UNIVERSITY OF CALIFORNIA

Los Angeles

Toward Customizable Wi-Fi

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Chi-Yu Li

2015

ABSTRACT OF THE DISSERTATION

Toward Customizable Wi-Fi

by

Chi-Yu Li

Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2015
Professor Songwu Lu, Chair

As the Wi-Fi technology becomes pervasive in reality, its usage pattern is also turning highly diversified in operation settings and application scenarios. This consequently leads to new design requirements for Wi-Fi networking solutions in terms of various combinations in energy efficiency and latency, in addition to the throughput. However, the state-of-the-art solutions typically chase for high speed in the different generations of Wi-Fi technologies (from 802.11a/g to 802.11n/ac) at the cost of other metrics. For example, Multiple-Input Multiple-Output (MIMO) is widely used to boost speed, yet it consumes much more power. Higher throughput may yield long-tail loss behaviors and compromise the perceived latency for data transfer. Fundamentally, we believe that both the user demand pull and the technology push call for customizable Wi-Fi solutions.

In this dissertation, we describe our effort on customizable Wi-Fi technology. We propose solutions that seek to meet the diverse requirements (i.e., energy, throughput, and latency). Specifically, we have come up with technical results on three topics. The first one explores to use rate adaptation (RA), the mechanism critical to performance yet unspecified by the 802.11 standards, for energy efficiency. It is shown that current MIMO RA algorithms in 802.11n/ac are not energy efficient despite ensuring high throughput. Marginal throughput gain is achieved at much higher energy cost. We then propose EERA, an energy-efficient RA solution. It balances throughput for energy savings while meeting the data rate quest by applications. In the second topic, we target home Wi-Fi scenario and interactive gaming applications. We examine the millisecond-level latency requirements of such applications. We show that current solutions work well for throughput but not for latency, due to the long tail of the packet delay distribution. We thus propose LLRA, a new latency-aware RA scheme that reduces the tail latency. It takes concerted design in rate control, frame aggregation scheduling and software/hardware retransmission dispatching. The implementation and evaluation confirm the viability of both EERA and LLRA. In the third topic, we propose HetRA, which can meet heterogeneous goals at each client. It explores a new theoretical approach, called piecewise linear wireless service curve. Using this new solution framework, we can meet minimum throughput on a per-flow basis, while simultaneously improving energy efficiency at the device level. We show in both analysis and experiments that, the resulting design outperforms all existing RA algorithms that only pursue a single goal. In all three concrete designs, we confirm that, the upcoming Wi-Fi technology has to be customized to the given usage scenario and designated goals, and this involves complex tradeoffs along multiple performance metrics of throughput, latency and energy saving and along various granularities of data flows, flow aggregates, and devices within a single user client and among multiple users.

The dissertation of Chi-Yu Li is approved.

Mario Gerla

Lixia Zhang

Babak Daneshrad

Songwu Lu, Committee Chair

University of California, Los Angeles

2015

To my parents and my wife Wei-Ju

and my children Aaron and Ayden

without whom I would not complete my PhD study.

TABLE OF CONTENTS

List of Figures

List of Tables

Acknowledgments

I am indebted to my advisor Professor Songwu Lu for his patient and insightful guidance through my Ph.D. study. He spent time on interacting with me frequently, and usually gave me valuable feedback and constructive suggestions. He carefully led me to go through the difficulties of thinking, writing, and experiments in doing scientific research. He also taught me how to do high quality of research and good presentation. I really appreciate his guidance and support throughout these years.

I am grateful to my committee members, Professors Lixia Zhang, Mario Gerla, and Babak Daneshrad, for their insightful comments on my research. Professor Zhang inspired me to think deeper in each work, and emphasize its insights and philosophy. She also taught me the importance of being precise on the wording and presentation. Professor Gerla helps me with his valuable comments and friendly discussions on my research projects. Professor Daneshrad's technical comments about his expertise of MIMO technologies, are very helpful for my MIMO-related work.

I also want to thank my colleagues in the Wireless Networking Group (WiNG) at UCLA. I am especially grateful to Yiannis Pefkianakis for his guidance on the early stage of my research. He motivates me to be aggressive and hard-working in the way of achieving my goals. I would like also to thank Chunyi Peng and Guan-Hua Tu for their cooperation on the projects of both Wi-Fi and cellular networks.

Vita

2000 - 2004	B.S., Computer and Information Science, NCTU, Taiwan
2000 - 2004	B.S. in Management, Management Science, NCTU, Taiwan
2004 - 2006	M.S., Computer Science and Engineering, NCTU, Taiwan
2006 - 2007	Second Lieutenant, Ministry of National Defense, Taiwan
2008 - 2009	Research Assistant, Computer Science, NCTU, Taiwan
2009 - 2013	M.S., Computer Science, UCLA
2009 - 2014	Graduate Student Researcher, Computer Science, UCLA

Publications

Guan-Hua Tu, Yuanjie Li, Chunyi Peng, **Chi-Yu Li**, Songwu Lu, ”Detecting Problematic Control-Plane Protocol Interactions in Mobile Networks,” *To appear in IEEE/ACM Transactions on Networking.*

Chi-Yu Li, Chunyi Peng, Songwu Lu, Xinbing Wang, Ranveer Chandra, ”Latency-Aware Rate Adaptation in 802.11n Home Networks,” *IEEE INFOCOM*, Hong Kong, Apr. 2015.

Chunyi Peng, **Chi-Yu Li**, Hongyi Wang, Guan-Hua Tu, Songwu Lu, ”Real Threats to Your Data Bills: Security Loopholes and Defense in Mobile Data Charging,” *ACM CCS*, Scottsdale, Arizona, Nov. 2014.

Guan-Hua Tu, Yuanjie Li, Chunyi Peng, **Chi-Yu Li**, Hongyi Wang, Songwu Lu, ”Control-plane Protocol Interactions in Cellular Networks,” *ACM SIGCOMM*, Chicago, Illinois, Aug. 2014.

Ioannis Pefkianakis, **Chi-Yu Li**, Chunyi Peng, Suk-Bok Lee, Songwu Lu, ”CMES: Collaborative Energy Save for MIMO 802.11 Wireless Networks,” *IEEE ICNP*, Gottingen, Germany, Oct. 2013.

Guan-Hua Tu, Chunyi Peng, Hongyi Wang, **Chi-Yu Li**, Songwu Lu, ”How Voice Calls Affect Data in Operational LTE Networks,” *ACM MOBICOM*, Miami, Florida, Sept. 2013.

Guan-Hua Tu, Chunyi Peng, **Chi-Yu Li**, Xingyu Ma, Songwu Lu, ”Accounting for Roaming Users on Mobile Data Access: Issues and Root Causes,” *ACM MOBICOM*, Taipei, Taiwan, Jun. 2013.

Chunyi Peng, **Chi-Yu Li**, Guan-Hua Tu, Songwu Lu, Lixia Zhang, ”Mobile Data Charging: New Attacks and Countermeasures,” *ACM CCS*, Raleigh, NC, Oct. 2012.

Chunyi Peng, Guan-Hua Tu, **Chi-Yu Li**, Songwu Lu, ”Can We Pay for What We Get in 3G Data Access?” *ACM MOBICOM*, Istanbul, Turkey, Aug. 2012.

Chi-Yu Li, Chunyi Peng, Songwu Lu, Xingbin Wang, ”EERA: Energy-efficient Rate Adaptation for 802.11n Devices,” *ACM MOBICOM*, Istanbul, Turkey, Aug. 2012.

Chi-Yu Li, Ioannis Pefkianakis, Bojie Li, Chenghui Peng, Wei Zhang, Songwu Lu, ”A Multimedia Service Migration Protocol for Single User Multiple Devices,” *IEEE ICC*, Ottawa, Canada, Jun. 2012.

Ioannis Pefkianakis, **Chi-Yu Li**, Songwu Lu, ”What is Wrong/Right with IEEE 802.11n Spatial Multiplexing Power Save Feature?” *IEEE ICNP*, Vancouver, Canada, Oct. 2011.

CHAPTER 1

Introduction

Wi-Fi devices are increasingly popular. The global shipments grew 19 percent year-over-year, and reached 1.9 billion units in 2013. Now, 4 billion devices with built-in Wi-Fi are in use across the world. It is expected to have over 7 billion devices being used globally by 2017 [Ana14]. They include various types of devices, such as PC, labtop, smartphone and tablet. Diverse demands thus arise. The applications running on them have different requirements (e.g., throughput and delay). Mobile devices further require energy efficiency to prolong their battery life. As a result, the functionality of Wi-Fi networks not only becomes much more important, but also needs to support nowadays diverse demands.

Mobile devices become the dominant population in the Wi-Fi networks. Wi-Fi enabled smartphones and tablets accounted for a combined 59% of all Wi-Fi device shipments in 2013. The shipments of these mobile devices are expected to pass 1.8 billion in 2017, and become 83% of the total [Cor13], as shown in Figure 1.1. Thus, more attention is paid to the Wi-Fi performance of mobile devices. In addition to the conventional performance goal, speed, there exists other important goals for mobile devices. At the device-level, the battery life is the major concern, so the energy efficiency of Wi-Fi operation is one of the concerned performance aspects. At the application-level, mobile applications exhibit other performance goals. For example, a Youtube or NetFlix streaming video requires minimum throughput, whereas VoIP tools (e.g., Skype) and gaming applications are concerned about timely delivery. The trend that mobile devices dominate Wi-Fi networks calls for the Wi-Fi operation satisfying their performance requirements.

Home networking environment has been the major scenario of Wi-Fi networks. In the US, 61% of households had Wi-Fi networks installed at home in April 2012; by 2016, about 800 million

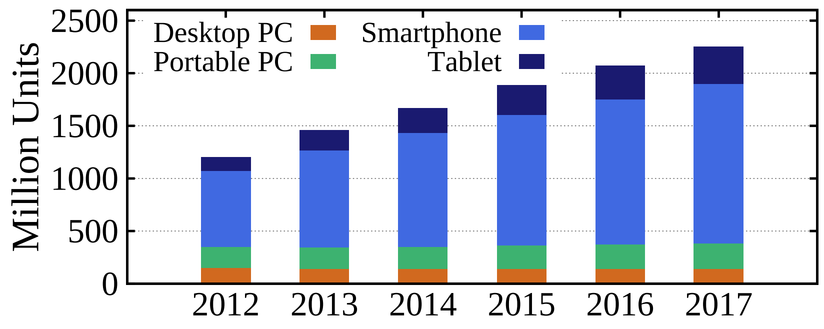


Figure 1.1: Worldwide smart device shipments from IDC (International Data Corporation) [Cor13].

households worldwide (about 42% of global penetration) will have Wi-Fi access at home [wif12]. In a typical home Wi-Fi deployment scenario today, an AP is used to serve multiple Wi-Fi devices, including laptops, tablets, smartphones, game consoles and other gadgets. Home users not only use them for the Internet access, but also play games, access high-fidelity music, watch online videos, etc.. According to the report [Ana14], there are 7 Wi-Fi devices averagely for every US Wi-Fi household now; by 2017, the number will become 11. Such large population of Wi-Fi devices in the same Wi-Fi network may easily cause the devices to interfere with each other. Their performance may suffer from the interference, especially for the real-time and interactive applications (e.g., gaming, screen mirroring, etc.). They impose stringent latency requirement over a single wireless hop.

The Wi-Fi standard has pushed the speed from Megabit to Gigabit, while paying little attention to the requirements of mobile Wi-Fi devices. The maximum speed increases from the legacy's 54 Mbps (802.11a/b/g), to the IEEE 802.11n standard's 600 Mbps and the IEEE 802.11ac standard's 6.7 Gbps. The major technology used to boost the speed is Multiple-Input Multiple-Output (MIMO). It uses multiple antennas to achieve faster transmission by delivering independent information bits in parallel. The standards of 802.11n and 802.11ac support up to 4 and 8 antennas, respectively. However, MIMO is a very power-hungry technology, especially for mobile devices with small battery. Without careful design of the MIMO operation, the battery can be drained quickly. Moreover, purely pursuing high speed may sacrifice latency. Aggressively chasing after the highest speed may result in higher packet loss than stably staying at a moderate speed. The packets encountering loss may be further rescheduled carelessly, thereby experiencing long delay.

In this dissertation, we seek to move toward the customizable Wi-Fi, which can achieve diverse goals. We thus make three major contributions which satisfy the diverse requirements of mobile devices and applications in Wi-Fi networks. We achieve them by focusing on the mechanism of rate adaptation (RA), since enhancing it does not require any hardware modification or firmware upgrade on the commodity Wi-Fi devices. Moreover, it plays the leading role to boost the Wi-Fi speed. In the first contribution, we explore the root cause why current RA algorithms cannot always achieve energy efficiency, and then propose a solution of energy-efficient RA (EERA) to save energy at the NIC. Second, we examine how the long tail latency comes from the 802.11 link layer, and further propose the latency-aware RA (LLRA), which achieves millisecond-level latency in a typical, multi-client home Wi-Fi network. Third, to satisfy the heterogeneous goals, we propose HetRA, which offers assurances in terms of minimum throughput on a per-flow basis, while simultaneously improving energy efficiency at the device level. We implement all these three solutions on the 802.11n commodity devices, and the evaluation confirms their viability.

CHAPTER 2

Performance-oriented Wi-Fi Technology

The evolvement of Wi-Fi technology aims to achieve the speed as high as possible. The new Wi-Fi technology (e.g., 802.11n/ac) thus seeks to improve throughput performance over the legacy 802.11a/b/g. However, the current diverse demands make the usage of Wi-Fi technology to be not always throughput driven. The current Wi-Fi features and operations, which aim to optimize the achievable throughput, may not work well with emerging demands: energy efficiency and short delay, as well as concurrent goals of throughput and energy. Moreover, current state-of-the-art work may be either not able to satisfy them.

In what follows, we first introduce the major features and operations of Wi-Fi technology. We further present the current diverse demands for Wi-Fi and the related state-of-the-art.

2.1 Wi-Fi Features and Operations

We consider the most popular Wi-Fi standard, 802.11n. We introduce the major features proposed to boost speed, and the power-saving mechanisms which come with those power-hungry speed boost features. We also present the major link-layer operations which are related to energy efficiency and small delay.

2.1.1 Speed Boost Features

The current 802.11n standard introduces three major features to boost speed: Multiple-Input Multiple-Output (MIMO), channel bonding, and frame aggregation. It also proposes two power-saving mechanisms. The first one is Spatial Multiplexing Power Save (SMPS), which reduces

MIMO energy consumption. The second mechanism is Power Save Multi-Poll (PSMP), which saves energy at the client by letting it sleep during its non-active period (e.g., when the AP transmits to other clients).

MIMO The MIMO technology uses multiple antennas to enhance the performance. It supports two operation modes: Spatial diversity and Spatial multiplexing. In the former, each antenna at the transmitter transmits a replica of the same data stream. The signal strength can be enhanced at the receiver by the leverage of multiple antenna links with independent fading. In the latter, each antenna transmits independent and separately encoded spatial streams. The speed thus increases proportionally with the number of streams. The 802.11n standard supports up to four antennas and four data streams. The new 802.11ac standard further enhances the speed to Gigabit by supporting up to eight antennas/streams.

Channel bonding This feature in 802.11n doubles the transmission rate by simultaneously using two separate channels. In the legacy Wi-Fi (802.11a/b/g), only a single 20 MHz channel can be used. However, the 802.11n can operate in the 40 MHz mode over two adjacent 20 MHz channels. It is thus able to double the rate of the same modulation and coding scheme (MCS) used by the legacy. The new 802.11ac standard supports up to 160 MHz channel.

Frame aggregation Frame aggregation increases throughput by reducing the MAC overhead (see [SNC08] for a tutorial). It packs multiple data frames into a single aggregated frame. Two types of frame aggregation are defined: Aggregate MAC Service Data Unit (A-MSDU) and Aggregate MAC Protocol Data Unit (A-MPDU). The former wraps multiple frames in a single 802.11n frame so that one A-MSDU contains a single MAC header, whereas the latter wraps each frame in an 802.11n MAC header. Although A-MPDU is less efficient than A-MSDU, it may be more efficient with high error rates due to a mechanism called block acknowledgement (Block-ACK). This mechanism allows each of the aggregated data frames in one A-MPDU to be individually acknowledged. A Block-ACK is sent if at least one frame in the A-MPDU is received without error. Each Block-ACK can acknowledge multiple packets. The maximum size of an A-MPDU is bound by the maximum transmission time of one frame (here, 4 ms), and thus increases with rates.

SMPS SMPS reduces MIMO energy consumption at a client by letting it retain only one receive chain for a period of time. It supports two operation modes. In the static mode, the client statically retains only a single receive chain. In the dynamic mode, the client activates multiple receive chains upon the upcoming received frame which is sent by any multi-stream rate. This activation is triggered by a RTS/CTS handshake which proceeds the frame. It switches back to one receive chain right after the frame transmission ends.

PSMP PSMP diminishes the energy consumption of the client's non-active period by making it sleep as long as possible. It supports two modes: Scheduled PSMP (S-PSMP) and Unscheduled PSMP (U-PSMP). In S-PSMP, the AP periodically initiates a PSMP sequence to schedule the transmission. During a PSMP sequence, a client shall not be able to receive and transmit frames at the times outside its scheduled periods. Thus, the client is allowed to sleep during the periods in which it has no interest. In U-PSMP, the AP starts an unscheduled sequence and delivers to those wakeup clients. The clients are awake when they notify the AP of their interests on receiving or/and transmitting frames.

2.1.2 Link-layer Operations

We present the background of rate adaptation and retransmission in the 802.11n networks. The former is not specified by the standards, whereas the latter's operation is either not completely addressed.

Rate adaptation Rate adaptation (RA) offers an effective mechanism to exploit the multi-rate, adaptive modulation capability at the physical layer (PHY). It adapts the PHY configuration to wireless channel dynamics. The design of RA is more complex in 802.11n than in the legacy 802.11a/b/g. Given the wireless channel condition, it has to select the appropriate configuration along three dimensions, the modulation and coding scheme (MCS), the chain setting (i.e., the chosen numbers of transmit and receive antennas), and the number of spatial streams. In contrast, for the legacy 802.11a/b/g, RA only needs to select the best MCS option.

The 802.11n specification defines a large parameter space, thus posing the scaling issue for

RA design. The MCS rates span from 6Mbps to 600Mbps when each sender/receiver can activate one to four transmit/receive antennas. The standard also supports multiple-stream operations, i.e., single-stream (SS), double-stream (DS), triple-stream (TS), and quadruple-stream (QS). The number of streams is bounded by the smaller number of transmit and receive antennas. Table 2.1 gives an example of feasible configurations with up to three transmit/receive antennas. Each setting is denoted by $N_t \times N_r/RN_{SS}$, with N_t and N_r being the number of transmit and receive antennas, R being the MCS rate, and N_{SS} being the number of streams. For example, the 3$\times$3/13.5SS setting defines the configuration using a single stream on three transmit/receive antennas each, with the lowest MCS rate being 13.5 Mbps. Note that different chain numbers would be allowed for the same stream mode, for example, the 3x3, 3x2, 2x3, and 2x2 chain settings can all support the DS mode.

Retransmission Retransmission is needed to recover from packet losses. There are two types: hardware retry and software reschedule. Figure 2.1 shows an example of both mechanisms at an AP. Initially, each packet is placed into the software queue (SQ) once it arrives at the MAC layer. It is not placed to the hardware queue (HQ) until those in other higher-priority SQs (for different clients) are scheduled ahead and transmitted. Once the HQ is available, packets (here, packets 1-3) are aggregated to form an aggregated frame. It is then pushed into the HQ and transmitted over the air. Retransmission for unsuccessful packets depends on whether the ACK is received.

If the ACK is received but not the whole frame succeeds, software reschedule is applied to retransmitting unsuccessful packets. A Block-ACK indicates those successful packets and then those failed ones (here, packet 2) are placed back to its original SQ (at the head). Retransmission has to wait for its turn from the scheduling among SQs. The unsuccessful ones may be aggregated with other packets (here, packets 4 and 5). In contrast, if no ACK is received (all aggregated packets fail), hardware retry is used. The frame stays in the HQ for retransmission. The maximum number of hardware retries is platform dependent and can be configured. If all the hardware retries fail, software reschedule is eventually employed.

Setting	**#Streams**	**MCS Index**	**Date Rate (Mbps)**
1x1/13.5SS	1	0 (BPSK, 1/2)	13.5
1x1/27SS	1	1 (QPSK, 1/2)	27.0
1x1/40.5SS	1	2 (QPSK, 3/4)	40.5
1x1/54SS	1	3 (16-QAM, 1/2)	54.0
1x1/81SS	1	4 (16-QAM, 3/4)	81.0
1x1/108SS	1	5 (64-QAM, 2/3)	108.0
1x1/121.5SS	1	6 (64-QAM, 3/4)	121.5
1x1/135SS	1	7 (64-QAM, 5/6)	135.0
2x2/27DS	2	8 (BPSK, 1/2)	27.0
2x2/54DS	2	9 (QPSK, 1/2)	54.0
...	...	...	...
2x2/270DS	2	15 (64-QAM, 5/6)	270.0
3x3/40.5TS	3	16 (BPSK, 1/2)	40.5
...	...	...	...
3x3/405TS	3	23 (64-QAM, 5/6)	405.0

Table 2.1: Example of available settings for 802.11n RA with one to three transmit/receive antennas.

