공학박사 학위논문

Cooperative System Design and Implementation of Bluetooth and Wi-Fi

블루투스와 무선랜 프로토콜 간 협력 시스템의 설계 및 구현

2017년 8월

서울대학교 대학원
전기·컴퓨터 공학부
한 종 훈

Abstract

We are living in a ubiquitous world. People bring a massive number of mobile devices such as smartphones, tablets, watches, wearable bands and they are connected among each other via various wireless communication technologies. Even though some of them are equipped with a cellular interface, BT and Wi-Fi protocols are still the most widely used communication technologies due to their free usage.

The co-existence issue of them has been investigated by industries and academia for a long time due to the fact that they operate in the same frequency band. Since BT and Wi-Fi protocols are completely different in terms of the physical and medium access control layer design, previous researches assume that the both protocol stacks cannot understand the signals of each other. We define the approaches based on the assumption as *blind-to-the-other* type approaches. However, this assumption is not applicable to state-of-the-art handhelds, since they include the both protocols in a device, i.e., *co-located*.

Co-located devices are able to interpret BT and Wi-Fi signals, and thus lead to novel architecture design that the both protocol stacks share the their information via inter protocol stack messages. We define a novel type of approaches based on the assumption that BT and Wi-Fi signals are mutually understandable as *aware-of-the-other* type approaches. With the co-located device, therefore, we can propose better solutions not only for the traditional BT/Wi-Fi co-existence issues, but also for the other issues through the aid of aware-of-the-other approaches.

In this dissertation, we design three different protocol designs. First, we tackle the performance of devices equipped with the shared antenna for BT and Wi-Fi protocol stacks. We show that co-location of both protocols provides new opportunity for one protocol to better understand the other and to operate in harmony with the other to avoid mutual interference. We develop an *Opportunistic Bluetooth Transmission*

(OBT) scheme that enables a dual stack device having the integrated module to exploit previously-unused waiting times of the Wi-Fi protocol. We evaluate its performance through not only a model-based analysis, but also a practical implementation of a prototype testbed.

Second, we bring a problem of Wi-Fi scanning overhead through measurements. Due to visiting every channel characteristics of 802.11 Wi-Fi scanning procedure, time and energy consumption of Wi-Fi scanning procedure are significant. To reduce the scanning overhead, we design *SplitScan* architecture which makes stations split the scanning channels and share the results among neighboring stations. SplitScan enables stations to select its own scanning channels in a distributed way, and let stations in proximity share their information via BT packets. We compare the performance of SplitScan with 802.11 standard scanning procedure and show our SplitScan significantly reduces Wi-Fi scanning overhead through simulation and implementation.

Third, we assume that pre-installed communication infrastructures are damaged and do not operate properly. In this scenario, a Wi-Fi multi-hop network is a feasible solution for delivering streaming traffic owing to its capability of high data rate support without requiring any infrastructure. We tackle that IEEE 802.11 medium access control (MAC) shows poor end-to-end throughput performance due to its use of carrier-sensing multiple access with collision avoidance (CSMA/CA), especially in multi-hop networks. We develop a fully distributed pipelining time division multiple access (TDMA) MAC, termed DP-MAC, that aims to reduce the impact of hidden nodes and unnecessary contentions. We evaluate its performance through extensive ns-3 simulation and show that our DP-MAC outperforms 802.11 MAC in various multi-hop network scenarios.

Keywords: Bluetooth, Wi-Fi, co-location, co-operation, MAC,
dual-stack device, Wi-Fi scan, multi-hop

Student number: 2012-30236

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Bluetooth (BT) and Wi-Fi are the most widely used standards for wireless communication in the industrial, scientific, and medical (ISM) bands. The former is managed by the BT special interest group (SIG) [1], and standardized as 802.15.1 specification by Institute of Electrical and Electronics Engineers (IEEE). From BT v1.0 in 1999, BT SIG keeps introducing the newer version of BT and recently v5.0 is announced in 2016. The main target of BT is connecting and exchanging small amount of data over short distance. As the advance of the internet of things (IoT) accelerates, BT will be promising solution of connecting billions of devices and sensors. The latter, Wi-Fi, is a wireless technology based on the IEEE 802.11 specification. Wi-Fi released its first standard in 1997, and has been developed through 802.11b, 802.11a, 802.11g, 802.11n. The latest generation of Wi-Fi is 802.11ac, which delivers up to multi gigabit per second data. Different from BT, it enables devices to exchange high data-rate traffic, such as multimedia streaming, rapid file transfer, etc. According to ABI research, the leader in transformative technology innovation market intelligence, forecasts that Bluetooth will be in 60% of total devices and Wi-Fi will be found in 47% of all devices including wearables, mobile phones, automotive, etc, in 2017.

Since BT and Wi-Fi operate on the same frequency band, the co-existence of them has been investigated for a long time by industries and academia. The main issue in the co-existence of both protocols was interference. Since BT and Wi-Fi are completely different from each other in terms of physical design and medium access control, the one network assumed to be blind to the other network thereby generating mutual interference. In other words, BT protocol stack cannot understand Wi-Fi signals and vice versa. Since two protocols were not mutually understandable, most previous efforts on enhancing performance (mainly throughput) of co-existence scenario had fundamental limitations. We denote these limited approaches as *blind-to-the-other* type approaches.

In todays' mobile devices, the both wireless protocols are included for wireless communication. For example, we can download files through Wi-Fi connection, while listening music through BT headset with our smartphones equipped with BT and Wi-Fi modules. In this *co-located* device, in contrast to the *blind-to-the-other* type approaches we are able to assume that the device exploits acquired information of the other protocol as well as fully understands the both signals. We define the approaches based on this assumption as *aware-of-the-other* type approaches. We believe these novel approaches enable to overcome the fundamental limitation of co-existence problems and they lead to synergistic cooperation in various scenarios.

In this dissertation, we design three novel architecture designs according to the various scenarios and objectives. The designs are based on the cooperation of BT and Wi-Fi, especially *aware-of-the-other* type of cooperation assuming that devices are *co-located* like smartphones and tablets. We first verify the feasibility of information exchange and estimate the possible achievable gain of it in various scenarios by extensive measurements of real-world traffic. Based on the measurement, we carefully design the protocols to enhance the performance exploiting synergy of *aware-of-the-other* type cooperation.

1.2 Related work

1.2.1 BT and Wi-Fi co-existence

The coexistence problems of BT and WLAN protocols can be categorized into two types: *network-level* and *device-level*. On the network-level coexistence, most previous works have focused on interference management. Yomo *et al.* proposed an adaptive frequency rolling mechanism which enables frequency-hopping protocols such that BT efficiently overcomes frequency-static interference [2]. Chek *et al.* proposed an approach called interference source oriented adaptive frequency hopping (ISOAFH) [3]. Hsu *et al.* introduced dynamic packet fragmentation to increase WLAN throughput in the presence of nearby BT devices [4]. Wang *et al.* proposed the WiCop framework that reserves resources for wireless personal area networks [5]. Chiasserini and Rao developed overlap avoidance (OLA) schemes that adjust packet length and perform simple traffic scheduling between BT and WLAN [6]. Aforementioned works help mitigate mutual interference in heterogeneous networks, but they may not operate effectively for dual-stack devices where BT and WLAN protocol stacks contend within a device.

For the device-level coexistence, Ophir *et al.* introduced a TD transmission scheduling method for collocated BT and WLAN protocol stacks [7], under which, however, a device may not receive traffic during the *BT period* since the sender does not know the status of the receiver. Packet traffic arbitration (PTA) has been proposed by the IEEE 802.15.2 task group [8]. It supports their coexistence by prioritizing packet transmission and providing control signals between the two protocol stacks for information exchange. This approach, however, assign time resources exclusively to the two protocols, thus causes inefficiency in spectrum utilization. In the case multiple antennas are available, Jeon *et al.* showed a method for simultaneous transmission and reception of BT and WLAN packets by using an interference cancellation technique [9]. These device-level approaches do not consider antenna sharing of the two protocols, thus they are inappropriate to be applied directly to dual-stack devices.

Most off-the-shelf dual-stack devices utilize custom TD scheduling techniques [10, 11, 12, 13]. Details related to signaling and scheduling operations are often unknown and thus hard to discern and evaluate. Meylan and Yan proposed a TD scheduling technique using Unscheduled Automatic Power Save Delivery (U-APSD) which has been widely adopted in commercial dual-stack devices [14]. Sun and Xhafa evaluated the performance of TD scheduling with the U-APSD through simulation and proposed a CTS (clear to send)-to-Self mechanism to reduce overhead [15]. However, U-APSD may suffer from performance degradation when the two protocol stacks try to transmit simultaneously. Some studies investigated simultaneous reception of BT and WLAN through a single antenna [16, 17]. They designed a shared low-noise amplifier (LNA) for BT and WLAN, and achieved simultaneous reception of the both signals via appropriate gain control. The technique, however, is limited to reception only, and the two protocol stacks still exclusively use the antenna for transmission.

1.2.2 Wi-Fi scanning

Many literatures are interested in scanning related operations of Wi-Fi protocol and they are categorized into two types according to their purpose: reducing handoff delay and reducing scanning overhead. On the first category, Brik *et al.* [18] eliminated handoff latency by exploiting multiple Wi-Fi radios, which is not common in today's mobile devices. Chen and Qiao [19] introduced a fast handoff scheme called HaND. HaND achieves zero channel dwell time of handoff station, however, it requires communication between APs and the handoff decision is made by the infrastructure. Xu and *et al.* [20] suggested D-SCAN, which reduces scanning time by eavesdropping hidden wireless traffic which contains the abundant AP information. Purushothaman and Roy [21] developed FastScan scheme, which reduces scanning delay by using a client-based database. Since it is history-based approach, the performance of scheme depends on the accuracy of database and it requires some time to build the accurate database. Wu and *et al.* [22] proposed a fast handoff scheme called Proactive Scan, which decouples the time-consuming channel scan from the actual handoff. Proactive Scan successfully provides fast handoff and satisfactory performance, but it does not reduce scanning time/energy overhead.

For reducing scanning overhead, Li and *et al.* [23] developed *WiScan* architecture. They found the primary culprit of scanning energy overhead is the main processor and offloaded scans to the Wi-Fi radio. Kim and *et al.* [24] introduced *WiFisense*, a system that employs user mobility information from low-power sensors to conserve battery power. Lee and *et al.* [25] also proposed a scanning method employing a motion sensor to suppress unnecessary scanning overhead. Ananthanarayanan and Stoica [26] suggested *Blue-Fi*, a system that predicts the availability of the Wi-Fi connectivity by using a combination of Bluetooth contact-patterns and cell-tower information. Choi [27] developed *WidthSense*, which also detects Wi-Fi signals using WPAN radio. Above-mentioned schemes run on each Wi-Fi station and reduce its own scanning overhead. However, there is no work considering cooperation among neighboring Wi-Fi stations for reducing scanning overhead.

1.2.3 Multi-hop transmissions

There are several previous works dealing with efficient multi-hop transmissions. Zhao *et al.* [28] modeled the intra-flow contention problem considering hidden stations in multi-hop networks. Gabale *et al.* [29] proposed LiT-MAC, a full-fledged TDMA based MAC protocol for real-time applications, and implemented it on the 802.15.4 platform. Sevani *et al.* [30] implemented this on the 802.11 platform. Raman *et al.* [31] presented Packets in Pipe (PIP), a connection-oriented multi-hop, multi-channel TDMA-based MAC for high throughput bulk transfer based on 802.15.4. However, LiT-MAC and PIP require separate resources for scheduling dissemination over the whole network due to their centralized scheduling feature. This requires an overhead, especially when the schedule varies with time. Koutsonikolas *et al.* [32] implemented TDM-MAC for wireless mesh networks, and Djukic *et al.* [33] designed Soft-TDMAC, a software TDMA based MAC protocol over 802.11 hardware. However, these literatures only focused on the global time synchronization issue, without considering the issues about scheduling and spatial reuse.

1.3 Contributions and Outline

Owing to their free usage in ISM bands BT and Wi-Fi protocols has become the most popular wireless technologies in the world. Since they are operating in the same frequency band, there has been many performance issues and corresponding solutions proposed in industries and academia. The basic assumption of these approaches was that BT and Wi-Fi protocol stacks cannot understand each other due to their PHY/MAC design difference.

Nowadays, the two protocols are embedded together in smart phones, tablets, and laptops. Since they are co-located, we are able to make a new assumption: the both protocol stacks are mutually understandable via inter-protocol cooperation. This assumption leads us to the possibility of aware-of-the-other approaches breaking the limitation of previous blind-to-the-other approaches. Aware-of-the-other approaches are able to improve the traditional co-existence problems and also provide basis to design novel schemes for existing BT and Wi-Fi protocol operations.

This dissertation is organized as follows. In Chapter 2, we tackle the performance of devices equipped with the shared antenna for BT and Wi-Fi protocol stacks. We identify performance issues associated with the device and verify the corresponding reasons of performance issues. We develop a standard compatible aware-to-the-other scheme, which exploits intrinsic waiting times of Wi-Fi MAC. We evaluate the performance of the proposed schemes via simulation and prove the feasibility of the proposed schemes via prototype as a proof-of-concept.

In Chapter 3, we propose a novel architecture for Wi-Fi scanning which utilizes BT protocols as a message of cooperation. We show that the proposed architecture outperforms in terms of time and energy consumption via simulation compared to the 802.11 standard scanning procedure. Furthermore, we also show that the proposed architecture works well on the real device through implementation on linux-based laptops.

In Chapter 4, we propose a fully distributed pipelining TDMA MAC for Wi-Fi multi-hop networks and implement it on ns-3 simulator. The proposed MAC solves two fundamental challenges - scheduling dissemination and spatial reuse - of TDMA multi-hop networks. According to extensive simulation, the proposed MAC outperforms in terms of end-to-end throughput compared to 802.11 ad-hoc MAC in various scenarios.

We conclude the dissertation in Chapter 5.

Chapter 2

Opportunistic Bluetooth Transmission Schemes in dual-stack devices

2.1 Introduction

Many personal mobile devices are equipped with multiple communication modules for different wireless communication technologies [34, 35, 36, 37]. Bluetooth (BT) and Wireless LAN (WLAN) are two of the most popular wireless technologies due to their free use of unlicensed spectra and support for a wide-range of applications from high-rate data transfer to real-time streaming [38]. Due to the cost and form factor issues, hardware manufacturers of handsets have implemented BT and WLAN protocol stacks on a single integrated circuit [39, 40]. This results in sharing of an antenna as shown in Fig. 2.1. In this paper, we denote the device equipped with a single-chip single-antenna communication module for BT and WLAN protocols as a *dual-stack* device.

Since the dual-stack device operates across two different networks, careful coordination for resource allocation and protocol operations is imperative for spectrum efficiency and system performance. Currently, the dual-stack device takes a time-division multiplexing approach, and alternates the working protocol between BT and WLAN. Let us define the *mode* of a dual-stack device to denote the working protocol. In this

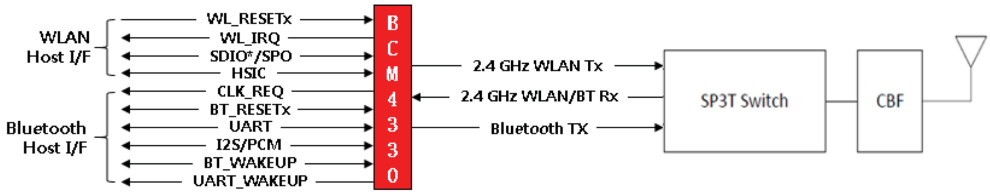


Figure 2.1: Block diagram of BCM4330 chip equipped in Apple iPhone 4S and Google Nexus 7.

alternating approach, we identify a couple of significant problems due to the heterogeneous nature of the two protocols. First, a dual-stack device may suffer from significant WLAN throughput degradation due to concurrent BT traffic. In order to meet BT's Quality of Service (QoS) requirements (e.g., isochronous streaming), the device stays more in BT mode because it serves best-effort service in WLAN mode. Second, as a dual-stack device often spends a significant amount of time in WLAN mode waiting for a specific WLAN channel to be idle even when it can utilize unused channels in the other mode.

To address the both problems, we take an overlay approach and develop an Opportunistic Bluetooth Transmission scheme, termed OBT, that allows a dual-stack device to exploit the waiting time in WLAN mode for BT transmissions. We note that our technique makes use of new opportunity provided by the protocol stack integration. As the dual-stack device can access information across both protocol stacks and has a holistic view of TD-based protocol operations, it can utilize niche times in WLAN mode, thereby improving spectrum efficiency. Our contributions in this paper are three-fold:

- We identify two performance issues associated with a dual-stack device through real-world measurements. We first show that there is a significant bias in bandwidth sharing between BT and WLAN protocols. Then we show that WLAN operation incurs a significant amount of waiting time for idle channel, which may be exploited in BT mode.

- We develop two standard-compliant schemes: OBT-BE and OBT-QoS. They utilize unused time resources in WLAN mode to transmit BT packets of best-effort traffic and QoS-sensitive traffic, respectively.

- We evaluate the performance of OBT through analysis and implementation in a testbed.

The rest of this chapter is organized as follows. Section 2.2 presents our motivation. We develop the proposed schemes in Section 2.3, and provide analytical results in Section 2.4. Further, performance evaluation is discussed in Section 2.5, followed by concluding remarks in Section 2.6.

2.2 Background and motivation

We briefly overview protocol operations of BT specification 4.0 [1] and 802.11n for WLAN. Then, we discuss the state-of-the-art scheduling method in dual-stack devices, and identify its limitation through experiments.

2.2.1 Overview of BT and WLAN operations

Over the 2.4GHz unlicensed band, BT uses frequency-hopping [41, 42] to combat interference and fading. BT devices construct a piconet with one master and multiple slaves that are synchronized with the master, where all devices employ the same frequency-hopping pattern. By default, the hopping pattern is determined over a set of 79 1MHz-channels or its subset depending on channel states. The latter is called *adaptive frequency hopping*. Time is slotted with the duration of $625\mu s$, and multiple time slots (e.g., 1, 3, 5) are allocated for a single BT transmission. A BT device, once it has a packet to send, transmits the packet immediately without carrier-sensing or waiting.

IEEE 802.11 WLAN has 11 partially overlapping 20MHz-channels in the same 2.4 GHz unlicensed band and operates over one or two channels (i.e., 20 MHz or 40

MHz in 802.11n). The standard adopts Distributed Coordination Function (DCF) with Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), which requires that a backlogged device, with a packet for sending, senses the channel, waits for distributed inter-frame space (DIFS) duration of idle channel state, picks a random backoff counter to reduce multiple access contention with other devices, and counts down in unit of *slottime* ($9\mu s$).

If the channel is sensed busy for a slottime, the device freezes its backoff counter. Otherwise, the device reduces the back-off counter by 1 and transmits the packet when the counter reaches 0. The transmission duration for a packet varies according to channel state, packet size, and frame aggregation policy.

2.2.2 Mode-switching of a dual-stack device

The state-of-the-art dual-stack devices provide connectivity to both BT and WLAN networks by scheduling the resources in a time-interleaving manner. A widely used method is to assign time to BT or WLAN protocols in an alternating manner by using U-APSD (standardized in the IEEE 802.11e specification). A brief overview of U-APSD operation is as follows. Suppose that a dual-stack device with U-APSD has non-empty queues (i.e., backlogged) for both of the protocols, and currently in WLAN mode. When it needs to use the BT network, it first signals the AP by sending a null data frame with the power management bit set in the frame control field of the header. The AP responds to the device with an ACK frame regarding as sleeping for power saving and holds any downlink packets destined to the device. The dual-stack device switches its mode and starts transmitting/receiving BT packets in BT mode. To avoid interference with other WLAN devices, the dual-stack device excludes the frequency band that is used in the WLAN mode from the BT frequency hopping set. Once the dual-stack device finishes its BT operation, it switches back to WLAN mode and transmits another null frame or data frame to the AP with the power management bit *disabled*. The AP then forwards the buffered WLAN packets to the dual-stack device.

Figure 2.2: Topology of a dual-stack handset device for experiments.

2.2.3 Performance degradation of BT-WLAN dual-stack device

Under the current resource scheduling with U-APSD signaling, the two protocols do sharing in an alternating manner, which may cause service interruption. We show that this commonly occurs. Fig. 2.2 shows a typical network topology used in experiments. A dual-stack device (i.e., smartphone in the middle) is associated with the AP (on the left) and also provides an audio stream to a BT headset (on the right). There is no other WLAN device. The AP is equipped with the RTL8192CE chipset, and the BT headset is equipped with BT module CSR8635, which supports the BT specification 4.0 with adaptive frequency hopping. The headset supports SubBand Coding (SBC) [43], the most common codec for BT audio streaming, with bit rate 328 kbps. The distance between AP and the device is 5 m, and the distance between the device and the headset is set to 0.5m. This configuration is for a user holding a smartphone and listening to music with a BT headset, while downloading files (e.g., mastering quality sound) via WLAN. We measure the performance of file transfer through WLAN, with and without BT audio streaming. We repeated each experiment 10 times, and tested four different dual-stack devices listed in Table 2.1.

Fig. 2.3 shows average WLAN throughput (i.e., file transfer size (1.36 GB) / measured completion time) for each dual-stack device. The WLAN throughput significantly degrades in the presence of BT audio streaming, by 60% on average. *In stark contrast to the small bit rate (328 kbps) consumed by the BT headset codec, the throughput degradation experienced by WLAN is two orders of magnitude greater.* It clearly shows that the trade-off between BT and WLAN throughput in dual-stack devices is disproportional and their operations are poorly coordinated.

Table 2.1: Testing Devices

Device	**Operating System**	**Dual-stack Module**
Apple iPhone6	iOS 9.0.1	Broadcom BCM4339
LG Nexus5	Android 4.4.4	Broadcom BCM4339
Samsung Galaxy S6 Edge	Android 5.0.2	Broadcom BCM4358
Laptop	Ubuntu 14.04.3	Atheros AR9462

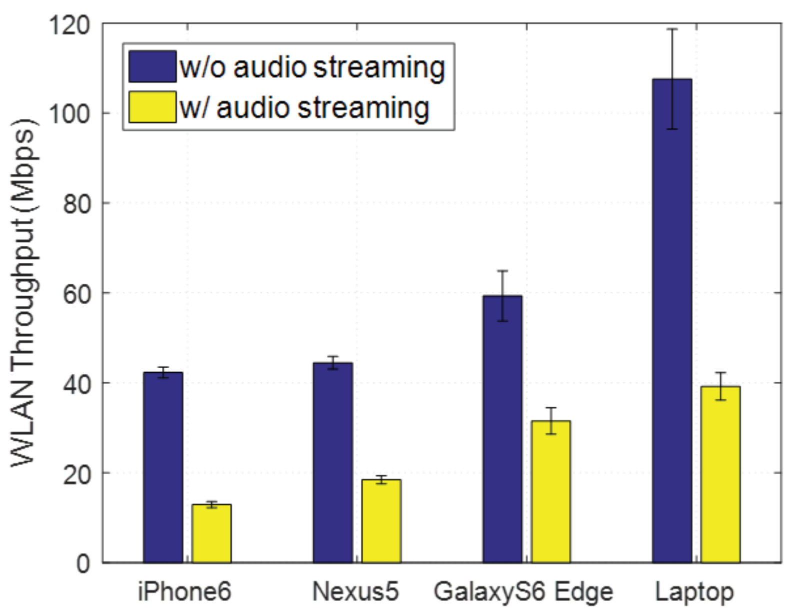


Figure 2.3: WLAN throughput performance of different dual-stack devices.

