residual energy. Now in the next part of the evaluation, the residual energy of *R2* is rapidly dropped by removing the DC power supply and the residual energy in the battery was close to 10%. As it can be seen, *S1* recognizes this in the next part (yellow shade) and prefers *R3* so that lifetime of the network can be extended. We remind the readers that the lifetime of the network is defined as the duration of network operation till the first node in the network is completely depleted of residual energy. Therefore, the nodes use the utility function described earlier in Section 3 trying to jointly consider multiple parameters including residual energy. Therefore, one goal is to distribute traffic over multiple paths so that nodes in the network can share the load based on available energy resources to extend the lifetime of the network. This is the exact intuitive behavior observed here in this region of Fig. 18. Thereafter, in the next part, we now move the *R3* node behind the source *S1* as shown in Fig. 16 such that it is no more a feasible next-hop that provides forward progress. This information is quickly realized by the source *S1* through the beacon packets as in all the previous cases. In this situation, with *R1* congested, *R3* is no more the possible choice for forward progress, *S1* must return to *R2* with a lower battery since that is the only possible choice to relay the packets to the destination (*GW*). Hence in the red region of Fig. 18, it can be seen that the traffic through *R2* picks up while the traffic through *R3* drops. Another interesting observation is that when the R3 initially changes location it still has some packets remaining for the *GW*, and *GW* is still within its transmission range. Accordingly, after the initial drop in traffic, it still is able to get those packets to the *GW* which can also be seen as a spike in the total traffic momentarily to overcome the dip it experienced a few moments prior. In the final section of Fig. 18, the backlog of *R1* is cleared. Now, in this case, the *S1* realizes that it does not have to use the *R2* with lower residual energy anymore and can share the traffic load through *R1*. As a result, it can be seen that now traffic through *R1* increases while the traffic through *R2* reduces. *It is also interesting to see that even with all these changes, the total packets/s at the GW (represented at the "total" legend in Fig. 18) remains very stable demonstrating the rapid adaptability of the cross-layer optimized nodes.* In this experiment, we have demonstrated the gamut of routing decisions the nodes can make in a distributed manner by just gathering information from their immediate neighbors demonstrating the effectiveness of the cross-layer optimized routing using an intuitively representative experiment.

4.5. 10-Node network experiments — network capacity

In this experiment, we evaluate how the network handles the increasing number of sessions and source (packet generation) rate. The 10-node topology is shown in Fig. 17. The experiments were conducted at 1 Mbps of data rate, the source rate was varied from $[10, 20, 40, 80]$ packets/s, and the number of sessions (independent sources generating traffic to the gateway node) was set to 1, 2 and 4. This is motivated by a typical use case for our intended deployment scenario. As expected, Fig. 19 demonstrates that the normalized throughput increases and gets saturated both when the source rate increases and the number of sessions increase. There is no drop in performance even when there are multiple sessions (4) with high source rates (80) and the normalized throughput is maintained at the saturated value (~ 0.6). This shows the efficiency of the network in handling multiple traffics and session rates as expected even when the traffic increases.

5. Conclusion and future direction

This article introduces the first-known fieldable cross-layer optimized, embedded software-defined radio (e-SDR) that has been designed, developed, and matured to provide high throughput and reliability by implementing the protocol stack (except the physical layer) on an embedded ARM processor. To emphasize the contribution of the article, we survey the literature to highlight that the majority of cross-layer optimized techniques have been either limited to simulations or extended to hardware testbeds. We point out some of the key hurdles that might have led to the halt in the maturity of these solutions. The survey also demonstrates the large scale of applicable domains for cross-layer optimization if the solutions are designed and matured to be deployed in real-life applications.

First, to overcome the hardware-related challenges, we discuss how a COTS e-SDR can be customized and re-configured for deploying cross-layer protocol stack. We then provide a detailed discussion on the software architecture and design choices that ensured the feasibility of such an endeavor. Ensuring the right balance between modular and flexible architecture along with efficiency is the key takeaway for this discussion. Most importantly, we have demonstrated through extensive outdoor field experiments the capability, range, and dynamic routing under varying network conditions.

In the future, cross-layer optimized networking can be a key enabler for several applications in IoT, wireless sensor networks, 5G, and beyond. For example, device-to-device communication can be enhanced by cross-layer optimization that could be implemented on resource constrained edge platforms. Additionally, several of the tactical applications rely on ad hoc networking to provide the much needed flexibility during tactical operations. We hope this successful demonstration of cross-layer optimized e-SDR will provide directions for future cross-layer optimized solutions to benefit tactical and commercial applications.