2.2 Current Diverse Demands for Wi-Fi

We introduce the current diverse demands for Wi-Fi technology in three aspects: energy efficiency, delay requirement in a home Wi-Fi network, and the concurrent goals of minimum throughput and energy efficiency. We then present that the state-of-the-art RAs cannot satisfy these diverse demands.

Energy Efficiency An 802.11n NIC consumes much more power than its legacy 802.11a/b/g one. Our measurement shows that, an 802.11n 3x3 MIMO receiver consumes about twice the power of 802.11a during active transmission, and 1.5 times power when idle. Therefore, energy

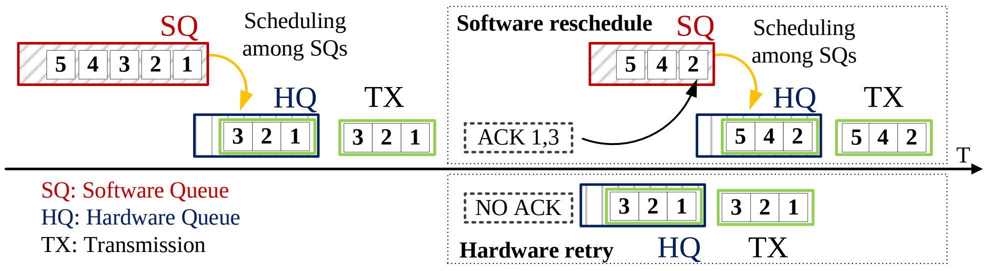


Figure 2.1: Two retransmission types: hardware retry and software reschedule.

efficiency becomes a critical issue for 802.11n NIC operations.

Delay Requirement We focus on serving latency-sensitive data flows that quest for millisecond-level delays in a home Wi-Fi network. Each flow denotes a stream of packets from a source to a destination. The per packet delay is the duration from the time a packet arrives at the link layer to the time its acknowledgement (ACK) is received. For each latency-sensitive flow, we seek to improve the long tail of its latency distribution. Our specific goal is to reduce the tail latency of a flow, which is defined as the packet latency at the α percentile (say, $\alpha = 90^{th}$ of all packets) over a window of packet bursts of the flow.

This aspect of Wi-Fi demand is highly motivated by the increasing popularity of real-time or interactive applications in home Wi-Fi networks. They impose stringent (millisecond-level) latency requirement over a single wireless hop. Such applications include wireless console gaming [con], mobile-to-mobile gaming [MAZ11], streaming gaming [chrb], screen mirroring and TV remote play (e.g., via Chromecast [chra]), etc.. For wireless gaming, timely delivery of commands (from the consoles, phones or other peripherals) and multimedia data if applicable, is critical to the liveness and interactiveness for gaming. It thus requires the latency within several milliseconds or even 1 ms [con]. So is the screen mirroring which projects what we see on the phone or tablet onto TV. The user experience of these applications is sensitive to the tail of the latency distribution (expressed in α percentile). Outside the home Wi-Fi network, some applications also have low latency requirement over a single wireless hop. For instance, high-fidelity music playback expects a latency within 11.5ms [CGL04], whereas 1-3ms delay is highly desirable in augmented reality [PJ00].

Concurrent Goals of Minimum Throughput and Energy Efficiency The Wi-Fi usage becomes highly diversified. In a typical operation scenario, various mobile applications running on different mobile devices coexist in a single access point (AP) or a WLAN. These applications exhibit different requirements, for example, a Youtube or NetFlix streaming video requires minimum throughput, whereas Web browsing on a smartphone cares more about its battery consumption. Three factors further accelerate this trend. First, mobile applications have grown exponentially. Second, more mobile devices have built-in Wi-Fi interfaces. Third, Wi-Fi access has become the norm rather than exception in real-life scenarios of home, campus, corporate and public-space.

2.2.1 Rate Adaptation

Numerous RA algorithms have been proposed in the literature [WGB08,WYL06,PHW10,HHS10, VBJ09,ASB10,GK11,HVB,CK08,REK,KM97,Bic05,LMT04,KDC13,CLS11]. Most of them aim to achieve high goodput. Only a few pursue other goals. [KDC13] seeks for energy efficiency, whereas [CLS11] is proposed to reduce latency in video streaming. The proposed EERA is the first study of achieving energy efficiency from the RA perspective, and then the study [KDC13] follows it. Moreover, no RA has appeared to reduce tail latency or satisfy multiple goals (e.g., latency, goodput, energy) simultaneously. In this dissertation, the LLRA and HetRA are proposed to achieve them, respectively.

RA seeks to select the best rate for its goal upon the wireless channel condition. There are two major approaches to achieve it. The first one is to directly measure the channel condition using the SNR metric and then use it to infer the best rate. It requires the receiver to gauge SNR or channel state information (CSI) and then feedback it to the transmitter. This mechanism is not supported in the current 802.11 platforms, so we do not take this approach in this dissertation. We adopt the practical approach as the second one, in which the transmitter probes the channel condition with different rates and selects the one with the best performance. The performance is always estimated based on the loss statistics obtained from the probing. We next introduce the existing RAs of these

SNR-based and probing-based designs, as well as other designs which do not belong to those two categories.

SNR-based designs The SNR-based designs [HVB, CK08, REK, HHS10] differ in how to effectively obtain the SNR or CSI information which associates with rate selection. RBAR [HVB] relies on the RTS/CTS exchange to gauge SNR at the receiver, and picks the transmit rate accordingly. CHARM [CK08] estimates average SNR at the receiver by utilizing the wireless channel's reciprocity. It relies on the packets overhead only from the receiver, instead of RTS/CTS exchange. FARA [REK] measures SNR on per-frequency basis, and thus allows a transmitter to use different rates across different OFDM subbands. ESNR [HHS10] is designed for MIMO 802.11 systems. It measures the CSI at the receiver, and then maps CSI to the transmit MIMO rate. Another work [KDC13] seeks to construct a model of energy efficiency based on the SNR, and then it picks the energy-efficient transmit rate based on the measured SNR.

Probing-based designs The probing-based solutions [KM97, Bic05, WGB08, WYL06, LMT04, GK11, PHW10] are different in two ways: probing approach and the utilization of loss statistics. ARF [KM97] and AARF [LMT04] attempt to use a higher transmission rate after a series of successful transmissions at a given rate, and switches back to a lower rate after 1 or 2 consecutive failures. SampleRate [Bic05] maintains the expected transmission time for each rate, and updates it after each transmission. The rate with the smallest time is considered the current best transmit rate. RRAA [WYL06] uses a short-term loss statistics to access the channel and opportunistically adapts the transmit rate to dynamic channel conditions. The above designs are proposed for the legacy 802.11a/b/g, but do not work for the MIMO 802.11n/ac, which support a large set of MIMO rates.

ARA [WGB08] and MiRA [PHW10] are proposed to support MIMO rates. The former excludes half of rates to speed up probing, whereas the latter uses zigzag probing which switches between different MIMO modes. Both of them select the rate with the highest estimated goodput. However, they use sequential search; the search does not scale well to 802.11n/ac where search space is bigger, thereby resulting in slower convergence (see the finding in Section 4.1.2).

Other designs There are two recent proposals [VBJ09, GK11] which do not belong to the above two types of designs. Strider [GK11] uses rateless codes to achieve an optimal, collision-resilient RA. It does not require neither SNR feedback nor probing. SoftRate [VBJ09] uses confidence information calculated by the physical layer to estimate the prevailing channel bit error rate (BER). The transmitter can pick the best rate based on this BER estimation. Both of them require the supports from the MAC and physical layers, which are not standard compliant, so they are not feasible for the commercial 802.11 NICs. However, this dissertation focuses on the RA solutions which are practical and standard-compliant.

2.3 State-of-the-Art for Diverse Demands

There are two major approaches to satisfy the diverse demands in the Wi-Fi networks, in terms of protocol stack. One is to focus on the transport layer and/or application layer, but it cannot leverage the Wi-Fi characteristics to save energy or achieve low-latency. The other is to target on the MAC layer. However, even if it can address the issues, but it may not be standard-compliant, thereby requiring hardware update on the devices. Due to these limitations, we seek to sit in the middle by taking the RA approach. It can not only leverage the Wi-Fi features, but also require no update on the device's hardware.

2.3.1 Energy Saving

The dissertation focuses on the energy-saving of Wi-Fi networks in the infrastructure mode. This area has been widely studied. The related work can be classified into three categories based on the energy-saving target: infrastructure [MCR06, JIP07, JPB09] and client [ZGK11, ACW07, LZ08, ZS11, DSP10, KMG11, QCJ03, GCN01, JHS11, PLL11] sides, as well as both [PLP13]. The dissertation mainly addresses the energy-saving issue for the client side, at which mobile devices always reside. We also present other studies which are related to the MIMO energy efficiency [CGB05, GD11, KCD09].

Infrastructure side Several solutions [MCR06, JIP07, JPB09] seek to save energy at the APs. The similar approach they take is to power off the APs which do not serve any traffic, but the difference is made on how to make power-off decision. In the Wake-on-WLAN [MCR06] work, each AP powers itself off when it does not see any client in its vicinity. However, SEAR [JIP07, JPB09] clusters APs that are in close proximity, and then power on and off APs upon the traffic demand. Some APs can be powered off under the low traffic load. The power-off decision is made by a central controller, rather than the AP itself in the Wake-on-WLAN work.

Client side The existing studies for the client's energy saving can be classified into three main dimensions: non-active, transmit, and MIMO energy savings. The work in the last category may also consider the energy-saving targets of the first two categories. To single out the MIMO factor as a category is because this dissertation focuses on the energy-saving solutions for the MIMO Wi-Fi.

- **Non-active energy saving** Clients can save energy by staying at the non-active status because the non-active power is smaller than the active one. In the legacy 802.11a/b/g Power Saving Mode (PSM), clients can go to sleep during the non-active period. They wake up only when the AP has the buffer of their downlink packets or they intend to transmit. However, PSM lacks the flexibility of scheduling the clients to sleep. The clients may stay awake for a long time to wait for their transmission. The current 802.11n standard [11n09] further proposes the Power Save Multi-Poll (PSMP) feature, as introduced in Section 2.1. It allows the clients to go to sleep during the periods of others' transmission. However, in order to support some applications which require timely delivery, clients may not be able to go to sleep. They thus need to stay in the idle mode during the non-active period, but the idle power is much larger than the sleep power.

  In order to save more energy, some proposals [ACW07, ZS11] seek to reduce the non-active power, whereas others [ZGK11, LZ08, DSP10] increase the length of the non-active period. [ACW07] attempts to minimize energy consumption of the Wi-Fi interface for the usage of VoIP call. It powers off the interface when there is no VoIP call in progress. It powers it on only on the reception of an incoming VoIP call through the always-on cellular radio. [ZS11] seeks to reduce the idle power consumption by adaptively downclocking the radio. [ZGK11] delivers downlink packets to the PSM clients in an optimal sequence that minimizes the time they are forced to stay awake. [LZ08] enables the clients to go to sleep more fine-grained, even for the small interval between MAC frames. [DSP10] combines small gaps between packets into meaningful sleep intervals, thereby allowing the clients to sleep as much as possible.

- **Transmit energy saving** Some proposals [QCJ03, GCN01] seek to save energy by adjusting the transmit power, since the power consumption of the transmit amplifier is much larger than the receive and idle power. Given the same MCS rate, reducing transmit power may result in lower signal strength at the receiver, thereby higher packet loss and lower speed. For energy-saving, they attempt to reduce transmit power while keeping the speed performance being not affected.

- **MIMO energy saving** Since MIMO is power hungry, the current 802.11n standard [11n09] specifies a new MIMO power-saving mechanism, Spatial Multiplexing Power Save (SMPS), while newly introducing MIMO in the Wi-Fi context. As introduced in Section 2.1, SMPS gives the clients the flexibility of retaining only one receive antenna for a period of time. MRES [PLL11] examines the strength and limitation of SMPS, and proposes an energy-saving solution through dynamically adjusting antenna settings. Snooze [JHS11] schedules client sleep time and configures antenna settings for energy savings.

The energy-saving design in this dissertation targets the MIMO Wi-Fi at the client side. All the efforts along this direction do not address the problem from the RA perspective. Thus, they are unable to achieve the most energy efficiency by completely trading off speed for energy savings. Although MRES [PLL11] and Snooze [JHS11] seek to adjust antenna settings for energy savings, their underlying RAs which pursue HG by default may not give the most energy-efficient rate setting. The decoupling of antenna switch and rate selection may also hurt performance due to the slow convergence of locating the best setting. However, the proposed EERA design seeks for energy savings by considering most components which affect both speed and power consumption: transmit/receive antenna settings, number of data streams, MCS rates.

Both sides CMES [PLP13] seeks to minimize the energy efficiency of the 802.11 MIMO system, which includes both AP and client. It identifies the most energy-efficient antenna setting of the system (transmitter-receiver pair).

MIMO system Recent theoretical studies on energy-efficient MIMO systems [CGB05, KCD09], seek to find the crossover point, which trades off MIMO gains at the cost of increased power consumption. [GD11] seeks to find the most energy-efficient settings only for transmission period, using their MIMO-OFDM based software-defined radio. However, they are either unable to be used on commodity platforms, or not 802.11n standard compliant.

2.3.2 Wireless Delay

There are numerous efforts to pursue low latency through different techniques, such as packet scheduling [CM03, Xu10], MAC protocol [NK12], transport-layer optimization [ZLC11, WSB13], redundancy [VGM13], and optimization for specific applications like video streaming [CLS11, FPK13, Sch06] and data centers [AKE12, AL13]. All of them either target to reduce the latency that is an order of magnitude or more (several hundreds of milliseconds or seconds) or focus on different networks (e.g., mesh networks, ad hoc networks, cellular networks, data centers).

The LLRA work in the dissertation seeks to leverage MAC mechanisms (rate control and frame aggregation scheduling) to reduce millisecond-level latency. This complements some proposed techniques on packet scheduling and higher-layer optimization.

CHAPTER 3

Overview

In this chapter, we first present the limitations in the current Wi-Fi network, and our goal of meeting diverse demands. To achieve it, we rely on the mechanism of rate adaptation (RA), since it does not require any hardware modification or firmware upgrade on Wi-Fi devices. Moreover, it plays the leading role to boost the Wi-Fi speed. We then introduce our contributions and the roadmap.

3.1 Limitations of Current Wi-Fi Network

We explore the limitations of the current Wi-Fi from the RA perspective. The traditional goal of RA is to achieve high speed/goodput (i.e., effective throughput). However, it may not be able to satisfy current diverse demands, energy efficiency, latency requirement, and concurrent goals of minimum throughput and energy efficiency.

Energy Efficiency Existing RA solutions are effective to ensure high goodput but not energy savings. Our study shows that, two popular 802.11n RA algorithms ARA [WGB08] and MiRA [PHW10] incur per-bit energy waste at an NIC as large as 54.5% and 52.9%, respectively, compared with the most energy-efficient fixed setting. The energy waste still exists with/without the power-saving mechanisms of SMPS and PSMP. The root cause is that, current RAs obtain high goodput at whatever energy cost. By powering on more antennas, more streams, or higher MCS rates, marginal goodput gain is realized at the cost of high energy consumption. Definitely, we also observe that MiRA and ARA ensure higher goodput during active data transmissions, about 52.5Mbps and 52.4Mbps, respectively, compared with the EE setting that yields only 35.4Mbps.

Latency Requirement At first glimpse, it seems that current Wi-Fi can already meet the latency

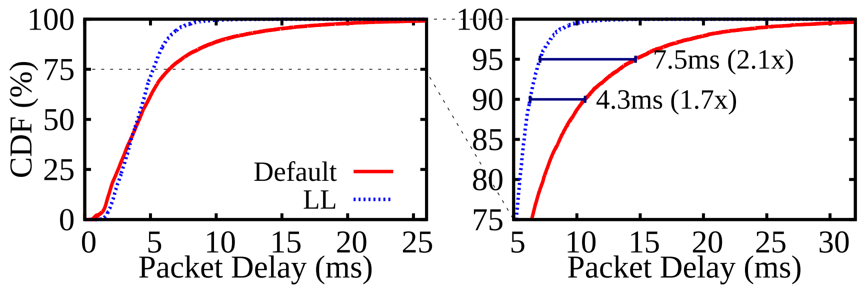


Figure 3.1: The cumulative distribution function (CDF) of packet delay for the default setting and the low-latency (LL) one at the observed client in an 802.11n home network.

requirement. With up to 600Mbps speed at an 802.11n AP, higher data rate implies shorter transmission time, and thereby lower latency. However, our study shows this is not true in general: Highest speed does not always yield shortest latency. Figure 3.1 plots the cumulative distribution functions (CDFs) of packet latency for two rate settings in 802.11n. The default setting achieves the highest speed whereas the other, denoted as low-latency (LL) one, yields shorter per packet delay. In terms of throughput, the former achieves 108.0 Mbps and does outperform the latter (1.2x of 87.4 Mbps by LL). However, it incurs 4–8ms extra delay for the 90^{th} or 95^{th} latency than LL. Specifically, the default setting takes 10.6 ms (90^{th}) and 14.6 ms (95^{th}), compared with 6.3 ms and 7.1ms by LL. The gap of the maximum even reaches up to 38.9 ms (58.3ms versus 19.4ms). The root cause lies in that, highest rate reduces the average or medium latency, but not necessarily the tail latency. In the example, both rates contribute to similar medium packet delay (4.0 ms), but the highest rate setting generates a much longer tail, which hurts quality of experience for applications.

Concurrent Goals of Minimum Throughput and Energy Efficiency Current RA solutions cannot either achieve energy efficiency while simultaneously ensuring a given minimum throughput. It is because they are mostly single-goal driven design. However, this demand involves multiple goals.

3.1.1 Goals

In this dissertation, we have three goals. The first is to ensure high energy efficiency for 802.11n NICs. The second is to reduce the tail latency of a flow in a typical, multi-client home Wi-Fi network. The third is to provide the minimum assurance in throughput on a per-flow basis, while simultaneously improving energy efficiency at the 802.11n NICs.

3.2 Contributions

We have three major contributions: energy-efficient RA (EERA), latency-aware RA (LLRA), and HetRA, which offers assurances in terms of minimum throughput on a per-flow basis, while simultaneously improving energy efficiency at the device level.

3.2.1 EERA: Toward Energy Efficiency in 802.11n MIMO Networks

We present EERA, a new RA algorithm that trades off goodput for energy savings at an 802.11n client NIC. EERA searches for the 802.11n setting consuming less per-bit energy, rather than purely achieving higher goodput. EERA supports both single-client and multi-client operations. In the former case, it poses the RA problem as multi-dimensional tree-based search, spanning MCS rates, the numbers of transmit and receive antennas, and the number of streams. EERA exploits ternary search and MIMO characteristics to prune multiple branches simultaneously to speed up its runtime convergence. The latter case builds on top of single-client design. Each client is periodically allocated a fair share of airtime, and can only use up this airtime share but no more. Fair sharing of extra airtime protects each client through isolation, thus enabling coexistence of EERA and other MIMO RA clients.

Moreover, EERA is configurable. Each client specifies a tuning knob on how much it is willing to slow down to save its NIC energy. It controls the balance between energy and goodput at NIC and handles multi-client interference. EERA reverts to traditional goodput-optimizing RA if needed. As a result, the EERA algorithm has low complexity and ensures high energy savings.

In our test scenarios, EERA consistently outperforms other RA algorithms. It saves about 30% energy compared with ARA and MiRA in all cases. It saves 6-36% in static settings and 20-24% in mobility and field tests, compared with another energy-saving proposal MRES [PLL11].

3.2.2 LLRA: Latency-Aware Rate Adaptation in 802.11n Home Networks

We focus on serving latency-sensitive data flows that quest for millisecond-level delays in a home Wi-Fi network. Each flow denotes a stream of packets from a source to a destination. The per packet delay is the duration from the time a packet arrives at the link layer to the time its acknowledgement (ACK) is received. For each latency-sensitive flow, we seek to improve the long tail of its latency distribution. Our specific goal is to reduce the tail latency of a flow, which is defined as the packet latency at the α percentile (say, $\alpha = 90^{th}$ of all packets) over a window of packet bursts of the flow.

Given the fact that the long tail mainly comes from the 802.11 link layer, other higher-layer approaches such as network packet scheduling or transport congestion control manage latency at course grains and are deemed less effective. Here, we reduce the tail latency from the link rate adaptation perspective. Our proposed solution LLRA is a first RA algorithm that achieves millisecond-level latency in a typical, multi-client home 802.11n network. The goal of LLRA is to find a LL rate setting with the lowest latency at the given α percentile. The target tail latency is thus reduced as much as possible. LLRA is the concerted design of three components: rate control, frame aggregation scheduling, and retransmission dispatching. The rate control searches for the LL rate, which balances between initial queue delay and service time, thereby achieving lowest latency. The aggressive aggregation scheduling is further applied to reduce the former. To curtail the latter, the retransmission dispatching prioritizes retransmissions.