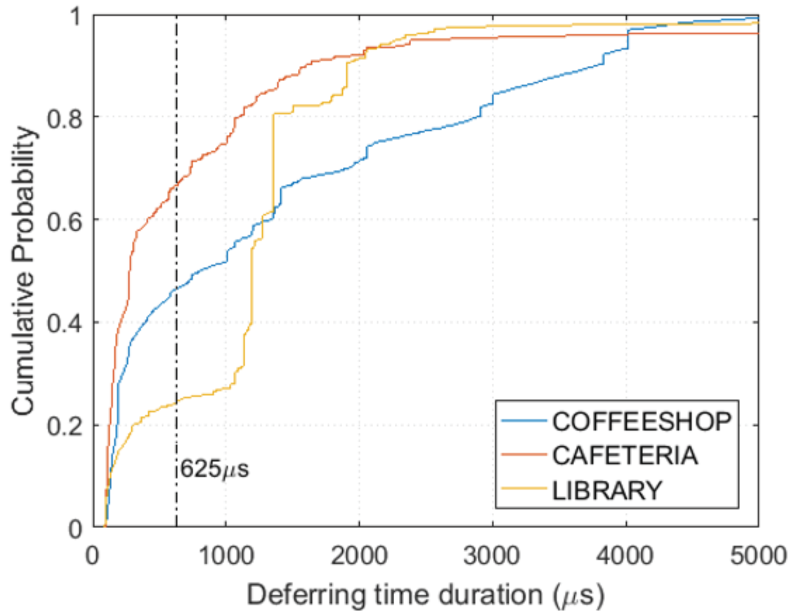
2.3 Opportunistic Bluetooth Transmission

We solve the problems by introducing the Opportunistic Bluetooth Transmission (OBT) scheme. Suppose that a dual-stack device in WLAN mode, and overhears a WLAN transmission that includes its *transmission duration*. Instead of waiting for the duration in WLAN mode, we let the device temporarily switch to BT mode and transmit BT packets within the given duration. Adaptive frequency hopping will be applied to avoid the channels that are already occupied by the WLAN networks. After the duration, the device switches back to the WLAN mode as if it has stayed there.

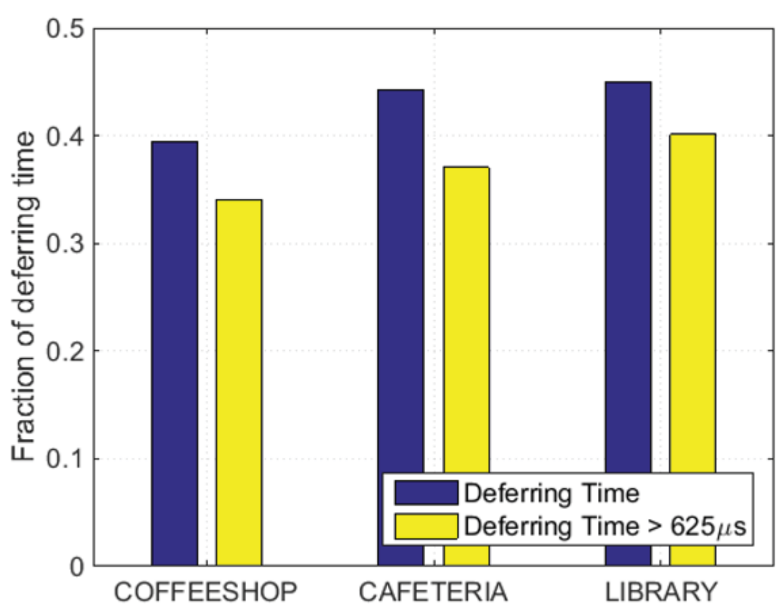
We first verify the feasibility of OBT from measurements of real-world WLAN environments. Let *deferring time* denote the time that a dual-stack device is in the WLAN mode waiting for the channel to be idle. The duration of the deferring time depends on the packet length, channel state, and modulation and coding scheme (MCS). If it is longer than the smallest time σ for a BT transmission (i.e., 625μs), it may be used for OBT.[1] We use an Intel laptop with Dual Band Wireless-AC3160 adapter to sniff WLAN packets and measure their deferring times for an hour in three different WLAN places: coffee shop, cafeteria, and library in the daytime. Fig. 2.4(a) shows the cumulative distribution in each place. We find that about 54%, 33%, 76% of the deferring times are longer than σ in the coffee shop, cafeteria, and library hotspots, respectively. In addition to per-packet distribution, we also measure the (normalized) total deferring time ($= \frac{1}{t}\sum_i t^i_{def}$, where t^i_{def} denotes the deferring time of the i-th packet and t is the measurement period), and the total OBT-doable deferring time ($= \frac{1}{t}\sum_i t^i_{def} \cdot 1_{\{t^i_{def} \geq \sigma\}}$, where $1_{\{\cdot\}}$ denotes the indicator function). Fig. 2.4(b) shows that more than 84% of the deferring time can be exploited by OBT, which is equal to additional 34% of time resources. This can be considered a practical bound on the OBT's performance gain.

To enable OBT, it requires that the dual-stack device i) knows the deferring time

[1] In the case of synchronous BT traffic, a dual-stack device may need longer deferring time for synchronized transmission.



(a) CDF of deferring time



(b) Deferring time per second (on average)

Figure 2.4: Statistics of deferring time in three real-world environments.

and ii) quickly switches between the two modes. In the former, a dual-stack device obtains the period of channel busy from the network allocation vector (NAV) defined in the 802.11 virtual carrier sensing mechanism. Originally, NAV is intended for a WLAN device to save energy by refraining from physical carrier sensing when the channel is busy for sure. We use this information as the deferring time duration. The latter can be considered an implementation overhead. We note that, in commercial devices, the switching time in dual-stack RF hardware is less than $1\mu s$ [44] and the Tx/Rx turnaround time is $150\mu s$ for BT module and $5\mu s$ for WLAN module. Thus, the switching overhead is marginal. In the following, we develop two different OBT schemes: one for best-effort BT traffic (OBT-BE), and the other for QoS-sensitive BT traffic (OBT-QoS).

OBT-BE: This scheme handles best-effort BT traffic such as file transfer. Since there is no QoS requirement for BT traffic, we give priority to WLAN traffic, and passively serve BT traffic through OBT.

Under OBT-BE, a dual-stack device starts in WLAN mode and waits for WLAN packet transmissions from other WLAN devices. If there is a transmission and the NAV value is updated, it checks whether the deferring time t_{def} is long enough to transmit a BT packet.

If t_{def} is no smaller than a threshold T_{OBT}^{thrs}, which equals one slot time $\sigma(= 625\mu s)$ for OBT-BE, the device selects an acceptable BT packet length accordingly: 1 if $t_{def} \in [\sigma, 3\sigma)$, 3 if $t_{def} \in [3\sigma, 5\sigma)$, and 5 if $t_{def} \geq 5\sigma$. Then, it switches to BT mode and transmits a BT packet of the chosen (or smaller) packet length. After the transmission, it checks again if the remaining deferring time is no smaller than σ. If it is not, the procedure of length selection and BT packet transmission repeats. Otherwise, the device switches back to WLAN mode and continues normal WLAN operation. Algorithm 1 shows the algorithm in detail.

OBT-BE is intended to achieve high spectrum efficiency under *heavy* WLAN traffic. We note that OBT-BE may suffer from low BT throughput under *light* WLAN

Algorithm 1 OBT scheme for best-effort BT traffic (OBT-BE)

Input: T_{OBT}^{thrs}
Monitoring NAV from WLAN
if *NAV* updated **then**
    $t_{def} \leftarrow NAV - t_w$
endif
while $t_{def} \geq T_{OBT}^{thrs}$ **do**
    Select a BT packet P fitting in t_{def}
    Transmit(P)
    $t_{def} \leftarrow t_{def} - T_{TX}(P)$
endwhile

traffic, and need further modification for practical use.

OBT-QoS: This scheme targets BT traffic with strict QoS requirements such as telephony and multimedia streaming. Specifically, we assume BT traffic with periodic time structure, denoted as a cycle. QoS-sensitive BT applications with isochronous bit rate requirements (e.g., voice or music streaming) often have constraints of throughput requirement R_{BT}^{req} and *target cycle* T_{cycle} due to their delay bound. In the state-of-the-art approach, the dual-stack device serves the QoS BT traffic first (i.e., the device starts each cycle with BT mode), and then within the remaining time of the cycle, it switches to WLAN mode and transmits WLAN packets. This may result in WLAN performance degradation when there is a large amount of BT traffic.

Under OBT-QoS, the device starts a cycle in WLAN mode, and transmits WLAN packets. In the meantime, it transmits BT packets through OBT if possible. It keeps track of the remaining cycle time and switches to BT mode when necessary. We denote the switching instance as the *critical time* of the cycle.

An example of operation is depicted in Fig. 2.5, and the details are as follows. Let t denote the time after a cycle starts at time 0. The variable r_{BT}^{ach} represents the achieved

Algorithm 2 OBT scheme for QoS BT traffic (OBT-QoS)

Input: T_{OBT}^{thrs}, R_{BT}^{req}, T_{cycle}
$r_{BT}^{ach} \leftarrow 0$, $t \leftarrow 0$, Initialize t_{BT}^{start}
while $t < t_{BT}^{start}$ **do**
    Monitoring NAV from WLAN
    if *NAV* updated **then**
        $t_{def} \leftarrow NAV - t_w$
    endif
    while $t_{def} \geq T_{OBT}^{thrs}$ and $t + t_{def} < t_{BT}^{start}$ **do**
        Select a BT packet P fitting in t_{def}.
        Transmit(P)
        $t_{def} \leftarrow t_{def} - T_{TX}(P)$
        Update r_{BT}^{ach}, t_{BT}^{start}
        if $R_{BT}^{req} = r_{BT}^{ach}$ **then** break;
        endif
    endwhile
endwhile
while $r_{BT}^{ach} < R_{BT}^{req}$ **do**
    Send BT traffic
    Update r_{BT}^{ach}
endwhile
if $t < T_{cycle}$ **then**
    Wait until $t = T_{cycle}$
endif

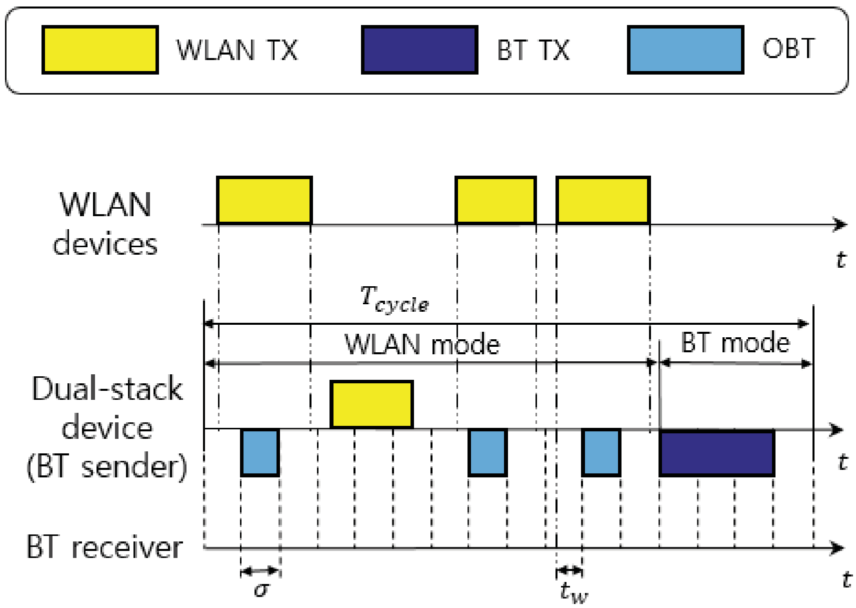


Figure 2.5: Example of operation depicting OBT-QoS scheme.

BT throughput up to t, and t_{BT}^{start} denotes the expected time for mode change. Initially the two variables are set to $r_{BT}^{ach} = 0$ and $t_{BT}^{start} = (1 - \frac{R_{BT}^{req}}{R_{BT}}) \cdot T_{cycle}$, respectively, where R_{BT} is the BT throughput achieved during the last BT mode. The dual-stack device starts in WLAN mode and monitors NAV update. If it detects a sufficiently long NAV, it triggers an opportunistic BT packet transmission. The dual-stack device stays in WLAN mode when $t < t_{BT}^{start}$. Suppose the device transmits a BT packet through OBT at time t. After the transmission, it updates achieved throughput and the expected time for mode change as follows:

$$r_{BT}^{ach} \leftarrow (\text{amount of transmitted BT traffic in this cycle})/t$$

$$t_{BT}^{start} \leftarrow t_{BT}^{start} + \frac{r_{BT}^{ach} \cdot t}{R_{BT}}.$$

When $t \geq t_{BT}^{start}$, the mode is changed to BT and subsequently only BT packets are transmitted until the end of the cycle. For each BT transmission, r_{BT}^{ach} is updated. The cycle ends when $r_{BT}^{ach} \geq R_{BT}^{req}$ and $t \geq T_{cycle}$.

There is a technical challenge in estimating the critical time due to the time-varying property of wireless channels. To address this, we take a more flexible time structure; after the device switches to BT mode, if all the BT packets for the cycle are transmitted before the end of the cycle, it switches back to WLAN early for the remaining time in the cycle. Otherwise (i.e., there remain BT packets for the cycle after the end of the cycle), it remains in BT mode occupying part of the next cycle, and makes a delayed switch to WLAN mode after finishing BT transmissions. Algorithm 2 presents the pseudo-code.

2.4 Modeling and Analysis

In this section, we characterize the performance of OBT-BE and OBT-QoS schemes. We consider a network scenario of n WLAN devices with one dual-stack device, and evaluate the performance of the dual-stack device with and without OBT schemes. Using OBT-BE scheme, the dual-stack device has BT traffic backlog and opportunistically transmits BT packets without affecting WLAN operation. Thus, it will achieve higher BT throughput and comparable WLAN throughput compared to the device without OBT. Using OBT-QoS scheme, the dual-stack device has a fixed amount of BT traffic in each cycle. As it opportunistically transmits more BT packets through OBT, it saves time in BT mode and transmits more WLAN packets. Thus it is expected to achieve comparable BT throughput and higher WLAN throughput than the device without OBT.

Suppose that all WLAN devices (including the dual-stack device) have backlogged WLAN traffic. The dual-stack device also has a BT traffic backlog under OBT-BE scheme and a fixed amount of BT traffic (in each cycle) under OBT-QoS scheme. Suppose that the dual-stack device always has packets to send over BT[2] and WLAN networks, and each WLAN device also has saturated traffic to send. Consequently, the

[2]This assumption is also valid under OBT-QoS where a fixed amount of BT traffic is provided in each cycle. Since OBT-QoS tries to maximize the duration in WLAN mode, it is likely to have BT packets at the end of the cycle.

collision probability and the number of competing devices are constant. We assume that the switching delay for the mode change between BT and WLAN is negligible. Then the dual-stack device has an *OBT opportunity* when the following three conditions are satisfied.

1. A WLAN packet is transmitted by a WLAN device.

2. The dual-stack device is not the receiver of the packet, and successfully overhears the packet header.

3. The duration of the deferring time obtained from the overheard WLAN packet is longer than σ.

Let S_{OBT} denote the expected BT throughput via OBT. We can write S_{OBT} as

$$S_{OBT} = N_{OBT} \cdot L_{OBT} \cdot \omega \quad (0 \leq \omega \leq 1), \tag{2.1}$$

where N_{OBT} denotes the average number of OBT opportunities in the WLAN mode per unit time, L_{OBT} denotes the average amount of BT data transmitted through one OBT opportunity, and ω denotes the time fraction that the device stays in the WLAN mode during a cycle. In the following, we derive each of N_{OBT}, L_{OBT}, and ω.

First, let T_{int} denote the time interval between two ticks of the backoff counter of the dual-stack device. Then the number of OBT opportunities per unit time can be expressed as

$$N_{OBT} = \frac{E[\text{Number of OBT opportunities within } T_{int}]}{E[T_{int}]}. \tag{2.2}$$

Let P_{tr} denote the probability that there exists at least one WLAN transmission from any devices in the network within T_{int}, and P_s denote the conditional probability that there is only one WLAN transmission given the event of at least one WLAN transmission in T_{int}. Similarly, let P_{tr}^w and P_s^w denote the probability that there exists at least one WLAN transmission from any *WLAN* devices (i.e., excluding the dual-stack device), and the conditional probability of only one WLAN transmission given the event, respectively. Then we analytically obtain the probabilities P_{tr}, P_s, P^w_{tr}, and P^w_s, which are referred to in Appendix A. Let P_{OBT} denote the probability that there is an OBT opportunity given successful transmission from one of the WLAN devices. Then the numerator of (2.2) is equal to $P^w_{tr} P^w_{tr} P_{OBT}$ and we have

$$N_{OBT} = \frac{P^w_{tr} P^w_s P_{OBT}}{P_{tr} P_s T_s + (1 - P_s) P_{tr} T_c + (1 - P_{tr}) T_{idle}}, \tag{2.3}$$

where T_s, T_c, and T_{idle} denote the time durations for a successful WLAN transmission, a collision, and idle channel, respectively. In the denominator, the first term is the expected time for a successful WLAN transmission, the second term is the expected time for a collision, and the third term is the expected time for idle channel (between two ticks of the backoff counter of the dual-stack device). From (2.3) and our results in Appendix A, we can obtain N_{OBT}.

Second, we estimate L_{OBT}, the average amount of BT data transmitted through an OBT opportunity. This will be determined by deferring time t_{def} and Signal to Noise Ratio (SNR) of the BT channel. Given deferring time t and SNR $\gamma \in [\gamma_{min}, \gamma_{max}]$, let $L_{OBT}(t, \gamma)$ denote the amount of transmitted BT data. We can obtain $L_{OBT}(t, \gamma)$ by modifying the BT throughput analysis in [45]. Fig. 2.6 illustrates $L_{OBT}(t, \gamma)$. The results show that the amount of transmitted data increases with t_{def} in a discrete manner, which is due to the discrete nature of BT packet transmission. Then we obtain L_{OBT} as

$$L_{OBT} = \int_{T^{thrs}_{OBT}}^{\infty} \int_{\gamma_{min}}^{\gamma_{max}} L_{OBT}(t, \gamma) \cdot f_T(t) \; d\gamma dt, \tag{2.4}$$

where $f_T(t)$ is the distribution of deferring time t_{def}, which is determined by WLAN packet length distribution. We assume that $f_T(t)$ is known in advance or can be obtained by overhearing.

Lastly, we consider ω, the time fraction that the device stays in the WLAN mode during a cycle. For OBT-BE, we have $\omega = 1$, since it does not have BT mode and transmits BT packets only through OBT. In this case, the OBT throughput can be directly obtained from (2.1), (2.3), and (2.4). For OBT-QoS, the OBT throughput depends on

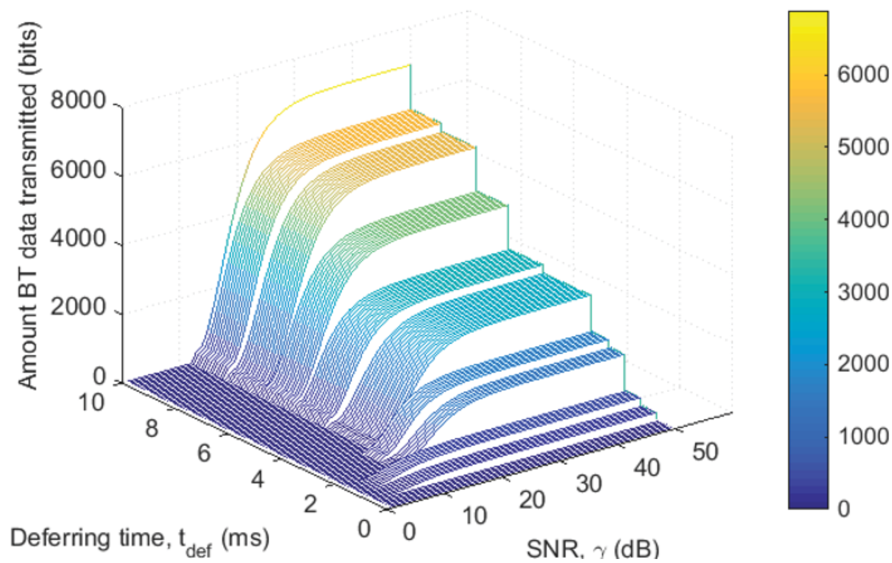


Figure 2.6: Amount of BT data transmitted through an OBT opportunity.

the required BT throughput R_{BT}^{req} or the total amount of BT data transmitted during a cycle. Assuming that the long-term average of a cycle duration equals T_{cycle}, our algorithm successfully transmits all BT packets in the corresponding cycle. In other words, we have

$$R_{BT}^{req} \cdot T_{cycle} \approx S_{OBT} \cdot T_{cycle} + S_{BT} \cdot T_{cycle}^{BT}, \tag{2.5}$$

where S_{BT} and T_{cycle}^{BT} denote the BT throughput in the BT mode, and the time duration that the dual-stack device is in the BT mode, respectively. The left side denotes the amount of BT data that should be transmitted during a cycle. On the right side, the first term denotes the amount of BT data transmitted through OBT, and the second term denotes the amount of BT data transmitted in the BT mode. Rearranging the equation, we have $\frac{T_{cycle}^{BT}}{T_{cycle}} = \frac{(R_{BT}^{req} - S_{OBT})}{S_{BT}}$, which equals $(1 - \omega)$ by definition. Hence, we obtain

$$S_{OBT} = R_{BT}^{req} - (1 - \omega) S_{BT}. \tag{2.6}$$

Combining (2.1) and (2.6), we have

$$S_{OBT} = \frac{R_{BT}^{req} - S_{BT}}{1 - \frac{S_{BT}}{N_{OBT} \cdot L_{OBT}}}, \tag{2.7}$$

which is the expected BT throughput under OBT-QoS scheme.

Besides the OBT performance, another interesting performance metric is WLAN throughput under the OBT schemes. For OBT-BE, the dual-stack device transmits BT traffic only through OBT, which does not affect the WLAN throughput. Thus, one can obtain the WLAN throughput of the dual-stack device, following the results in [46].

For OBT-QoS, the dual-stack device stays in the WLAN mode no shorter than the dual-stack device without OBT, thereby achieving higher WLAN throughput. To calculate the WLAN throughput of the dual-stack device, we use the transmission probability τ from one WLAN device during T_{int}. For the calculation of τ, please refer to Appendix A. Suppose that the dual-stack device transmits a WLAN packet. Given the transmission, the conditional probability P_s^d of successful transmission equals the probability that all the other WLAN devices do not transmit, i.e., $P_s^d = (1 - \tau)^n$. Let S_{WLAN} denote the WLAN throughput of the dual-stack device with OBT-QoS, and N_{WLAN} denote the number of WLAN transmissions in WLAN mode in a unit time. Then we obtain S_{WLAN} as

$$\begin{aligned} S_{WLAN} &= N_{WLAN} \cdot E[L] \cdot \omega, \\ N_{WLAN} &= \frac{\tau P_s^d}{P_{tr} P_s T_s + (1 - P_s) P_{tr} T_c + (1 - P_{tr}) T_{idle}}, \end{aligned} \tag{2.8}$$

where L is the WLAN payload size of transmission. The numerator of N_{WLAN} denotes the probability of successful WLAN transmission of the dual-stack device during T_{int}.

According to [46], the channel access behavior of WLAN devices with saturated traffic can be modeled by a Markov chain. Since OBT opportunity can be obtained only in WLAN mode, we focus on the duration that the network operates in WLAN mode. In the WLAN mode, since a dual-stack device accesses the channel like a WLAN device, it can be also modeled by a Markov chain. Then, in a saturated network of n WLAN devices and one dual-stack device, the packet transmission probability τ in a slottime and the packet collision probability p are calculated as

$$\tau = \frac{2(1-2p)(1-p)}{(1-2p)(W_{min}+1)+pW_{min}(1-(2p)^m)} \qquad (2.9)$$
$$p = 1-(1-\tau)^n,$$

respectively, where W_{min} denotes the minimum contention window, and m denotes the maximum backoff stage (set to 7 by default). Then we obtain τ by solving the equation (2.9) numerically.[3]

Let P_{tr}^w and P_s^w denote the probability that there exists WLAN transmission from one of the WLAN devices excluding the dual-stack device, and the conditional probability that there is only one WLAN transmission in T_{int}, respectively, where T_{int} denotes the time interval of backoff counter decrement in the dual-stack device. Using the obtained τ, we have

$$P_{tr}^w = 1-(1-\tau)^n \qquad (2.10)$$
$$P_s^w = \frac{n\tau(1-\tau)^n}{P_{tr}^w},$$

since only one WLAN device should transmit and the others including the dual-stack device should stay silent for a successful transmission. Thus, the probability that there is a successful packet transmission from the WLAN devices in T_{int}, i.e., the first condition of *OBT opportunity* in Section 5, is expressed as $P_{tr}^w P_s^w$.