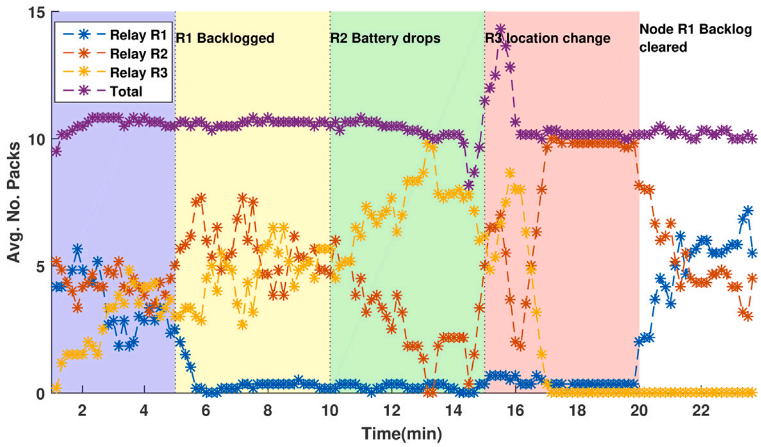


Fig. 18. 5-node Experiment (dynamic routing).

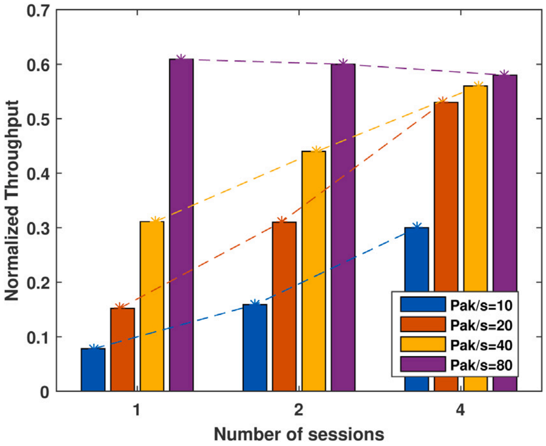


Fig. 19. 10-node Experiment.

CRediT authorship contribution statement

Jithin Jagannath: Made equal contributions and shared responsibility as the first authors and PI/Co-PI of the effort leading the design and development process, Supported by Andrew Drozd throughout the duration of the effort. **Anu Jagannath:** Made equal contributions and shared responsibility as the first authors and PI/Co-PI of the effort leading the design and development process, Supported by Andrew Drozd throughout the duration of the effort. **Justin Henney:** Part of the core team that performed development and testing of the solutions. **Tyler Gwin:** Part of the core team that performed development and testing of the solutions. **Zackary Kane:** Part of the core team that performed development and testing of the solutions. **Noor Biswas:** Part of the core team that performed development and testing of the solutions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment and disclaimer

The authors would like to thank John Orlando, Jeff Porter of Epiq Solutions for their support, Raymond Shaw of Spectrum Bullpen for help with Spectrum Supportability Risk Assessments, and Dan O' Connor of ANDRO Computational Solutions for his help during the outdoor testing.

(a) Contractor acknowledges the Government's support in the publication of this paper. This material is based upon work supported by the US Army Contract No. W15P7T-20-C-0006. (b) Any opinions, findings, and conclusions or recommendation expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Army.

References

[1] M. Katz, M. Matinmikko-Blue, M. Latva-Aho, 6Genesis flagship program: Building the bridges towards 6G-enabled wireless smart society and ecosystem, in: Proc. of IEEE Latin-American Conference on Communications, LATINCOM, 2018, pp. 1–9.

[2] Y. Zhao, J. Zhao, W. Zhai, S. Sun, D. Niyato, K.-Y. Lam, A survey of 6G wireless communications: Emerging technologies, 2020, arXiv, arXiv:2004.08549.

[3] A. Jagannath, J. Jagannath, T. Melodia, Redefining wireless communication for 6G: Signal processing meets deep learning, IEEE Trans. Artif. Intell. 2 (6) (2021) 528–536.

[4] E. Calvanese Strinati, S. Barbarossa, J.L. Gonzalez-Jimenez, D. Ktenas, N. Cassiau, L. Maret, C. Dehos, 6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication, IEEE Veh. Technol. Mag. 14 (3) (2019) 42–50.