We further take a novel probing-based approach to LL rate search to make it work in real systems. Association rule-based pruning is used to eliminate high-loss rate settings by correlating results at different settings under similar channel conditions. It consequently offers a light-weight solution to protect LLRA from excessive probing cost. In all tested scenarios, it consistently reduces the tail latency over other algorithms. For the latency at the 90^{th} and 95^{th} percentiles, LLRA reduces tail latency by 30.7%-90.8% and 21.8%-66.2% over ARA [WGB08] and MiRA [PHW10], respectively. For example, while ARA and MiRA observe tail latency of 28.0 ms and 14.4 ms respectively, LLRA keeps it below 10 ms.

3.2.3 HetRA: Meeting Diverse Requirements in 802.11n Networks

We design solutions that seek to meet diverse requirements by mobile applications and devices. The goal is to provide the minimum assurance in throughput on a per-flow basis, while simultaneously improving energy efficiency at the device network interface. The target setting is an 802.11n AP, which is the dominant AP shipment and delivers up to 600Mbps at PHY via Multiple-input and Multiple-output (MIMO). The fundamental challenge is to devise a practical solution to this wireless quality-of-service (QoS) problem.

To this end, we explore a new solution approach of wireless service curve (WSC). WSC adapts the service curve concept in the Internet QoS [Cru92, Cru95, SZN97] to the wireless 802.11n domain. It serves as the building block to specify the minimum amount of service (both bandwidth and delay) for a given flow, while managing energy efficiency for mobile devices. A fundamental difference from the Internet case is that, wireless link capacity is time varying, highly dependent on the channel condition and the used PHY rate. To meet heterogenous requirements, we propose piecewise linear WSC (PL-WSC), which allows for using different PHY rates to achieve diverse goals, while collectively reducing energy consumption and meeting the goodput and/or delay constraints. It turns out that, the PL-WSC framework can be readily implemented via the popular rate adaptation algorithm for 802.11n devices and the airtime scheduler.

At the core of the solution is HetRA, which offers a novel rate adaptation (RA) algorithm based on PL-WSC. The key insight is that HetRA uses nonlinear service curve, whereas existing algorithms follow linear service curve that is a special case of PL-WSC. We show that the nonlinear WSC option outperforms its linear counterpart. To handle diverse requirements, HetRA works in concert with the airtime scheduler. We further uses a credit/debit mechanism in airtime scheduling to handle dynamics due to channel variations, mobility, and dynamic traffic source. In a nutshell, together with the airtime scheduler, HetRA achieves two-tier adaptive service. The base tier ensures minimum throughput, while the upper tier fairly distributes extra free airtime to improve energy savings for the battery-constrained clients. The evaluation based on our prototype shows that, HetRA can meet the diverse requirements by application flows while simultaneously saving energy. It outperforms existing algorithms MiRA and EERA by up to 35.4% and 20.1%, respectively. The extra airtime can be fairly shared among the clients under 0.5% difference. Our design is also readily extended to the upcoming 802.11ac, which supports up to 8-antenna MIMO [11a11].

3.3 Dissertation Organization

The rest of the dissertation is structured as follows.

Chapter 4 presents EERA. Section 4.1 uses a case study to examine the limitations of 802.11n RA in NIC energy efficiency, and Section 4.2 models the 802.11n NIC energy. Section 4.3 describes the design of EERA. Section 4.4 presents the implementation and evaluation of EERA.

Chapter 5 describes LLRA. Section 5.1 showcases the impacts of RA on latency. Sections 5.2 presents the design of LLRA. Its implementation and evaluation are described in Section 5.3.

Chapter 6 introduces HetRA. Section 6.1 describes the problem, the motivation and the challenges. Section 6.2 illustrates the new WSC approach via experimental examples. Sections 6.3, 6.4.1 and 6.4 present its design, implementation and evaluation. Section 6.3.4 discusses miscellaneous design issues.

Chapter 7 summarizes this dissertation and presents our future work.

CHAPTER 4

EERA: Energy-Efficient Rate Adaptation

In this chapter, we explore to design RA for 802.11n NICs from an energy efficiency perspective. We use a case study to examine the limitations of 802.11n RA in NIC energy efficiency, and model the 802.11n NIC energy consumption. We then present the design of EERA, as well as the implementation and the evaluation of it.

4.1 802.11n RA Limitation: An Energy Efficiency Perspective

In this section, we present a case study to examine the limitations of current 802.11n RA algorithms in terms of energy efficiency. We show that, current solutions are effective to achieve high goodput, but may not ensure energy efficiency. There exists fundamental conflicts between the best goodput setting and the most energy-efficient setting for 802.11n NICs. We start the case study from its experimental setting.

4.1.1 Experimental Setting

We first quantify the energy consumption of 802.11n NICs under various RA algorithms. We select two existing 802.11n RA algorithms. Atheros RA (ARA) algorithm [WGB08] is used by Atheros 802.11n NICs, while MiRA [PHW10] is recently proposed for 802.11n radios. Both apply sequential search to probe different settings and locate the best setting eventually. ARA probes from the medium setting (i.e., the highest MCS rate in the DS mode). If the probe succeeds, it switches to the TS mode; otherwise, it goes down through MCS rates in DS and SS modes. Its implementation excludes half of rates to speed up probing. In contrast, MiRA uses zigzag probing

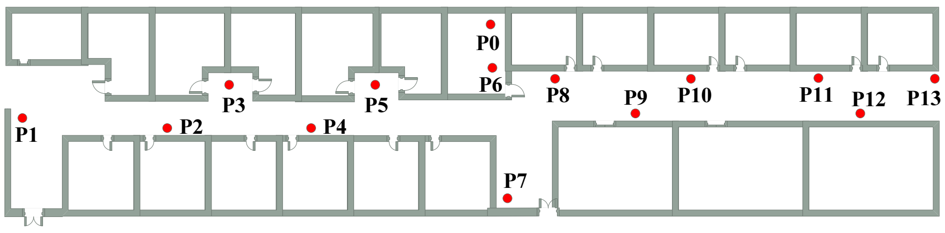


Figure 4.1: Experimental floorplan.

and starts from the highest MCS in the TS mode, and then switches to DS and SS until it succeeds. Both algorithms can be implemented in the current platforms using available 802.11n chipsets. We also considered other MIMO RA algorithms, e.g., ESNR [HHS10] and SoftRate [VBJ09]. They cannot be implemented on current 802.11n systems.

We conduct the experiments in a controlled laboratory setting (see Figure 4.1 for the floor plan) over the 5GHz band without external interference. Spots P1 to P13 denote locations for the client, whereas AP is always at P0. We consider the infrastructure mode only, the dominant deployment in reality. Both the AP and clients are programmable 802.11n devices, which use Atheros AR9380 2.4/5 GHz MIMO chipset and support three antennas. The platform supports SS, DS, and TS modes, with transmission rates up to 450Mbps over 40MHz bands. We also use Intel 5300 wireless NIC at the client. We test downlink transmissions, with the AP being the transmitter and the client being the receiver. In each test, we send the UDP traffic generated by iperf at constant source rate (say, 30 Mbps for the tests in Table 4.1) for 120 seconds and collect measurements over five runs.

We use power meter Agilent 34401A to record the consumed power. For PSMP, we collect traces and use the ideal doze power to simulate its energy value, since PSMP is not implemented in current drivers. Note that, energy saving from PSMP is thus overestimated without counting its processing overhead. We use per-bit energy consumption (i.e., energy-per-bit) E_b to quantify the energy efficiency of RA algorithms. It is defined as the consumed energy when exchanging each bit at a given setting. This metric counts the consumed energy during both active and non-active periods [PLL11].

	EE	ARA	MiRA
E_b (nJ/bit)	19.2	29.7	29.4
Gap (%)	–	54.5%	52.9%
Goodput (Mbps)	35.4	52.4	52.5
#Bits (Mbits)	3598	3576	3598
Energy (J)	69.0	106.2	105.6

Table 4.1: Energy-per-bit, goodput, number of bits, and energy consumption for ARA, MiRA and the optimal EE setting at Location P1. The UDP source rate is 30 Mbps.

4.1.2 Case Findings

Our study shows that, both ARA and MiRA incur large energy waste, compared with the most energy-efficient fixed setting. Table 4.1 gives the energy-per-bit by both ARA and MiRA, as well as the best configuration (called "EE" in the table) achieving highest energy efficiency among all settings at P1. The results show that, ARA and MiRA incur per-bit energy waste as much as 54.5% and 52.9%, respectively, compared with the best setting. Definitely, we also observe that MiRA and ARA ensure higher goodput during active data transmissions, about 52.5Mbps and 52.4Mbps, respectively, compared with the EE setting that yields only 35.4Mbps. Note that all can sustain the 30Mbps UDP data source.

We next find why current RA algorithms incur energy waste for NICs. It turns out that, though both ARA and MiRA are able to achieve high goodput during the active period, these settings also incur high energy consumption. To this end, we plot the goodput, as well as the energy-per-bit for those selected settings in Figure 4.2. The setting yielding highest goodput (marked with "HG" in the figure) is 3x3/81DS. It is clearly seen that this HG setting is not the most energy-efficient one (i.e., 3x1/40.5SS, marked with "EE"). The energy-per-bit gap between these two settings reaches 11.1 nJ/bit, resulting in energy waste as large as 57.8% when using the HG setting only. Figure 4.3 further plots the rate distribution (in percentage) of ARA and MiRA. We see that both algorithms are effective in reaching high goodput. This observation is consistent with the primary goal of their design. ARA mainly selects two settings, 3x3/81DS and 3x3/108DS, whereas the selection

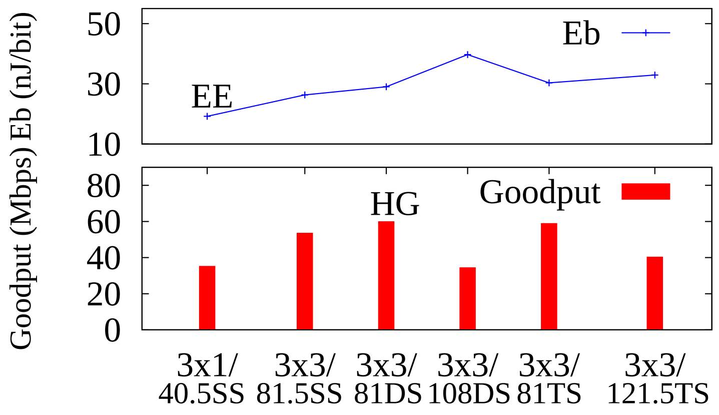


Figure 4.2: Goodput and energy-per-bit under various settings at P1.

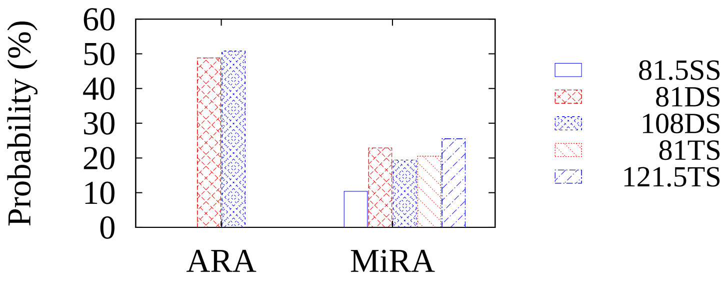


Figure 4.3: Rate distribution of MiRA and ARA algorithms at P1. All the settings use three receive antennas.

of MiRA spreads over 5 settings. ARA chooses fewer settings because it considers only half of the rate settings. However, these high-goodput settings consume more energy per bit, as large as 39.7 nJ/bit (3x3/108DS), about twice the per-bit energy consumption of the EE setting 3x1/40.5SS. Finally, they make ARA and MiRA deviate from the EE setting by as large as 54.5% and 52.9% energy waste, respectively, in Table 4.1.

We further explore why the highest-goodput settings cannot ensure best energy efficiency. It turns out that, at these high-goodput settings, the per-bit energy cost to obtain the marginal goodput gain is pretty high. To achieve higher goodput, we have to activate more antennas and more streams no matter how much extra power could be consumed. Here, the “HG” setting (3x3/81DS) consumes additional 394.4 mW and 224.4 mW in active and non-active periods, compared with the

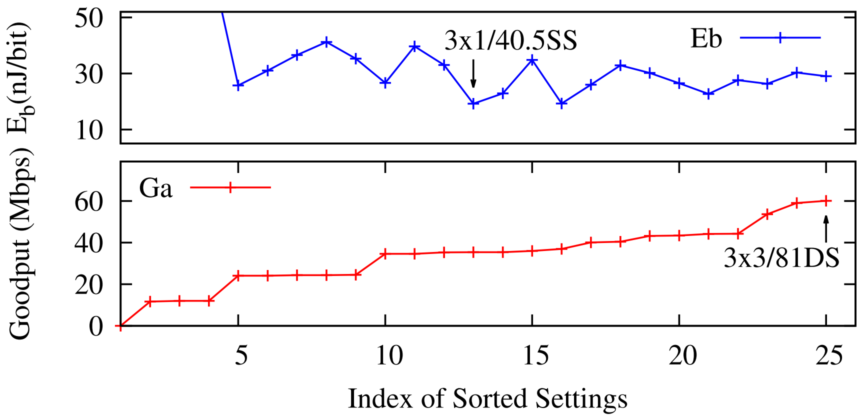


Figure 4.4: Goodput and energy-per-bit energy under different settings at P1.

Setting	3x1/ 40.5SS	3x3/ 81SS	3x3/ 81DS	3x3/ 108DS	3x3/ 81TS	3x3/ 121.5TS
Active Power (mW)	580.6	812.3	975.0	982.5	1046.4	1063.4
Idle Power (mW)	541.2	765.6				

Table 4.2: Power consumption of the most energy-efficient setting and those used by MiRA and ARA at P1.

"EE" setting (3x1/40.5SS) (shown in Table 4.2). Figure 4.4 plots goodput and energy-per-bit for various settings at P1. It shows that, E_b does not monotonically decrease as the goodput increases in the upper plot of Figure 4.4. Several dips appear in these settings. The marginal goodput gain at the cost of extra energy consumption becomes smaller or even negative for some high-rate settings. Consequently, it becomes less energy efficient to chase for higher goodput.

Moreover, our study reveals that current RA algorithms may not provide fast convergence when locating the best setting. This is because both ARA and MiRA apply sequential search to probe feasible settings and result in slower convergence. This can be illustrated by the detailed probing process of ARA and MiRA shown in Figure 4.5. The sequential search in ARA and MiRA further faces the scaling issue when the number of settings becomes larger. In case of three antennas at the AP and the client each, the client thus supports $1 + 2 + 3 = 6$ (i.e., $M_{Nr}(M_{Nr} + 1)/2$, where M_{Nr} is the maximum number of receive antennas) modes, and each mode allows for multiple MCSes (8

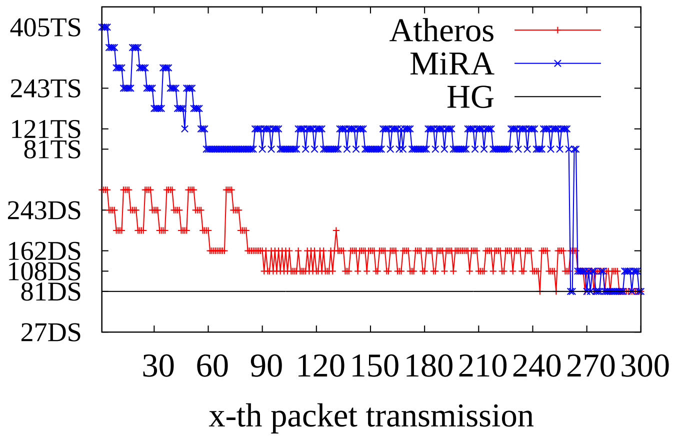


Figure 4.5: The probing process of MiRA and ARA at P1.

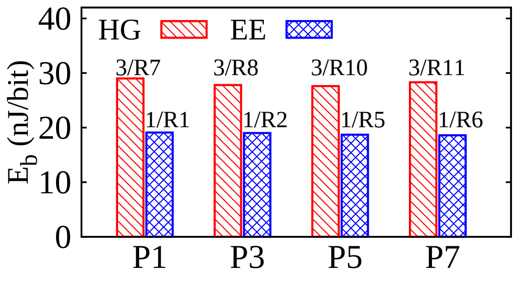
MCS options for 802.11n). The total number of settings can reach $6 \times 8 = 48$. The scaling issue becomes more prominent when the number of antenna grows to eight and the number of MCSes per mode reaches ten in the upcoming 802.11ac [11a11]. This leads to the overall search space of 360 (i.e., $10 \times 8(8+1)/2$) choices.

4.1.3 Dynamics of Energy-Efficient Settings

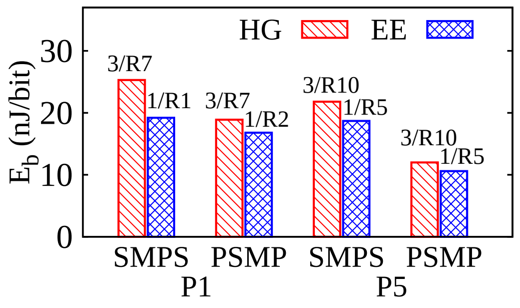
We next show that, the most energy-efficient (EE) setting varies with several factors, including location, data source, power-saving schemes, and the number of activated AP antennas. In contrast, current RAs usually stay at the HG setting, which is typically different from the EE setting.

Location dependence We now show that the energy waste by current RA schemes is location dependent. The fundamental reason is that, the EE setting varies with locations, but it is not the same as the HG one in general. Figure 4.6(a) shows energy-per-bit for HG and EE settings at various locations; the data source is set as 30 Mbps and AP uses three transmit antennas. It shows that the HG setting underperforms the EE one in terms of energy efficiency. Specifically, HG consumes 51.8%, 46.3%, 47.6%, and 52.2% extra energy compared with EE at P1, P3, P5, and P7, respectively.

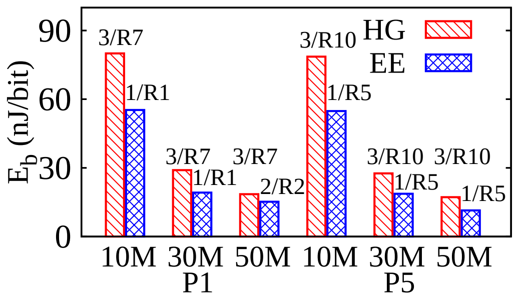
Effect of power-saving schemes We next show that current RA schemes still incur energy waste,



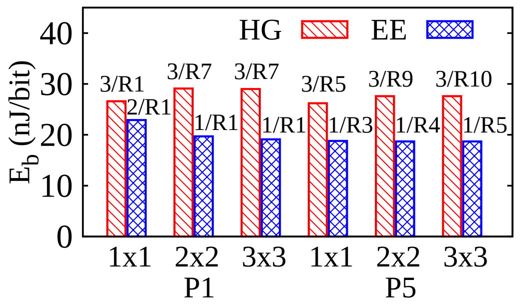
(a) Client locations



(b) Power saving schemes



(c) Source rates



(d) Number of AP antennas

Figure 4.6: Per-bit energy consumption of the HG and EE settings under varying factors. The setting is represented by $N_r/Rate$, where the rate of each label is as follows. For the number of transmit antennas (N_t), the AP always uses three antennas by default in the first three cases, whereas the last case varies with different secnarios. (R1: 40.5SS, R2: 54SS, R3: 81SS, R4: 108SS, R5: 121.5SS, R6: 135SS, R7: 81DS, R8: 162DS, R9: 216DS, R10: 270DS, R11: 324TS)

whether or not we use the power-saving schemes. However, power-saving schemes may reduce the waste percentage because of reduced power during the non-active (idle/sleep) state. The root cause is that, the EE setting varies with the use of power-saving schemes, while the HG setting is invariant as long as the channel condition remains unchanged. For example, when we use different power-saving schemes (i.e., SMPS and PSMP), the EE setting at P1 turns into 3x1/40.5SS and 3x1/54SS, respectively, still different from the HG setting (3x3/81DS). Note that, the energy-per-bit of the HG and EE settings indeed decreases due to smaller idle/sleep energy consumption. The difference thus becomes smaller. However, the gap is still as large as 31.8% at these two locations as shown in Figure 4.6(b).

Effect of data source rate We also study the impact of source rates on energy efficiency. We note that the energy inefficiency of current RAs also varies with data source. The reason is that, the EE setting varies with source rates, while the HG setting remains unchanged. For instance, when the source rate increases from 10 Mbps to 50 Mbps, the EE setting at P1 changes from 3x1/40.5SS to 3x2/54SS. The energy waste by the HG still reaches 44.7%, 51% and 21.7%, for three source rates, as shown in Figure 4.6(c).

Effect of activated AP antennas The energy efficiency of current RA schemes also changes with the number of activated antennas. When the number of AP antennas reduces from 3 to 1, both the HG and EE settings change accordingly. For instance, the HG setting at P5 changes from 270DS to 216DS, and finally to 121.5SS using three receive antennas, while the EE setting changes from 121.5SS to 108SS and finally to 81SS, using only one receive antenna. Therefore, the difference of energy-per-bit between HG and EE is still as large as 47.6%, 47.6% and 39.4%, respectively, as shown in Figure 4.6(d). In general, the fewer the number of AP antennas, the smaller the gap between HG and EE settings. Fundamentally, it depends on how much more receive antennas contribute to goodput improvement and power increase. When the receiver is closer to the AP, the marginal goodput gain is small when activating an extra receive antenna; it is ineffective to use more receive antennas in terms of energy efficiency. However, an extra receive antenna may bring higher marginal gain when the client is far away from the AP.