Let P_{OBT}, reflecting the second and third conditions of OBT opportunity, denote the probability that there is an OBT opportunity given a successful transmission from one of the WLAN devices. It can be calculated as,

$$P_{OBT} = g(n) \cdot P[t_{def} \geq T_{OBT}^{thrs}] = g(n) \cdot \int_{T_{OBT}^{thrs}}^{\infty} F_T(t)dt, \qquad (2.11)$$

[3]The obtained τ and p can be inaccurate in a network with a very low number of devices (1 to 3) as mentioned in the original study. This is due to the fact that the assumption of constant τ is somewhat inaccurate in a small-sized network.

where $g(n)$ is the probability that the dual-stack device is not the receiver of the transmission given a successful transmission from one of the WLAN devices, t_{def} is the duration of deferring time, and $F_T(t)$ is the cumulative distribution function of t_{def} which can be empirically obtained by the dual-stack device.

Let P_{tr} and P_s denote the probability that there exists WLAN transmission from any devices in the network, and the conditional probability that there is only one WLAN transmission given WLAN transmission in T_{int}, respectively. $(n + 1)$ devices (including the dual-stack device) are contending on the channel, and each transmits with probability τ. Then we obtain

$$\begin{aligned} P_{tr} &= 1 - (1-\tau)^{n+1} \\ P_s &= \frac{(n+1)\tau(1-\tau)^n}{P_{tr}}. \end{aligned} \tag{2.12}$$

To this end, we claim that a successful transmission occurs with probability $P_{tr}P_s$, a collision with probability $P_{tr}(1 - P_s)$, and no transmission with probability $(1 - P_{tr})$ in T_{int}. Let T_s, T_c, and T_{idle} denote the time durations for a successful WLAN transmission, a collision, and idle channel, respectively. Then we obtain

$$T_s = \begin{cases} T_h + x + T_{SIFS} + T_{ACK} + T_{DIFS} & \text{(no RTS/CTS)} \\ T_h + x + T_{SIFS} + T_{ACK} + T_{DIFS} & \\ +T_{RTS} + T_{SIFS} + T_{CTS} + T_{SIFS} & \text{(RTS/CTS enabled)} \end{cases} \tag{2.13}$$

$$T_c = \begin{cases} T_h + x + T_{DIFS} & \text{(no RTS/CTS)} \\ T_{RTS} + T_{DIFS} & \text{(RTS/CTS enabled)} \end{cases} \tag{2.14}$$

$$T_{idle} = \text{slottime } (9\mu s), \tag{2.15}$$

where T_h is 802.11 header duration, x is WLAN packet duration, and $T_{SIFS}, T_{ACK}, T_{DIFS}$ are constant values indicating the SIFS, ACK, DIFS duration, respectively.

Then, from (2.2) in the paper and (2.10)-(2.15), we obtain

$$N_{OBT} = \frac{P_{tr}^w P_s^w P_{OBT}}{P_{tr}P_sT_s + (1-P_s)P_{tr}T_c + (1-P_{tr})T_{idle}}.$$

Table 2.2: Network Configuration Parameters

	WLAN parameters
Payload size	1500 bytes (No A-MPDU), variable (A-MPDU)
A-MPDU duration	1024 or 10240 μs
Preamble duration	20 μs
SIFS, DIFS	10, 28 μs
CW$_{min}$, CW$_{max}$	16, 1024
	BT parameters
Duration of slot	625 μs
CW$_{min}$, CW$_{max}$	16, 1024
Packet type	DH
Packet duration	1, 3, or 5 slots
Payload size	27, 183, or 339 bytes
Target cycle	50 ms

2.5 Performance Evaluation

2.5.1 Simulation

We evaluate the performance of OBT schemes through simulation, and examine their throughput in various settings. We consider a network scenario of n WLAN devices and one dual-stack device. All $(n+1)$ devices are assumed to have backlogged WLAN traffic and their destinations are assumed to be evenly distributed over all devices in the network. The dual-stack device has either BE or QoS-sensitive BT traffic. We set the target cycle to 50ms due to two reasons: One is that the atheros device driver running on the conventional dual-stack device supports BT streaming with a TD scheduling period of 45ms. The other is that the target cycle needs to be a multiple of σ for efficient OBT operation.

To observe OBT performance on various settings, we vary WLAN packet duration

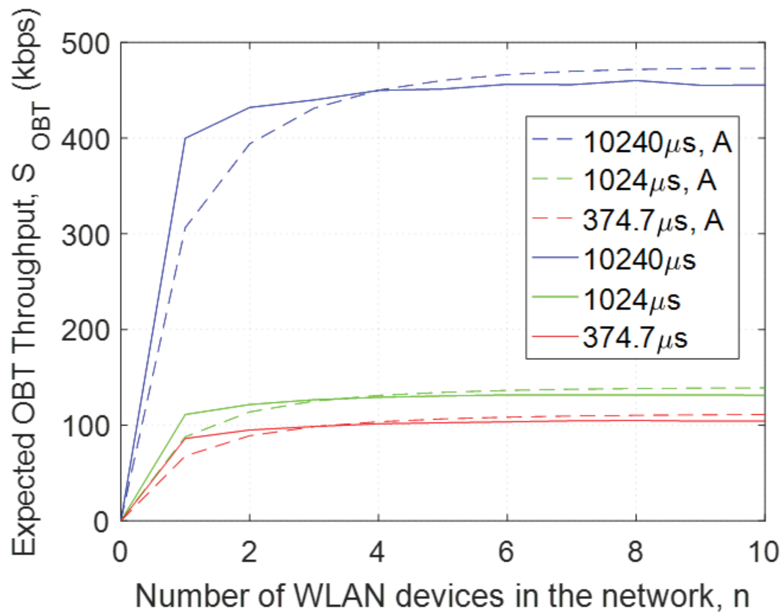
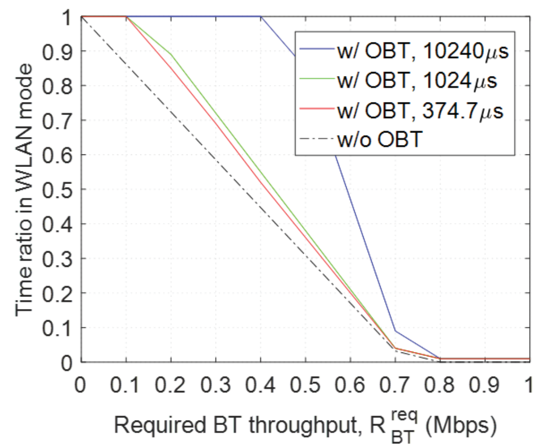


Figure 2.7: OBT throughput of the dual-stack device with OBT-BE scheme ($\gamma = 30$, no RTS/CTS).

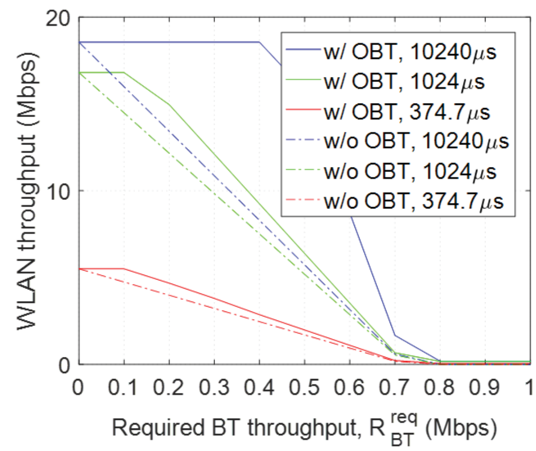
and the number of WLAN devices. For WLAN packet duration control, we apply the aggregated medium access control (MAC) protocol data unit (A-MPDU), which has been widely used for WLAN networks. Without A-MPDU setting, we fix the payload size to 1500 bytes, in which case the WLAN packet duration varies according to the MCS (374.7μs on average). With A-MPDU setting, we fix the WLAN packet duration to 1024μs or 10240μs, in order to show the impact of the WLAN packet duration on the OBT throughput. We present a scheme with the duration (in μs) of A-MPDU.

The performance of OBT-BE is illustrated in Fig. 2.7. In this figure, the results marked with 'A' are obtained from our analysis, and the rest from simulation. This figure shows the expected OBT throughput S_{OBT} as we increase the number of WLAN devices, n. The airtime occupied by WLAN packets increases with n. This allows more deferring time for the dual-stack device, resulting in larger OBT throughput. The results show that OBT-BE achieves the performance close to the analysis and verify our analysis. The gap between analysis and simulation is from the assumption that a WLAN packet collides with constant probability p regardless of the number of retransmissions suffered. This assumption is not valid when n and W_{min} are small, as mentioned in [46].

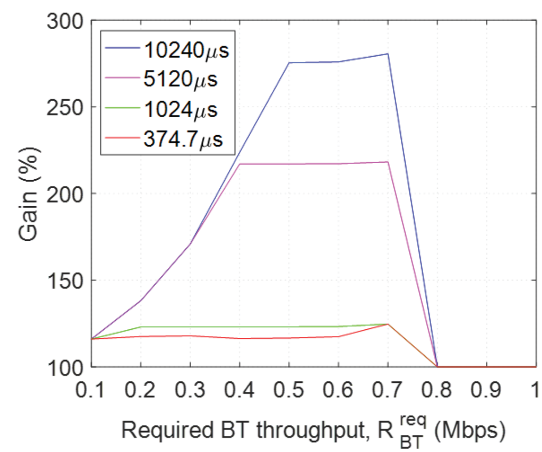
If the dual-stack device adopts the conventional TD scheduling, it obtains BT throughput by sacrificing WLAN throughput as we see in Section 2.2.2. On the other hand, the device with OBT-BE scheme achieves S_{OBT} without sacrificing any WLAN performance via exploiting deferring times.



(a) Time ratio in WLAN mode for the dual-stack device



(b) WLAN throughput of the dual-stack device



(c) WLAN throughput gain of the dual-stack device compared to conventional TD scheduling

Figure 2.8: Performance of the dual-stack device with OBT-QoS scheme ($\gamma = 30$, no RTS/CTS, $n = 5$).

Next, we measure the performance of the dual-stack device with and without OBT-QoS, under different throughput requirements R_{BT}^{req} of BT QoS traffic. We omit the analysis results, since the gap between analysis and simulation is smaller than 3.5% when $n \geq 3$. Fig. 2.8(a) shows the ratio ω of the duration in BT mode to a cycle. Without OBT, for non-zero R_{BT}^{req}, the ratio immediately decreases to transmit BT packets. In contrast, with OBT-QoS, the decrease of ω starts later (i.e., after $R_{BT}^{req} \geq 0.1$), since a small amount of BT traffic can be fully delivered through OBT. Throughout the simulation, we observe that with OBT, the ratio is better (larger) than that without OBT. Also the results show that the gain becomes larger with the A-MPDU size. For example, when A-MPDUs are of length 10240 μs, the BT traffic rate up to 0.4 Mbps can be supported by the OBT scheme.

Fig. 2.8(b) shows that the WLAN throughput according to the required BT throughput. Basically, the WLAN throughput follows the same pattern as in Fig. 2.8(a), since it depends on the time spent in the WLAN mode per cycle. Without OBT, for non-zero R_{BT}^{req}, the WLAN throughput immediately decreases to transmit BT packets. On the other hand, with OBT-QoS, it starts to decrease from a certain R_{BT}^{req}, since a small R_{BT}^{req} can be achieved through OBT. The results show that WLAN throughput is always better with OBT than that without OBT and the gain grows with the A-MPDU size, which is the similar results with the previous one.

Fig. 2.8(c) illustrates the throughput gain of OBT-QoS scheme. The throughput gain is defined as WLAN throughput of OBT-QoS over that of conventional TD scheduling (without OBT). OBT-QoS achieves larger WLAN throughput with longer A-MPDU (up to 288%) owing to the longer duration of deferring time.

2.5.2 Implementation and experiments

In this section, we implement the OBT scheme on a commercial device and evaluate its performance. The BT protocol stack consists of two parts: host part and controller part. The former is implemented in the Linux kernel and the latter is on a hardware chipset or a firmware. An ideal implementation is to embed the OBT scheme in the controller part, since it needs rapid discovery of deferring time and OBT within a deferring time. However, due to the limited accessibility to hardware and firmware[4] we implement OBT-BE on the host part.

Our implementation in the Linux kernel is depicted in Fig. 2.9. In a laptop with an AR9280 802.11n WLAN card and a CSR USB dongle that supports BT 4.0, we modify the kernel and set up the laptop to operate as a dual-stack device with OBT-BE. Without accessing the firmware, we are unable to have a precise time control for scheduling, and face a slower transition problem between the two protocols (compared to the case of implementation in the controller), which incurs additional overhead for OBT. Nonetheless, we show the gain of OBT is substantial and our implementation serves as a good proof-of-concept in practice.

We modify the atheros *ath9k* open source driver [47] to obtain NAV value, which is updated every frame reception. Our BT host generates a BT packet at the start of an OBT opportunity, and the BT controller is responsible for the BT transmission of the packet. In order to transmit multiple BT packets within a single OBT opportunity, the BT host needs to know when the previous transmission ends. Unfortunately, this information is not directly available at the BT host, and we obtain the information by observing the number of completed packets (NumCP) events from the controller. Since the interval of this event is manufacturer specific, we need additional (and random) delay to get the information. The host part in our OBT-BE implementation operates as follows.

1. The host waits until the NAV value is larger than the OBT threshold. If this happens, it switches to BT mode and sets the deferring time to the NAV.

2. It schedules a BT packet transmission that fits in the deferring time, and waits

[4] A lot of dual-stack device are equipped with a BT/WLAN dual-stack module, which integrates BT and WLAN into a single chip. The technique for scheduling both traffic is manufacturer specific and confidential in general since it is a key area of competition among vendors.

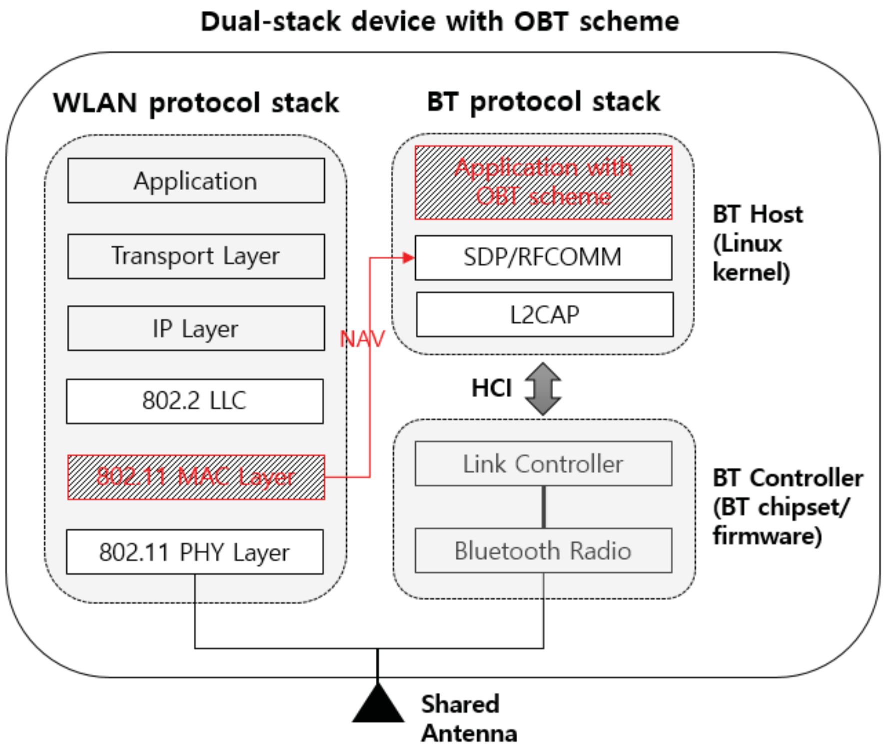
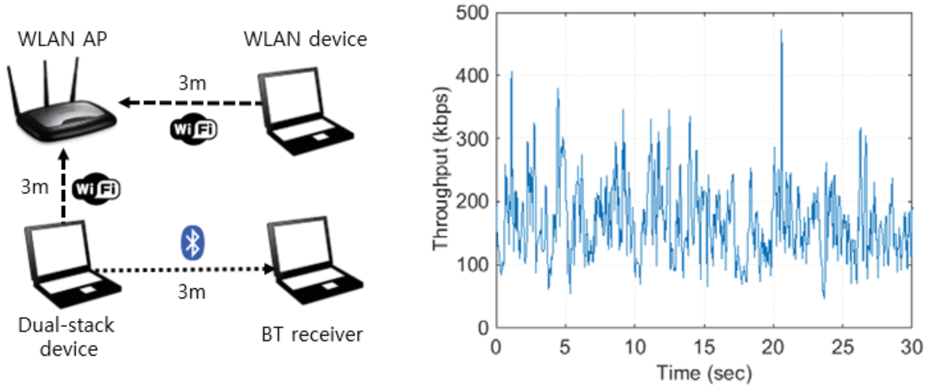


Figure 2.9: Overview of OBT implementation.

for a NumCP event.

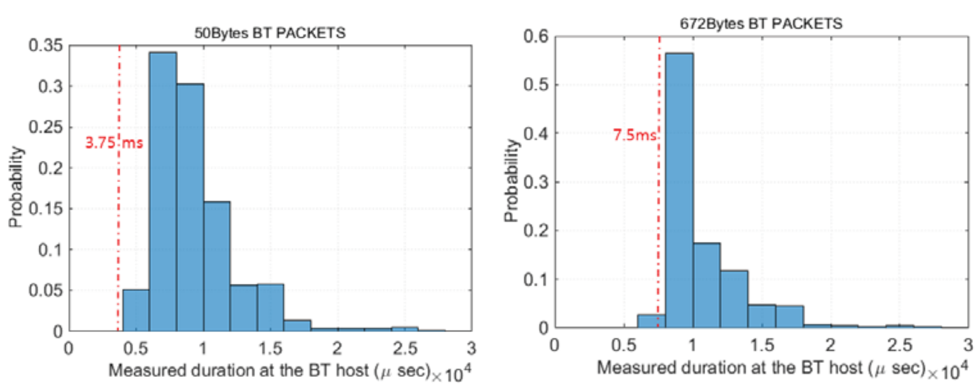
3. If it observes a NumCP event before the deferring time expires, updates the deferring time and goes to Step 2. Otherwise, it assumes that the BT transmission failed, switches back to the WLAN mode, and goes to Step 1.

We consider the network topology shown in Fig. 2.10(a) and measure the OBT throughput of the laptop (dual-stack device) with OBT-BE scheme. There is one (non-dual-stack) WLAN device that transmits WLAN packets to AP with A-MPDU length of 10240μs. The dual-stack device generates WLAN uplink traffic with the same A-MPDU length and transmits BT packets only through OBT. The WLAN device and the dual-stack device are 3m away from the AP and the BT receiver is also 3m away from the dual-stack device. Fig. 2.10(b) illustrates a trace of OBT throughput for 30



(a) Evaluation topology (b) OBT throughput trace

Figure 2.10: Network configuration and the resulting OBT throughput.



(a) 50-byte BT Packets (b) 672-byte BT Packets

Figure 2.11: Distribution of measured transmission durations for different size of BT packets in CSR BT USB dongle.

seconds. The dual-stack device reports its average OBT throughput of the previous interval, which is set to 50 ms We repeat the measurements 10 times. On average, OBT throughput is 183.3kbps, and 42.2% of OBT attempts are successful. We consider a BT packet scheduled at the host as a successful transmission if we observe a NumCP event within the deferring time, and as a failed transmission otherwise. Note that a missed NumCP event means that the host is not sure whether the scheduled BT packet is successfully transmitted within the deferring time. Compared to the simulation result of 399 kbps OBT throughput (when n=1, 10240μs A-MPDU) in Fig. 2.7, the dual-stack device achieves only 45.9% of the throughput. The fluctuation in OBT throughput is mainly due to frequent failures of OBT attempts.

To find the reason for frequent OBT failures, we further investigate the OBT implementation overhead at the BT host. We measure the time until a NumCP event is observed after its scheduling, i.e., transmission time observed at the host. Fig. 2.11 shows the time distribution. In theory, BT transmissions of 50-byte packet and 672-byte packet take less than 3.75ms and 7.5ms, respectively. However, the results show that 95% of 50byte packets take the transmission time of larger than 6ms, and 97% of 672byte packets larger than 8ms, due to the lack of precise time control for scheduling at the BT host. The overhead is mainly due to the waiting for NumCP events at the host and this overhead causes frequent OBT failures. The generation of NumCP events highly depends on chipset manufacturers. In additional experiments with a Broadcom USB dongle, we observed that the waiting time for a NumCP event is about 250ms, which significantly restricts BT transmissions in an OBT opportunity.[5]

2.6 Conclusion

Recent technology advances allow sharing of an antenna between two different network protocols in dual-stack devices. In particular, BT and WLAN protocol stacks are

[5]If one can implement OBT schemes *on the BT controller*, it is possible to exactly know the completion of BT packet transmission at the BT radio without using NumCP events.

commonly combined into a single chip for form factor and cost purposes. Despite independent operation of the two protocols, the ease of information sharing between them affords opportunities to overcome inefficiency arising from TD based scheduling. We propose opportunistic BT transmission (OBT) as a means to increase the capacity of a dual-stack BT-WLAN system in which BT packets are transmitted while it waits for the WLAN channel to be idle in WLAN mode. By opportunistically utilizing intrinsic deferring times of WLAN MAC operation and avoiding interference through adaptive frequency hopping, we showed that significant performance gain is achievable.

Motivated by the measurements from three real-world hotspots, we developed two OBT schemes, each for best-effort traffic and QoS-sensitive (i.e., streaming) traffic, respectively, and evaluated their performance through analysis and simulation. Further, we verified feasibility by implementing the preliminary version of OBT-BE scheme in the kernel space under limited hardware accessibility. Although our proposed OBT schemes are compliant with the standards, there is a room for enhancing our prototype by using fine-grained BT transmission timing information. Since it requires accessibility to the firmware, it remains an interesting implementation problem.

Chapter 3

SplitScan: Distributed Wi-Fi Scanning over Neighboring Stations via Bluetooth Low Energy

3.1 Introduction

Today's mobile devices support various kinds of wireless connectivity, such as long-term evolution (LTE), Bluetooth, Wi-Fi, etc. Especially, Wi-Fi is the most popular wireless protocol due to their free use of unlicensed spectra and support for high-data rate applications. In order to enjoy applications through Wi-Fi connectivity, a mobile device needs to maintain a connection to Wi-Fi access point (AP), and all uplink and downlink traffics pass through the AP. [1] Accordingly, discovering nearby APs and maintaining the list of them are salient issues for Wi-Fi devices to retain Wi-Fi connection or handoff to another better AP.

To establish a Wi-Fi connection with an AP, a mobile device, operating as a Wi-Fi station, requires information of the target AP, such as service set identifier (SSID), supported rates, capability information, etc. 802.11 specification defines *scanning* procedure for a station to obtain necessary information of nearby APs. In a single scanning procedure, a station is required to visit all available Wi-Fi channels one-by-one

[1] We consider only infrastructure mode, since ad-hoc mode is rarely used.

and collects the information from APs in each visiting channel. The frequency of the scanning procedure is implementation specific, however, the procedure should be done consistently since the scanning procedure is essential for maintaining the list of nearby APs which may be a potential candidate of handoff.

The energy consumption of mobile devices has been an important issue since they have limited power. A scanning procedure is one of the energy consuming Wi-Fi operations, since it spends considerable amount of energy to discover APs on every Wi-Fi channel. Furthermore, the scanning procedure delays handoff to another AP by letting the station spend much time on visiting every Wi-Fi channel [48]. Through experiments, we identify that the periodic scanning procedure consumes plenty of time and energy. To address this problem, we propose *SplitScan* protocol, which enables Wi-Fi stations to share their scanning information via Bluetooth Low Energy (BLE) packet exchange. This is possible since the majority of mobile devices, such as smartphones, tablets, and laptops, are equipped with the both Bluetooth and Wi-Fi interface. By sharing their scanning information through SplitScan[2], stations are able to reduce the number of their scanning channels, thereby saving time and energy consumption on Wi-Fi scanning procedure.