[5] G. Berardinelli, N.H. Mahmood, I. Rodriguez, P. Mogensen, Beyond 5G wireless IRT for industry 4.0: Design principles and spectrum aspects, in: Proc. of IEEE Globecom Workshops, 2018, pp. 1–6.

[6] S. Dang, O. Amin, B. Shihada, M.-S. Alouini, What should 6G be? Nat. Electron. 3 (2019) 20–29.

[7] W. Saad, M. Bennis, M. Chen, A vision of 6G wireless systems: Applications, trends, technologies, and open research problems, IEEE Netw. 34 (3) (2020) 134–142.

[8] L. Bonati, M. Polese, S. D'Oro, S. Basagni, T. Melodia, Open, programmable, and virtualized 5G networks: State-of-the-art and the road ahead, Comput. Netw. 182 (2020) 1–28.

[9] B. Fu, Y. Xiao, H. Deng, H. Zeng, A survey of cross-layer designs in wireless networks, IEEE Commun. Surv. Tutor. 16 (1) (2014) 110–126.

[10] F. Foukalas, V. Gazis, N. Alonistioti, Cross-layer design proposals for wireless mobile networks: A survey and taxonomy, Commun. Surveys Tuts. 10 (1) (2008) 70–85.

[11] V. Srivastava, M. Motani, Cross-layer design: a survey and the road ahead, IEEE Commun. Mag. 43 (12) (2005) 112–119.

[12] I. Al-Anbagi, M. Erol-Kantarci, H.T. Mouftah, A survey on cross-layer quality-of-service approaches in WSNs for delay and reliability-aware applications, IEEE Commun. Surv. Tutor. 18 (1) (2016) 525–552.

[13] X. Lin, N.B. Shroff, R. Srikant, A tutorial on cross-layer optimization in wireless networks, IEEE J. Sel. Areas Comms 24 (8) (2006) 1452–1463.

[14] S. Shakkottai, T. Rappaport, P. Karlsson, Cross-layer design for wireless networks, IEEE Commun. Mag. 41 (10) (2003) 74–80.

[15] J. Jagannath, A. Jagannath, Z. Henney, N. Biswas, T. Gwin, A. Drozd, Fieldable cross-layer optimized embedded software defined radio is finally here!, in: Proc. of IEEE Military Communications Conference, MILCOM, 2021.

[16] J. Jagannath, Cross-layer Optimized Networking for Next-generation 5G Ad Hoc Networks (Ph.D. thesis), Northeastern University, 2019.

[17] L.D. Mendes, J.J. Rodrigues, A survey on cross-layer solutions for wireless sensor networks, J. Netw. Comput. Appl. 34 (2) (2011) 523–534.

[18] X. Wang, K. Kar, Cross-layer rate optimization for proportional fairness in multihop wireless networks with random access, IEEE J. Sel. Areas Comms 24 (2006) 1548–1559.

[19] I. Nosheen, S.A. Khan, F. Khalique, A mathematical model for cross layer protocol optimizing performance of software-defined radios in tactical networks, IEEE Access 7 (2019) 20520–20530.

[20] J. Jagannath, T. Melodia, A. Drozd, DRS: DIstributed deadline-based joint routing and spectrum allocation for tactical ad-hoc networks, in: Proc. of IEEE Global Communications Conference, GLOBECOM, Washington, DC USA, 2016.

[21] T. Zhu, X. Chen, L. Chen, W. Wang, G. Wei, GCLR: GNN-based cross layer optimization for multipath TCP by routing, IEEE Access 8 (2020) 17060–17070.

[22] M.J. Herrmann, G.G. Messier, Cross-layer lifetime optimization for practical industrial wireless networks: A petroleum refinery case study, IEEE Trans. Ind. Inf. 14 (8) (2018) 3559–3566.

[23] J. Jagannath, S. Furman, A. Jagannath, A. Drozd, Energy efficient ad hoc networking devices for off-the-grid public safety networks, in: Proc. of IEEE Consumer Communications & Networking Conference, CCNC, Las Vegas, NV, USA, 2019.

[24] S. Barmpounakis, N. Maroulis, M. Papadakis, G. Tsiatsios, D. Soukaras, N. Alonistioti, Network slicing - enabled RAN management for 5G: Cross layer control based on SDN and SDR, Comput. Netw. 166 (2020) 106987.