4.1.4 Impact of EE settings

The EE setting deviates from the HG setting, thus slowing down data transmission and prolonging the delivery time. This affects both the client itself and other clients.

EE client itself The EE setting used by a client affects its throughput and delay. Take the client at P3 of Section 4.1.2 for instance. The portion of active transmission period increases from 29.9% to 68.6%, when the EE setting 3x1/54SS, not the HG setting 3x3/162DS, is used. This slowdown reduces the client's throughput from 100.0 Mbps to 43.9 Mbps. Moreover, it increases both the transmission time and the queuing delay. The packet delay increases from 0.25 ms to 0.41 ms at

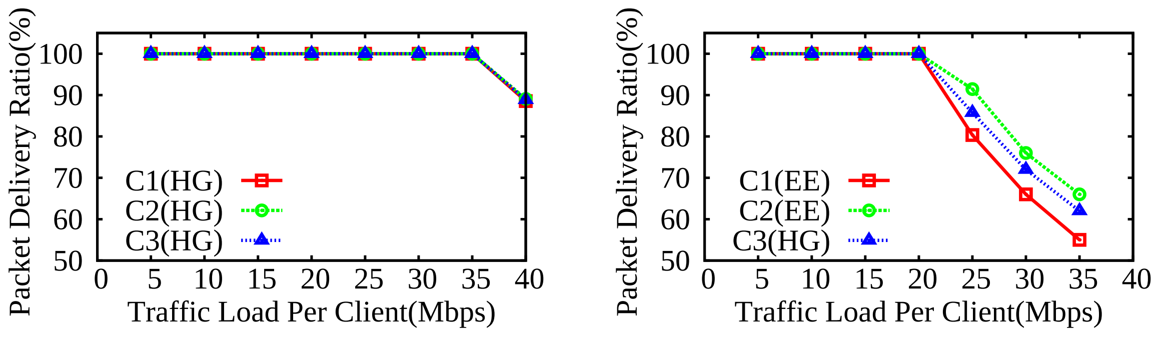


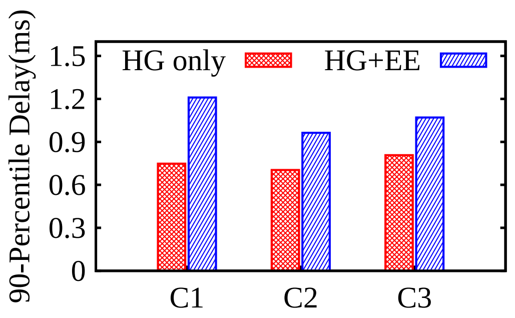
Figure 4.7: Packet delivery ratio varies with traffic load at each client. Left: all the clients are HG. Right: two clients (C1, C2) are EE, whereas the other (C3) is HG.

the medium, and from 0.75 ms to 1.21 ms at the 90-th percentile.

Other HG clients We conduct a three-client study to access the impact of EE settings in the multi-client case. We let two clients (C_1 and C_2) use their EE settings, 3x1/54SS and 3x1/108SS, respectively. The third (C_3) uses its HG setting, 3x3/81.5SS. We compare it with the default case where all clients use their HG settings. Note that the HG settings of C_1 and C_2 are 3x3/162DS and 3x3/324TS, respectively.

For each client, we plot in Figure 4.7 on how its packet delivery ratio (i.e., the percentage of successfully delivered packets out of source traffic load) varies with source traffic load. It shows that, the affordable traffic load decreases in the EE case (right plot), compared with the default HG case (left plot). The HG case supports three clients up to 35 Mbps traffic load each. However, when two clients use their EE settings, the affordable load of each client becomes 20 Mbps. This is because the slowdown due to the EE settings reduces the available air-time for other clients.

We also study the delay impact. Take C_3 as an example. As shown in Figure 4.8, when all clients use HG, C_3 gets the delay, 0.81 ms and 1.07 ms, at the 90-th and 95-th percentiles, respectively. However, they become 0.96 ms and 1.15 ms, respectively, when both C_1 and C_2 adopt EE settings.



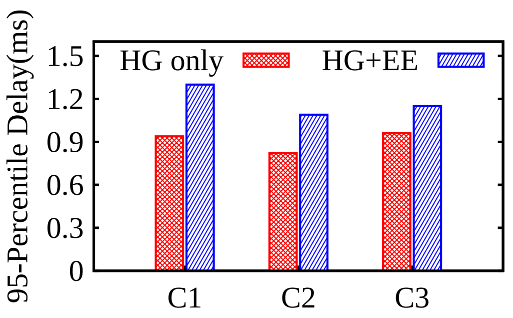


Figure 4.8: The 90-th(Left) and 95-th(Right) percentile delay varies with clients in the two conditions as in Figure 4.7.

4.2 Understanding Energy Efficiency in 802.11n

To better understand where energy savings can be obtained, we now measure and model the power consumption of an 802.11n NIC.

4.2.1 Per-bit Energy

The NIC energy efficiency is quantified by the metric of per-bit-energy, i.e., the consumed energy while transmitting/receiving each single bit. Given the time interval of interest T, the per-bit energy is calculated as

$$E_b = Energy/N_{Bits} = (P \times T)/(G \times T_a), \tag{4.1}$$

where P represents the average power consumption during T and G represents the average goodput during active time T_a. When the traffic can be accommodated by the setting, namely, the data source rate S is no larger than the achieved goodput G (i.e., $S \leq G$), the client may experience both active and non-active (i.e., idle/sleep) modes. We have $P \cdot (T_a + T_{na}) = P_a \cdot T_a + P_{na} \cdot T_{na}$, where P_a and P_{na} are the active and non-active power consumption by the 802.11n receiver. Since data delivery only occurs during the active period, we also have $G \cdot T_a = S \cdot T$. Therefore, E_b can be computed as

$$E_b = \frac{P_a \times T_a + P_{na} \times T_{na}}{S \times T} = \frac{P_a - P_{na}}{G} + \frac{P_{na}}{S}. \tag{4.2}$$

Clearly, the data source S is determined by higher-layer applications, the power consumption (P_a and P_{na}) depends on the NIC state including both rate settings and power-saving modes, and the goodput G is decided by rate settings and wireless channels. For example, Section 4.1.3 shows that client locations and the number of AP antennas both affect the achieved goodput. The goodput under dynamic wireless channels has been well studied in conventional RA work [WGB08, WYL06, PHW10, HHS10, VBJ09, ASB10, CQY07, GK11]. Here, we examine the active and non-active power consumptions of an 802.11n NIC.

4.2.2 Power Model of an 802.11n NIC

We next model and measure the receiver's power consumption in active, idle and sleep states, and the transmitter's active power.

Active receive power The power consumption of an active 802.11n receiver can be decoupled as $P_{rx} = P_{rc} + P_{rb}$, where P_{rc} is the power consumption of MIMO circuitry, and P_{rb} is that of baseband signal processing [CGB05]. P_{rc} includes power consumption of all circuit paths, each of which contains all the circuits from RF to analog-to-digital converter (ADC), e.g., frequency synthesizer, low/band pass filter, mixer, low noise amplifier, and variable gain amplifier. Based on the MIMO power model of [CGB05, GD11], the ADC power can be estimated as a linear function of bandwidth whereas the remaining circuits consume constant power. The baseband power consumption P_{rb} scales with bandwidth and also depends on the number of receive chains. The decoder power consumption correlates with the number of streams and rate settings; P_{rb} can be approximated as a linear function of the number of receive antennas, channel width, and rate. Therefore, it can be formulated as

$$P_{ra} = (\alpha_1 \cdot N_r + f(N_{ss})) \times BW + \alpha_2 \cdot N_r + \alpha_3 \cdot R + P_f,$$

where BW is channel width (MHz), P_f is constant power consumption (mW) and $\alpha_1, \alpha_2, \alpha_3$ are model coefficients; they are platform dependent [LPL12a].

Table 4.3 lists the model coefficients for Atheros 9380 and Intel 5300 NICs based on our measurement. Figure 4.9 plots the measured and estimated values of an Atheros 9380 receiver under

Platform	α_1	α_2	α_3	$f(N_{ss})$			P_f	i_1	i_2
				SS	DS	TS	(mW)		
Atheros 9380	2.31	19.8	0.3	0.6	4.6	7	429.0	2.31	19.8
Intel 5300	2.95	195	0.33	3.3	4.1	4.3	496.8	2.9	195

Table 4.3: Receive power models for Atheros 9380 and Intel 5300.

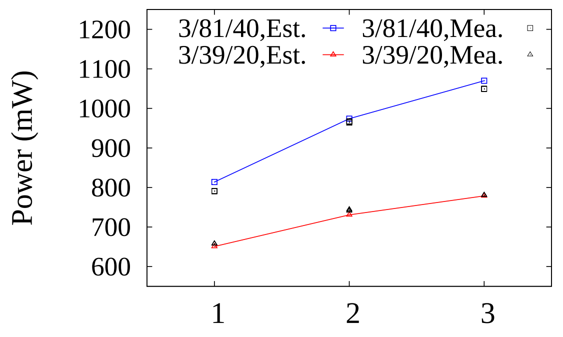
various factors. It shows that this power model matches well with actual power consumption; the gap between estimated and measured values is within 3%. Therefore, active power grows in proportion to the number of receive chains, rate, or channel width when other factors are fixed. Figure 4.9(a) shows that the active power (i.e., $f(N_{ss})$) grows sublinearly as a function of the number of streams. The active power spreads between 500∼1200 mW. We also note that, active power consumption increases significantly with the number of streams/receive-antennas. However, given the same N_r and N_{ss}, the difference of active power using different MCS indices is negligible. For instance, 3x3/13.5SS and 3x3/135SS consume about 798.8 mW and 823.1 mW, respectively, close to 812.3 mW for 3x3/81SS. This is understandable since α_3 is much smaller than other factors such as α_1 and α_2.

Idle power The idle power P_{ri} roughly equals to the circuitry power (P_{rc}) since almost no data processing is used during idle time. It can thus be estimated as

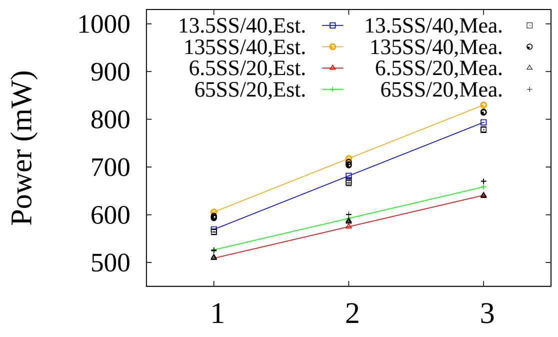
$$P_{ri} = i_1 \cdot N_r \times BW + i_2 \cdot N_r + P_f,$$

where i_1 and i_2 are idle power coefficients. The measured parameters for Atheros 9380 and Intel 5300 wireless NICs are also given in Table 4.3. Figure 4.10(a) plots the estimated and measured idle power values for each setting. They differ within 1%. Therefore, the idle power is only determined by the number of antennas and channel width, independent of the number of streams. For example, Atheros 9380 consumes about 541.2 mW, 653.4 mW and 765.6 mW while using one, two, and three receive antennas during idle (BW = 40MHz).

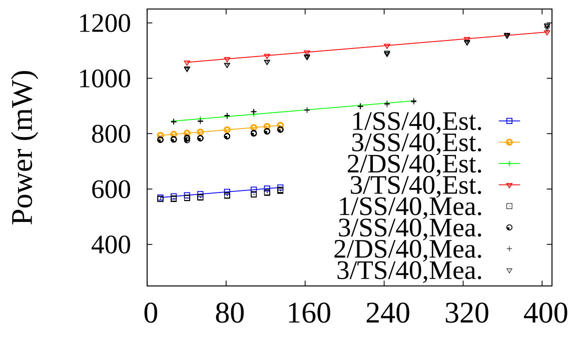
Sleep power In the sleep mode, we find that few components remain active. Both Atheros 9380 and Intel 5300 cards consume constant power (158.4 mW and 166.5 mW). It saves at least $2/3$ of energy, compared with the idle mode.



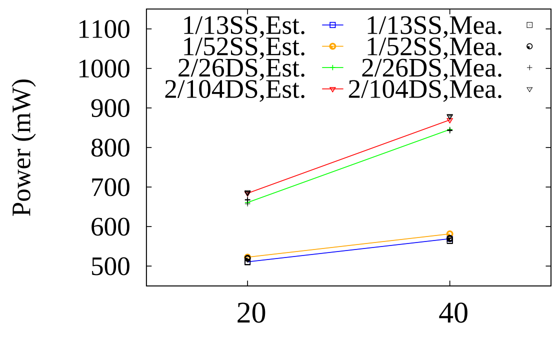
(a) Power versus number of streams



(b) Power versus number of chains



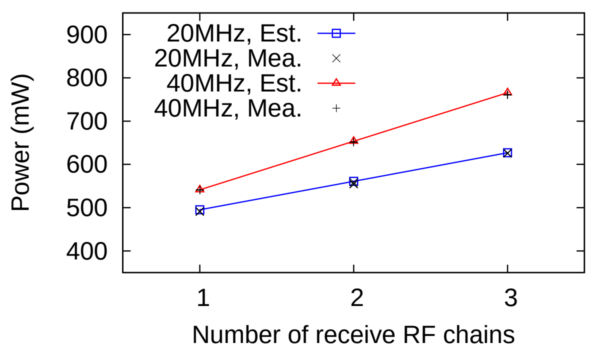
(c) Power versus rate



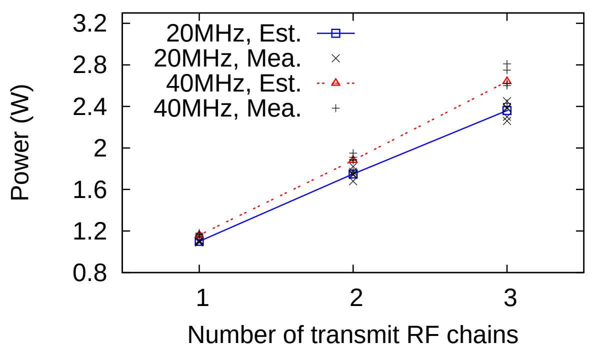
(d) Power versus bandwidth

Figure 4.9: The measured and estimated active receive power consumption for an Atheros NIC. varies with number of streams, number of receive chains, rate, and bandwidth. Each setting in these four plots is denoted by $N_r/R/BW$, RN_{ss}/BW, $N_r/N_{ss}/BW$ and N_r/RN_{ss}, respectively.

Transmit power The 802.11n transmit power can be decoupled into $P_{ta} = P_{PA} + P_{tc} + P_{tb}$, where P_{tc} and P_{tb} are consumed power for transmit circuitry and baseband processing, and P_{PA} is for power amplifier (PA) that dominates the transmit power. Our measurement shows that transmit power for Atheros 9300 NIC is in proportion to the number of transmit antennas and varies with channel width (as shown in Figure 4.10(b)), but remains almost invariant to MCS and the number of streams. The measured values for Atheros 9380 are 1.16 W, 1.88 W and 2.64 W in case of one, two, and three transmit antennas (BW = 40MHz). The transmit power for Intel 5300 is given in [HGS10].



(a) Idle power



(b) Transmit power

Figure 4.10: The measured and estimated idle/transmit power of an Atheros NIC.

4.3 EERA Design

EERA trades off goodput for energy savings at an 802.11n client NIC. It locates the MIMO setting consuming less per-bit energy at NIC, rather than one reaching higher goodput. The overall idea of EERA is to let each client select the most energy-efficient setting from its feasible candidates while slowing down. However, this slowdown is constrained by two conditions: EERA must do its best to accommodate the data source, and not affect other clients when they were to choose their highest-goodput settings.

EERA supports both single-client and multi-client operations. In the single-client case, it abstracts the problem as multi-dimensional search, and exploits ternary search and MIMO characteristics to speed up its runtime convergence. The multi-client case builds on top of the single-client design, but requires additional operations. Note that each client cannot slow down too much to affect others. This is done by setting a limit, expressed in airtime share, for each client. A client cannot select a setting such that its extra communication time (due to slowdown) exceeds its airtime share, no matter how energy efficient this setting can be. In addition, EERA allows each client i to specify a threshold parameter $R_{c,i}$, which defines the minimum goodput that EERA cannot go below when selecting settings. It configures how much a client is willing to slow down to save its NIC energy. To this end, AP first calculates the temporal fair share (defined in airtime) for each EERA client based on per-client parameter $R_{c,i}$ and actual traffic demands and channel conditions.

Assume that each client uses its highest-goodput rate and extra air time is available thereafter. The extra air time is then fairly allocated among all active clients. Each client then uses the single-client algorithm to minimize energy, given its fair airtime share. This way, an EERA client can achieve energy-efficient delivery but cannot be arbitrarily slow to impede others. We now present detailed design on both single-client and multi-client cases.

4.3.1 Single-Client Case

We first consider the simple, single-client scenario. In this case, EERA formulates the energy-efficient RA as a multi-dimensional search problem that locates the low-energy MIMO setting. It organizes settings into a multi-level tree, and then applies the ternary search scheme over each branch. At each setting, EERA uses probing to obtain the per-bit energy. The probing is "in band" by using multiple data frames sent from the AP to the client. By further exploiting the MIMO communication features, EERA can simultaneously prune multiple branches at runtime, thus eliminating those probings deemed unnecessary. The solution also works with/without complementary power-saving schemes (e.g., SMPS and PSMP). There are three key issues: (a) How to organize the settings into a search graph for fast lookup? (b) How to prune branches and reduce probing at runtime? and (c) How to estimate the per-bit energy for each setting? We next elaborate on these details.

4.3.1.1 RA as Multidimensional Search

In addition to MCS rates, MIMO RA in 802.11n has to consider more dimensions: the number of transmit and receive antennas activated, and the number of data streams used. We thus abstract the problem of RA as multidimensional search. The goal is to find the setting given pre-specified optimality criteria. The traditional objective for a RA is high goodput, whereas the goal for EERA is reduced energy consumption. Since low-energy settings also depend on the data source (shown in Section 3.3), we pose RA as the following problem: Given the data source, EERA searches for lower-energy settings that can sustain the source along four dimensions: the number of transmit antennas N_t, the number of receive antennas N_r, the number of data streams N_{ss}, and the various

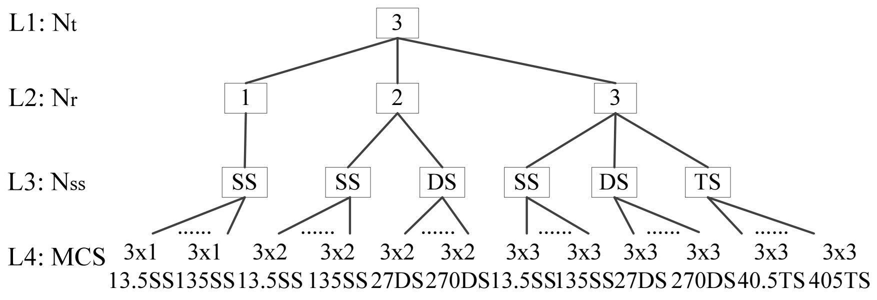


Figure 4.11: An example of the MIMO search tree.

MCS options N_{MCS}.

We organize the search graph as a four-level tree, where each node denotes a setting with its estimated per-bit energy. As shown in Figure 4.11, the hierarchy of the tree is built following the order of N_t, N_r, N_{ss}, and N_{MCS}. As the first heuristic, the AP uses the maximum number of antennas. This eliminates the top level and reduces to a three-level tree. The rationale is as follows. Our goal is to reduce energy consumption at the client with full collaboration from the AP. Therefore, it is easy to show that, given the maximum number of antennas activated at AP, the client has the largest number of choices, thus leading to better search results on energy savings.

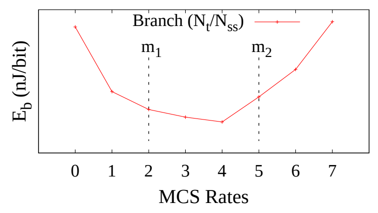
The search tree-based abstraction also illustrates how traditional RAs work. They typically follow sequential search (e.g., MiRA and ARA) or randomized search (e.g., the MIMO version of SampleRate [PHW10]). Consequently, these algorithms have the complexity of $O(N_t N_r N_{ss} N_{MCS})$. Take MiRA at P1 in Section 4.1.2 as an example. The energy-per-bit of all the settings is given in Figure 4.4. MiRA will go all steps shown in Table 4.4 to reach the low-energy setting 3x1/40.5SS. It takes 35 steps. Given each branch with the same N_r and N_{ss}, sequential search needs to keep on probing until reaching the setting after the optimal one. In contrast, EERA uses a novel solution technique, called ternary search with simultaneous pruning, to locate low-energy setting in EERA. The resulting algorithm is more efficient than sequential or randomized search.