Reducing scanning overhead also have received attention from hardware manufacturers. Some commercial Wi-Fi stations intelligently control the interval of the scanning procedure to reduce scanning overhead. For example, having a stable connection with an AP, the station increases the interval of the scanning procedure. This might save scanning overhead, but approaches to increasing the scanning interval include a potential danger of increasing handoff delay due to outdated scanning information when the existing connection is lost unexpectedly. Our solution, SplitScan, tries to reduce scanning overhead, i.e., time and energy consumption, by limiting the number of scanning channels via sharing scanning information with neighboring stations instead of increasing scanning interval. Hence, the Wi-Fi stations running SplitScan maintain

[2] In this paper, we use SplitScan as an abbreviation of SplitScan protocol.

up-to-date AP-existing channel information even with the reduced scanning overhead. Our contributions in this paper are two-fold:

- We propose SplitScan protocol. It enables stations to reduce their scanning overhead in terms of time and energy consumption. Eventually, SplitScan extends a life-time of Wi-Fi station and leads to fast association or handoff.

- We evaluate the performance of SplitScan through simulation and implementation in a testbed.

The rest of this chapter is organized as follows. Section 3.2 presents scanning background and our motivation. We develop the proposed protocol, SplitScan, in Section 3.3. We provide performance evaluation in Section 3.4, followed by concluding remarks in Section 3.5.

3.2 Background and motivation

We briefly overview Wi-Fi scanning procedure operations of 802.11 specification. Then, we identify its limitation through experiments.

3.2.1 802.11 standard scanning

Two kinds of scanning mode, i.e. active scanning and passive scanning, are defined in 802.11 specification. Using passive scanning, the station shall listen to the channel for no longer than *MaxChannelTime*. If it hears a beacon from AP, then it will obtain the information of the AP included in the received beacon. Using active scanning, the station sends the probe request frames and waits for the corresponding probe response frames from nearby APs. When the station sends the probe request frames, it set and start a probe timer. If the station receives at least one probe response before the the probe timer reaches *MinChannelTime*, then it waits more probe responses on the channel until the probe timer reaches *MaxChannelTime*. On the other hand, if the

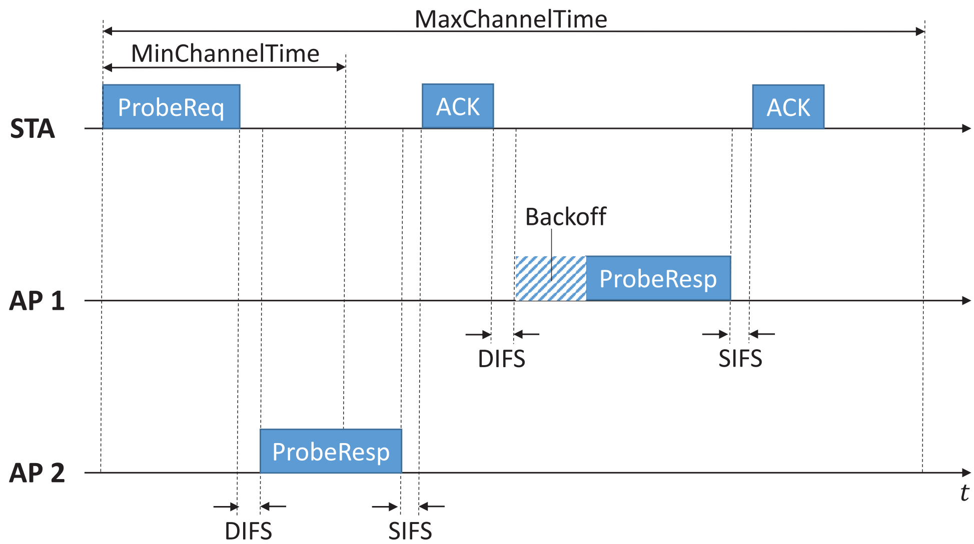


Figure 3.1: Active scanning operation

probe timer reaches *MinChannelTime* without receiving any probe responses, then the station scans the next channel. Fig. 3.1 shows the details of active scanning operation. Active scanning is faster than the other since it triggers response from APs instead of passively waiting beacons, but prohibited in some frequency bands or some Wi-Fi regulatory domains.

Since a station wants to obtain the information of nearby APs in all Wi-Fi channels in general, a single scanning procedure requires a station to scan every channel one by one. In each channel, the station should employ either of the scanning mode. Since active scanning is preferred due to its speed, the station uses active scanning unless it is prohibited by the regulation. A station may scan only a subset of Wi-Fi channels instead of scanning all channels. This might reduce the time spending on the Wi-Fi scanning procedure, but result in loss of AP information on the non-scanned Wi-Fi channels. The *scanning time* of the station is defined as the sum of time consumption in each channel during the scanning procedure and it depends on the set of scanning channels and the regulation of the channels.

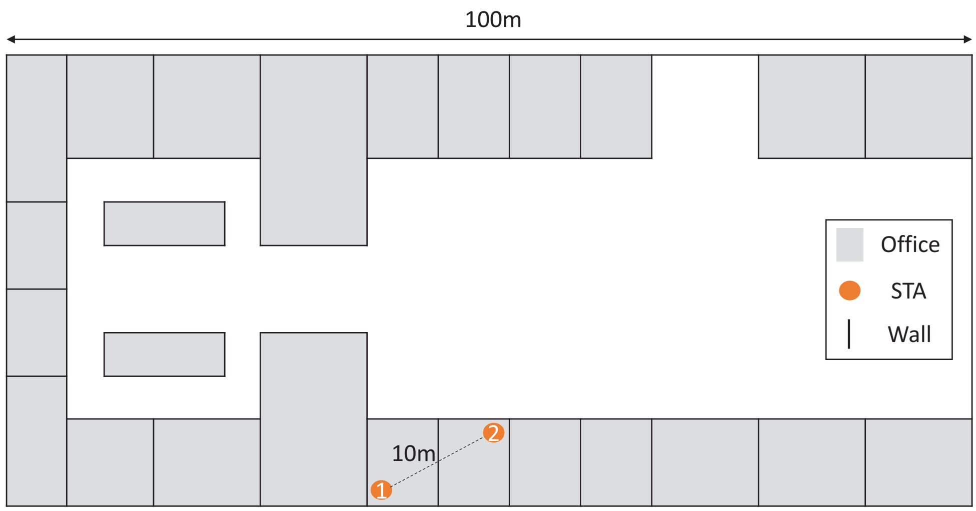
Table 3.1: Scanning overhead of the station

	Average
Scanning time (s)	2.891
Wasted scanning time (s)	2.094
Energy consumption (mJ)	793.1

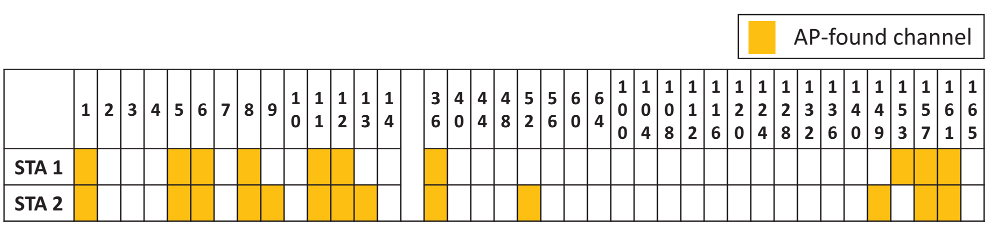
3.2.2 Scanning performance of 802.11 standard scanning

Under the 802.11 standard scanning, a station scans all Wi-Fi channels, and it may cause time and energy waste. We show that this wastage commonly occurs through measurement. We put one laptop equipped with Qualcomm Atheros AR9580 Wi-Fi network adapter in the office. The laptop is unassociated and trigger its scanning procedure with the fixed scanning period. The *scanning period*, defined as a time interval between two consecutive scanning procedure, of the laptop is set to 60 s (default setting of mac80211). According to the original 4.4.33 kernel codes, the laptop stays for the channel for 56 ms for a active scanning, 108 ms for a passive scanning. We change the channel staying time for a active scanning to 30 ms [18] by modifying *mac80211* kernel code, since the default value is not realistic. The experiment results are averaged over 24 hours.

Fig. 3.1 shows the scanning overhead of the laptop. It consumes around 4.8% (2.891 s per 60 s) of its operation time. We calculates the energy consumption during the scanning procedure [49] and BT power consumption parameters are obtained from [50]. by measuring the transmission time of probe requests, receiving time of probe responses, and the idle listening time. The laptop consumes significant amount of energy (793.1 mJ) on the scanning procedure, mostly in idle listening state [51] for waiting probe responses or beacons. We calculate the wasted scanning time where the wasted scanning time denotes the sum of scanning time on the channel without AP. The laptop shows high wasted scanning ratio (larger than 72%), since it scans all Wi-Fi channels.



(a) Topology map



(b) AP-found channel list of two stations

Figure 3.2: Similar AP-found channel list of two geographically close stations

3.3 SplitScan Protocol

We solve the problem of scanning overhead by introducing the SplitScan protocol. Suppose that stations are allowed to transmit packets via BLE interface containing their own Wi-Fi scanning information. Then, instead of scanning all Wi-Fi channels, stations will scan the reduced set of channels using information from nearby stations. Since BLE advertising packets are broadcast, any station around the sender can receive the packet. We first verify the feasibility of SplitScan from real-world measurements and explain the SplitScan in detail.

3.3.1 Potential of SplitScan

To identify the potential gain of cooperation between stations, we add one more laptop 10 m away from the existing one as illustrated in the Fig. 3.2(a). We compare the the AP-found channel list, the set of channels that the laptop found at least one AP on, of the two stations obtained from the their scanning procedures. As illustrated in the Fig. 3.2(b), the both laptops find APs in the almost same channels. Since they are geographically close each other, they are likely to see the almost same set of APs. Thus, one Wi-Fi station is able to share its AP-found channel list to the other station, then the other station may reduce the set of scanning channel set using the shared information, and vice versa.

We also measured the number of stations seen by the laptop 1 to verify how many Wi-Fi stations are able to share their scanning results potentially. The measurement is done by capturing probe requests whose Wi-Fi signal strength larger than -70 dBm (close enough to sharing information through BLE) in the Wi-Fi channel 1. We count the number of different senders of probe requests per 60 s for a day. Fig. 3.3 shows that the station 1 has 4.3 neighboring stations in average. In the daytime, the number of neighboring stations fluctuate since people with mobile devices wander around. Since stations currently follows 802.11 standard, they independently run its own scanning

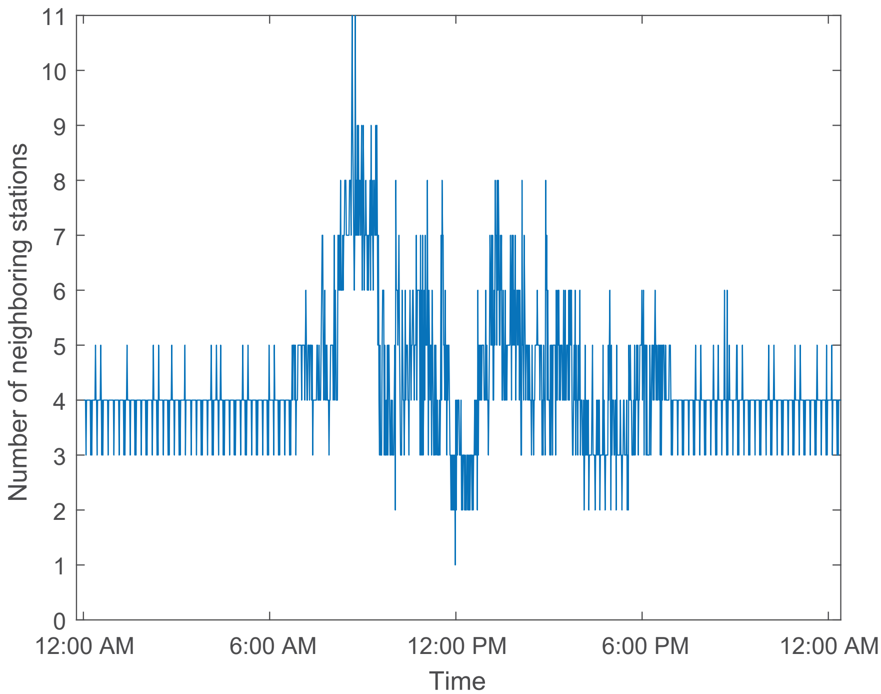


Figure 3.3: Average number of neighboring stations

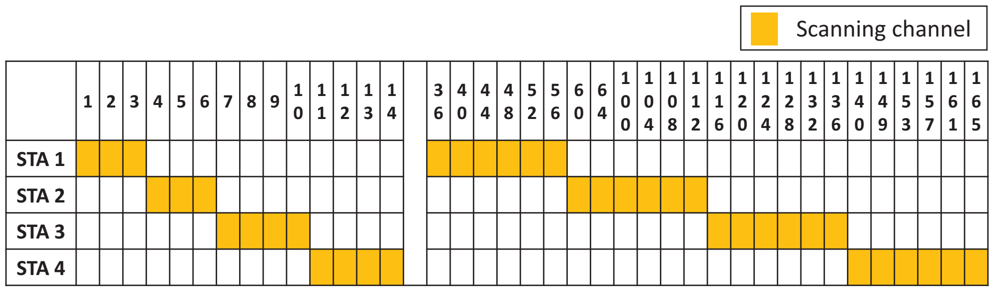


Figure 3.4: Example of sharing scanning information among 4 stations

procedure without sharing information with neighboring stations even though they may see the similar set of APs.

If multiple devices share their scanning result efficiently, then the scanning time dramatically reduces because the devices reduce the number of scanning channels. For instance, 4 stations scan only $1/4$ of the total Wi-Fi channels as depicted in the Fig. 3.4 and share the results through BLE thereby saving $3/4$ of the scanning time and the corresponding scanning energy compared to scanning all Wi-Fi channels.

In summary, a Wi-Fi station consumes much time and energy for scanning procedure, especially in 802.11 standard scanning. We believe the amount of scanning channels can be reduced by sharing scanning information with neighboring stations. This approach is feasible since geographically close stations are likely to see the same APs when the density of stations is high enough to share scanning information through BLE.

3.3.2 Overview of SplitScan

SplitScan protocol is designed to distribute Wi-Fi channels needed to be scanned into neighboring stations and let them share the scanning results via BLE packets to reduce scanning time/energy overhead as depicted in the Fig. 3.5. There are three principles to design SplitScan protocol. First, SplitScan protocol operates in a distributed way. A centralized approach is effective on fair scanning channel distribution among

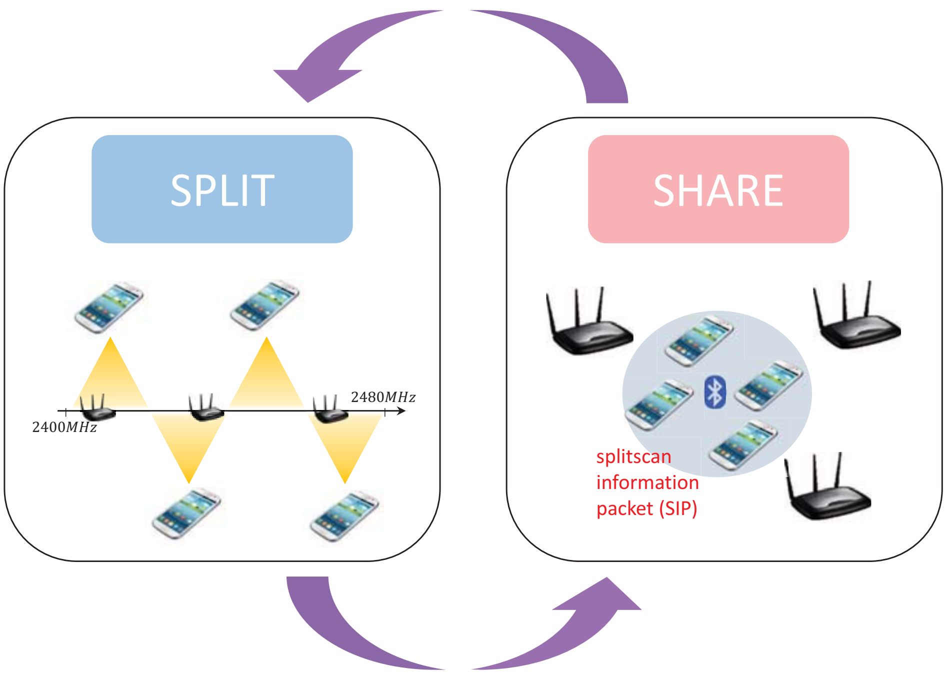


Figure 3.5: Overview of SplitScan

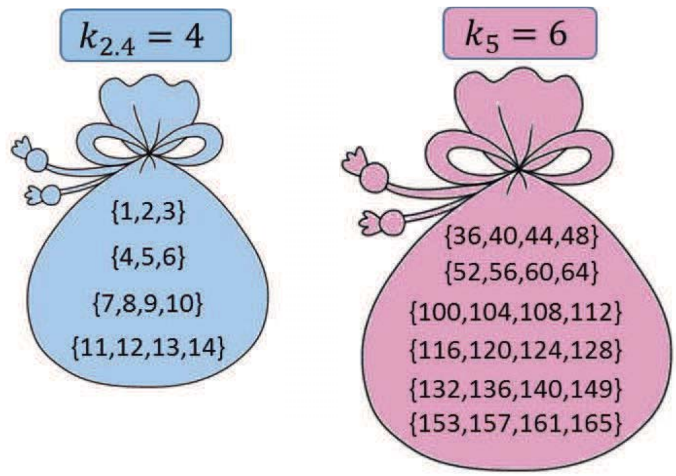


Figure 3.6: Definition of channel set

stations, but the management of the coordinator and the dissemination of scanning channel assignment to the other stations are huge burden on the network. Second, a single SplitScan procedure should update the scanning result of all Wi-Fi channels. In other words, one SplitScan procedure should not miss the scanning result of any channels. Third, SplitScan protocol fairly distributes the total Wi-Fi scanning channels among stations. For example, if 4 stations are sharing their scanning information via SplitScan, they scan $\frac{1}{4}$ amount of Wi-Fi channels each. In the following, we explain the details of SplitScan protocol.

In 802.11 standard scan, a station scans 38 Wi-Fi channels, 14 channels at 2.4 GHz and 24 channels at 5 GHz [3]. We call these 38 Wi-Fi channels as the total Wi-Fi channel set. For convenience, we divide the total Wi-Fi channel set into $k = k_{2.4} + k_5$ subsets, $k_{2.4}$ subsets at 2.4 GHz and k_5 subsets at 5 GHz. For example, when $k_{2.4} = 4$ and $k_5 = 6$, the total Wi-Fi channel set is divided as $\{\{1,2,3\},\{4,5,6\},\{7,8,9,10\},\{11,12,13,14\}\}$ at 2.4 GHz and $\{\{36,40,44,48\},\{52,56,60,64\},\{100,104,108,112\},\{116,120,124,128\},$ $\{132,136,140,149\},\{153,157,161,165\}\}$ at 5 GHz where the elements in the sets

[3]We use United States FCC regulatory domain

represent the Wi-Fi channel index. Let us define these subsets as *channel sets* (CSs). Fig. 3.6 shows the channel set definition when $k_{2.4} = 4, k_5 = 6$.

Instead of scanning the total Wi-Fi channel set, a station with SplitScan protocol scans the channels only included in two group of channels. One is defined as *scanning channel set* (SCS), which is the channel set the station takes charge of. The SCS of the station is determined as a union of CSs selected by itself among all k CSs. The other is *received scanning result* (rx-SR), which is defined as the channels notified by nearby station that there is at least one AP in the channel. We express SCS of the station as k-bit boolean value, the first $k_{2.4}$ bits for 2.4 GHz channels and the rest k_5 bits for 5 GHz channels. For example, when $k_{2.4} = 4$ and $k_5 = 6$, then the SCS of the station can be expressed as 1100111000 and the SCS value means that the station in charge of scanning the first two CSs in 2.4 GHz and the first three CSs in 5 GHz. Thus, the station will scan the channel index from 1 to 6 and from 36 to 112 with the SCS, 1100111000. We express rx-SR of the station as a 38 bit boolean value. The least significant bit of rx-SR maps to the first channel, channel 1, and the most significant bit to the last channel, channel 165.

Initially, rx-SR of the station is set to all zero bits. Every time the station receives SplitScan information packet (SIP) through BLE, it updates rx-SR and *received SCS* (rx-SCS). The former is updated as an bit-wise OR operation of existing rx-SR and the received scan result (SR) field in the SIP payload. The latter is also updated as an bit-wise OR operation of rx-SCS and the SCS field in the SIP payload. Before starting its scanning procedure, the station checks rx-SR. It needs to scan the channel whose bit is enabled in rx-SR since it is believed that there is at least one AP in the channel. Moreover, the station also need to scan the channels included in its own SCS, which is derived from the rx-SCS via SCS selection algorithm. Therefore, the station generates 38-bit a *scanning channels* (SC) value, which is the union of the channels included in the SCS and the channels enabled in rx-SR and then only scans the channels in SC.

After finishing SplitScan procedure, the station should transmit SIP via BLE. The

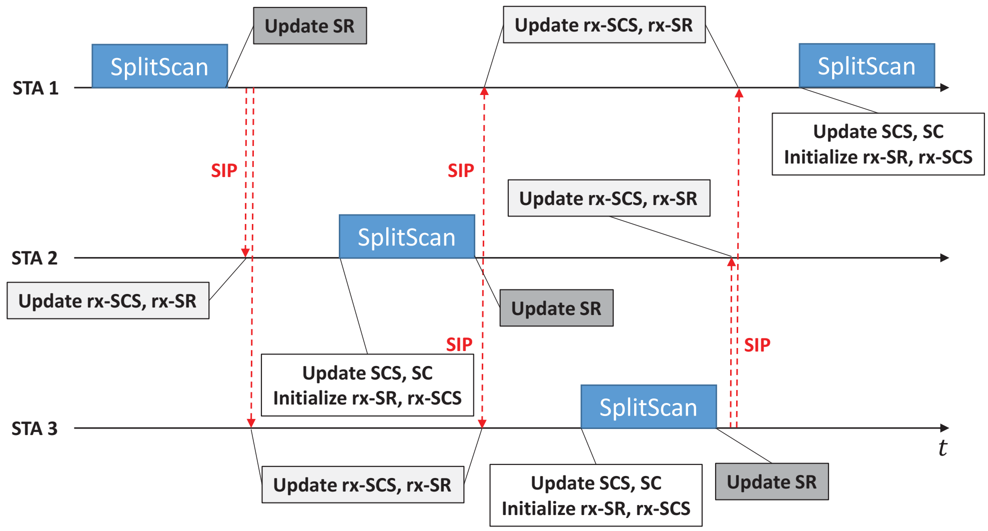


Figure 3.7: Example of SplitScan protocol operation

SIP contains its SCS and SR. SR is also expressed as 38-bit boolean array, and represents the AP existence of each channel obtained from the SplitScan procedure. The enabled i-th bit in SR means there is at least one AP in the corresponding channel i.

Fig. 3.7 shows the example of SplitScan protocol operation when three stations are running SplitScan protocol. In summary, a station maintains two boolean arrays, rx-SR and rx-SCS, and update them whenever it receives SIP. Before starting SplitScan procedure, the station update SCS from rx-SCS and and scans the union of channels included in the SCS and rx-SR, named SC.

3.3.3 Scanning channel set (SCS) selection algorithm

The most important part in SplitScan is how to assign the Wi-Fi channels fairly to the stations in close proximity in a distributed way. We achieve it through *SCS selection algorithm*. In SplitScan protocol, a station maintains two values, rx-SCS and N_i, the set of neighboring stations who send SIP to station i, in a previous SplitScan period. The station i decides its own SCS based on the rx-SCS and N_i through the SCS selection algorithm.