[25] L. Zhang, X. Zhou, Joint cross-layer optimised routing and dynamic power allocation in deep space information networks under predictable contacts, IET Commun. 7 (5) (2013) 417–429.

[26] A. Jagannath, J. Jagannath, A. Drozd, Artificial intelligence-based cognitive cross-layer decision engine for next-generation space mission, in: Proc. of IEEE Cognitive Communication for Aerospace Applications, CCAA Workshop, Cleveland, OH, USA, 2019.

[27] Y. Zhou, H. Yang, Y.-H. Hu, S.-Y. Kung, Cross-layer network lifetime maximization in underwater wireless sensor networks, IEEE Syst. J. 14 (1) (2020) 220–231.

[28] D. Pompili, I.F. Akyildiz, A multimedia cross-layer protocol for underwater acoustic sensor networks, IEEE Trans. Wireless Commun. 9 (9) (2010) 2924–2933.

[29] L.-C. Kuo, T. Melodia, Cross-layer routing on MIMO-ofdm underwater acoustic links, in: 2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON, IEEE, 2012, pp. 227–235.

[30] J. Liu, M. Yu, X. Wang, Y. Liu, X. Wei, J. Cui, RECRP: An underwater reliable energy-efficient cross-layer routing protocol, Sensors 18 (12) (2018) 4148.

[31] M.C. Vuran, I.F. Akyildiz, Cross-layer packet size optimization for wireless terrestrial, underwater, and underground sensor networks, in: IEEE INFOCOM 2008-the 27th Conference on Computer Communications, IEEE, 2008, pp. 226–230.

[32] J.M. Jornet, M. Stojanovic, M. Zorzi, On joint frequency and power allocation in a cross-layer protocol for underwater acoustic networks, IEEE J. Ocean. Eng. 35 (4) (2010) 936–947.

[33] N. Cen, J. Jagannath, S. Moretti, Z. Guan, T. Melodia, LANET: VIsible-light ad hoc networks, Ad Hoc Netw. 84 (2019).

[34] M.S. Demir, M. Uysal, A cross-layer design for dynamic resource management of VLC networks, IEEE Trans. Commun. 69 (3) (2021) 1858–1867.

[35] J. Jagannath, T. Melodia, VL-ROUTE: A Cross-layer routing protocol for visible light ad hoc network, in: Proc. of IEEE Symp. on a World of Wireless, Mobile, and Multimedia Network, WoWMoM, Washington D.C., USA, 2019.

[36] N.T. Dang, V.V. Mai, A PHY/MAC cross-layer analysis for IEEE 802.15. 7 uplink visible local area network, IEEE Photonics J. 11 (3) (2019) 1–17.

[37] J. Jagannath, T. Melodia, An opportunistic medium access control protocol for visible light ad hoc networks, in: Proc. of International Conference on Computing, Networking and Communications, ICNC, Maui, Hawaii, USA, 2017.

[38] L. Ding, T. Melodia, S. Batalama, J. Matyjas, M. Medley, Cross-layer routing and dynamic spectrum allocation in cognitive radio ad hoc networks, IEEE Trans. Veh. Technol. 59 (2010) 1969–1979.

[39] L. Ding, K. Gao, T. Melodia, S. Batalama, D. Pados, J. Matyjas, All-spectrum cognitive networking through jointly optimal distributed channelization and routing, IEEE Trans. Wireless Commun. 12 (11) (2013) 5394–5405.

[40] C. She, C. Yang, T.Q.S. Quek, Cross-layer optimization for ultra-reliable and low-latency radio access networks, IEEE Trans. Wireless Commun. 17 (1) (2018) 127–141.

[41] J. Jagannath, S. Furman, T. Melodia, A. Drozd, Design and experimental evaluation of a cross-layer deadline-based joint routing and spectrum allocation algorithm, IEEE Trans. Mob. Comput. 18 (8) (2019) 1774–1788.

[42] E.-C. Park, D.-Y. Kim, H. Kim, C.-H. Choi, A cross-layer approach for per-station fairness in TCP over WLANs, IEEE Trans. Mob. Comput. 7 (7) (2008) 898–911.

[43] C. Xu, Z. Li, J. Li, H. Zhang, G.-M. Muntean, Cross-layer fairness-driven concurrent multipath video delivery over heterogeneous wireless networks, IEEE Trans. Circuits Syst. Video Technol. 25 (7) (2015) 1175–1189.