4.3.1.2 Ternary Search in Each Branch

The proposed ternary search uses the following property: (the proof is in [LPL12a])

Branch	Probing Sequence: MCS (E_b nJ/bit)	Steps
*3x1/SS	135SS(∞) $\cdots\rightarrow$ **40.5**SS(19.2) $\rightarrow$ 27SS(25.7)	7
3x2/DS	270DS(∞) $\cdots\rightarrow$ **54**DS(27.6) $\rightarrow$ 27DS(36.6)	8
3x2/SS	108SS(∞) $\cdots\rightarrow$ **54**SS(22.7) $\rightarrow$ 40.5SS(22.9)	4
3x3/TS	405TS(∞) $\cdots\rightarrow$ **81**TS(30.3) $\rightarrow$ 40.5TS(33.0)	8
3x3/DS	162DS(∞) $\cdots\rightarrow$ **81**DS(29) $\rightarrow$ 54DS(30.2)	4
3x3/SS	121.5SS(∞) $\cdots\rightarrow$ **81**SS(26.4) $\rightarrow$ 54SS(26.5)	4
Total		35

Table 4.4: Search steps of MiRA sequential search at P1.



Branch	Probing Sequence: MCS (E_b nJ/bit)	Steps
*3x1/SS	**40.5**SS(19.2) $\rightarrow$ 108SS(∞) $\rightarrow$ 54SS(19.3) $\rightarrow$ 27SS(25.7)	4
3x2/DS	**81**DS(22.9) $\rightarrow$ 216DS(∞) $\rightarrow$ 108DS(∞) $\rightarrow$ 54DS(27.6)	4
3x2/SS	40.5SS(22.9) $\rightarrow$ 108SS(∞) $\rightarrow$ **54**SS(22.7) $\rightarrow$ 81SS(26.0)	4
3x3/TS	121.5TS(32.9) $\rightarrow$ 324TS(∞) $\rightarrow$ 162TS(∞) $\rightarrow$ **81**TS(30.3) $\rightarrow$ 40.5TS(33.0)	5
3x3/DS	**81**DS(29) $\rightarrow$ 216DS(∞) $\rightarrow$ 108DS(39.7) $\rightarrow$ 54DS(30.2)	4
3x3/SS	40.5SS(26.6) $\rightarrow$ 108SS(∞) $\rightarrow$ 54SS(26.5) $\rightarrow$ **81**SS(26.4)	4
Total		25

Figure 4.12: Ternary search. Figure 4.13: Search steps of EERA ternary search at P1.

Property 1. *The per-bit energy E_b is a unimodal function with respect to the MCS rate, given fixed number of chains and fixed number of streams.*

This property makes case for ternary search. Note that binary search cannot be applied since E_b is not a monotonic function with respect to MCS rates. We sort the MCS rates in the increasing order based on their indices, say, $[l, r]$, and find the MCS rate that yields lower per-bit energy. In ternary search, we select two intermediate points that partition the interval into three equal segments, i.e., $m_1 = l + (r - l)/3$; $m_2 = r - (r - l)/3$, as shown in Figure 4.12. There are three cases: (1) if $f(m_1) < f(m_2)$, then the minimum cannot be on the right side $[m_1 + 1, r]$. We then search only the left side $[l, m_1]$; (2) if $f(m_1) > f(m_2)$, then the situation is similar. The minimum cannot be on the left $[l, m_2 - 1]$, so we go to the right side - $[m_2; r]$; (3) If $f(m_1) = f(m_2)$, then the search should be conducted in $[m_1, m_2]$. It can be solved recursively by referring to the first two cases.

An illustrative example is given in Figure 4.13. We apply ternary search on each branch. Take the branch 3x1/SS as an example. Initially, the indices of the MCS rates are $[0, 7]$, and we choose two intermediate rate settings, 40.5SS and 108SS, to partition the branch into three segments, i.e., $m_1 = 2$; $m_2 = 5$. The former setting can achieve 19.2 nJ/bit, whereas the latter gets high per-bit energy due to high loss. Therefore, the minimum of per-bit energy is located in the interval $[0, 4]$. Then, the next intermediate points picked for probing are 54SS and 27SS. Finally, we can locate the optimal setting 40.5SS over this branch. After each branch is traversed, the best setting, 3x1/40.5SS, can be reached. The number of total search steps is 25.

4.3.1.3 Simultaneous Pruning of Branches

It turns out that EERA can be more efficient than the above tree-based scheme, which uses ternary search over each lowest level. The technique is to simultaneously prune the search space (across multiple branches) and reduce probing at each step. Consequently, it not only eliminates some branches for further lookup based on runtime search results, but also reduces the range for the ternary search (i.e., l or r). The technique exploits the MIMO communication characteristics. It has two concrete cases at each search step, depending on whether the MCS rate used by the setting has exceeded the channel capacity.

The first case is when the probing of the rate at the current search step does not result in high packet loss, i.e., it yields reasonable goodput. In such a case, we can apply the low-loss-pruning rule to eliminate some lower MCS rates from further ternary search, given the fixed numbers of chains and streams. It uses the property I that E_b monotonically decreases with respect to MCS in loss-free cases. Specifically, whenever current probing finds a new setting with lower per-bit energy, we use this rule to eliminate those lower-MCS-rate settings, which cannot have the same or lower per-bit energy even in the loss-free case. The rationale is that, when a setting achieves its maximum goodput (in the loss-free case), it obtains the lowest per-bit energy. If such bounds cannot beat the current setting, these MCS rates can be removed from the search process.

The second case is when the probe of the rate incurs high packet loss (say, larger than a threshold such as 90%), thus giving very low goodput. This tells us that the current MCS rate exceeds the channel capacity. We then apply the high-loss-pruning rule to eliminate some settings that yield higher loss (i.e., further exceeding the channel capacity) from search. It is based on the following property:

Property 2. *Loss monotonically increases with (1) MCS rate R, given the same N_r, N_{ss}; (2) decreasing N_r, given the same R and N_{ss}; (3) N_{ss}, given the same R and N_r.*

When the probe at a setting (R, N_r, N_{ss}) fails, we can then eliminate two more scenarios. The first is the settings with the MCS no smaller than R and the number of chains lower than N_r, given the same N_{ss}. The second scenario concerns those settings with MCS no smaller than R, the number of streams higher than N_{ss}, and the number of chains no larger than N_r. All these settings would fail.

With both pruning heuristics, we further reduce the search steps in the above example to 17, as shown in Table 4.5. During the search of the 3x3/SS branch, we prune up to 23 settings at other branches. For example, high-loss pruning at 3x3/108SS triggers the following 15 settings to be removed based on Property 2: those higher than or equal to 3x3/216DS, 3x3/324TS, 3x2/108SS, 3x2/216DS, and 3x1/108SS. Moreover, based on the low-loss-pruning rule, the per-bit energy of 3x3/81SS (i.e., 26.4 nJ/bit) helps to remove 8 settings, with their loss-free goodput lower than 3x3/81SS: the ones no larger than 3x3/54DS, 3x3/81TS, 3x2/27SS, 3x2/27DS, and 3x1/13.5SS. The pruning incurred by the 3x3/SS branch results in smaller search space, $[2, 4]$, at 3x3/DS. The continuous pruning reduces the remaining search space at each branch to only 2 to 3 MCS rates.

4.3.1.4 Estimation at Each Setting

To facilitate EERA, we need to calculate the per-bit energy for each setting, and the data source rate. For a given setting, we compute instantaneous energy-per-bit E_b upon receiving every aggregate frame at the receiver. The active and non-active power at a given setting can be measured a priori (see Section 4.2), since they do not change at runtime. The goodput is computed upon the

Branch	$[l, r]$	Steps	# Pruned Settings
3x3/SS	$[0, 7]$	4	23
3x3/DS	$[2, 4]$	3	2
3x3/TS	$[2, 3]$	2	0
3x2/SS	$[2, 4]$	3	2
3x2/DS	$[2, 3]$	2	0
3x1/SS	$[2, 4]$	3	0
Total		17	27

Table 4.5: Ternary search steps with simultaneous pruning at P1.

arrival of an aggregate frame. The source rate is also estimated following the procedure described next. Once we obtain the instantaneous E_b for each probe frame, we estimate the moving average E_b at time t, denoted by $\overline{E_b}(t)$, using the instantaneous per-bit energy $E_b(t)$ and the standard procedure $\overline{E_b}(t) = (1 - \alpha) \cdot \overline{E_b}(t - 1) + \alpha \cdot E_b(t)$, where $\alpha = \frac{1}{8}$ is the weighting factor. This can smoothen out transient variations while tracking the evolving trend.

We also estimate the data source rate based on the transmitter buffer. Upon each frame arrival or departure at the buffer, the instantaneous source rate can be estimated as $S(t) = G(t) + \Delta Q(t)$, where $G(t)$ is the outgoing goodput, and $\Delta Q(t)$ is the buffer change at t. We then compute the moving average of $S(t)$, using procedures similar to $\overline{E_b}(t)$. This way, we estimate the source rate by monitoring the change of data buffer and outgoing goodput.

EERA works with/without power-saving schemes such as SMPS and PSMP, which only change the per-bit energy by having smaller non-active power P_{na}. EERA and these mechanisms complement with each other by primarily managing the active and non-active periods, respectively. EERA also has mobility and interference handling mechanisms, similar to [PHW10].

4.3.1.5 Analytical Properties

Finally, we show that EERA has runtime complexity as described in Theorem 3.

Theorem 3. *(Search Complexity) Assume that the power increase due to MCS rates at the MIMO* *receiver is negligible. EERA has the worst-case search complexity no worse than* $O(\log N_{MCS} \cdot N_r \cdot N_{ss})$.

Proof The asymptotic complexity of ternary search over a branch with fixed N_r and N_{ss} is $O(\log N_{MCS})$. The search complexity of EERA considering three dimensions, N_r, N_{ss}, and N_{MCS}, is thus $O(\log N_{MCS} \cdot N_r \cdot N_{ss})$. $\square$

4.3.2 Multi-Client Case

In the multi-client scenario, EERA uses additional mechanisms to prevent its clients from impeding others (running EERA or conventional RA schemes such as ARA/MiRA). An EERA client selects lower-goodput (but more energy-efficient) settings only if it does not affect other clients' transmission when they were to use their highest-goodput rates. Specifically, a client is given certain amount of extra airtime to slow down for energy savings. The extra airtime is allocated through a temporal fair share, which helps to isolate one client from another during transmission. Each client can only use up its fair share of airtime to slow down for energy savings, but cannot spend more time designated for other clients.

Specifically, EERA runs over regular time intervals (called epoch, and its duration is T_{ep}) periodically. During each epoch, it has three phases of operations. In the first phase, AP probes each client for its highest-goodput setting. This can be done via a traditional MIMO RA algorithm such as ARA or MiRA. We can refine ARA and MiRA by eliminating their sequential search. We use binary search over each tree brunch of the multi-dimensional search, similar to Section 4.3.1, but for goodput instead of energy. During the second phase, AP calculates the temporal fair share for each client. The fair airtime share stipulates how much extra time each client may spend when slowing down to save energy. During the third phase, EERA selects the most energy-efficient setting given the constraints set by the airtime share and pre-configured threshold $R_{c,i}$ for client i.

The fair share calculation is as follows. Assume every client uses its highest-goodput rate setting during epoch k. Given the highest goodput $G_{c,i}$ and the source rate $S_{k,i}$ for client i, its used airtime percentage is given by $\frac{S_{k,i}}{G_{c,i}}$. The unused airtime (in percentage) by all n clients during epoch k is thus obtained as $1 - \sum_{i=1}^{n} \frac{S_{k,i}}{G_{c,i}}$. EERA equally allocates this extra airtime among all n clients. Therefore, during epoch k with duration T_{cp}, each client i is allocated airtime share $F_{k,i}$ as $(1 - \sum_{i=1}^{n} \frac{S_{k,i}}{G_{c,i}}) \cdot \frac{T_{ep}}{n}$. If client i cannot use up its airtime share (say, limited by its parameter $R_{c,i}$), we then allocate fair share based on the celebrated max-min fairness [Hah91]. Note that other fairness index (e.g., proportional fairness) may also be used to allocate the extra airtime in EERA.

Once each client is allocated its airtime share, it can effectively apply operations during each epoch, similar to the single-client case. The minor difference is that, tree branches can be further pruned by both parameters of pre-configured threshold $R_{c,i}$ and fair share $F_{k,i}$. The rule is that the selected setting cannot exceed the airtime share, nor yields goodput lower than $R_{c,i}$, compared with its highest-goodput setting during current epoch.

The threshold parameter $R_{c,i}$ is specified by each client in advance. It defines how much a client is willing to slow down to ensure NIC energy efficiency. It can be given as absolute value, or be set in the percentage (say, 90%) of the highest goodput to the client. When $R_{c,i}$ is specified in percentage and chosen as 100%, EERA reverts to traditional goodput-optimizing RA. In fact, the per-client parameter $R_{c,i}$ serves as a tuning knob for EERA. It offers flexible tradeoff between goodput and energy savings. It may facilitate to mitigate cross-client interference. Since it limits on how much an EERA client may slow down, a client can configure this parameter to reduce effect on other traditional RA clients. Moreover, this parameter may take into account application requirements (e.g., minimum throughput needed by video streaming).

4.3.3 Other Issues

There are a few more design issues in EERA.

Coexistence of EERA and non-EERA clients EERA may coexist with clients running other conventional 802.11n RA algorithms. EERA clients are allowed to slow down to save NIC energy only when extra airtime is available. When non-EERA clients are greedy and no extra airtime is available, EERA reverts to goodput-optimizing RA mode.

Greedy clients Greedy clients (e.g., those running TCP flows) always demand highest goodput.

For these clients, we set the pre-configured parameter $R_{c,i}$ as a fixed percentage (say, 90%). The value $1 - R_{c,i}$ is interpreted as the percentage of goodput the client is willing to give up for NIC energy savings during each epoch.

Uplink case EERA can be extended to support the uplink case. Using client transmit power model, EERA minimizes $\frac{P_{ta}-P_{na}}{G_{UL}}$, where P_{ta} is the active transmit power, P_{na}, is the non-active (idle/sleep) power, and G_{UL} is the uplink goodput. AP computes and notifies fair share for each client over each epoch. Note that transmit power is dominated by the power amplifier, and thus power consumption of different MCS rates with the same N_t varies slightly, less than 5% based on our measurement. The mixed uplink and downlink case can also be supported similarly. AP acts as the coordinator by collecting the required information to calculate fair share for each uplink/downlink client. It then notifies each uplink client its airtime share.

Ad-hoc mode EERA currently does not support ad-hoc operations. There are two associated challenges: (1) How to allocate fair share of airtime in the multihop setting? (2) How to coordinate RA operations among multiple clients in a fully distributed manner? We plan to study these issues in the future.

4.4 Implementation and Evaluation

We implement EERA in the open-source driver, ath9k, for Atheros 802.11n Wi-Fi chipsets, and compare it with ARA, MiRA, and MRES [PLL11]. MRES is a recent proposal to improve MIMO energy efficiency by adjusting only the number of RF chains on top of RA. Both MiRA and MRES work with up to two receive chains and DS mode [PHW10, PLL11]. We extend them to support three chains with TS mode. We next present the implementation and performance evaluation.

4.4.1 Implementation

We implement EERA at the transmitter. To save energy at the client, AP coordinates clients to adjust their receive RF chains for downlink transmissions at runtime, whereas each client adapts its transmit chains for uplink traffic. During the association phase, a client enables EERA at AP by issuing a request with mandatory parameters, including the maximal number of receive antennas, power parameters, and minimum goodput threshold. All such messages are exchanged by a 802.11n action frame.

We resolve two major technical issues in EERA operations. First, we enable EERA to work with power-saving schemes. Once a client enables any power-saving scheme (e.g., SMPS) by notifying its AP (e.g., via a SMPS action frame), the EERA at AP automatically updates its power parameters associated with the power-saving scheme and re-estimates its per-bit energy. Second, transient loss may occur during the switching process of the client's receive chains due to inconsistent views on both sides. For example, after the client switches its receive chain setting from three to two, AP may still use the TS rates for its transmission. Then, the client cannot decode these packets with only two receive chains. To address this issue, the rates accepted by both settings are always used during chain switching.

4.4.2 Evaluation

We conduct extensive experiments in static office environment (see Figure 4.1), with a variety of factors on client location, wireless configuration and traffic pattern. We also examine EERA in scenarios of mobility, interference, uplink traffic, multiple clients, and field trials. In the experiment, both AP and clients have three antennas, working on 40MHz channel over 5GHz band. The default setting uses downlink UDP transmission without enabling any power-saving mode.

A brief summary of EERA performance is as follows. In all test scenarios, EERA consistently outperforms other algorithms in terms of NIC energy efficiency. Table 4.6 summarizes its energy-saving percentage in major test settings, which also include field trials in an office building. In general, EERA saves about 30% energy compared with ARA/MiRA in most cases (equivalent to energy waste of 43% by ARA/MiRA). Compared with MRES, EERA saves about 6-36% energy in static settings, as well as 20-24% in mobility and field tests.

	ARA	MiRA	MRES
Static UDP	(13.4-35.6) %	(14.3-36.1) %	(5.8-26.8) %
Static TCP	(5.1-20.5) %	(10.4-32.3) %	(7.3-23.8) %
Application	(26.5-33.9) %	(26.6-35.2) %	(6.7-36.5) %
Mobility	27.8 %	30.1 %	20.3 %
Field Trials	31.7 %	33.1 %	24.1 %

Table 4.6: EERA energy savings over other designs.

4.4.2.1 An Example of EERA Performance

We first assess EERA at a static location P5 with hybrid traffic patterns. It runs 210 seconds, including the first minute with 60 Mbps UDP traffic, the second minute with 500 MB TCP-based file downloading, the third minute with 10 Mbps UDP traffic, and the last 30 seconds for ten small ($<$10MB) file downloading. Figure 4.14 plots the traces over time for both ARA and EERA, which include the delivered bits, the selected major rate settings, energy consumption, and per-bit energy. As shown in Table 4.7, the energy saving of EERA over ARA and MiRA ranges between 29.2–35.1%, whereas that over MRES is 6.8–20.8%. Thus, EERA outperforms all three other algorithms in terms of energy consumption. This is because EERA uses energy-efficient (EE) settings (typically lower-rate settings, 3x1/135SS, 3x1/121SS, and 3x1/108SS). Moreover, another reason that EERA outperforms MRES is its capability of locating the EE settings faster. Note that the process of reaching the EE setting is slower for MRES, which runs on top of conventional RA algorithms, thereby hurting its performance.

4.4.2.2 Single Client Under Various Factors

We now examine the performance of EERA in various single-client scenarios: different locations, different wireless configurations (e.g., the number of AP antennas and power-saving modes), a variety of traffic sources (e.g., source rates, UDP/TCP and applications), and others (e.g., mobility, interference and uplink transmissions).

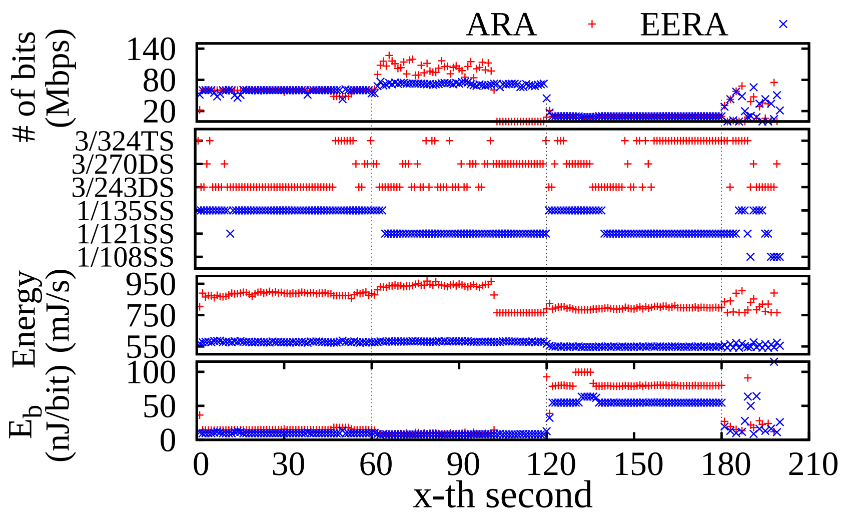


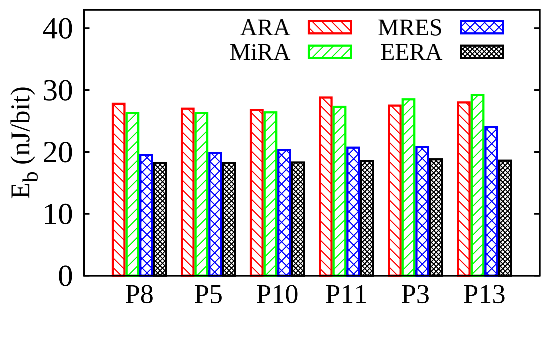
Figure 4.14: Performance trace of EERA and ARA in a hybrid-traffic example.

	ARA	MiRA	MRES
60M UDP (1-60s)	35.1 %	33.4 %	12.2 %
500MB TCP (61s-120s)	34.3 %	29.2 %	9.0 %
10M UDP (121s-180s)	32.5 %	31.8 %	6.8 %
10 files $\leq$ 10MB (181s-210s)	31.4 %	32.0 %	20.8 %

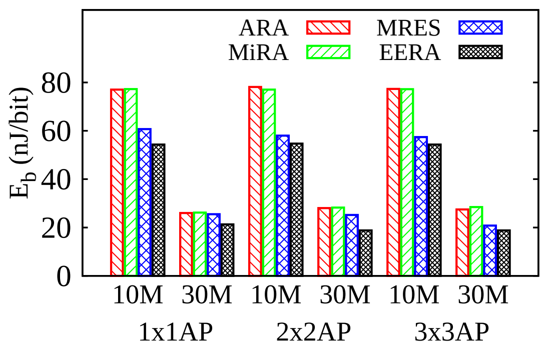
Table 4.7: Energy savings of EERA over alternative designs in a hybrid-traffic example.