The SCS selection algorithm is run by each stations and its goal is to minimize the size of SCS, i.e. the number of scanning channels in the SCS, since the time and energy consumption of Wi-Fi scanning is directly proportional to the number of scanning channels. According to the SplitScan design principles, there are two important constraints. First, the station i should obtain new scanning results of all Wi-Fi channels. This can be expressed as the equation below.

$$SCS_{all}^{m} = rxSCS_{i}^{m} \mid SCS_{i}^{m} \quad (m = 2.4 \text{ or } 5) \tag{3.1}$$

where

$$rxSCS_{i}^{m} = \big|_{j \in N_i} \; SCS_{j}^{m}, \tag{3.2}$$

SCS_{all}^{m} denotes the k_m bits of 1 s, $rxSCS_{i}^{m}$ represents the received SCS of station i at m GHz band during the previous scanning interval, SCS_{i}^{m} represents the SCS of station i at m GHz band, and the symbol | denotes the bit-wise OR operation. Equation 3.1 means that a certain SCS not scanned by station i's neighboring stations should be scanned by the station i, i.e. should be included in the SCS of station i, to obtain new scanning results of the all Wi-Fi channels. Equation 3.2 means that $rxSCS_{i}^{m}$ is an accumulated information of received SCSs from neighboring stations $j \in N_i$. That is, if a certain SCS is scanned by at least one of its neighboring stations j, the station i classify it as the scanned CS and will not scan again.

Second, the station i should fairly share the SCSs with its neighboring stations. This can be expressed as

$$n(SCS_{i}^{m}) \geq round(\frac{k_m}{size(N_i) + 1}) \tag{3.3}$$

where $n(X)$ is the number of 1s in set X and the $size(S)$ denotes the number of elements in the set S. Equation 3.3 means that the station i scans at least $\frac{k_m}{size(N_i)+1}$ number of SCSs, which is the average number of SCSs, assuming that k_m CSs are evenly divided into station i and its neighboring stations who send SIPs to the station i.

Station i runs the SCS selection algorithm right before starting its SplitScan procedure. Based on the constraints, the algorithm operates as follows. Station i applies the first constraint and generates candidates of SCS_i^m. The required input for applying the first constraint is N_i and $SCS_j^m(j \in N_i)$, the received SCS from neighboring stations. Then, station i calculates the minimum number of SCSs of station i on the m GHz band, $n_{min}(SCS_i^m)$, which satisfies the Equation 3.3 and minimizes $n(SCS_i^m)$. Lastly, among all SCS_i^m candidates, station i eliminates the candidates whose $n(SCS_i^m)$ are not equal to $n_{min}(SCS_i^m)$. If the remaining candidates of SCS_i^m are multiple then the station i randomly chooses one of them as its SCS_i^m. Algorithm 1 presents the pseudo code of the algorithm.

Algorithm 3 SCS selection algorithm

```
Input: N_i, rxSCS_i^m, k_m for m = 2.4 and 5
Output: SCS_i^m for m = 2.4 and 5
C = ∅
n_min(SCS_i^m) ← round(k_m / (size(N_i)+1)) for m = 2.4 and 5
C ={ SCS_i^m | SCS_all^m = (rxSCS_i^m | SCS_i^m) for m = 2.4 and 5}
for all element c ∈ C do
    Eliminate c if n(c) ≠ n_min(SCS_i^m)
endfor
Select one element as SCS_i^m in C
```

We briefly explain how the station operates with SplitScan protocol. Suppose that the station i periodically triggers SplitScan procedure. It collects SCS_j^m and SR_j^m $(j \in N_i)$ from neighboring stations by receiving SIPs between two consecutive SplitScan procedures. The collected SCS_j^m and SR_j^m $(j \in N_i)$ are accumulated in $rxSCS_i^m$ and $rxSR_i^m$ whenever the station i receives SIP. Before starting the SplitScan procedure, the station i determines SCS_i^m using the SCS selection algorithm. It starts SplitScan procedure on the channels, i.e., visiting channel one by one, included in

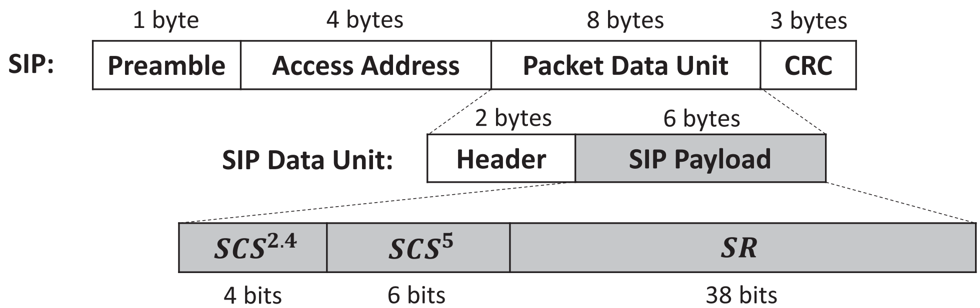


Figure 3.8: SplitScan information packet format

SCS_i^m and enabled in SR_i^m. After finishing the SplitScan procedure, the station i sends its SCS_i^m and its scanning result, SR_i, using SIP through BLE. Neighboring station j of station i updates their $rxSCS_j^m$ and $rxSR_j^m$ if they receive the SIP from station i. Every SplitScan-enabled station independently operates the same as station i.

3.3.4 SplitScan Information Packet

As a means of exchanging Wi-Fi scanning information, we design a SIP. It contains two values, SCS and the SR. As mentioned before, SCS requires k bits and SR requires 38 bits, so the required payload length is 38+k bits. We select $k_{2.4} = 4$ and $k_5 = 6$. Thus, SIP payload requires 48 bits (= 6 bytes). The SIP packet format is illustrates in Fig. 3.8.

The SIP can be generated and interpreted as follows. The SIP sender enables the s-th bit of $SCS^{2.4}$ and SCS^5 if it scans the corresponding SCS and also enables the l-th bit of SR if it found at least one AP on the channel l. The SIP receiver believes that the SCS are scanned by the SIP sender if the corresponding bits are enabled in the received SCS field and the AP exists on the channel l if the l-th bit enabled in the received SR field. For example, if a station receives SIP payload 1000 100000 1010...000, that the station believes that the SIP sender scans the first SCS, i.e., first 3 channels, in 2.4 GHz, the first SCS, i.e., the first 4 channels, in 5 GHz and there are APs only in the

Table 3.2: Simulation configuration parameters

Settings	Value
Simulation Time	600 s
Number of Wi-Fi Channels	14 in 2.4 GHz, 24 in 5 GHz
Number of APs	10
Number of Stations	30 (3 per AP)
Transmission Range of BLE/Wi-Fi	30 m / 100 m
Wi-Fi Scanning Interval	60 s
Active / Passive Channel Time	30 ms / 108 ms
Min / Max Channel Time	30 ms / 30 ms
Number of SCS	4 in 2.4 GHz, 6 in 5 GHz
BLE Advertising Interval/Timeout	170 ms / 60 s
BLE Scan Window/Interval	170 ms / 60 s

channel 1 and 3.

3.4 Performance Evaluation

3.4.1 Simulation

We evaluate the performance of SplitScan protocol through simulation, and examine its performance. We consider a network scenario of 10 APs and 30 stations, 3 associated stations per AP. APs are deployed randomly in 100 m radius from the origin and stations are deployed randomly in 100 m from the their associated APs as depicted in Fig. 3.9. Operating channels of APs are randomly selected among 38 channels. Stations are assumed to scan periodically with 60 sec interval and total simulation time is 600 sec. BLE operation parameters - advertising interval/timeout and scan window/interval are derived from [52] to minimize energy consumption while guaranteeing BLE packet delivery. We assume that there are no BLE/Wi-Fi channel loss. The

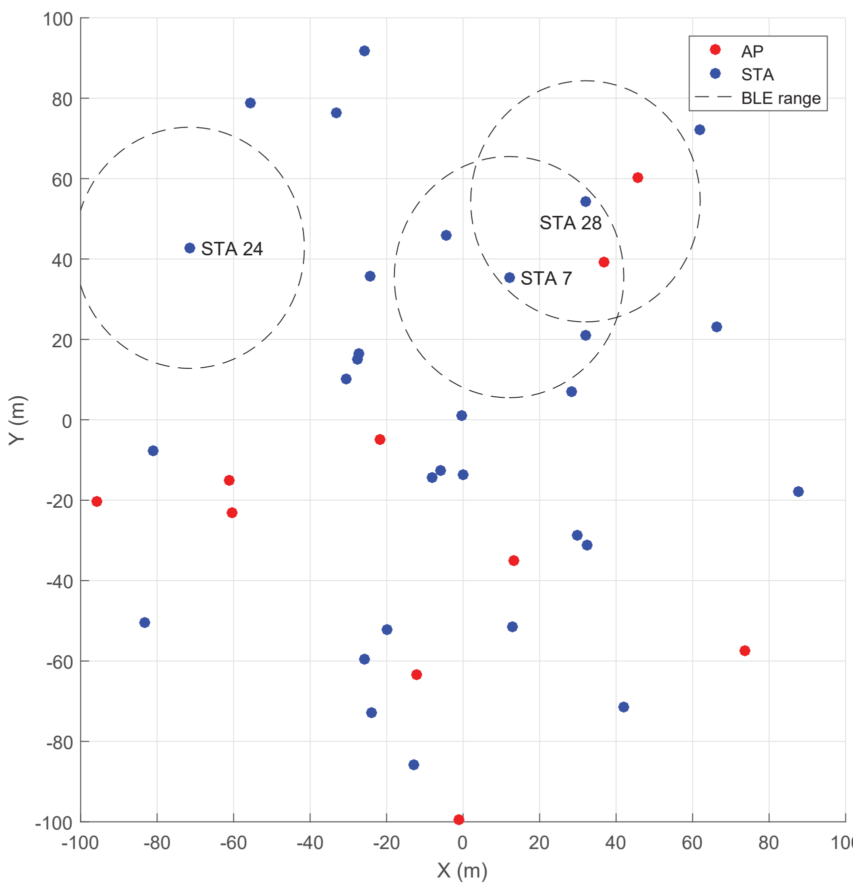


Figure 3.9: Simulation topology

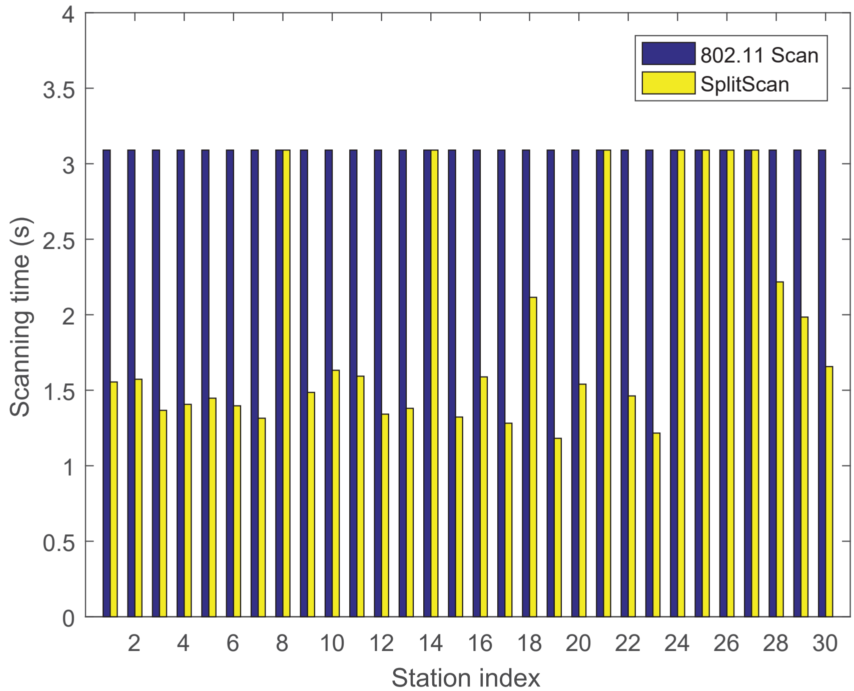


Figure 3.10: Average scanning time of every station

detailed configuration parameters are listed in Table 3.2.

Fig. 3.10 shows the average scanning time of all stations with two different scanning procedures, 802.11 standard scan and SplitScan. In average, 802.11 standard scan procedure consumes 3090 ms, while SplitScan procedure consumes only 1890 ms, 61.15% of standard scan. Hence, stations spend 5.15% of their operation time for scanning procedure with 802.11 standard scan, but spend less than 3.2% with SplitScan in average. There are some interesting cases in these figures. For example, station 24 does not obtain any scanning time gain from SplitScan since it does not have any neighboring stations in the BLE transmission range. Station 8, 14, 21, 25, 26, 27 also do not obtain scanning time gain with SplitScan due to the same reason with station 24. Station 28 obtains a relatively small scanning time gain due to topological reason. The station 28 may expect to scan half of Wi-Fi SCSs, since it has only one neighbor sta-

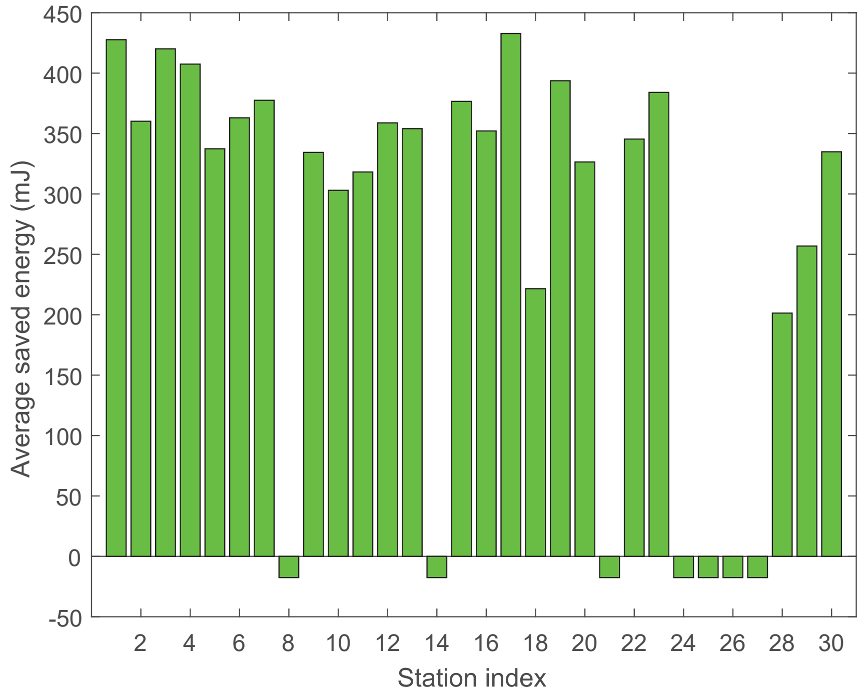


Figure 3.11: Average saved energy of each station

tion, station 7. However, the station 7 scans only 1/4 of Wi-Fi SCSs, since it receives 3 SIP packets from neighboring stations. Thus, station 28 need to scan the rest of 3/4 of Wi-Fi SCSs not scanned by station 7. As a result, The gain of station 28 becomes relatively small.

Next, we measure the average saved energy of the scanning procedures. It is calculated as the $\frac{\text{Energy consumption in 802.11 standard scan} - \text{Energy consumption in SplitScan}}{\text{Number of scanning procedures}}$. We use the energy model used in the Section 3.2.2. Fig. 3.11 shows the energy consumption of each station. The average energy gain per scan 262.1 mJ, i.e., the average power gain is 4.37 mW ($\simeq$ 262.1 mJ$/$60 s). Some stations such as station 8, 14, and so on, cannot achieve positive power gain, since they has no neighboring station. The station 28 saves only 43.08 mJ of energy per scanning procedure since it has only one neighbor station and scans 3/4 of Wi-Fi SCS due to topological reason. Stations with small num-

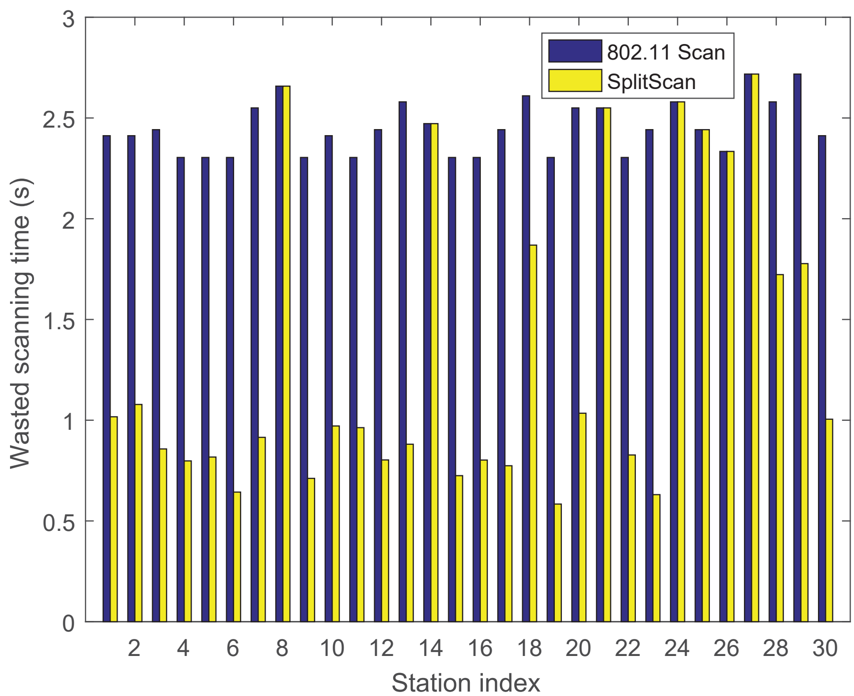


Figure 3.12: Wasted scanning time of every station

ber of neighboring stations spend energy on transmitting, waiting and listening SIP via BLE, but likely to obtain a little Wi-Fi information from neighboring stations to get enough scanning time gain.

Fig. 3.12 illustrates the wasted scanning time of each station. Compared to SplitScan, 802.11 standard scan wasted 83.92% of time more on scanning the channels without APs. Since stations with SplitScan do not scan the channels notified by neighboring station that there is no AP, thus the scanning time wastes of stations reduces.

We evaluate the performance of the SCS selection algorithm by calculating the fairness index of $n(SCS)$ of stations. The fairness index of scanning channel set is defined as

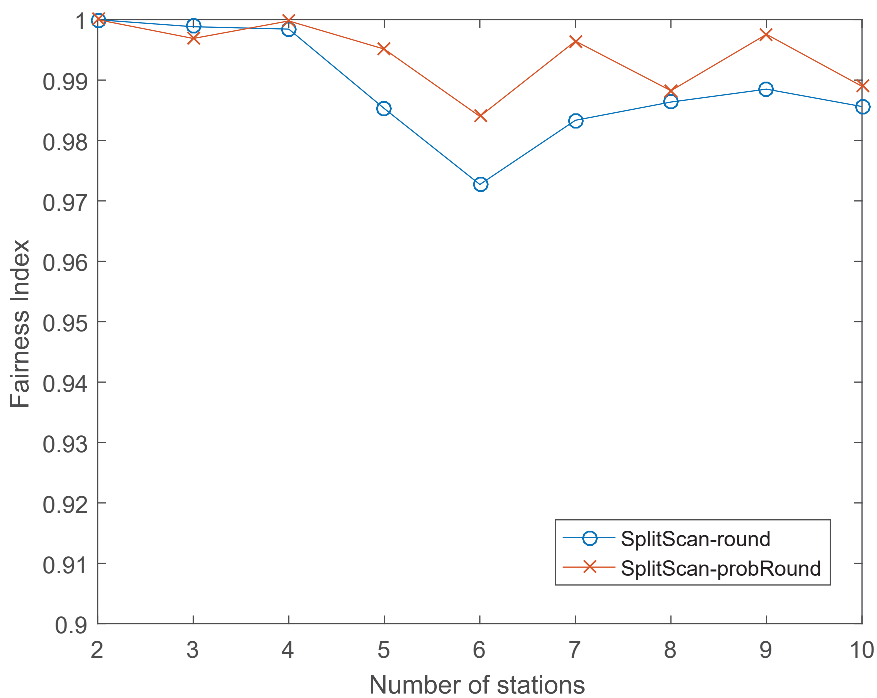


Figure 3.13: Fairness index of the number of SCSs of n SplitScan stations

Table 3.3: Unfair SCS saturation example in 2.4 GHz band

Time instance	Station 1	Station 2	Station 3
t_0	1100	0011	-
t_3	-	-	1111
t_1	0001	-	-
t_2	-	0010	-
$t_3 + T$	-	-	1100
$t_1 + T$	0001	-	-
$t_2 + T$	-	0010	-
$t_3 + 2T$	-		1100

$$\rho = \frac{\{\sum_{i=1}^{n} SCS_i\}^2}{n \sum_{i=1}^{n} {SCS_i}^2} \tag{3.4}$$

where the allocation of $n(SCS)$ is given as $\{n(SCS_1), n(SCS_2), n(SCS_3), ..., n(SCS_n)\}$. ρ is called Jain's fairness index and one of the quantitative fairness measures. The value is in the range of $[1/n\ 1]$. If all values are equal, $\rho = 1$. If a certain one takes all, and the others have nothing, then $\rho = 1/n$.

Since the number of SCS affects the scanning time and energy, fair distribution of SCS is important issue of SplitScan performance. In the aforementioned topology, the fairness index is 0.7942. Note that a station with many neighbor stations scans small number of SCSs, but one with small number of neighbor stations scans large number of SCSs. Thus, the fairness index of $n(SCS)$ depends on the topology a lot and the fairness index will be larger, if the density of stations are uniform, i.e., evenly distributed like a grid topology.

Fig. 3.13 shows the fairness index of $n(SCS)$ according to number of neighboring stations, n, assuming that all n stations are in the BLE range, i.e., all stations receive $n - 1$ SIPs from neighboring stations in their scanning interval. There are two results labeled with 'SCS-$orig$' and 'SCS-$probRound$'. The former is the result applying the original SCS selection algorithm. In the simulation, we found some corner cases, which result in a unfair SCS allocation. We call this problem as *unfair SCS saturation*. According to the examination, the problem only happens when $\frac{k_m}{size(N_i)+1}$ is not an integer.

The exact reason of this problem is illustrated in the Table. 3.3. Suppose that station 3 starts the SplitScan procedure when station 1 and station 2 fairly scan half of the total Wi-Fi channels. We assume that the scanning period, T, of stations are the same. Let us denote t_0 as the initial time instance and t_i as the time instance of starting the scanning procedure of station i. The scanning order is given as station 3 $\rightarrow$ station 1 $\rightarrow$ station 2, i.e., $t_0 < t_3 < t_1 < t_2$.

At t_0, there are only station 1 and station 2, and they evenly distribute their SCSs. Station 3 start its SplitScan procedure with $SCS_{all}^{2.4} = 1111$ (initial SCS setting) and scans all SCSs at t_3 in 2.4 GHz. Since the station 1 receives 2 SIPs from station 2 and 3, $size(N_i) = 2$ at t_1. Since $k_m = 4$ in 2.4 GHz, $n(SCS_1^{2.4}) = round(\frac{4}{2+1}) = 1$. Thus, the station 1 chooses $SCS_1^{2.4} = 0001$ among 4 candidates, $0001, 0010, 0100, 1000$, by applying the SCS selection algorithm. The station 2 also receives 2 SIPs from station 1 and 3, $n(SCS_2^{2.4}) = round(\frac{4}{2+1}) = 1$. Thus, the station 2 selects $SCS_1^{2.4} = 0010$ among 4 candidates at t_2. At $t_3 + T$, the station 3 receives two SIPs from station 1 and 2. Since they scans only one CS each, the station 3 need to scan the rest CSs to obtain the results of all Wi-Fi channels. As a results, $SCS_3^{2.4} = 1100$. Once the SCS distribution is determined to $0001, 0010, 1100$, respectively for stations, then it cannot be changed as time goes by. As a result, station 3 always scan more CSs than the others. This is how the unfair SCS saturation occurs.