[44] C. Dai, W. Li, L. Wang, M. Song, Y. Zhang, J. Xiao, A cross layer routing protocol for green communication network, in: IET International Conference on Communication Technology and Application, ICCTA 2011, 2011, pp. 910–914.

[45] H.B. Salameh, M.B. Irshaid, A.A. Ajlouni, M. Aloqaily, Energy-efficient cross-layer spectrum sharing in CR green IoT networks, IEEE Trans. Green Commun. Netw. (2021) 1.

[46] M. Al-Jemeli, F.A. Hussin, An energy efficient cross-layer network operation model for ieee 802.15.4-based mobile wireless sensor networks, IEEE Sens. J. 15 (2) (2015) 684–692.

[47] D. Xue, E. Ekici, Delay-guaranteed cross-layer scheduling in multihop wireless networks, IEEE/ACM Trans. Netw. 21 (6) (2013) 1696–1707.

[48] X. Liu, K. Hua, Z. Chen, A.S. Alghamdi, M.N. Ali, An efficient cross-layer approach for throughput-maximal and delay-minimal green vehicular networks, in: 2018 International Conference on Computing, Networking and Communications, ICNC, 2018, pp. 652–658.

[49] I. Jemili, D. Ghrab, A. Belghith, M. Mosbah, Cross-layer adaptive multipath routing for multimedia wireless sensor networks under duty cycle mode, Ad Hoc Netw. 109 (2020) 102292, URL https://www.sciencedirect.com/science/article/pii/S157087052030651X.

[50] M. Cheng, Q. Ye, L. Cai, Cross-layer schemes for reducing delay in multihop wireless networks, IEEE Trans. Wireless Commun. 12 (2) (2013) 928–937.

[51] H. Xiong, R. Li, A. Eryilmaz, E. Ekici, Delay-aware cross-layer design for network utility maximization in multi-hop networks, IEEE J. Sel. Areas Commun. 29 (5) (2011) 951–959.

[52] I. Jemili, D. Ghrab, A. Belghith, M. Mosbah, S. Al-Ahmadi, Cross-layer multipath approach for critical traffic in duty-cycled wireless sensor networks, J. Netw. Comput. Appl. 191 (2021) 103154, URL https://www.sciencedirect.com/science/article/pii/S1084804521001685.

[53] S.R. Lahane, K.N. Jariwala, Secured cross-layer cross-domain routing in dense wireless sensor network: A new hybrid based clustering approach, Int. J. Intell. Syst. 36 (2021) 3789–3812.

[54] American National Standard ANSI/ISA-100.11a-2011, ANSI/ISA-100.11A-2011 wireless systems for industrial automation: Process control and related applications, 2011.

[55] D. Chen, M. Nixon, A. Mok, WirelessHART: Real-Time Mesh Network for Industrial Automation, first ed., Springer Publishing Company, Incorporated, 2010.

[56] P. Shome, M. Yan, S.M. Najafabad, N. Mastronarde, A. Sprintson, Crossflow: A cross-layer architecture for SDR using SDN principles, in: NFV-SDN (Ed.), 2015 IEEE Conference on Network Function Virtualization and Software Defined Network, 2015, pp. 37–39.

[57] E. Demirors, G. Sklivanitis, T. Melodia, S.N. Batalama, RcUBe: Real-time reconfigurable radio framework with self-optimization capabilitites, in: Proc. of IEEE Intl. Conf. on Sensing, Communication, and Networking, SECON, Seattle, WA, 2015.

[58] J. Jagannath, H. Saarinen, W. Timothy, J. O'Brien, S. Furman, T. Melodia, A. Drozd, Combat: Cross-layer based testbed with analysis tool implemented using software defined radios, in: Proc. of IEEE Conf. on Military Comm., MILCOM, Baltimore, MD, 2016.

[59] S. Soltani, Y.E. Sagduyu, S. Scanlon, Y. Shi, J.H. Li, J. Feldman, J.D. Matyjas, Cross-layer design and SDR implementation of DSA, backpressure routing and network coding, 2019, arXiv preprint arXiv:1912.05658.

[60] G. Sklivanitis, E. Demirors, A. Gannon, S.N. Batalama, D.A. Pados, S.N. Batalama, All-spectrum cognitive channelization around narrowband and wideband primary stations, in: Proc. of IEEE Global Communications Conf., GLOBECOM, San Diego, CA, 2015.