Client Locations We examine how EERA performs under various wireless channels, by placing the client at different locations. Figure 4.15(a) plots the per-bit-energy (E_b) with 30 Mbps traffic source. It shows that EERA consistently outperforms other algorithms, with more than 30% energy saving over both ARA and MiRA, and 8.6–22.2% saving over MRES. It is because EERA is able to adjust the used setting to the EE one under various wireless channels, and stay at them. Moreover, only less than 1% frames are used for probing, so it also keeps the impact of probing overhead being little on its energy performance.

Wireless configuration We evaluate EERA with different configurations in two aspects, the number of AP antennas and power-saving mode. We first vary the number of AP antennas, and observe the impact on the performance of EERA. Figure 4.15(b) shows E_b of the client at P3

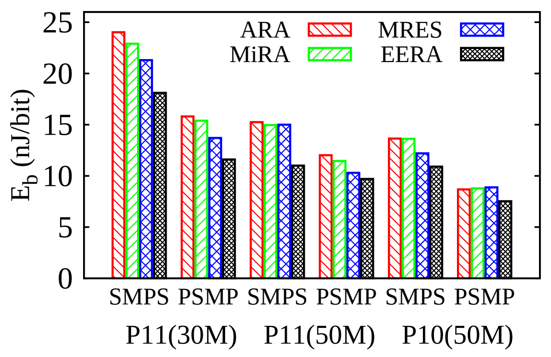


(a) Client locations

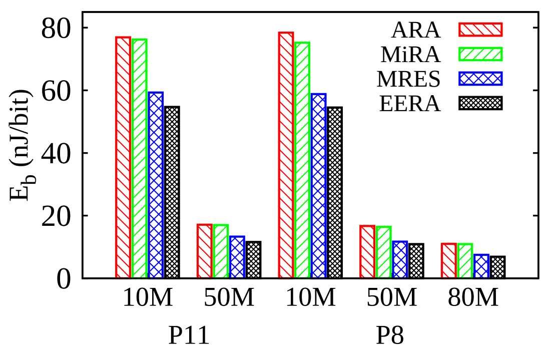


(b) Number of AP antennas

Figure 4.15: Per-bit energy consumption for static clients under various factors.



(a) Power saving modes

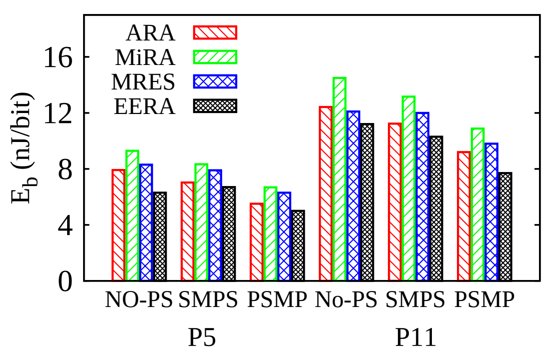


(b) Source rates

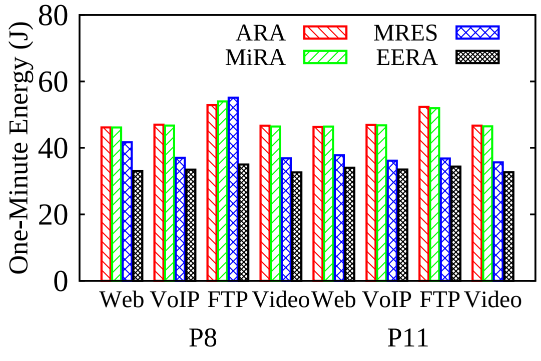
Figure 4.16: Per-bit energy consumption for static clients under various factors.

with 10 Mbps and 30 Mbps source rates. EERA consistently outperforms other algorithms. For instance, compared with ARA and MiRA, the gain is more than 18.5%, 33.0%, and 33.0%, when the client with 30 Mbps is associated with the 1x1, 2x2, and 3x3 APs, respectively. It is respectively 16.5%, 25.2% and 9.6% for MRES.

We then evaluate EERA, when the client enables any power-saving mode (i.e., SMPS or PSMP). Figure 4.16(a) plots E_b of the client with two source rates at P10 and P11. Compared with ARA, MiRA and MRES, EERA still yields energy savings 13.4-28.0%, 14.3-26.6%, and 6.0-26.8%, respectively. Different from the case without any power-saving mode, the smaller power consumption at the idle/sleep state results in the reduction of energy savings. The smaller non-



(a) TCP



(b) Various applications

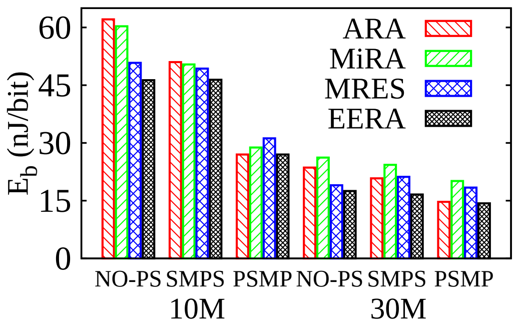
Figure 4.17: Per-bit energy consumption for static clients under various factors.

active power makes the client to favor faster transmission so that it experiences longer non-active state to save energy. However, we note that, racing to sleep cannot achieve the most energy efficiency due to two reasons. First, if the setting for the active state is not selected properly, the energy waste accumulated during the active period cannot be offset by the reduction of energy consumption in the non-active period (i.e., idle or sleep). Second, for some real-time applications (e.g., VoIP, Gaming, Video streaming), the sleep mode is prevented from affecting their real-time performance.

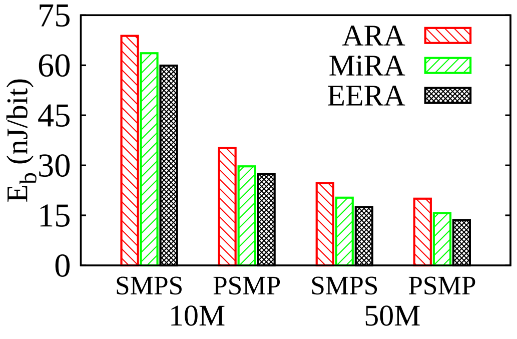
Traffic sources We now evaluate EERA under various traffic sources. We first consider UDP traffic and vary its source rates from 10 Mbps to 80 Mbps at P8, and from 10 Mbps to 50 Mbps at P11. As shown in Figure 4.16(b), EERA consistently saves more than 30% energy over both ARA and MiRA, and outperforms MRES with energy saving 5.1-12.4%. We further observe that the packet size has little impact on the energy efficiency of EERA.

We then test with TCP flows. In this case, source rates fluctuate under dynamic wireless channels due to congestion control. Figure 4.17(a) demonstrates that EERA consistently outperforms others. It produces energy savings from 5.1% to 20.5% over ARA, more than 19% over MiRA, and from 7.3% to 23.8% over MRES. It implies that EERA is able to estimate dynamic TCP source rates well and then fast adapt the EE setting, no matter whether any power-saving mode is enabled.

Finally, we gauge the performance of EERA for four popular applications: (I) Web: fetching a



(a) Interference



(b) Uplink UDP

Figure 4.18: Energy consumption in interference and uplink traffic scenarios.

3.8 MB webpage five times within a minute; (II) VoIP: chatting for two minutes; (III) FTP: downloading a 721.9 MB file; and (IV) Video streaming: playing a 10-minute 1080p HD video. Figure 4.17(b) plots E_b of the client at P8 and P11 for these applications. EERA outperforms all other algorithms, since it adapts the EE setting to different source rates from a variety of application traffic patterns. Its energy savings range between 26.5-33.9%, 26.6-35.2%, and 6.7-36.5%, compared with ARA, MiRA and MRES, respectively.

Other scenarios We now evaluate EERA in the cases of mobility, interference and uplink transmission. In mobility test, we move a client from P6 to P1 through P4 and P2, and then go back to P6 at approximately constant, pedestrian speed of 1 m/s. During mobility, the AP sends 30 Mbps UDP source to the client. Table 4.8 lists E_b and the major rate settings selected by all RA algorithms, at each sub-path. EERA outperforms ARA, MiRA, MRES with energy savings, 27.8%, 30.1%, and 20.3%, respectively. We further study the probing cost. EERA, as well as MiRA and MRES, uses a single aggregate frame to probe each setting. Our trace analysis shows that, EERA requires small probing overhead since it excludes most less-EE settings with simultaneous pruning. For example, in order to reach the EE one, 3x1/108SS, at P4, EERA needs only 7 frame transmissions for the probing. However, 15 and 29 frame transmissions are required by MiRA and MRES, respectively, to reach the best settings they consider. Such low probing overhead also contributes to energy efficiency of EERA over MRES.

We gauge the performance of EERA in the interference scenario by placing the client into

	P6 → P4 → P2 → P1 → P2 → P4 →P6	E_b
EERA	3x1/108SS → 3x1/108SS → 3x1/81SS → 3x2/54SS → 3x1/81SS → 3x1/108SS → 3x1/108SS	19.7
MRES	3x1/135SS → 3x1/121.5SS → 3x2/108DS → 3x2/54SS → 3x1/81SS → 3x1/121.5SS → 3x1/135SS	24.7
ARA	3x3/324TS → 3x3/243DS → 3x3/162DS → 3x3/108DS → 3x3/108DS → 3x3/216DS → 3x3/324TS	27.3
MiRA	3x3/324TS → 3x3/243TS → 3x3/162DS → 3x3/108DS → 3x3/162DS → 3x3/243DS → 3x3/324TS	28.2

Table 4.8: Chosen rate settings vary with sub-paths during mobility.

the crowded 2.4GHz band (Channel 11), on which we observe more than 10 APs. The channel width is switched from 40 MHz to 20 MHz. We evaluate this case for different traffic sources and power-saving schemes at P12, as shown in Figure 4.18(a). Compared with ARA, MiRA and MRES, the energy savings still reach up to 25.8%, 33.3% and 22.1%, respectively, though external interference incurs more packet losses and collisions.

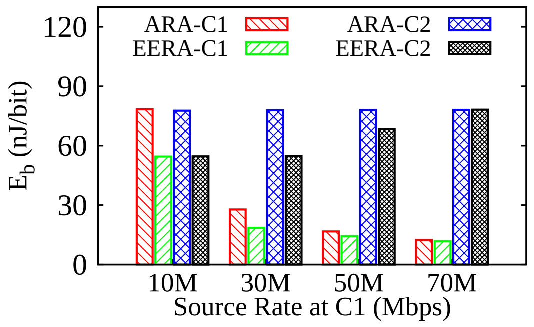
We also evaluate EERA for uplink traffic in the power-saving modes. The goal of enabling power-saving modes is to reduce the effect of the non-active period on the energy. It is because the energy-saving from the non-active period has been considered in the above cases of downlink traffic. We let the client at P8 send UDP traffic to the AP. Figure 4.18(b) shows that, EERA outperforms others with energy savings up to 32%. This is because it is able to employ one transmit chain for its uplink traffic, whereas ARA and MiRA always activate three transmit chains, which are much more power-hungry.

4.4.2.3 Multi-Client Scenarios

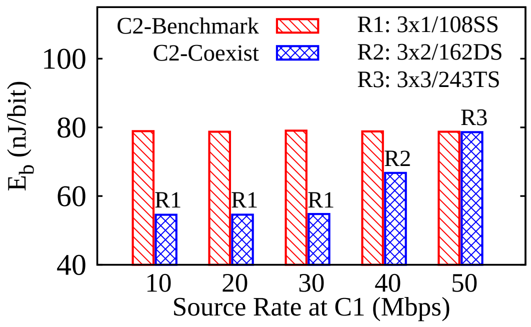
We now examine how EERA performs in multi-client scenarios. The goal is to see whether an EERA client affects others when slowing down its transmission, and how it saves energy when the available airtime decreases. We consider two scenarios: (1) only EERA clients associated with the AP; (2) both EERA and non-EERA (ARA is used here) clients coexist. We test with two clients and three clients in these two scenarios. For the comparison, we also test one benchmark setting, in which all the clients use ARA.

Multiple EERA clients We examine how multiple EERA clients perform, when they coexist and contend for extra airtime to slow down. In the experiment, clients $C1$, $C2$ and $C3$ are placed at P6, P8, and P11, respectively. In order to consider different amount of extra airtime, we vary traffic source from 10 Mbps to 70 Mbps at $C1$. The other clients are loaded with 10 Mbps. Figure 4.19(a) plots the per-bit energy for both clients in the two-client scenario, compared with the benchmark setting. It is seen that, as the source rate increases, the energy-saving gain decreases. At the 10 Mbps source, the $C1$ and $C2$ EERA clients have energy savings, 30.5% and 29.7%, respectively, compared with the benchmark setting of ARA clients. However, the energy savings for both clients at the 70 Mbps source are close to zero, since very less extra airtime can be used by them for slowdown. Similar results are observed in the case of three clients. It is understood that EERA pursues energy saving when the extra airtime is sufficient for its slowdown, and performs similarly to conventional goodput-chasing RAs when it lacks extra airtime.

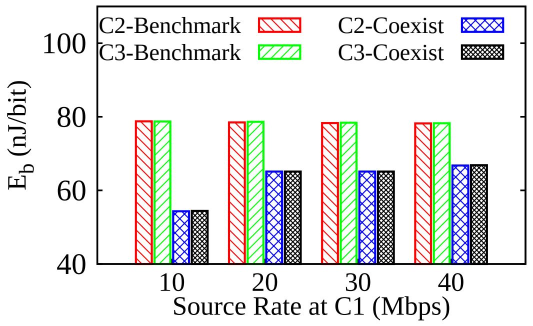
Coexistence of EERA and ARA clients We seek to examine whether EERA clients affect other HG clients, and explore how their energy gains are affected while the traffic demand of the HG clients increases. In this scenario, $C1$ is considered to be HG client running ARA, whereas the others run EERA. Figure 4.19(b) plots the per-bit energy of $C2$ (EERA) varies with the source rates at $C1$ (ARA), compared with the benchmark setting ($C2$ runs ARA). We make two observations. First, the HG client ($C1$) is not affected by the EERA client ($C2$). When its source rate increases, its delivery ratio is always 100% since $C2$ is only allowed to use extra airtime for energy saving. Second, the energy saving of $C2$ decreases as the traffic demand grows. It reduces from 30.8% to 0.1%. Its rate setting switches from 3x1/108SS to 3x2/162DS and 3x3/243TS, when the source rate increases from 30 Mbps to 40 Mbps and 50 Mbps, respectively. It is because its assigned extra airtime becomes less and less. The similar result is observed in the three-client case, as shown in Figure 4.19(c). The energy-saving gain decreases from 31.0% to 14.6% when the source rate



(a) 2 EERA Clients



(b) 1 ARA and 1 EERA Clients



(c) 1 ARA and 2 EERA Clients

Figure 4.19: Per-bit energy consumption for EERA clients in multi-client scenarios.

increases from 10 Mbps to 40 Mbps. Moreover, the traces show that the extra airtime is fairly allocated to $C2$ and $C3$; that is, they get the equal extra airtime share over time.

4.4.2.4 Field Trials

We conduct uncontrolled field trials in our office building during working hours, where various applications from different users dynamically coexist in a complex manner. Two clients are initially placed at P8 and P5. We run TCP flows on both clients for about 30 minutes. During the first half period, they are static; In the remaining period of time, the client at P8 moves to P11, while the other client follows the mobility pattern of Section 4.4.2.2. The result shows that, EERA outperforms ARA, MiRA and MRES with energy savings, 31.7%, 33.1%, and 24.1%, respectively.

CHAPTER 5

LLRA: Latency-Aware Rate Adaptation

In this chapter, we examine the packet latency of 802.11n from both theoretical analysis and empirical study. We show that the high-goodput (HG) operations may fail to minimize tail latency (90^{th} and 95^{th} percentiles). Based on the obtained insights, we design LLRA to reduce tail latency. It offers a unified operation of three components: rate controller, frame aggregation scheduler, and retransmission dispatcher. We implement the design using the standard-compliant approach, and then evaluate it in the home environment.

5.1 Packet Latency in 802.11n

The latency of a packet is the duration from the time a packet arrives at the link layer to the time its acknowledgement (ACK) is received. It consists of three parts:

$$D = T_q + T_{mac} + T_{air}, \tag{5.1}$$

where T_q is the queuing delay, T_{mac} is the duration for the MAC overhead, and T_{air} is the transmission time on the air.

• T_q: the queuing delay includes the initial queuing delay and possible subsequent values due to retransmissions. Its latency comes from two queues: the software queue (T_{SQ}) and the hardware queue (T_{HQ}). T_{SQ} is used to wait for being scheduled to the hardware queue, while T_{HQ} is used to wait for its turn to be delivered once in the HQ. All the packets experience the initial queueing delay, since it counts the waiting period from the packet arrival to its first transmission attempt. For the delay incurred by retransmission, a hardware retry only requires T_{HQ}, whereas a software reschedule includes both, thereby resulting in longer delay.

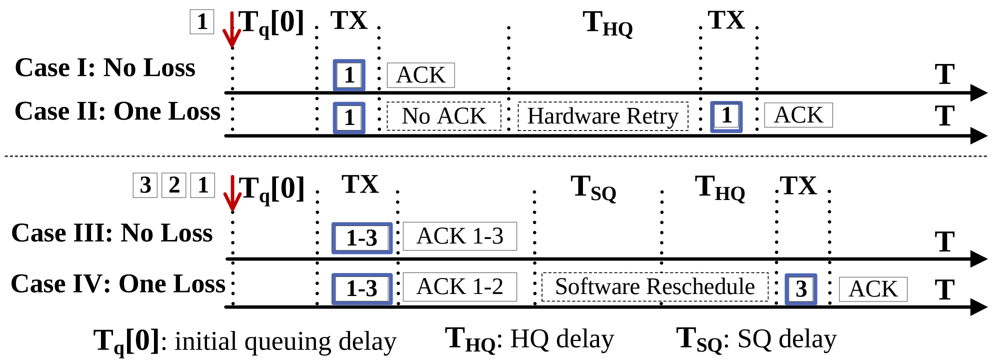


Figure 5.1: Four illustrative examples of the last packet's latency.

• T_{mac}: It counts the overall MAC overhead for all transmission attempts including retransmissions. For each transmission attempt, the MAC overhead covers the time for contention (backoff), inter-frame space (e.g., DIFS/SIFS) and ACK.

• T_{air}: It sums up transmission airtime for all attempts. For a packet in a FA frame, its airtime is determined by the length of the entire frame and the used data rate, since all packets are bundled in their transmission.

5.1.1 Illustrative Examples

We now show how packet latency varies with two rate settings in non-FA and FA cases. Figure 5.1 depicts four illustrative examples. Consider an AP serving multiple clients. For one observed client, assume that the low-rate setting R_L has small PER so that the initial transmission succeeds. Its per-packet transmission airtime is 0.2 ms. The high rate R_H results in smaller airtime (0.15ms) but larger PER; its first transmission fails but the delivery succeeds after one retransmission. Transmission from other clients of interest takes 2 ms before each transmission of AP. Assume the HQ holds only one frame, so 2 ms can be considered as T_{HQ}. The initial queueing delay is 1 ms. The MAC overhead for each transmission attempt is 0.2 ms. Table 5.1 shows the breakdown of the last packet's latency in four cases.

Cases I and II: non-FA case As shown in Table 5.1, Packet 1 gets the latency 1.4 ms and 3.7 ms in Cases I and II, respectively. In Case I, it is successfully delivered during the first attempt.

Case	$T_q[0]$	T_{SQ}	T_{HQ}	T_{air}	T_{mac}	L (ms)
I	1	0	0	0.2	0.2	1.4 (100%)
II	1	0	**2**	0.15+0.15	0.2+0.2	3.7 (264%)
III	1	0	0	0.2*3	0.2	1.8 (100%)
IV	1	**5**	**2**	0.45+**0**.15	0.2+0.2	9.0 (500%)

Table 5.1: Latency breakdown of the last packet in four cases.

However, in Case II, one retransmission via hardware retry is needed, resulting in 2.3 ms extra latency. This shows that PER plays a critical role, yet the airtime saving from higher-goodput rates is almost negligible.

Insight 1: Packet latency is dominated by loss-induced retransmission when other factors have similar effect.

Cases III and IV: FA case When FA is used, multiple packets are transmitted together, thus saving MAC overhead. It can be seen that three packets share the MAC overhead in 0.2 ms (Case III). For the last packet (Packet 3), its latency is 1.8 ms and 9.0 ms in both cases. In case IV, retransmission is still the dominant factor, but much longer latency is incurred compared with Case II. This is due to software reschedule, which is triggered when partial aggregation frame succeeds. In such a case, both T_{SQ} and T_{HQ} contribute to the delay. T_{SQ} is obtained by assuming two 2-packet frames from the SQs for other clients prior to this retransmission. Therefore, T_{SQ} is $2 \times 2.5 = 5.0$ ms (each frame requires $T_{HQ} + T_{mac} + T_{air} = 2 + 0.2 + 0.3 = 2.5$ ms). The overall latency is thus 9.0 ms.