To resolve the unfair SCS saturation, we introduce a simple function $probRound$. It probabilistically determines whether ceiling or flooring the input as the output. The definition of $probRound$ function is as follows,

$$probRound(x) = \begin{cases} \lfloor x \rfloor & p(x) = 1 \\ \lceil x \rceil & p(x) = 0 \end{cases}$$
$$p_X(x) = \begin{cases} 1 & u \leq x - \lfloor x \rfloor \\ 0 & otherwise \end{cases}, \tag{3.5}$$

where u is the uniform random variable between [0 1). When calculating the minimum limit of $n(SCS_i^m)$ in equation 3.3, we use $probRound$ function instead of $round$ function. There are two reasons for using $probRound$ instead of $round$. First, it can prevent the unfair SCS saturation since it probabilistically determines the output with the identical input. Second, the expected value of $probRound(x)$ is design to be x. If we use $round(x)$, then it always results in $\lfloor x \rfloor$ or $\lceil x \rceil$ and this is inappropriate. Fig. 3.13 shows that SCS-$probRound$ outperforms in terms of fairness since it eliminates the unfair SCS saturation problem. Moreover, the fairness index is always larger than 0.94 with SCS-$probRound$, which means the SCS algorithm fairly distributes SCSs among stations.

3.4.2 Implementation and experiments

In this section, we implement SplitScan protocol on linux-based laptops and evaluate its performance. The laptops are equipped with an AR9280 802.11n WLAN card and a CSR USB dongle that supports BT 4.0 [1]. Our implementation in the Linux kernel is depicted in Fig. 3.14. We add SplitScan layer between Wi-Fi medium access control layer (especially *mac80211* module) and Bluetooth host controller interface (HCI) layer and add functionalities of managing essential variables to mac80211 layer such as SCS, SR and SC. Since the mac80211 module and HCI layer are common to all linux-based Bluetooth/Wi-Fi equipped devices, our implementation is not network interface card dependent. SplitScan layer takes in charge of three functionalities.

1. SplitScan layer generates a SIP packet from the station's SCS and SR and passes

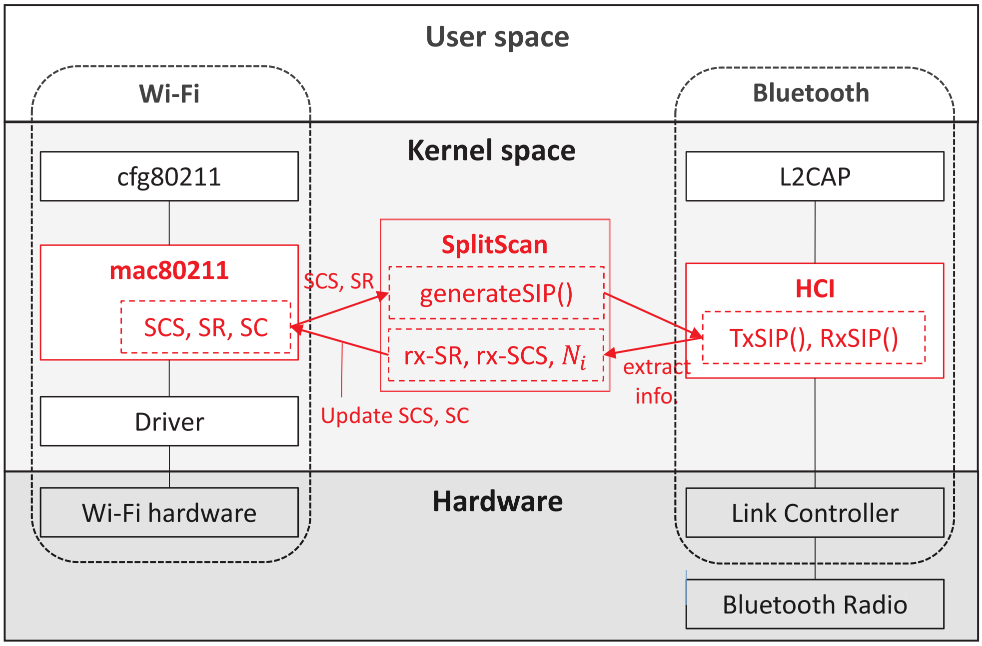


Figure 3.14: Overview of SplitScan implementation

it to HCI layer.

2. SplitScan layer extracts the received SCS and the received SR, i.e., rx-SCS and rx-SR, from the received SIP packet.

3. SplitScan layer updates SCS from rx-SCS through SCS selection algorithm, and also updates SC from rx-SCS and rx-SR.

To send and receive SIP packets, we use BLE advertising packets. In the BLE advertising procedure, there are 4 important operation parameters, scan/advertising interval, scan window, and advertising timeout. We set the advertising interval as same as the Wi-Fi scanning period, since a station transmits SIP after every SplitScan procedure. The other parameters are derived to guarantee a reliable delivery and minimize the energy consumption of station according to [52].

Table 3.4: Scanning overhead of the station

	802.11 standard scan	SplitScan
Scanning time (s)	2.891	1.671
Wasted scanning time (s)	2.094	1.048
Energy consumption (mJ)	793.1	475.11

Two stations experiments

We consider the network topology shown as the same with the topology in Section 3.2.2 and measure the performance of stations with and without SplitScan. The scanning period, channel time parameters, the number of SCS, and BLE operation parameters follow the table 3.2. The results, averaged over 5 hours, summarized in the Table 3.4. A station with SplitScan spends 57.8% scanning time compared to 802.11 standard scan, resulting in reduced energy consumption. Energy consumption of SplitScan reduced to 60% of standard scan. Energy consumption gain is a bit smaller than the time gain, since stations consumes energy on BLE operation, transmitting, waiting and receiving SIP. SplitScan also outperforms in terms of wasted scanning time, since it eliminates scanning the channel without AP notified by the neighboring station.

To verify the detail operation of SplitScan protocol, we take another 6 hour trace in the scenario that the station 1 is running SplitScan while the station 2 turns Wi-Fi on and off in the topology depicted in Fig. 3.2(a). The scanning trace is depicted in the Fig. 3.15. Initially, station 1 is the only station running SplitScan protocol, thus it scans all channels. The station 2 turns on its Wi-Fi and runs SplitScan protocol at 5:55pm, so station 1 and 2 share their Wi-Fi scanning information thereby reducing their scanning time. Note that the scanning time of two stations differ from each other due to asymmetric AP existence. Even though they designed to scan each half of the SCS, their scanning time can be different when AP-existing channels are more in station 1's SCS. In this experiments, all SCSs are evenly distributed to two stations, but SCS of station 2 contains 9 AP-existing channels while the SCS of station 1 contains

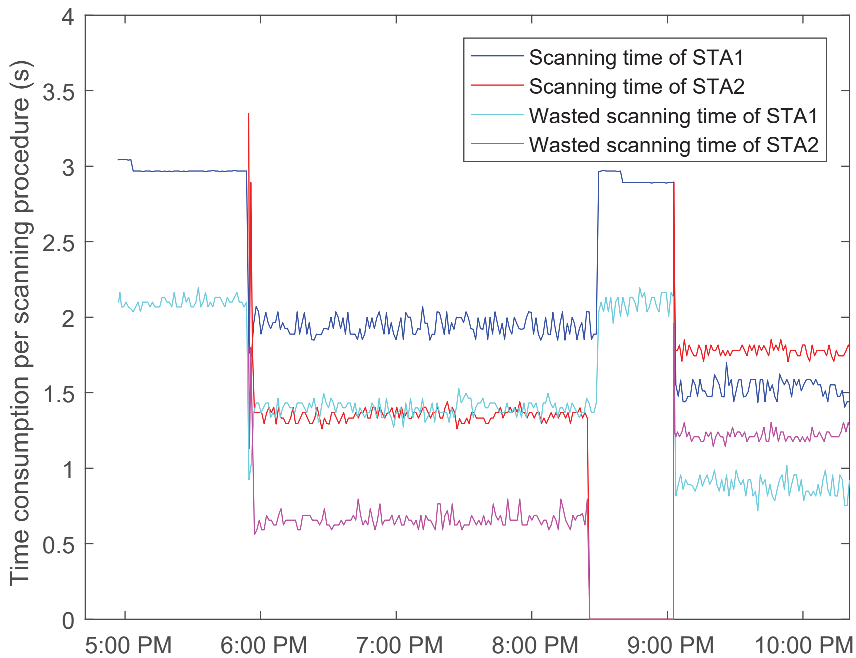


Figure 3.15: Scanning trace of SplitScan

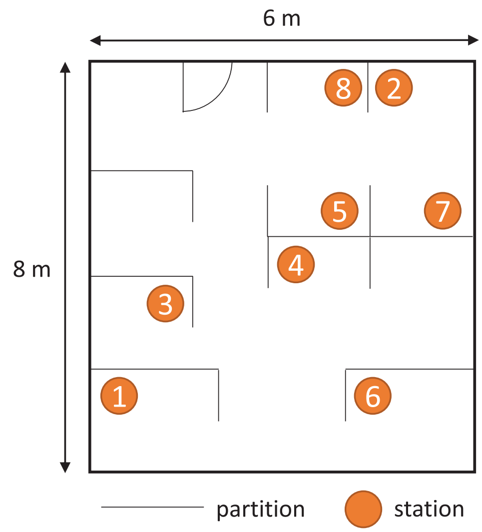


Figure 3.16: Station-dense topology

only 3 AP-existing channels until the station 2 leaves at 8:25pm. As a result, the station 1 need to scan 6 more channels per scanning procedure than station 2, and this makes scanning time difference between them. After the station 2 turns off its Wi-Fi module, the station 1 start scanning all channels again. At 9:00pm, the station 2 turns on its Wi-Fi interface and start SplitScan procedure. In this time, the scanning time of the station 1 is less than the station 2, since the selected SCSs of stations are different from the previous case. The a little fluctuation of scanning time comes from the rx-SR variation. In SplitScan, stations scan the channels enabled in their SCS and rx-SR bit. Since rx-SR from the neighboring station varies with time due to scanning error, the small fluctuation of scanning time occurs. The wasted scanning times are also reduced when two stations are running SplitScan. The amount of wasted scanning times depends on the number of AP-existing channels, thus they are affected by the topology and AP channel configuration.

Table 3.5: Scanning overhead of the station

	802.11 standard scan	SplitScan-2	SplitScan-8
Scanning time (s)	2.891	1.671	0.604
Energy consumption (mJ)	793.1	475.11	175.25

Station-dense experiments

We also consider the network topology shown in the Fig. 3.16 and measure the performance of stations with and without SplitScan. All settings are the same with two station experiments except for the number of stations. In the topology, all stations are close enough to exchange SIP via BLE. In other words, any station receives 7 SIPs from the neighboring stations. Let SplitScan-k denote the scenario with k stations running SplitScan protocol. As summarized in the Table 3.5, a station with SplitScan spends only 20.9% of scanning time compared to 802.11 standard scan, resulting in reduced energy consumption. Energy consumption of the station reduced to 22% of 802.11 standard scan. As expected, the gain of SplitScan increases according to the number of neighboring stations. This is natural since SplitScan splits the scanning channels into neighboring stations. In summary, SplitScan works more efficiently in station-dense environment.

3.5 Summary

We are living in a ubiquitous world. A massive number of mobile devices such as smartphone, tablet, watch, wearable band, etc, are released in the market and people brings them in their lives. Since the mobile devices are power constrained, extending their lifetime is critical issue in various research field. Wi-Fi, the most popular wireless technology due to its license-free usage in these days, also consumes considerable amount of energy in mobile devices. In this paper, we focus on the Wi-Fi scanning procedure which is time/energy consuming, but necessary to maintain wireless connectivity. Motivated by the measurements, we propose SplitScan protocol which reduces scanning time and energy of a Wi-Fi station by sharing information with its neighboring Wi-Fi stations via BLE. Making use of information share, Wi-Fi stations reduce the number of scanning channels thereby saving time and energy consumption of Wi-Fi scanning procedure. We evaluated the performance of SplitScan protocol through the simulation and verified feasibility by implementing it in the kernel space of the commercial devices. In the both simulation and implementation, SplitScan outperforms in terms of scanning time and energy consumption. SplitScan do not need any changes in the infrastructure (APs) and requires only a small available memory to add SplitScan layer at the station. Although our SplitScan protocol is compliant with the standards, there is a room for enhancing its energy efficiency by controlling BLE operation time intelligently. Moreover, the performance in heterogeneous scanning period among stations remains as an interesting work.

Chapter 4

DP-MAC: Uni-directional TDMA MAC for Real-time Streaming in Multi-hop Wi-Fi Networks

4.1 Introduction

Concerns about public safety and disaster relief are rapidly growing since various kinds of disasters such as earthquakes, hurricanes and terrorisms occur regardless of place and time. In such disasters, pre-installed communication infrastructure is vulnerable and it takes time to restore once damaged or destroyed. Under such circumstances, real-time streaming is essential to understand and manage emergency situations. Since many personal mobile devices are equipped with Wi-Fi modules, Wi-Fi multi-hop networking can be a viable option to deliver streaming traffic to the disaster control center or nearby rescuers, since it provides high data rate without having any infrastructure [53]. But IEEE 802.11 MAC in multi-hop networks performs poorly, and its performance is not high enough to support streaming services.

To address these problems, we propose a fully distributed pipelining TDMA MAC, termed *DP-MAC*, for Wi-Fi multi-hop networks. It aims to achieve high end-to-end throughput under various hop distance scenarios through in-sequence scheduling and appropriate spatial reuse. Our contributions in this paper are three-fold:

- We identify the performance problem of IEEE 802.11 MAC in multi-hop networks and demonstrate that the reason behind the problem is interference from hidden stations.

- We propose DP-MAC, which improves Wi-Fi end-to-end throughput performance. Furthermore, it is highly scalable due to its fully distributed medium access control, and adaptive to varying hop distance.

- We evaluate the performance of DP-MAC through *ns*-3 simulation and show that DP-MAC outperforms 802.11 MAC in multi-hop networks.

The rest of this chapter is organized as follows. Section 4.2 presents the background and our motivation. We develop the proposed DP-MAC in Section 4.3. Then, we provide performance evaluation in Section 4.4, followed by concluding remarks in Section 4.5.

4.2 Background and Motivation

We first briefly overview IEEE 802.11 MAC operation and identify its limitations in multi-hop networks through simulation.

4.2.1 Medium access control of 802.11

IEEE 802.11 Wi-Fi has 11 partially overlapping 20 MHz channels in the 2.4 GHz unlicensed band and operates over one or two channels (i.e., 20 MHz or 40 MHz) [54]. The standard adopts Distributed Coordination Function (DCF) with CSMA/CA, which requires a backlogged device to sense the channel, waits for distributed inter-frame space duration of idle channel state, picks a random backoff counter to lower multiple access collision with other devices, and counts down the back-off counter. If the channel is sensed busy for a slot time, the device freezes its backoff counter. Otherwise, the device reduces the back-off counter by 1 and transmits the packet when the counter

Table 4.1: Simulation Parameters

Settings	Value
Simulation time	5 min
Number of iterations	10
Bandwidth	20 MHz
Rate adaptation	Minstrel
Payload size	1420 bytes
Tx power	18 dBm
CW_{min}, CW_{max}	16, 1024
CCA threshold	-92 dBm
Propagation Model	Log-normal

reaches 0. The transmission time for a packet varies according to the channel state, packet size, and frame aggregation policy.

4.2.2 Limitations of 802.11 MAC in multi-hop networks

In multi-hop ad-hoc networks, 802.11 DCF based MAC causes severe end-to-end performance degradation. We identify this phenomenon through *ns*-3 simulation. The simulation parameters are summarized in Table 4.1. Fig. 4.1 shows a typical multi-hop chain topology used in simulation. It consists of 8 stations of 802.11g. Each station is indexed by the hop count from the leftmost station (i.e, source node). The hop distance between two neighboring stations is denoted as d. We assume the traffic is unidirectional and the routing path is fixed as $0 \to 1 \to \ldots \to 7$ in advance. Stations communicate with each other without resort to an access point since they are configured in ad-hoc mode. Station 0 is the streaming source with backlogged traffic, and station 7 is the destination. This means, the streaming traffic is unidirectional. We

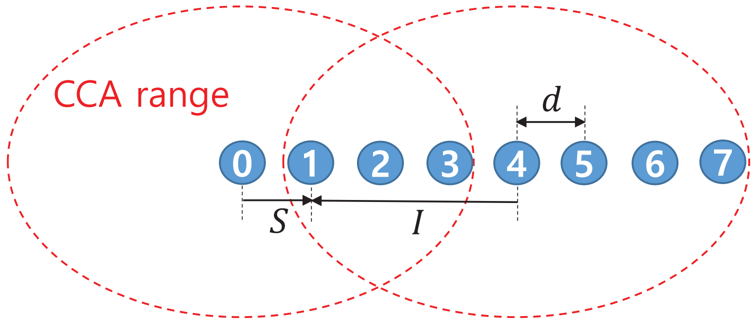


Figure 4.1: Multi-hop chain topology used in simulation.

measure the end-to-end throughput with varying d.[1] We repeated each experiment 10 times for 5 minutes.

Fig. 4.2 shows the average end-to-end throughput and frame retransmission ratio according to the hop distance d. The throughput significantly degrades as d increases. According to [55], the end-to-end throughput of 1.5∼2 Mbps (for $d > 50$) can support 480p streaming only. The figure also shows the frame retransmission ratio, which is defined as the ratio of the number of retransmitted frames to the total number of transmitted frames. We calculate the Pearson correlation coefficient $-1 \leq \rho \leq 1$, which is a measure of the linear correlation between end-to-end throughput and frame retransmission ratio. We have $\rho = -0.9375$, i.e. a huge negative correlation. Based on this, we can say that the main cause of throughput degradation is frame retransmissions in the network.

Next, we investigate the reason for frame retransmission increase in Fig. 4.1 by focusing on station 1. For small d, as shown in the figure, station 0 is able to transmit simultaneously with station 4. Then, the target signal strength S at station 1 is sufficient to overcome the interfering signal strength I from station 4. In general, a larger distance d makes the SIR at the receiver smaller, which is shown in Fig. 4.3. For $d \geq 48$, station 0 can transmit simultaneously with station 2, so the SIR at station 1

[1] We also measured the end-to-end delay and jitter, which were around 1 s and smaller than 10 ms, respectively. Therefore, the measured delay and jitter do not degrade video streaming quality greatly.

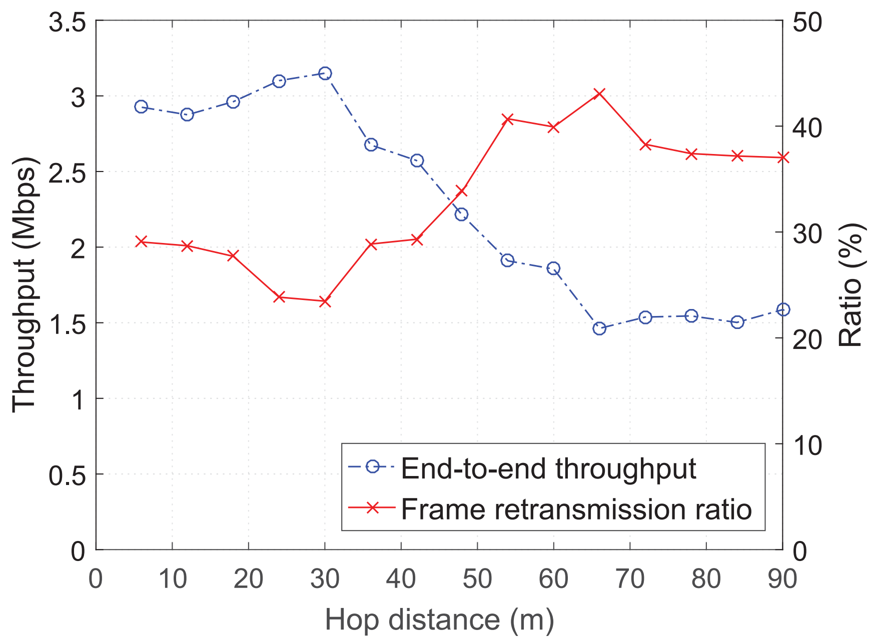


Figure 4.2: Throughput and frame retransmission ratio in 802.11 MAC vs. hop distance.

becomes 0 (i.e., $S = I$).

In other words, the main reason for low end-to-end throughput is due to hidden stations, since simultaneous transmissions from hidden stations cause low SIR, resulting in high retransmission ratio. This phenomenon is inevitable in CSMA/CA based MAC because Wi-Fi stations utilize energy detection based channel sensing with the clear channel assessment (CCA) threshold.

4.3 DP-MAC: Distributed Pipelining MAC

In this section, we propose DP-MAC that aims to achieve high end-to-end throughput in multi-hop wireless networks. We first describe the overall DP-MAC design and explain the core components of DP-MAC in detail.

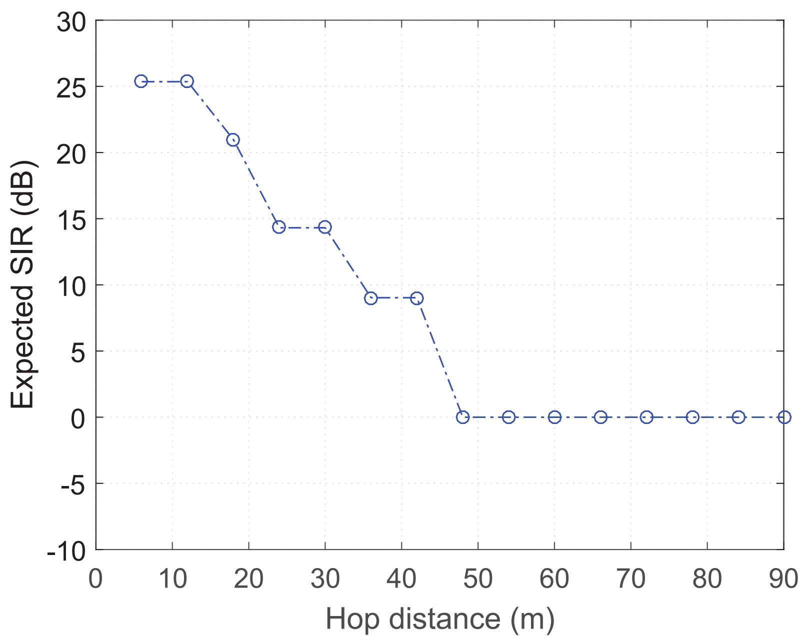


Figure 4.3: Expected SIR when a station is influenced by the interference from a closest hidden station vs. hop distance.

4.3.1 Overall DP-MAC Design

A TDMA MAC is a potential solution that reduces the impact of hidden stations by scheduling the transmission of each station. In doing this, it has two challenges. First, we should appropriately exploit spatial reuse in scheduling. In CSMA/CA MAC, stations out of the CCA range are allowed to transmit simultaneously. But in TDMA, a TDMA scheduler should decide which stations are allowed for simultaneous transmissions. Second, the scheduling information should be delivered to all the detectable stations which need to know the exact time for transmission and reception.

We solve these two challenges with *in-sequence scheduling* and *avoiding station selection*. The in-sequence scheduling makes the transmission of each station happen one after another. This means no spatial reuse, resulting in low throughput performance. Let an avoiding station of station k denote the farthest station of which transmission will interfere with station k's transmission, denoted as A_k. An avoiding station selection scheme helps the network maximally exploit spatial reuse. DP-MAC uses the avoiding station selection scheme which allows simultaneous transmissions as many as possible.

The number of simultaneous transmissions has a trade-off relation with the distance between communication pairs. If the selected pairs are too close to each other, the throughput will be decreased due to mutual interference. On the other hand, if they are far, their SIR may be reasonably good, but their end-to-end throughput degrades due to poor spatial reuse. Therefore the avoiding station selection scheme should be carefully designed to achieve high end-to-end throughput.