[61] G. Kakkavas, K. Tsitseklis, V. Karyotis, S. Papavassiliou, A software defined radio cross-layer resource allocation approach for cognitive radio networks: From theory to practice, IEEE Trans. Cogn. Commun. Netw. 6 (2) (2020) 740–755.

[62] ORCA-Orchestration and reconfiguration control architecture, 2021, URL https://www.orca-project.eu/. (Accessed August 2021).

[63] Iris-the reconfigurable radio testbed, 2021, URL http://iristestbed.eu/. (Accessed August 2021).

[64] Open-access research testbed for next-generation wireless networks (ORBIT), 2021, URL https://www.orbit-lab.org/. (Accessed August 2021).

[65] J. Jagannath, S. Furman, A. Jagannath, L. Ling, A. Burger, A. Drozd, HELPER: HEterogeneous efficient low power radio for enabling ad hoc emergency public safety networks, Ad Hoc Netw. (Elsevier) 89 (2019) 218–235.

[66] Sidekiq-Z2, epiq solutions, 2021, URL https://epiqsolutions.com/static/Epiq-Solutions-Sidekiq-Z2-8a9337a3ad0d6387032e52c2db623625.pdf. (Accessed 19 July 2021).

[67] X. Chen, Y. Xu, A. Liu, Cross layer design for optimizing transmission reliability, energy efficiency, and lifetime in body sensor networks, Sensors 17 (4) (2017) 900.

[68] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, in: Proc. of IEEE Conference on Decision and Control, CDC, Honolulu, HI, 1990.

[69] S. Liu, E. Ekici, L. Ying, Scheduling in multihop wireless networks without back-pressure, IEEE/ACM Trans. Netw. 22 (5) (2014) 1477–1488.

[70] K. Hao, Z. Jin, H. Shen, Y. Wang, An efficient and reliable geographic routing protocol based on partial network coding for underwater sensor networks, Sensors 15 (6) (2015) 12720–12735.

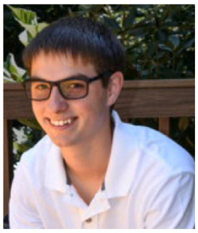
[71] J. Jagannath, A.L. Drozd, A. Jagannath, S.R. Furman, A.K. Burger, Routing data through distributed communications network, Google Patents, US Patent 10, 721, 162, 2020.

Jithin Jagannath (SM-IEEE'19) is the Chief Technology Scientist and Founding Director of the Marconi-Rosenblatt AI/ML Innovation Lab at ANDRO Computational Solutions. He is also the Adjunct Assistant Professor in the Department of Electrical Engineering at the University at Buffalo, State University of New York. Dr. Jagannath received his B. Tech in Electronics and Communication from Kerala University; M.S. degree in Electrical Engineering from University at Buffalo, The State University of New York; and received his Ph.D. degree in Electrical Engineering from Northeastern University. He is an IEEE Senior member and serves as IEEE Industry DSP Technology Standing Committee member. He also serves on the Federal Communication Commission's (FCC) Communications Security, Reliability, and Interoperability Council (CSRIC VIII) Working Group 1. Dr. Jagannath was the recipient of the 2021 IEEE Region 1 Technological Innovation Award with the citation, "For innovative contributions in machine learning techniques for the wireless domain".

Dr. Jagannath heads several of the ANDRO's research and development projects in the field of Beyond 5G, signal processing, RF signal intelligence, cognitive radio, cross-layer ad-hoc networks, Internet-of-Things, AI-enabled wireless, and machine learning. He has been the lead and Principal Investigator (PI) of several multi-million dollar research projects. This includes a Rapid Innovation Fund (RIF) and several Small Business Innovation Research (SBIR)s for several customers including the U.S. Army, U.S Navy, Department of Homeland Security (DHS), United States Special Operations Command (SOCOM). He is currently leading several teams developing commercial products such as SPEARLink™, DEEPSpec™ among others. He is the co-inventor of 11 U.S. Patents (granted, pending, and provisional). He has been invited to give various talks including Keynote on the topic of machine learning and Beyond 5G wireless communication. He has been invited to serve on the Technical Program Committee for several leading technical conferences.