Insight 2: When FA is enabled, retransmission (likely via software reschedule) is deferred longer than that via hardware retry. It also depends on how many packets have been waiting in the other software queues.

5.1.2 Empirical Study

We now use an empirical study to disclose how the HG operations fail to minimize tail latency (we consider both 90^{th} and 95^{th} percentiles) in 802.11n. The operations include rate control (that decides packet loss and retransmissions), frame aggregation (that affects initial queueing delay and results in large latency due to software reschedule), and retransmission. Experimental settings are described in Section 6.4. We consider the latency of the downlink flow from the AP to one observed client at P12, as shown in Figure 5.5.

Figure 5.2 plots the CDFs of packet delay for the client under two traffic patterns. In the left plot, there is only one observed downlink flow and no FA is used. In the right plot, there are six concurrent downlink flows to different clients, and the default, aggressive FA is adopted. We conduct exhaustive search to locate the rate settings yielding highest goodput (HG) and lowest latency (LL). In both cases, the HG and LL settings are 162DS[1] and 108DS. It is easy to see that HG performs worse given the 90^{th} (95^{th}) percentile latency goal.

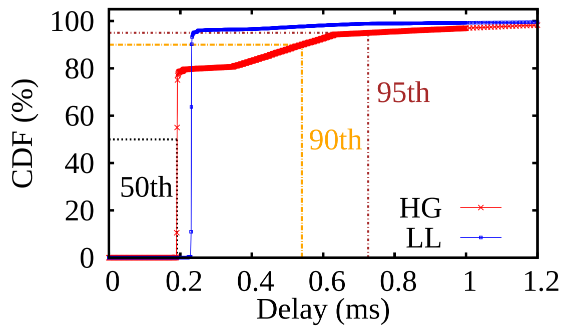
Insight 3: The default HG operations in 802.11n may fail to achieve lowest latency.

We make three observations. First, retransmission (due to high PER) is critical to the tail latency (e.g., the 90^{th}, 95^{th} percentile) (Insight 1). As expected, HG performs better in terms of the medium/average latency. However, for the tail percentile, packets experience retransmissions at the HG rate (its PER is 17.9% and throughput is 85.4 Mbps in the left plot). In contrast, LL has a much lower PER (1.7%) and negligible retransmissions, yielding shorter latency (e.g., 0.23ms versus 0.54ms at the 90^{th} percentile). Under such light traffic, initial queueing delay is negligible so that the time used for transmission and retransmission is more critical.

Insight 4: When initial queuing delay is negligible, the fastest rate setting among those requiring the minimum retransmissions at the considered percentile, is preferred since retransmissions dominate the perceived latency.

Second, FA aggravates the impact of retransmission. Under heavy traffic load (right plot), the

[1] 162DS denotes the 162 Mbps double-stream mode. Other MIMO modes include single-stream (SS), triple-stream (TS), and quadruple-stream (QS).



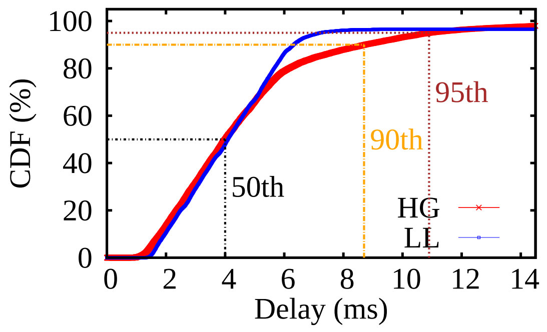


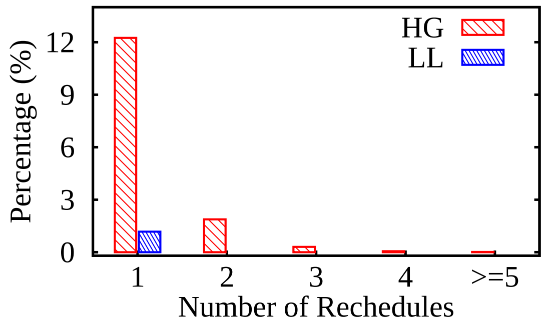
Figure 5.2: Latency in one-client (left) and six-client (right) cases.

latency gap becomes wider. For the 90^{th} percentile tail latency, it turns into 2.7ms, up from 0.31 ms in case of no FA. This is because a software reschedule is employed since FA is used (Insight 2). The left plot in Figure 5.3 further explains why. It shows the number of software reschedules in the six-client setting. Using the HG rate, 12.2% packets encounters one reschedule, and 2.2% have at least two. It induces longer queueing delay. This queuing delay can be even larger, when there are more transmitters (i.e., more clients have uplink traffic), which results in longer T_{HQ} for each transmission. Figure 3.1 confirms this when one more 10 Mbps uplink flow is added.

Insight 5: Retransmission dispatching is required to reduce latency for software reschedule in case of retransmissions (which cannot be avoided in the high-PER scenario such as mobility, interference, etc.).

Third, the initial queuing delay increases with the number of transmissions preceding the packet of interest, especially with the same flow's ones. It is because the considered packet needs to wait for the drain of them, each of which experiences both T_{SQ} and T_{HQ}. We examine the effect of the preceding transmission number on the latency by varying the limit of aggregation size for the observed client using the HG setting. The number of preceding transmissions increases with the decreasing of the size limit. As shown in the right plot of Figure 5.3, the latency greatly increases with the decreasing of the size limit in the 6-client case, compared with the aggressive FA (No limit case). Therefore, to reduce the initial delay of the considered packet, the number of the transmissions preceding the packet should be made as small as possible.

Insight 6: Aggressive FA is preferred to reduce initial queuing delay.



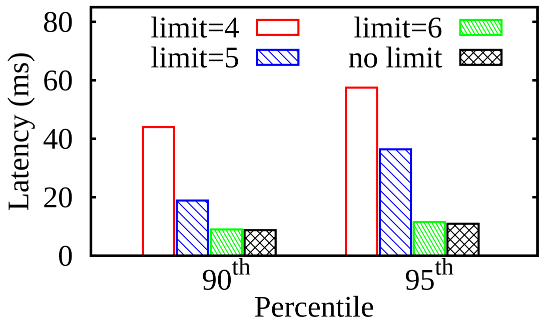


Figure 5.3: Left: Percentage of packets versus the reschedule number. Right: Latency versus the limit of aggregation size.

In summary, the interplay of these three components in RA (i.e., rate control, FA scheduling, retransmission dispatcher) exhibits inherent tradeoffs from the latency perspective. For rate control, the higher rate to use, the smaller airtime to transmit each packet, the possibly higher loss percentage to receive it. For FA scheduler, aggressive FA and the resulting larger aggregated frame size help to reduce the initial queueing delay but it also leads to the likelihood of an individual packet loss within a large frame. The resulting software reschedule by retransmission dispatcher generally incurs larger latency than hardware retry. We will address them in our design.

5.2 Design

The proposed RA has three components of rate controller, frame aggregation scheduler, and retransmission dispatcher, which work together to reduce tail latency. It runs at the AP. We assume that, an inter-client scheduler is responsible for allocating certain fair airtime to each client at the higher level. This can be readily achieved through temporal fair scheduling [CM03]. Note that ensuring low latency for uplink flows at one client is a subset of the problem at the AP, since the client only handles its own traffic, instead of the AP's multiple associated clients.

The overall solution offers a unified operation of three components. Given the $\alpha-$percentile latency requirement (say, α=90%), we split the considered N packets into two groups, one containing the packet with the latency at the $(\alpha \cdot N)^{th}$, and the other with the rest of packets. Each packet's latency consists of initial queuing delay (induced by packet arrival patterns) and service time (including both the airtime plus T_{mac} for transmission and the hold time due to retransmissions). Based on the six insights, we apply three rules:

- **Rule 1:** When queueing delay does not vary with rate settings (e.g., only one transmission is needed to drain the queue), the service time thus makes difference. The fastest rate setting among those requiring the least number of retransmissions is preferred. Otherwise, we should consider the initial queueing delay and service time together for each rate setting.
- **Rule 2:** Aggressive aggregation is applied to the first group of packets, but not together with the second group.
- **Rule 3:** To offset the tail delay of the first group, prioritized reschedule is applied to the group's packets that require software reschedule.

In rate control, we seek to balance between the initial queueing delay and the service time. The general idea works as follows. In the presence of light traffic (e.g., only one aggregate frame in the software queue), we apply rate control that selects the highest rate setting with low PER, rather than the highest-goodput rate that incurs higher PER, to the aggregate frame. This is to reduce the retransmission-induced latency, which typically dominates the service time. Under heavy traffic (e.g., several aggregate frames are required to drain the queue), we apply rate control that balances the queueing delay and the service time, to reduce the initial queueing delay for the large data burst. In this case, the faster yet with larger PER rate may be better, because it drains packets faster.

Both low-latency FA scheduling and retransmission dispatching help to further reduce the queueing and service delays. FA scheduling applies aggressive aggregation to the first group's packets, to minimize the time of draining packets, thereby reducing the queueing delay. It separately applies aggressive aggregation to the second group, to minimize its effect on the queuing delay of subsequent packets. The retransmission dispatcher prioritizes the retransmitted frames from the first group when the queue is empty. It thus reduces the group's tail delay incurred by software reschedule without hurting others' initial queuing delay.

We next elaborate on each component.

5.2.1 LLRC: Low-Latency Rate Control

LLRC searches for the LL rate setting across different MIMO modes, and protects itself from the penalty of excessive probing. LLRC probes those rates which may reach the lowest latency at the α percentile, while applying several pruning rules. The search is triggered by both time-driven and event-driven approaches. With the former, LLRC periodically searches for the best one. With the latter, the event is that the current rate becomes worse. In the penalty protection, in addition to rate pruning, LLRC introduces a novel light-weight probing mechanism, which probes rates' PERs with few packets. We now present the LLRC search and pruning. Probing is described in Section 5.2.3. Note that we use the interference handling mechanisms similar to [PHW10].

Pruning rules We first derive three pruning rules for tail latency. These rules can speed up the search process by eliminating ineligible rates. They are specified by three corresponding corollaries. Two prune lower and higher rate settings from the current best one R_{best} during search, respectively. The third rule is used for latency comparison between two rates. Given the packet latency as $D_{est} = T_q[0] + T_{srv}$. $T_q[0]$ is the initial queuing delay and depends on the number of preceding frames (N_{pre}) of the same flow. T_{srv} is the service time, and depends on the retransmission count (N_{rt}) and the time consumed by each transmission (T_{tx}). The corollaries are derived based on these parameters that vary with rate settings.

Theorem 4. *Given R_{best} with $N_{rt} = 0$, any low rate R (i.e., $R < R_{best}$) is pruned since it cannot perform better than R_{best} (i.e., $D_{est}(R) \geq D_{est}(R_{best})$).*

Proof. Since $R < R_{best}$ and R_{best} has low PER (due to $N_{rt} = 0$), R_{best} has smaller N_{pre} and T_{tx}. Moreover, it also has the smallest value of N_{rt}, 0, then $D_{est}(R) \geq D_{est}(R_{best})$ and R can be pruned. $\square$

Theorem 5. *Given a rate R which $R > R_{best}$ and $D_{est}(R) > D_{est}(R_{best})$, the higher rate R' (i.e., $R' > R$) in the same MIMO mode is pruned since it cannot perform better than $D_{est}(R_{best})$.*

Proof. Since $R > R_{best}$ and $D_{est}(R) > D_{est}(R_{best})$, R must have higher PER such that it has either of larger N_{pre} and larger N_{rt} or both. According to the steep increasing of PER between adjacent

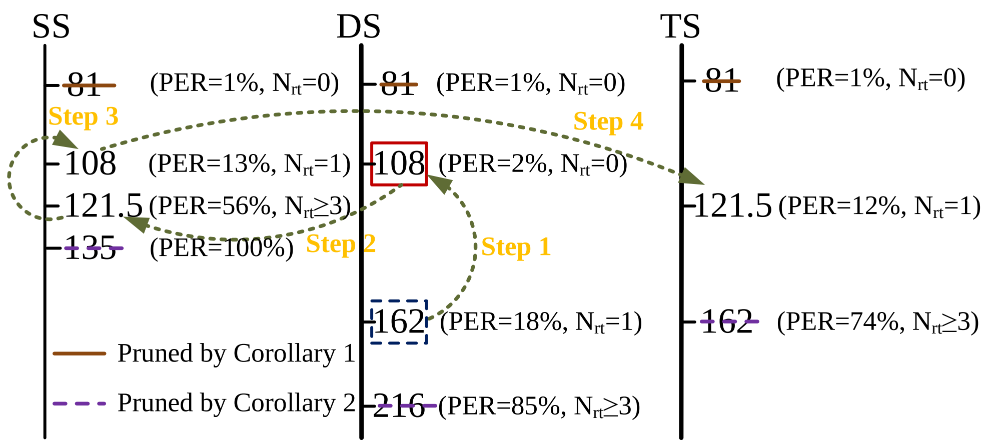


Figure 5.4: Search is triggered when 162DS becomes worse (N_{rt} changes from 0 to 1).

rates with non-negligible PER [PHW10], the higher rate R' with much higher PER which increases both of N_{pre} and N_{rt} cannot perform better and thus can be pruned. Here, we do not consider T_{tx}, since N_{pre} and N_{rt} dominate the queuing delay and the service time, respectively. □

Theorem 6. *Given two adjacent rates R_{low} and R_{high}. If they have identical N_{pre}, the one with smaller N_{rt} is better. If they also have the same N_{rt}, R_{high} is better. If they do not have identical N_{pre}, they need to be compared based on the estimated D_{est}.*

Proof. The same N_{pre} represents that R_{low} and R_{high} have the same initial queuing delay, so N_{rt} which dominates the service time leads the latency. Thus, the one with smaller N_{rt} has lower latency. When both N_{pre} and N_{rt} are the same, R_{high} transmits faster and thus has smaller T_{tx}, thereby achieving lower latency. □

Search algorithm LLRC differentiates MIMO modes (e.g., SS, DS, TS). Its search starts from the same MIMO mode of the current best rate setting (R_{best}). It proceeds along upward direction, and compares tail latency among adjacent rates by Theorem 6. LLRC replaces the current setting with a better one when one is found. This upward search continues until either the highest rate setting is reached or all higher rates are pruned by Theorem 5. It then queries downward from the original best rate setting if the number of retransmissions (denoted as N_{rt}) is one or more; otherwise, all lower rates are pruned ($N_{rt} = 0$) based on Theorem 4. It continues until either the lowest rate setting or the highest one with $N_{rt} = 0$ is reached. After the current MIMO mode, LLRC prunes the rate settings at other MIMO modes by using the highest rate with $N_{rt} = 0$. By Theorem 4, all rates lower than this one can be pruned. In each mode, the search starts from the rate which is next higher than the current best one. It searches upward and then downward, with the same stop conditions.

We illustrate the algorithm via the example of Figure 5.4. It is triggered by the event that 162DS becomes worse. Assume N_{rt} changes from 0 to 1 when the client moves from P10 to P12 (Figure 5.5). We consider the 90^{th} percentile tail latency. Assume that $N_{pre} = 0$. The initial queuing delays thus are the same among different settings. It first searches downward in the same MIMO mode, so it queries 108DS (Step 1) and changes the best rate to it, because its $N_{rt} = 0$ incurs lower latency than 162DS with $N_{rt} = 1$. With identical initial queuing delay, N_{rt} dominates the overall latency. The rates higher than 162DS can be pruned by Theorem 5. Moreover, by Theorem 4, all rates lower than 108DS can also be pruned among all MIMO modes. The search of this mode thus stops. LLRC then searches the next higher rate, 121.5SS, in the SS mode (Step 2). Since it is worse than 108DS due to $N_{rt} \geq 3$, the rates higher than 121.5SS are pruned. It stops at 108SS, which is worse due to $N_{rt} = 1$, (Step 3) in this mode. It further probes the next higher rate, 121.5TS, in the TS mode (Step 4), and prunes higher rate settings. The search completes with the best rate, 108DS.

5.2.2 Aggregation Scheduling and Retransmission Dispatching

We divide packets into two groups based on the retransmission count occurring at the packet of the α percentile. It dominates the latency metric, given the same queuing delay incurred by the current best rate. We collect N_{rt} and the latency of N packets in history, and obtain the retransmission count N_{rt} of the packet (say, β) at the α percentile. The packets with retransmission count no larger than β belong to the first group, which contains the packet with latency at the $(\alpha \cdot N)^{th}$. The remaining packets belong to the second group.

Aggregation scheduling packs the first group's packets as many as possible for transmission.

Once no more packets in the first group are queued, aggressive FA is applied to the second group's packets. If any packet in the first group encounters software reschedule, it is given the highest priority and scheduled to the hardware queue right after its failure. It thus reduces the tail delay of the first group without increasing the initial queuing delay of other packets in this group.

Take Case IV of Section 5.1.1 as an example. In Figure 5.1, we consider two cases: $\beta = 0$ and $\beta = 1$. Assume that the maximum aggregation size of the current rate is larger than 3. In both cases, aggressive aggregation is applied into the aggregation formation of packets 1, 2, and 3, since all have no retransmission count and belong to the first group. However, different actions are taken for the retransmission of Packet 3 in two cases. In the former case, it belongs to the second group due to $\beta = 0$. It is thus not prioritized for retransmission and takes 9.0 ms. In the latter case, it belongs to the first group due to $\beta = 1$. Its retransmission is then prioritized over other traffic. Consequently, it does not wait for other frames in the software queues, thereby taking only 4 ms.

5.2.3 Novel Light-weight probing

We now introduce a novel light-weight probing mechanism; the goal is to probe PER at a given rate. For example, the 95^{th} percentile requires PER to have accuracy no larger than 5% (i.e., at least 20 packets). There are two general approaches to PER estimation: probing-based [PHW10] and SNR-based [HHS10]. Most current real platforms do not support the SNR scheme. We thus take the probing-based approach, in order to implement and test it on real systems.

It consists of three components: slow-start pruning, association rule-based pruning, success inheritance. The first two eliminate probing packets for inapplicable rates, especially those high-loss ones. The last one reduces probing of applicable rates. A rate may inherit successful probings from the higher rate, thereby reducing its own probing. If other flows without latency requirements coexist with the latency-sensitive one, their packets are used for probing. The latency-sensitive flows accordingly have less probing overhead.

Slow-start pruning It detects an inapplicable rate with few packets. To probe a rate, it starts from the frame with one-packet size, and then exponentially increases the frame size (i.e., two-packet,

N_{rt}	1	2	3
$\alpha = 90^{th}$	10.0%	31.6%	46.4%
$\alpha = 95^{th}$	5.0%	22.3%	36.8%

Table 5.2: PER threshold versus retransmission count.

History Probing	121.5SS	162DS	121.5TS
1	1	1	1
2	1	1	0
3	0	0	0
4	1	0	1
5	1	1	1

Table 5.3: Example of five probing transactions for three rates when 108DS is being used. 0 and 1 denote applicable and inapplicable, respectively.

four-packet, and so on). It stops when either the collected results are sufficient or the packet loss count has verified the inapplicability of the rate. A rate is inapplicability when its PER exceeds certain threshold associated with the retransmission count, such that it can perform no better than the current best rate, according to Table 5.2 (to be derived later).

Association rule-based pruning It uses correlations among rates to infer a rate's inapplicability from other rates under similar channel conditions. The underlying premise is that, each rate setting behaves similarly among time-varying samples under the same channel condition, so rate settings can be correlated with each other in their performance. We apply association-rule learning to discover correlations between rate settings. Such correlations are learned from the historical probing statistics. We further obtain the confidence level of each correlation, which indicates its reliability. If the confidence level of one correlation μ is larger than a specified value (say, $\mu \geq 0.9$), the correlation is dependable. We only apply the dependable correlations in the work. Finally, once the dependable correlation of each rate pair is learned, one rate's result can be used to infer the other's. That is, one rate's failure infers that the other rate also fails.

Table 5.3 shows an example of five probing transactions for three candidate rates (108DS is the used, current rate). The inapplicable rates are marked as 1; otherwise, they are set to 0. The correlation "if 162DS fails (i.e., inapplicable), then 121.5SS fails" has the confidence of 1.0 ($\frac{fail(162DS \cup 121.5SS)}{fail(162DS)} = \frac{3}{3}$), and can be used. However, the one "if 162DS fails, then 121.5TS fails" has the confidence of 0.7 ($\frac{2}{3}$), and is not applicable. Consequently, if the probing of 162DS fails, we do not probe 121.5SS.

5.2.4 Parameter Estimation

We further estimate various parameters used in LLRA.

The initial queuing delay includes both software and hardware queueing delays. Assume that the α percentile's packet is always in the last frame, which empties the queue, and aggressive FA is applied. It can be estimated as follows, based on the average number of simultaneous packets in the queue and the rate's maximum aggregation size:

$$T_q[0] = (N_{pre} + 1) \times T_{sw,interval} + T_{HQ}, \tag{5.2}$$

where $T_{sw,interval}$ represents how often a frame at the head of the software queue (SQ) is scheduled to the hardware queue (HQ). The software queuing delay is estimated based on the number of preceding frames (N_{pre}) plus 1 (i.e., itself) and the scheduling interval ($T_{sw,interval}$). T_{HQ} denotes the hardware queuing delay.