4.3.2 In-sequence scheduling

Let N denote the number of packets transmitted per medium access. Fig. 4.4 illustrates an in-sequence scheduling example for a 5-hop linear topology. After station 0 successfully transmits N packets, station 1 forwards the received N packets to station 2. In this way, a transit station receives and transmits N packets per medium

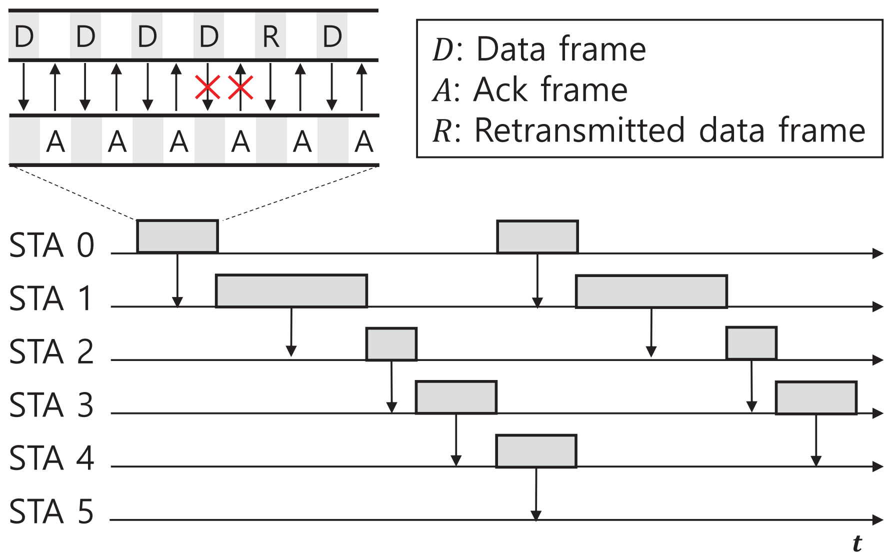


Figure 4.4: Example of in-sequence scheduling.

access. The transmission time for N packets per medium access varies depending on the wireless channel state. The in-sequence scheduling has three distinct advantages. First, it eliminates unnecessary contentions since a station is allowed to transmit when its previous stations have finished their transmissions. Second, the transmission time at each station is determined in a distributed way. Thus, concerns on synchronization disappear. Third, the scheduling information does not need to be disseminated over the whole network.

4.3.3 Avoiding station selection scheme

Since A_k denotes the avoiding station index of station k, in DP-MAC, each station k should know A_{k-1} to maintain its SIR higher than the SIR threshold, SIR_{th}, which is the minimum SIR for transmission. The expected SIR is a major factor that affects end-to-end throughput. So we select the avoiding station based on the expected SIR. Each station k runs the avoiding station selection scheme, which output is A_{k-1} and updated with period T_{update}. This means, A_{k-1} varies according to the network condition.

In an IEEE 802.11 ad-hoc network, the beacon frame generation is performed in a distributed manner. Under the avoiding node selection scheme, station k updates the signal strength of the previous and following stations (i.e. station $(k-1)$ and station j $(> k)$) from hearing 802.11 independent basic service set (IBSS) beacon frames. Thus, the signal strength information collection does not require any additional overhead.

Let $SIR_a(b)$ denote the SIR at station a when station $(a-1)$ and station b transmit simultaneously. Then station k calculates $SIR_k(l)$ for every detectable station l, assuming that the previous station $(k-1)$ and station l, i.e., one of the following detectable stations, transmit simultaneously. Then, station k compares $SIR_k(k+i)$ with SIR_{th}, starting from $i = 1$ in an increasing order of i. If an $SIR_k(k+i)$ exceeds SIR_{th}, station $(k+i-1)$ is selected as A_{k-1}. This means that station $(k-1)$ and station $(A_{k-1}+1)$ are allowed to transmit simultaneously. Algorithm 4 shows the avoiding station selection scheme in detail.

4.3.4 Medium Access State (MAS) Management

In DP-MAC, each station follows two global transmission rules to achieve in-sequence scheduling and spatial reuse. A station transmits if the following rules are satisfied.

1. The previous station has finished its transmission.

2. Its avoiding station has finished its transmission.

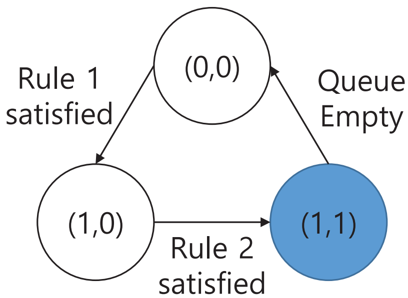
As mentioned in Section 4.3.1, the first one explains the in-sequence scheduling and the second one is for an appropriate spatial reuse. To apply these rules, each station k maintains an MAS state (b_p, b_a) where b_p and b_a denote the transmission end indicator of the previous station $(k-1)$ and the avoiding station A_k, respectively. That is, each indicates whether the transmission rule is satisfied or not. Fig. 4.5(a) shows the MAS diagram of a transit station which has three possible states. The station can access the medium only at state $(1,1)$ and stays there until the MAC queue becomes empty (i.e. successful reception of N packets from the previous station). Frame trans-

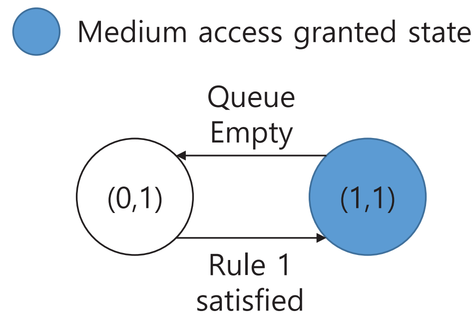
Algorithm 4 Avoiding station selection scheme at station k

Input: T_{update}, k, SIR_{th}
Output: A_{k-1}
$t \leftarrow 0$
while $t < T_{update}$ **do**
  Monitor the channel to hear IBSS beacon frames
  if a beacon frame from station l ($\geq (k-1)$) is heard **then**
    Update $S_k(l)$ with the received signal strength from station l.
  Update t as the current time
$i \leftarrow 1$
$SIR_k(i) \leftarrow S_k(k-1) - S_k(k+i)$
while $SIR_k(k+i) \leq SIR_{th}$ **do**
  $i \leftarrow i+1$
  $SIR_k(k+i) \leftarrow S_k(k-1) - S_k(k+i)$
$A_{k-1} \leftarrow k+i-1$
return A_{k-1}

mission in DP-MAC follows the same way as in 802.11. Fig. 4.5(b) shows the case that some stations close to the destination do not have their avoiding stations. Since rule 2 is always satisfied, they have only two states: $(0, 1)$ and $(1, 1)$.

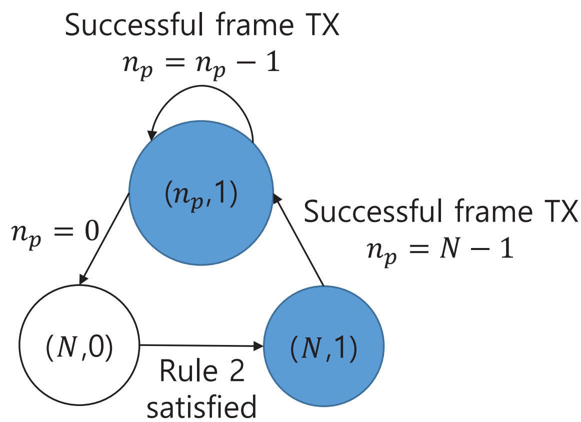
Source station 0 does not have its previous station, so we define its state slightly differently as MAS (n_p, b_a) where n_p is the number of frames for sending. Fig. 4.5(c) shows the MAS diagram of source station. Again N denotes the number of frames to transmit per medium access. Source station starts with state $(N, 1)$. For each successful transmission, it updates n_p to $(n_p - 1)$. If source finishes its transmission (i.e., $n_p = 0$), it initialize its state to $(N, 0)$. Then it approaches the medium again if the transmission rule 2 is satisfied. Note that source station accesses the medium only for $(n_p, 1)$ with $n_p > 0$.





(a) MAS of a transit station

(b) MAS of a station which does not have its avoiding station



(c) MAS of source station

Figure 4.5: MAS of each station.

4.3.5 Design of Signaling Frames

For MAS management, a station should know whether the transmission rules are satisfied or not. To help this, we define signaling frames which are compatible with the IEEE 802.11 standard.

For the transmission rule 1, DP-MAC uses the power management bit in the frame control field of the 802.11 header. In an 802.11 ad-hoc network, the power management bit is not used, thus utilizing this value does not affect other Wi-Fi operations. When a station transmits its last one of N frames it sends the frame with its power

Table 4.2: Newly defined control frames for DP-MAC

	EOT-REQ	**EOT-RES**	**EOT-NOTI**
Subtype	0001	0010	0011
Data rate	Basic	Basic	Basic / Supported

management bit enabled, denoted as *pData* frame. The receive station of *pData* frame believes that the transmission rule 1 is satisfied and let $b_p = 1$. After that, it sends the ACK with the power management bit enabled, denoted as *pACK* frame, to the send station. Receiving the *pACK* frame, the send station initializes its state to $(0, 0)$.

For the transmission rule 2, DP-MAC defines three new 802.11-compatible control frames, named end-of-transmission request (EOT-REQ), response (EOT-RES), and notify (EOT-NOTI) frames. Table 4.2 shows newly defined three frame types that use the reserved subtype field of the 802.11 control frame. Fig. 4.6 shows an signaling example of DP-MAC when station $(k + 3)$ is A_k. The solid line represents unicasting, and the dotted line represents overhearing.

A station k transmits the EOT-REQ frame destined for A_{k-1} when it overhears the *pACK* frame of the previous station $(k - 1)$. That is, station k wants to know whether A_{k-1} has finished its transmission or not. Station A_{k-1} (received the EOT-REQ frame) transmits the EOT-RES frame when it has finished its transmission. Otherwise, it does not reply. The EOT-REQ frame has a timeout value, denoted as REQ timeout.

Thus, if EOT-RES is not received in the REQ timeout, the sender retransmits the EOT-REQ frame. Receiving the EOT-RES frame, station k is aware that station $(k-1)$ can transmit since station A_{k-1} has finished its transmission, and transmits the EOT-NOTI frame to station $(k - 1)$. The station that received the EOT-NOTI frame believes that the transmission rule 2 is satisfied, and let $b_a = 1$. The EOT-NOTI frame also has a timeout value, denoted as NOTI timeout. If station $(k-1)$ does not start transmission within the NOTI timeout, station k retransmits the EOT-NOTI frame.

The EOT-REQ and EOT-RES frames are transmitted at the basic rate since they

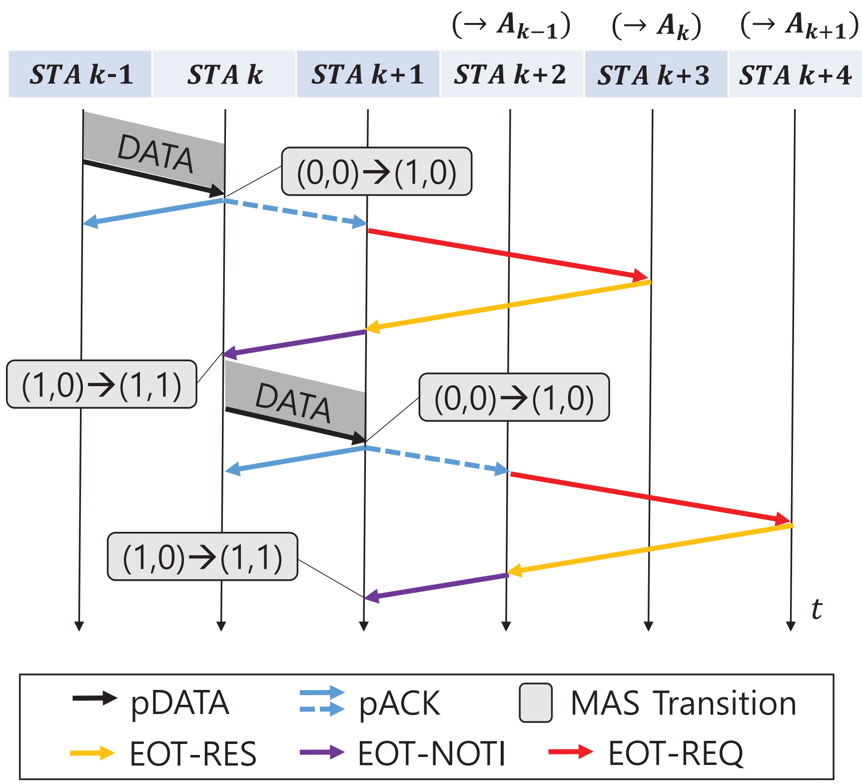


Figure 4.6: Signaling example of DP-MAC.

may traverse multiple hops. However, the EOT-NOTI frame is transmitted at a desired rate to reduce transmission time, since it is always sent to the previous station (i.e., one hop).

4.4 Evaluation

We have implemented DP-MAC in *ns*-3.26 [56]. In this section, we first show the DP-MAC architecture, and describe simulation settings followed by simulation results.

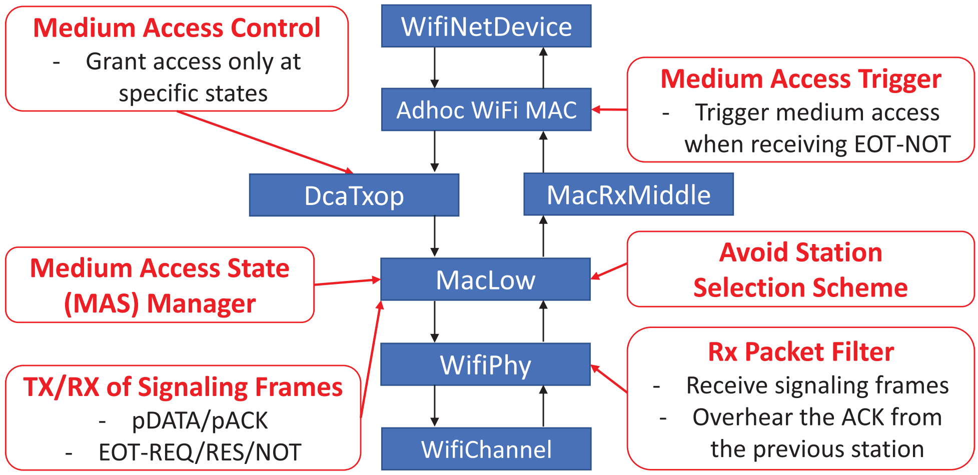


Figure 4.7: DP-MAC architecture in *ns*-3.

4.4.1 DP-MAC implementation

To implement DP-MAC, we modified PHY and MAC sublayers of *ns*-3 Wi-Fi MAC model. Since *ns*-3 Wi-Fi model has a modular structure, we modified four modules: *ad-hoc Wi-Fi MAC*, *DcaTxop*, *MacLow*, and *WifiPhy*. We define new signaling frames, add essential functionalities such as the avoiding station selection scheme, manage MAS according to the signaling, and finally control medium access based on the current MAS. The implemented DP-MAC architecture is illustrated in Fig. 4.7.

4.4.2 Simulation Settings

We consider a linear topology of 8 Wi-Fi stations as depicted in Fig. 4.1 and stations 0 and 7 are the source and destination of traffic, respectively. All the stations use 802.11 ad-hoc mode and operate at Wi-Fi channel 1 in 2.4 GHz. For DP-MAC, we use four operation parameters: T_{update}, N, SIR_{th}, *REQ timeout*, and *NOTI timeout*. DP-MAC obtains high end-to-end throughput and experiences high end-to-end delay with the increase of N. This means, it achieves low throughput and low delay with smaller N. We fix N to 10, since it shows a good balance between throughput and delay. Through

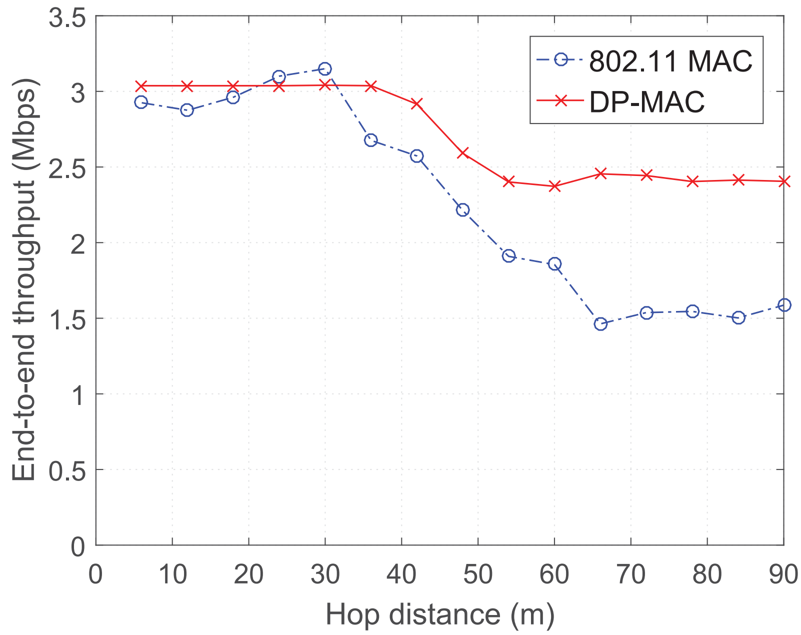


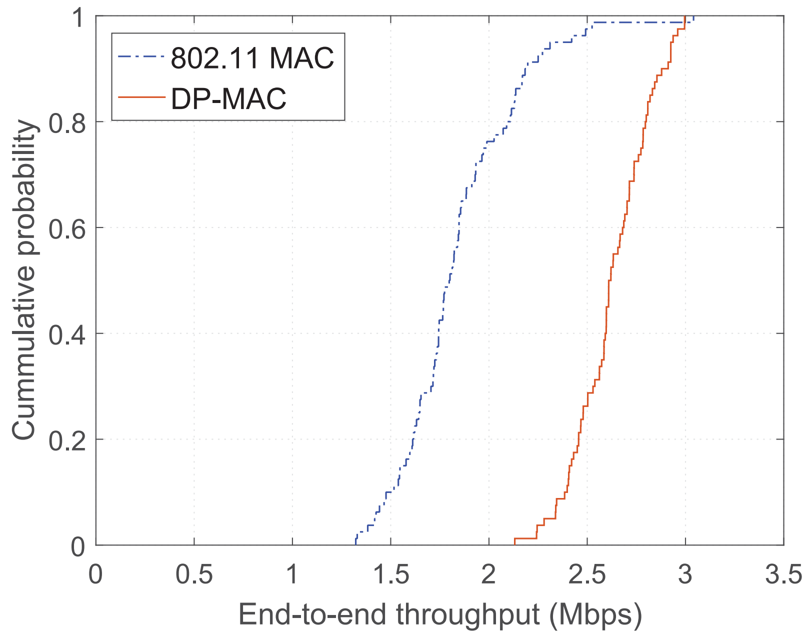
Figure 4.8: Throughput performance of 802.11 MAC and DP-MAC according to the hop distance in the fixed hop distance scenario.

extensive simulation, we determine T_{update}, SIR_{th}, REQ timeout, and NOTI timeout value as 1 s, 7 dB, 3 ms, and 4 ms, respectively. The detailed network simulation parameters are given in Table 4.1.

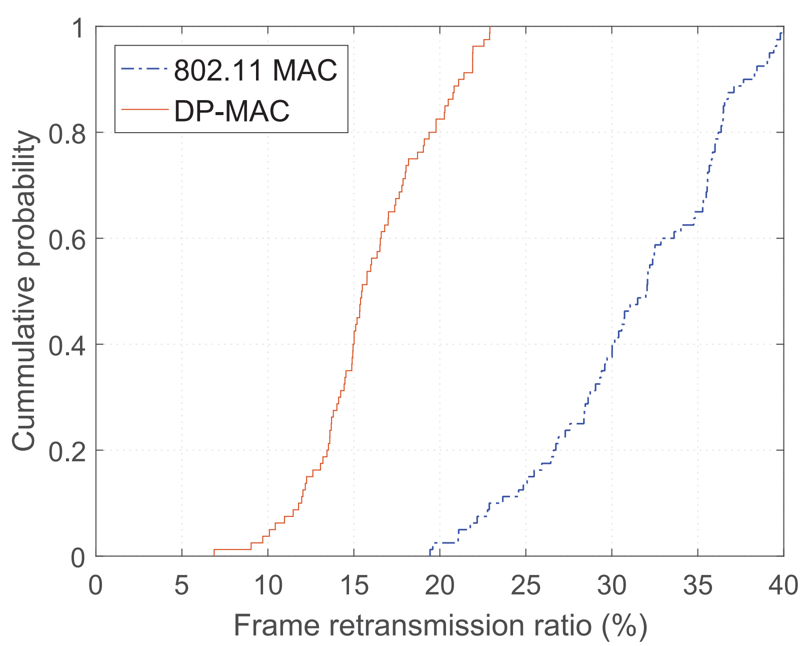
4.4.3 Simulation Results

To observe the performance, we evaluate DP-MAC in two different scenarios. One is the fixed hop distance scenario, i.e. all the hop-distances equal d. The other is the various hop distance scenario where d is uniformly distributed in [10, 80] m. We denote this scenario as the uniformly distributed hop distance scenario.

The end-to-end throughputs of DP-MAC and 802.11 MAC in the fixed hop distance scenario are illustrated in Fig. 4.8. The throughput of each MAC decreases as the hop distance increases. However, DP-MAC achieves the throughput gain of 67.8%



(a) Cummulative probability distribution functions of the end-to-end throughput



(b) Cummulative probability distribution functions of the frame retransmission ratio

Figure 4.9: Performance of 802.11 MAC and DP-MAC in the uniformly distributed hop distance scenario.

over 802.11 MAC at maximum even with a large hop distance. This is because, stations in DP-MAC are allowed to transmit simultaneously when they have the expected SIR of higher than SIR_{th}. Due to this, the average frame retransmission ratio in DP-MAC is 18% which is roughly half of that in 802.11 MAC. The overall gain from increased SIR overwhelms the signaling overhead of DP-MAC for exchanging EOT-REQ/RES/NOTI frames. But for $24 \leq d \leq 30$, 802.11 MAC shows better performance than DP-MAC due to the signaling overhead.

Fig. 4.9(a) is the cumulative probability distribution function of end-to-end throughput in 100 different topologies generated by uniformly distributed random hop distances. The average throughput gain of DP-MAC against 802.11 MAC is 42.8%. The average retransmission ratio in DP-MAC is only 51.2% of that in 802.11 MAC as depicted in the Fig. 4.9(b). Such a lower retransmission ratio in DP-MAC leads to better throughput, which is the main advantage of the avoiding station selection scheme.

Packet loss rate, end-to-end delay, and end-to-end throughput have a crucial impact on video streaming [57]. We measure the packet loss rate and the end-to-end delay. Simulation results show that the packet loss rate is always under 10^{-4}, thus barely affects the streaming quality. Moreover, the average end-to-end delay in 802.11 MAC (< 1 s) is good enough to guarantee the quality of streaming even though it is higher than that in DP-MAC. Thus, we omit these results due to lack of space.

4.5 Conclusion

In this paper, we designed a distributed pipelining MAC, termed DP-MAC, as a means to support real-time streaming in Wi-Fi multi-hop networks. In DP-MAC, Wi-Fi stations access the medium in a time-division way, while maintaining low interference in spite of simultaneous transmissions. In DP-MAC design, we tackled two challenges: transmission scheduling and dissemination of scheduling information over the whole network. DP-MAC solves these challenges by using in-sequence scheduling and avoiding station selection.

Our extensive simulation shows that DP-MAC outperforms 802.11 MAC in terms of end-to-end throughput under various topologies. It has distinct advantages in terms of scalability and adaptability, since it is fully distributed, adaptively selects avoiding stations, and supports channel state dependent scheduling. Future work is needed to extend this technique to mobile environments.

Chapter 5

Conclusion

5.1 Research Contributions

In this dissertation, we propose three designs for different scenarios and objectives.

First, we proposed OBT as a means to increase capacity of a dual-stack device. According to our extensive measurements, dual-stack devices experience a huge throughput drop when the BT and Wi-Fi are simultaneously used. To alleviate this problem, we proposed OBT, which allows the dual-stack device to send BT packets while waiting for the Wi-Fi channel to be idle in Wi-Fi mode. We developed two OBT schemes, each for best-effort BT traffic and QoS-sensitive BT traffic, respectively, and evaluated their performance through analysis and simulation. Furthermore, we verified feasibility by implementing the preliminary version of OBT-BE scheme in kernel space under limited hardware accessibility.