Anu Jagannath (SM-IEEE'21) received her MS degree (2013) from State University of New York at Buffalo in Electrical Engineering. She is currently a Senior Scientist at ANDRO Computational Solutions, LLC and serves as the Associate Director of Marconi-Rosenblatt AI/ML Innovation Lab at ANDRO. Her research focuses on MIMO communications, Deep Machine Learning, Reinforcement Learning, Adaptive signal processing, Software Defined Radios, spread spectrum systems, LPI/LPD communications, spectrum sensing, adaptive Physical layer, and cross-layer techniques, medium access control and routing protocols, underwater wireless sensor networks, signal intelligence and so on. She serves as author and coauthor for book chapters and multiple research publications in journals and conference proceedings. She has rendered her reviewing service for conferences such as IEEE Annual Consumer Communications & Networking Conference (CCNC) and IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). She is the co-Principal Investigator (co-PI) and Technical Lead in multiple Rapid Innovation Fund (RIF) and SBIR/STTR efforts involving developing embedded MIMO solutions, deep and reinforcement learning for wireless communications, signal intelligence, and mesh networking.

Justin Henney received his BS from the Rochester Institute of Technology in Electrical Engineering in 2019. He is currently an Associate Scientist/Engineer at ANDRO Computational Solutions, and has worked on developing the SPEARLink™ and ARROWLink™ products.



Tyler Gwin received the ir B.S. degree in Computer and Information Science from SUNY Polytechnic Institute. He is currently an Associate Scientist/Engineer at ANDRO Computational Solutions, LLC. During his work at ANDRO, he has been part of the team developing SPEARLink™ software, along with the research and development of small Unmanned Aerial Systems. Tyler has also assisted in the authoring of a book chapter titled "Deep Learning and Reinforcement Learning for Autonomous Unmanned Aerial Systems: Roadmap for Theory to Deployment."

Zackary Kane received a B.S. in Electrical and Computer Engineer from the State University of New York Polytechnic Institute. Zack is currently a Radio Frequency Engineer/Associate Scientist I at ANDRO Computational Solutions in ANDRO's Marconi-Rosenblatt lab. He has worked on two Small Business Innovation Research (SBIR) projects and the STRAIN Rapid Innovation Fund (RIF) project. While on STRAIN, Zack assisted in range testing the SPEARLink™ radio platform and worked on documentation for the SPEARLink™ radio. He is currently working on the DeepeRFind Phase I SBIR, generating an database of commercial radio devices, and the I-ROAM RIF, testing and improving the IROAM CDMA physical layer.

Noor Biswas received his BS degree in Computer Engineering from the State University of New York Polytechnic Institute in 2020. He is currently an Associate Scientist/Engineer at ANDRO Computational Solutions, and has contributed to projects involving software-defined radio networks.

Andrew Drozd received his Ph.D. from St. John Fisher College in Executive Leadership in Science & Technology Towards Advancements in Human and Machine Learning Using Theoretic Frameworks for AI/ML Applications. Dr. Drozd has extensive experience in electromagnetics science and technology with expertise in emerging spectrum technologies, radar cross section (RCS), computational electromagnetics (CEM), radar and multisensor data fusion, multitarget identification/tracking, target discrimination, cyber and electronic warfare (EW), and cosite modeling and simulation through his various working relationships over the past 43 years with DoD and Federal government agencies, including the U.S. Army, U.S. Air Force, U.S. Navy, Office of the Secretary of Defense (OSD), Missile Defense Agency (MDA), Defense Information Systems Agency/Department of Spectrum Operations (DISA/DSO), and Department of Homeland Security (DHS). Over the past several years, Dr. Drozd has led efforts to develop and mature new technologies under Small Business Innovative Research/Small-business Technology Transfer Research (SBIR/STTR) and Commercialization Readiness Program (CRP) contracts.

Dr. Drozd is an iNARTE (international Association of Radio and Telecommunications Engineers) Certified Engineer (1988–present); IEEE Life Fellow; past president of the IEEE EMC Society (2006–2007); past chairman of the IEEE EMC-S Standards Development and Education Committee (2007–2009); member of the FCC's Communications, Security, Reliability, Interoperability Council (CSRIC VIII - 2021-2023); member of the Alpha Gamma Omicron Chapter of Kappa Delta Pi Honor Society in Education, board member of the Project Fibonacci® Foundation, Inc. focused on STEM leadership education for workforce development; and a 2019 inductee in the Rome Academy of Sciences Hall of Fame. He holds 10 patents and patents pending including the areas of spectrum governance, signal detection/classification, and block chain auditing.