Second, we developed SplitScan architecture to save Wi-Fi scanning overhead. We identified that Wi-Fi scanning consumes considerable amount of time and energy via measurements on the laptop. Motivated by the measurements, we proposed SplitScan architecture which comprised of two core functionalities: splitting the scanning channels into neighboring stations and sharing the scanning information with neighboring stations. We suggested SCS selection algorithm for the former, SIP design for the latter. Through extensive simulation, we showed that SplitScan outperforms 802.11 standard scan in terms of time and energy consumption. We proved the feasibility of SplitScan via implementation on linux-based laptops. SplitScan layer is added in the kernel space, and several values and functions are added on mac80211 module and BT HCI layer. The testbeds with 2-station and 8-station environments, respectively, show that SplitScan successfully reduced Wi-Fi scanning overhead.

Third, we tackled low end-to-end throughput in multi-hop Wi-Fi network. Through simulation, we identified the major reason of low end-to-end throughput is the impact of hidden nodes. We proposed a distributed pipelining TDMA MAC, DP-MAC, to reduce the impact of hidden nodes. The proposed DP-MAC consists of the two core components: in-sequence scheduling and avoiding station selection. We implemented these on ns-3 Wi-Fi model and evaluated performance in various scenarios. Since DP-MAC guarantee the minimum SIR of every transmissions, it outperforms 802.11 ad-hoc MAC in terms of frame retransmission ratio and thus end-to-end throughput.

To summarize, the co-located devices open a new possibility of cooperation between BT and Wi-Fi. In the first and second works, we found problems by motivating measurements and developed proper solutions based on the cooperation between BT and Wi-Fi. The proposed solutions, OBT and SplitScan utilize direct communications between two protocol stacks, this aware-of-the-other design makes synergistic gain which does not exist in plenty of blind-to-the-other approaches. In the last work, we assumed that the Wi-Fi end-to-end multi-hop path is maintained by BT signaling since it requires lower energy than Wi-Fi. Based on this assumption, we developed DP-MAC achieving higher throughput than 802.11 ad-hoc MAC. If smartphone users are isolated in a disaster area, the smartphones find Wi-Fi end-to-end path via BT and stream video traffic using DP-MAC via Wi-Fi.

5.2 Future Research Directions

Based on the results of this dissertation, there are several topics to be further investigated.

Even though we implemented the proposed architecture OBT and SplitScan on the laptops, implementing them in the handhelds such as smartphones is still not accomplished. The difficulties of implementation on smartphone are basically from the manufacturer dependent device driver. Android operating system is most popular among state-of-the-art smartphones. It is based on the linux, the open source kernel. However, smartphone manufacturers do not utilize the original linux kernel, but customize it by themselves. Thus, modifying the original linux kernel is not enough to implement our schemes on smartphones. Therefore, analyzing the customized kernel and implementing our schemes on it would be great extension of this dissertation.

For the SplitScan, the performance variation in various scenarios worth investigating. In the real Wi-Fi networks, the scanning interval of devices are not identical. Assuming heterogeneous scanning interval among stations, the SplitScan results can be different from the results we obtained with assumption of the homogeneous scanning interval. The probability of finding nearby APs also worth analyzing since neighboring stations are not always seeing the same set of APs. This probability is expected to be varied according to BLE transmission range, and may affect the handoff performance of Wi-Fi stations.

Bibliography

[1] IEEE Std. 802.15.1 Version 4.2, Specification of the Bluetooth System, Bluetooth Special Interest Group (SIG), 2014.

[2] H. Yomo, P. Popovski, H.C. Nguyen, and R. Prasad, "Adaptive frequency rolling for coexistence in the unlicensed band," *IEEE Transactions on Wireless Communications*, vol. 6, no.2, pp. 598-608, 2007.

[3] M.C. Chek, Y. Kwok, "Design and evaluation of practical coexistence management schemes for Bluetooth and IEEE 802.11 b systems," *Computer Networks*, vol. 51, no.8, pp. 2086-2103, 2007.

[4] A.C. Hsu, D.S. Wei, and C.J. Kuo, "Coexistence Wi-Fi MAC design for mitigating interference caused by collocated Bluetooth," *IEEE Transactions on Computers*, vol. 64, no.2, pp. 342-352, 2015.

[5] Y. Wang, Q. Wang, G. Zheng, Z. Zeng, R. Zheng, and Q. Zhang, "WiCop: Engineering Wi-Fi temporal white-spaces for safe operations of wireless personal area networks in medical applications," *IEEE Transactions on Mobile Computing*, vol. 13, no.5, pp. 1145-1158, 2014.

[6] C.F. Chiasserini and R.R. Rao, "Coexistence mechanisms for interference mitigation in the 2.4-GHz ISM band," *IEEE Transactions on Wireless Communications*, vol. 2, no.5, pp. 964-975, 2003.

[7] L. Ophir, Y. Bitran, and I. Sherman, "Wi-Fi (IEEE 802.11) and Bluetooth coexistence: issues and solutions," in *Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Barcelona, Spain, Sep 2004. pp. 847-852.

[8] IEEE Std. 802.15.2-2003, Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands, IEEE Computer Society, 2003.

[9] W. Jeon, S. Han, J.W. Choi, and K.J. Park, "Harnessing self-cancellation for coexistence of Wi-Fi and Bluetooth," in *Proceedings of IEEE International Conference on Ubiquitous and Future Networks*, Da Nang, Vietnam, Jul 2013. pp. 294-299.

[10] P. Desai, and B. Ibrahim, "Method and system for sharing a single antenna on platforms with collocated Bluetooth and IEEE 802.11 b/g devices," U.S. Patent No. 9,504,056, November 22, 2016.

[11] I. Sherman, A.P. Shoham, and Y. Tzooreff, "Apparatus for and method of bluetooth and wireless local area network coexistence using a single antenna in a collocated device," U.S. Patent No. 8,045,922, October 25, 2011.

[12] L. Ophir, A. Klein, I. Sherman, and Y. Bitran, "Method of wireless local area network and Bluetooth network coexistence in a collocated device," U.S. Patent No. 20,060,292,987, October 25, 2011.

[13] E.J. Rivard, L.F. Tsaur, S.S. Ding, H.K. Wang, S. Jiang, and D. Liu, "Resource Sharing Priority," U.S. Patent No. 9,277,415, March 1, 2016.

[14] A. Meylan, and M. Yan, "Efficient operation for co-located WLAN and Bluetooth," U.S. Patent No. 7,899,396, March 1, 2011.

[15] Y. Sun and A.E. Xhafa, “Mechanisms for coexistence of collocated WLAN and bluetooth in the same device,” in *Proceedings of IEEE International Conference on Computing, Networkings and Communications*, San Diego, USA, Jan 2013. pp. 905-910.

[16] R. Winoto, M. He, Y. Lu, D. Signoff, E. Chan, C. H. Lin, W. Loeb, J. Park, and L. Lin, “A WLAN and Bluetooth combo transceiver with integrated WLAN power amplifier, transmit-receive switch and WLAN/Bluetooth shared low noise amplifier,” in *Proceedings of IEEE Radio Frequency Integrated Circuits Symposium*, Montreal, Canada, Jun 2012. pp. 395-398.

[17] N. Tran, T. Li, G.Y.Y. Wong, G. Lee, T. Tokubo, K. Yeung, and R.A. Chokshi, “Bluetooth and WLAN coexistence architecture having a shared low noise amplifier,” U.S. Patent No. 8,706,065, April 22, 2014.

[18] V. Brik, A. Mishra, and S. Banerjee, “Eliminating handoff latencies in 802.11 WLANs using multiple radios: Applications, experience, and evaluation,” in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (IMC)*, Anaheim, USA, Apr 2005. pp. 299-304.

[19] X. Chen, and D. Qiao, “Hand: Fast handoff with null dwell time for ieee 802.11 networks,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, San Diego, USA, Mar 2010. pp. 1-9.

[20] C. Xu, J. Teng, and W. Jia, “Enabling faster and smoother handoffs in AP-dense 802.11 wireless networks,” *Computer Communications*, vol. 33, no.15, pp. 1795-1803, 2010.

[21] I. Purushothaman and S. Roy, “Fastscan: a handoff scheme for voice over IEEE 802.11 WLANs,” *Wireless Networks*, vol. 16, no.7, pp. 2049-2063, 2010.

[22] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, “Proactive scan: Fast handoff with smart triggers for 802.11 wireless lan,” in *Proceedings of IEEE Conference on*

Computer Communications (INFOCOM), Anchorage, USA, May 2007. pp. 749-757.

[23] T. Li, C. An, R. Chandra, A.T. Campbell, and X. Zhou, "Low-power pervasive Wi-Fi connectivity using WiScan," in *Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, Osaka, Japan, Sep 2015. pp. 409-420.

[24] K.H. Kim, A.W. Min, D. Gupta, P. Mohapatra, and J.P. Singh, "Improving energy efficiency of Wi-Fi sensing on smartphones," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Shanghai, China, Apr 2011. pp. 2930-2938.

[25] S. Lee, M. Kim, S. Kang, K.Lee, and I. Jung, "Smart scanning for mobile devices in WLANs," in *Proceedings of IEEE International Conference on Communications (ICC)*, Ottawa, Canada, Jun 2012. pp. 4960-4964.

[26] G. Ananthanarayanan and I. Stoica, "Blue-Fi: Enhancing Wi-Fi performance using Bluetooth signals," in *Proceedings of ACM international conference on Mobile systems, applications, and services (MobiSys)*, Krakow, Poland, Jun 2009. pp. 249-262.

[27] J. Choi, "Widthsense: Wi-fi discovery via distance-based correlation analysis," *IEEE Communications Letters*, vol. 21, no.2, pp. 422-425, 2017.

[28] H. Zhao, S. Wang, Y. Xi, and J. Wei, "Modeling intra-flow contention problem in IEEE 802.11 wireless multi-hop networks," *IEEE Communications Letters*, vol. 14, no.1, pp. 18-20, 2010.

[29] V. Gabale, B. Raman, K. Chebrolu, and P. Kulkarni, "LiT MAC: Addressing the challenges of effective voice communication in a low cost, low power wireless mesh network," in *Proceedings of the First ACM Symposium on Computing for Development Article*, London, United Kingdom, Dec 2010. pp. 1-5.

[30] V. Sevani, B. Raman, and P. Joshi, "Implementation-based evaluation of a full-fledged multi-hop TDMA-MAC for Wi-Fi mesh networks," *IEEE Transactions on Mobile Computing*, vol. 13, no.2, pp. 392-406, 2014.

[31] B. Raman, K. Chebrolu, S. Bijwe, V. Gabale, "PIP: A connection-oriented, multi-hop, multi-channel TDMA-based MAC for high throughput bulk transfer," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Zürich, Switzerland, Nov 2010. pp. 15-28.

[32] D. Kotsonikolas, T. Salonidis, H. Lundgren, P. Leguyadec, Y.C. Hu, and I. Sheriff, "TDM MAC protocol design and implementation for wireless mesh networks," in *Proceedings of the 2008 ACM CoNEXT Conference*, Madrid, Spain, Dec 2008, pp. 28-39.

[33] P. Djukic, and P. Mohapatra, "Soft-TDMAC: A software-based 802.11 overlay TDMA MAC with microsecond synchronization," *IEEE Transactions on Mobile Computing*, vol. 11, no.3, pp. 478-491, 2012.

[34] Y. Wang, S. Jin, Y. Ni, and K.K. Wong, "Interference mitigation scheme by antenna selection in device-to-device communication underlaying cellular networks," *Journal of Communications and Networks*, vol. 18, no.3, pp. 429-438, 2016.

[35] S. Jeong, H.S. Kim, S.G. Yoon, and S. Bahk, "Q-bt: Queue-based burst transmission over an asynchronous duty-cycle MAC protocol," *IEEE Communications Letters*, vol. 20, no.4, pp. 812-815, 2016.

[36] J. Koo, K. Lee, W. Lee, Y. Park, S. Choi, "BattTracker: Enabling energy awareness for smartphone using Li-ion battery characteristics," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, San Francisco, USA, Apr 2016, pp. 1-9.

[37] H.S. Kim, H. Im, M.S. Lee, J. Paek, and S. Bahk, “A measurement study of TCP over RPL in low-power and lossy networks,” *Journal of Communications and Networks*, vol. 17, no.6, pp. 647-655, 2015.

[38] W. Lee, J. Koo, Y. Park, and S. Choi, “Transfer time, energy, and quota-aware multi-RAT operation scheme in smartphone,” *IEEE Transactions on Vehicular Technology*, vol. 65, no.1, pp. 307-317, 2016.

[39] R. Chokshi, “Yes! Wi-Fi and Bluetooth can coexist in handheld devices,” Marvell Semiconductor, Inc., Santa Clara, USA, White Paper, 2010.

[40] “How 802.11b/g WLAN and Bluetooth can play,” Philips Semiconductors, Koninklijke, Netherlands, White Paper, 2005.

[41] IEEE 802.15-00/367r0, Adaptive frequency hopping implementation proposals for IEEE 802.15. 1/2 WPAN, 2000.

[42] T. Lee, M.S. Lee, H.S. Kim, and S. Bahk, “A synergistic architecture for RPL over BLE,” in *Proceedings of IEEE International Conference on Sensing, Communication, and Networking (SECON)*, London, United Kingdom, Jun 2016, pp. 1-9.

[43] Advanced audio distribution profile specification adopted v1.0, Bluetooth SIG Audio Video Working Group, 2003.

[44] Skyworks Solutions Inc., SKY85203-11: 2.4 GHz, 802.11ac Switch/Low-Noise Amplifier Front-End, http://www.skyworksinc.com/uploads/documents /SKY85203_11_202958H.pdf.

[45] M.C. Valenti, M. Robert, and J.H. Reed, “On the throughput of Bluetooth data transmissions,” in *Proceedings of IEEE Wireless Communications and Networking Conference*, Orlando, USA, Mar 2002, pp. 119-123.

[46] G. Bianchi, “Performance analysis of the IEEE 802.11 distributed coordination function,” *IEEE Journals on Selected Areas in Communications*, vol. 18, no.3, pp. 535-547, 2000.

[47] Atheros Linux Wireless Driver for all Atheros 802.11n chipsets, https://wireless.kernel.org/en/users/drivers/ath9k.

[48] A. Mishra, M. Shin, and W. Arbaugh, “An empirical analysis of the IEEE 802.11 MAC layer handoff process,” *ACM SIGCOMM Computer Communication Review*, vol. 33, no.2, pp. 93-102, 2003.

[49] J. Liu, C. Chen, Y. Ma, and Y. Xu, “Energy analysis of device discovery for Bluetooth Low Energy” in *Proceedings of IEEE Vehicular Technology Conference*, Las Vegas, USA, Sep 2013, pp. 1-5.

[50] CYW43438 Single-chip IEEE 802.11ac b/g/n MAC/Baseband/Radio with Integrated Bluetooth 4.1 and FM Receiver, http://www.cypress.com/documentation/datasheets/ cyw43438-single-chip-ieee-80211ac-bgn-macbasebandradio-integrated-bluetooth.

[51] X. Zhang, K.G. Shin, “E-mili: energy-minimizing idle listening in wireless networks,” *IEEE Transactions on Mobile Computing*, vol. 11, no.9, pp. 1441-1454, 2012.

[52] T. Lee, J. Han, M.S. Lee, H.S. Kim, and S. Bahk, “CABLE: Connection Interval Adaptation for BLE in dynamic wireless environments” in *Proceedings of IEEE International Conference on Sensing, Communication, and Networking (SECON)*, San Diego, USA, June 2017, pp. 1-9.

[53] H. Roh, and J. Lee, “Channel assignment, link scheduling, routing, and rate control for multi-channel wireless mesh networks with directional antennas,” *Journal of Communications and Networks*, vol. 18, no.6, pp. 884-891, 2016.

[54] S. Jang, and S. Bahk, "A channel Allocation algorithm for reducing the channel sensing/receiving asymmetry in 802.11 ac networks," *IEEE Transactions on Mobile Computing*, vol. 14, no.3, pp. 458-472, 2015.

[55] Video Encoding Settings for H.264 Excellence, http://www.lighterra.com/ papers/videoencodingh264/

[56] ns-3 project, release ns-3.26, https://www.nsnam.org/ns-3-26/

[57] D. Wu, Y.T. Hou, W. Zhu, Y.Q. Zhang, and J.M. Peha, "Streaming video over the Internet: Approaches and directions," *IEEE Transactions on circuits and systems for video technology*, vol. 11, no.3, pp. 282-300, 2001.

초 록

우리는 유비쿼터스(Ubiquitous) 세계에 살고 있다. 오늘날 사람들은 스마트폰, 태블릿, 웨어러블 밴드 등 수많은 종류의 모바일 단말들을 사용하고 있고, 이들은 각각의 용례에 맞는 무선 통신 프로토콜을 기반으로 동작하여 전세계의 서로 다른 단말들과 연결된다. 현재까지 제안되고 발전된 수많은 무선 통신 프로토콜들 중에서 블루투스(Bluetooth)와 무선랜(Wi-Fi)은 비면허 대역에서 동작한다는 장점으로 인하여, 오늘날의 모바일 단말들에서 가장 널리 사용되는 두 가지의 통신 프로토콜이다.

비면허 대역에서 두 무선 통신 프로토콜의 공존은 필연적으로 간섭으로 인한 성능 문제를 야기하게 되고 이에 대한 대처 방법이나 해결 방안들은 학계는 물론, 산업계에서도 다양한 방식으로 연구되어 왔다. 그러나 블루투스와 무선랜의 공존에 대한 기존의 연구들은, 이 두 프로토콜이 서로의 신호를 수신(Receive)하거나 해석(Decode)할 수 없다는 것을 가정하고 진행되어 온 것이 대부분이다. 이러한 가정은 블루투스 네트워크가 무선랜 네트워크가 서로 독립적으로 구성되고, 한 단말이 무선랜 혹은 블루투스 중 하나의 프로토콜만 탑재하던 과거에는 합리적인 가정이었다. 그러나 이 가정을 바탕으로 진행된 연구들은 블루투스와 무선랜이 모듈이 서로의 신호를 수신 혹은 해석할 수 없기 때문에, 두 프로토콜이 서로 협력하여 동작하는데 근본적인 한계를 가지고 있었다.

과거의 가정들과는 다르게 최근의 모바일 단말들은 대부분 블루투스와 무선랜을 함께 탑재하고 있다. 블루트스와 무선랜은 서로 다른 유형의 어플리케이션을 목적으로 사용되기 때문에, 모바일 단말의 사용자가 두 프로토콜을 동시에 사용하는

경우 역시 잦아지게 되었다. 예를 들어 핸드폰에 블루투스 이어폰을 연결한 상태로 무선랜을 통해 유튜브 동영상을 감상하는 경우, 사용자는 무선랜과 블루투스를 동시에 활용하게 된다. 이렇게 한 단말에 무선랜과 블루투스가 함께 탑재되면서, 해당 단말은 두 무선 통신 프로토콜 스택을 모두 가지게 되었고, 결과적으로 이는 프로토콜 스택 간의 협력을 통해 서로 다른 프로토콜의 정보를 교환하여 성능을 높이는 새로운 형태의 협력에 대한 가능성을 열어 주었다.

본 논문에서는 기존의 연구들에서 다루어 졌던 문제들인 두 프로토콜의 공존 시 수율 저하 문제, 무선랜의 스캐닝 오버헤드 문제, 다중 홉에서의 수율 저하 문제에 대하여, 프로토콜 스택 간에 직접적 협력이라는 새로운 방식을 통해 해결하는 시스템들을 제안하였다. 또한, 제안하는 시스템을 실제로 프로토타입 형태로 구현하여, 단지 아이디어를 제안하는 수준에 그치지 않고, 실제로 활용될 수 있음을 보여주었다. 본 논문에서 제안하는 세 가지의 시스템은 다음과 같다.

첫째, 블루투스와 무선랜 프로토콜이 하나의 공유된 안테나로 연결된 듀얼 스택 단말의 성능 문제를 해결하는 시스템을 제안하였다. 본 연구에서는 실험을 통해 듀얼 스택 단말에서 블루투스와 무선랜이 동시에 사용될 때, 큰 수율 저하가 발생함을 확인하고, 해당 수율 저하를 완화시킬 수 있는 기회적 블루투스 전송 (OBT) 기법을 제안한다. 기회적 블루투스 전송은 무선랜 동작에 필연적으로 수반되는 대기 시간 (waiting time)을 블루투스 전송에 활용함으로써 수율 향상을 이끌어 내는 방법이다. 제안하는 기법은 블루투스와 무선랜 프로토콜 간의 직접적인 시그널링을 사용하여 무선랜의 대기 시간을 블루투스 프로토콜 스택이 활용할 수 있게끔 디자인되었다. 제안한 기법의 성능은 시뮬레이션을 통해 확인하였고, 일부 기법을 실제 노트북에 구현함으로써 제안 기법이 기존 기법에 비해 높은 수율을 얻을 수 있음을 확인하였다.

둘째, 무선랜의 스캐닝 동작에 소모되는 시간과 에너지 오버헤드를 줄이는 방법을 제안하였다. 먼저 기존의 무선랜 스캐닝 동작은 모든 채널을 방문하기 때문에 많은 오버헤드를 야기함을 실험을 통해서 확인하였다. 스캐닝 동작의 오버헤드를 줄이기 위해, SplitScan 아키텍처를 제안하였고, 제안하는 아키텍처는 크게 두 가지 동작(주변 무선랜 단말들과의 채널 분할 및 결과 공유)으로 구성되었다. SplitScan을

통하여 임의의 무선랜 단말은 주변의 단말과 스캐닝할 채널의 집합을 서로 분할할 수 있고, 그 결과를 블루투스를 통해 공유함으로써 스캐닝 오버헤드를 줄일 수 있다. SpliScan 시스템 역시 블루투스와 무선랜 간의 직접적인 협력을 이용하도록 구현되었으며, 이를 위해 리눅스 커널에 SplitScan 계층을 추가하였다. 우리는 SplitScan의 성능을 시뮬레이션을 통해 확인하였을 뿐만 아니라, 실제 노트북에 구현함으로써 그 성능을 증명하였다.

셋째, 다중홉 단방향 무선랜 네트워크에서 종단간 수율을 향상시키는 매체 접근 제어(Medium Access Control) 방식을 제안하였다. 재난으로 기존의 통신 인프라를 사용할 수 없는 상황에서, 재난 상황에 대한 비디오 정보를 제공하기 위해서는 무선랜 다중 홉 네트워크가 좋은 대안이 된다. 이 때, 블루투스를 통해 다중 홉 네트워크를 유지하고, 높은 수율을 요구하는 스트리밍과 같은 어플리케이션이 필요한 경우에만 일시적으로 무선랜을 사용하는 네트워크를 생각할 수 있다. 상기와 같은 환경에서, 기존의 802.11 에드혹(ad-hoc) 매체 접근 제어는 Carrier-sensing Multiple Access (CSMA)를 사용하기 때문에 근본적으로 히든 스테이션 문제에서 자유로울 수 없다. 시뮬레이션을 통하여 802.11 에드혹 매체 접근의 한계를 규명하고, 이를 해결하기 위한 방법으로 TDMA MAC인 DP-MAC을 제안한다. DP-MAC은 순차 전송 스케줄링(in-sequence scheduling)과 히든 스테이션 선택 기법을 바탕으로 히든 스테이션의 영향력을 줄임과 동시에 불필요한 경쟁을 TDMA를 통해 제거함으로써 종단간 수율을 향상시키도록 디자인되었다. ns-3 시뮬레이션을 통하여 다양한 시나리오에서 제안하는 DP-MAC이 기존의 802.11 에드혹 매체 접근보다 더 좋은 성능을 보임을 확인하였다.

주요어: 블루투스, 무선랜, 와이파이, 공존, 매체 접근 제어, 듀얼 스택 단말, 스캐닝, 다중 홉 전송
학번: 2012-30236