Service time depends on the retransmission count N_{rt} and type. Given the expected retransmission count at the α percentile with the given rate's PER. It can be estimated in both non-FA ($T_{srv,nofa}$) and FA ($T_{srv,fa}$) cases:

$$T_{srv,nofa} = T_{tx} + N_{rt} \times (T_{hw,interval} + T_{tx}), \tag{5.3}$$

$$T_{srv,fa} = T_{tx} + N_{rt} \times (T_{HQ} + T_{tx}), \tag{5.4}$$

where T_{tx} is the sum of airtime and MAC overhead for each transmission, and $T_{hw,interval}$ represents how often a frame at the head of HQ is scheduled for transmission. $T_{hw,interval}$ is usually smaller than T_{HQ}, since the size of HQ may be larger than one frame. In $T_{srv,fa}$, T_{HQ} is used for the retransmission of software reschedule. Low-latency retransmission dispatching places the rescheduled packets to the tail of HQ, instead of SQ, right after failure. For simplicity, T_{tx} is treated the same between different transmissions, since its variation is negligible compared with the number of retransmissions, which dominates the service time.

Given the PER at a given rate setting, we estimate the retransmission count (N_{rt}) for the packet at the α percentile. Assume packet error events are independent and have the same probability p. The probability that a successful transmission is preceded by i failures, is given by $p^i(1-p)$. This probability also represents the percentage of the successful frames after $i+1$ transmissions. If the percentage is larger than α, we compute N_{rt} as the smallest integer satisfying $\sum_{i=0}^{N_{rt}} p^i(1-p) \geq \alpha$. As a result, we can calculate the PER threshold for the retransmission count shown in Table 5.2.

Moving average estimation is applied for the following parameters: N_{pre}, T_{tx}, T_{HQ}, $T_{hw,interval}$, and $T_{sw,interval}$.

Given rate setting R, N_{pre} is estimated as $\left\lceil \frac{N_{sp} \times \rho}{maxFA(R)} \right\rceil - 1$, where N_{sp} is the number of simultaneous first-attempt packets appearing together at the queue, ρ represents the ratio of the first-attempt packets to the total number of packets with no more than three attempts, and $maxFA(R)$ is the maximum aggregation frame size of R. Note that the number of packets with more than three attempts is negligible. To estimate N_{sp}, we start to count packets when they come to the empty queue, but stop counting when no more first-attempt packet is in the queue after a frame is scheduled. T_{tx} is given by $\frac{avgFA(R) \times framesize}{tpt(R)} + T_{mac}$, where $avgFA(R)$ is the average aggregation frame size of R, $framesize$ is the average frame size, and $tpt(R)$ is the loss-free throughput of R. $avgFA(R)$ can be calculated as $\frac{N_{sp} \times \rho}{N_{pre}+1}$.

$T_{sw,interval}$ takes into account both the traffic load at the AP and that from other clients on the same channel. It is estimated based on the scheduling interval of two adjacent frames in the SQ. $T_{hw,interval}$ is estimated in two cases. If HQ does not have any frame ahead of the current transmission, we compute $T_{hw,interval}$ as the duration from the time the frame is placed into HQ to the time its ACK is received, while subtracting transmission airtime (T_{air}) and the MAC overhead (T_{mac}). If HQ has frames before the current transmission, $T_{hw,interval}$ is calculated as the duration between two consecutive ACKs for first-attempt transmissions, while also excluding T_{air} and T_{mac} of the second frame. Finally, for the hardware queueing delay T_{HQ}, it is the duration from the time the frame is just placed into HQ to the instant its ACK is received, which excluding T_{air} and T_{mac}.

5.2.5 Other Issues

There are some additional issues in LLRA.

Coexistence of LLRA and other RA goals at different clients LLRA may coexist with the clients pursuing other goals (e.g., highest goodput and energy efficiency) under the same AP. The LLRA clients may slow down from their high-goodput rate settings and grab more airtime, thereby possibly affecting the performance of other clients (e.g., lower goodput or less energy-saving). Nevertheless, the impact can be alleviated or eliminated through an inter-client scheduler. The airtime allocated to each client is constrained by the scheduling policy (e.g., fair airtime allocation via temporal fair scheduling [CM03]). Thus, each client is only able to use up its assigned airtime so that it does not affect the others no matter which goal it pursues. When LLRA clients do not have enough airtime for the slowdown of LL rate settings, they switch to the highest-goodput settings upon their backlogged queue. Note that the scheduling mechanisms and policies of airtime allocation are independent of this work.

Impact of LLRA on non-latency-sensitive flows at the same client LLRA may hurt the throughput of the non-latency-sensitive flows, when LLRA is simply applied to all the flows at one client. It can be addressed by considering latency-sensitive flows and the others separately. For example, a hybrid RA algorithm pursues the lowest latency for the former flows, while retaining the highest goodput for the latter ones. We leave it to the future work.

5.3 Implementation and Evaluation

We implement the design in Atheros 802.11n Wi-Fi driver (i.e., ath9k [ath]), and then evaluate it through extensive experiments in a commodity 802.11n network. We compare LLRA with two 802.11n RA algorithms – default Atheros RA (ARA) [WGB08] and MiRA [PHW10].

5.3.1 Implementation

At the initialization phase, the client MAC is notified of the low-latency requirement for a given flow based on its five-tuple flow ID. The wireless MAC then notifies the AP of this flow requirement via an action frame, a type of management frame. To measure the latency of a packet, we rely on the structure of its associated socket buffer to hold the start timestamp. The timestamp is logged once the packet is passed to the MAC from the upper layer. It is then traced via the sequence number in the MAC header, when it moves between hardware and software queues, transmission, and retransmission. When its ACK is received, the difference between the logged start timestamp and the current one is the overall latency. We execute the design mainly in two modules of the Wi-Fi driver: rate control and transmission modules. The former module implements rate control, whereas the latter takes charge of the FA scheduler and the retransmission dispatcher. Note that the implementation not only uses standard-compliant messages, but also follows the general Wi-Fi driver framework. It can be readily ported to other Wi-Fi drivers.

5.3.2 Evaluation

We examine the latency for a downlink flow in both single-client (non-interference, interference, and static and mobile scenarios) and multi-client cases. We test with spots in a typical single family home environment (the left plot of Figure 5.5) under various factor configurations. Findings are similar. We only present some representative results due to space limit. Our evaluation shows that LLRA consistently outperforms ARA and MiRA in all tested scenarios. For example, it reduces the 90^{th} percentile latency by 1–19 ms in interference and multi-client cases (about 22–68% reduction).

Experimental settings We conduct experiments in the infrastructure mode. The AP is located at the spot$_{AP}$, whereas clients are placed at fine-grained spots from P0 to P15. Both AP and the clients use Atheros dual-band (2.4/5 GHz) 802.11n chipset, supporting up to three antennas and TS mode (the right picture in Figure 5.5). We use iperf to generate constant-rate UDP traffic. Unless explicitly specified, the default latency goal is the 90^{th} percentile tail, each traffic flow has

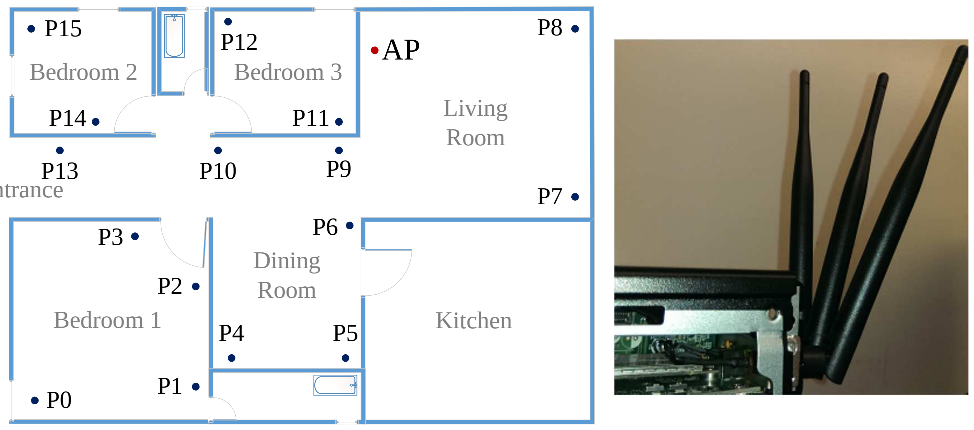


Figure 5.5: Left: home floorplan. Right: 802.11n platform.

10 Mbps, and the packet size is 1470 bytes. For each test, we run 25 times and each run is long enough to collect 30K-packet traces. We present the result with the median packet error rate (PER).

5.3.2.1 Single-client Case

We first consider the non-interference cases over the 5GHz band at various spots or with different latency goals. We further assess it in the interference and mobility cases.

Client Placements We study LLRA under various wireless channels by placing the client at different spots. Figure 5.6 shows the 90^{th} percentile tail latency and the achieved goodput at various spots. LLRA consistently outperforms ARA and MiRA by reducing latency by 30.7%–75.2% and 22.9%–63.4%, respectively. Moreover, it only sacrifices 0.5%-13.3% goodput. It is because LLRA chooses LL rates at low probing costs. Figure 5.7 plots the rate distribution at P3 and P6. The rates chosen by LLRA are a little slower than the HG rates but have lower PERs. Take P3 as an example. LLRA mainly stays at the LL rate, 121.5TS, which has comparatively low PER (1.9%), while also oscillating between two adjacent rates, 162DS and 108DS. In contrast, ARA and MiRA tend to choose the HG rate (162DS), which results in higher PERs ($\approx$ 12%).

Latency Goals We vary latency goals to examine how LLRA performs with different tail latency requirements. We use P14 as an example to demonstrate its impact. The results are similar to other

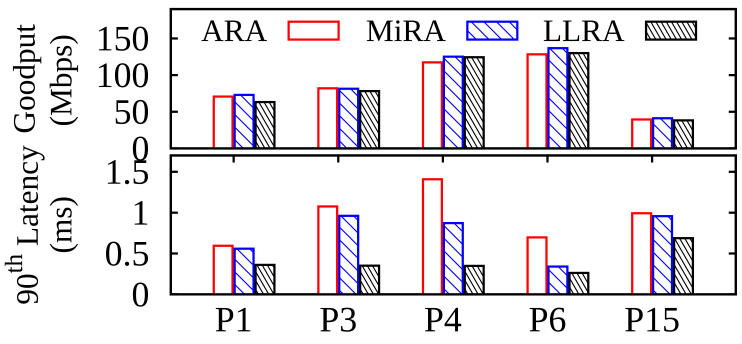


Figure 5.6: 90^{th} percentile latency and goodput vary with spots.

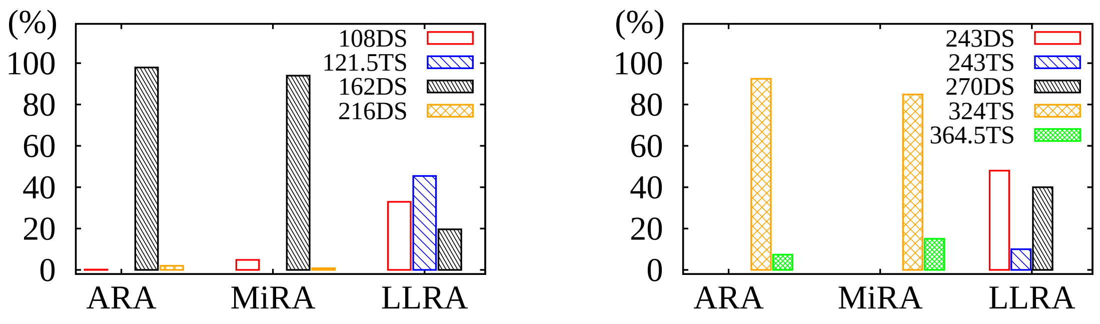
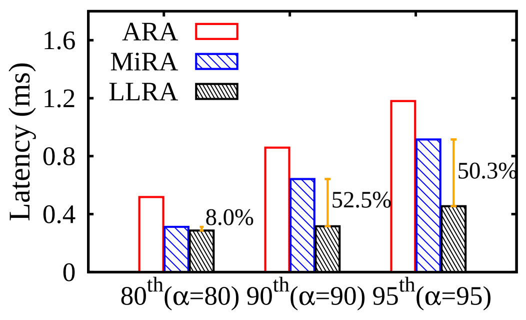


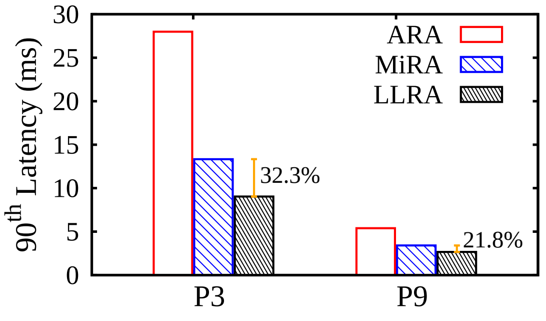
Figure 5.7: Rate distributions at P3 (Left) and P6 (Right).

spots. Figure 5.8(a) shows the latency gain for three percentiles, 80^{th}, 90^{th}, and 95^{th}. We have two observation. First, LLRA outperforms both ARA and MiRA for all three goals. Second, the gain becomes smaller at the lower percentile. The latency reduction decreases from 63.4% to 31.9% (ARA) and from 52.5% to 8.0% (MiRA), when the percentile changes from 95^{th} to 80^{th}. It is because the chosen LL rate is very close to the HG one, as the percentile decreases.

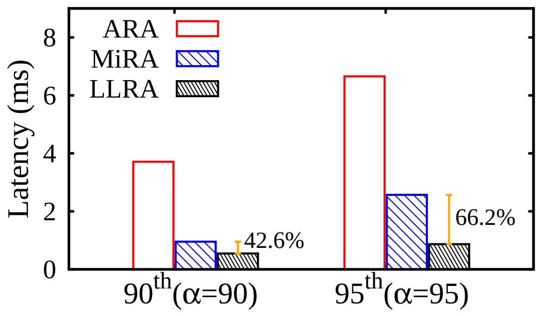
Interference We evaluate LLRA with rich interference using Channel 1 (20MHz) at the crowded 2.4GHz. It is observed that there are more than 10 APs from neighbors. Figure 5.8(b) shows the 90^{th} latency at P3 and P9. LLRA is still more effective in lowering latency. The reduction over ARA is 19 ms (67.7%) and 3 ms (50.6%), while the one over MiRA is 4 ms (32.3%) and 1 ms (21.8%). The most noticeable thing is that actual latency is much larger in this case, e.g., 28 ms for ARA, up from 1 ms in the non-interference case. The reason for much larger delay is that (1) interference brings longer queuing and contention time, and (2) RA algorithms tend to slow down



(a) Various Percentiles at P14



(b) Interference Case



(c) Mobility Case

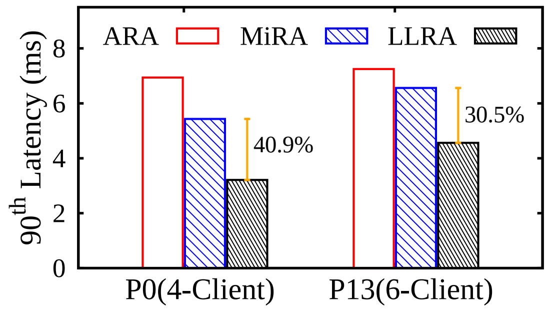
Figure 5.8: The tail latency varies with different scenarios in the single-client case.

due to interference-induced loss. Even so, LLRA performs robust against co-channel interference. It implies its effectiveness in competing scenarios.

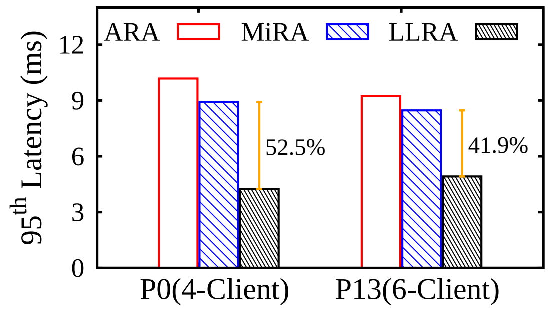
Mobility We move the client from P6 to P1 along the wall and then back to P6, at the constant pedestrian speed of 1 m/s. The client is loaded with 5 Mbps source. Figure 5.8(c) shows that LLRA outperforms ARA and MiRA by 85.3-90.8% and 42.6-66.2%, respectively. Table 5.4 lists the major rates selected by them for each pathway. It implies that LLRA is able to rapidly locate the LL rates during mobility.

5.3.2.2 Multi-client Case

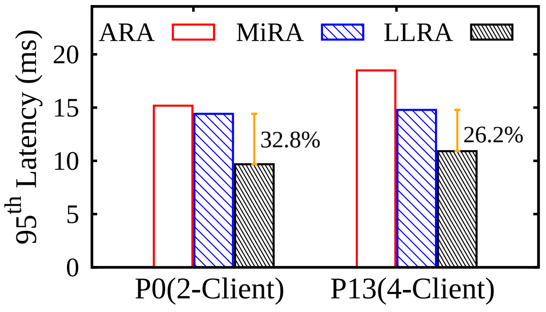
We further evaluate LLRA in the multi-client case. We consider two sub-cases: the observed client is placed in a far spot (P0) or a near spot (P13). Another five clients are deployed at P4, P6,



(a) $\alpha = 90$ at observed client



(b) $\alpha = 95$ at observed client



(c) $\alpha = 95$ with uplink traffic

Figure 5.9: Multi-client case. (a)(b): a downlink flow per client. (c): downlink flows plus one (P0) or two (P13) uplink flows.

P7, P9, and P10. Each client is loaded with a downlink traffic flow. We use up to 4 clients (P0) and 6 clients (P13). Figures 5.9(a) and 5.9(b) show the results for the 90^{th} and 95^{th} percentile latency. Even compared with MiRA (better than ARA), LLRA reduces latency by 30.5%–52.5% (2–5 ms). LLRA prefers to stay at the LL rate without incurring retransmissions at both 90^{th} and 95^{th} percentiles. However, ARA and MiRA stay at the HG rates and aggressively probe higher rates with high losses. Their high PERs (15.1-29.7%) result in one or two retransmissions, incurring larger delay than the single-client case.

In order to examine how retransmission dispatcher comes into play, we test LLRA with high loss in the multi-client case. We induce more possible collisions by adding one or two uplink flows. The uplink traffic also reduces the capacity for downlink flows; we then support 2-clients and 4-clients in the P0 and P13 cases. Figure 5.9(c) gives the 95^{th} percentile latency for the

RA	P6→P5	→ P3	→ P1	→ P3	→ P5	→ P6
ARA	243DS	216DS	162DS	108DS	162DS	162DS
MiRA	243DS	243TS	121.5TS	121.5TS	121.5TS	162DS
LLRA	243DS	162DS	108DS	121.5TS	108DS	162DS

Table 5.4: Chosen rates for each pathway during mobility.

observed client. LLRA continues to outperform them by saving 4–8 ms (26.2%-41.0%). It is mainly attributed to its retransmission dispatcher. Even though the rate controller chooses the rate with lower PER, the packet at the 95^{th} percentile for LLRA still experiences one retransmission via software reschedule. Retransmission dispatcher then helps to prioritizes retransmissions to have smaller queueing delay.

CHAPTER 6

HetRA: Heterogeneous Rate Adaptation

6.1 Problem

We now seek to design a solution that meets diverse requirements in an 802.11n WLAN setting. The proposed scheme needs to ensure minimum throughput on a per application flow basis, while improving the energy efficiency of the 802.11n NIC for those battery-constrained devices.

To highlight the main issues and solution ideas, we first focus on a simple usage scenario, but extend to more complex cases in Section 6.3.4. In the setting, an 802.11n AP serves multiple clients for downlink data flows. A flow denotes a stream of packets from a source to a given destination 802.11n client. Each flow f can specify its minimum throughput requirement B_f, e.g., if it is an FTP session for a large file download. In addition to meeting these hard requirements (i.e., the throughput constraints) for applications, we further seek to improve energy efficiency for the 802.11n network card (NIC). As shown by recent studies [HGS10, LPL12b], one way to save receive energy (defined in energy per bit) is through using fewer receive antennas and slowing down data transmissions in 802.11n devices.

In a nutshell, we seek to find an 802.11n solution, which improves energy efficiency at the device NiC while ensuring minimum throughput assurance on a per-flow basis at a receive client in an AP setting.

There are two main challenges. The first is to meet the per-flow requirements in terms of throughput, while simultaneously improving energy efficiency. It thus becomes a constrained optimization problem. Our goal is to find a practical solution to it. Note that the current 802.11n practical solutions (e.g., RA algorithms) are mostly single-goal driven design. However, our problem involves multiple goals. The devised solution has to support coexistence of these objectives that vary among flows and devices. These goals may lead to conflicting actions. For example, meeting the deadline requires the client to speed up, whereas saving energy suggests it to slow down and use fewer antennas. Second, flows and clients compete for the shared air-time. Since the channel conditions vary among clients and over time, the allocation of air-time has to be dynamic.

To address the above problem, a packet scheduling approach appears to suffice. Packet scheduling has been widely used to ensure throughput and delay guarantees in the Internet or wireless networks (e.g., [NLB99]). However, it is nontrivial to extend it to achieve energy efficiency at each receive client while ensuring minimum throughput bounds. We thus pursue another solution approach in this work. Our solution takes a novel wireless service curve approach, to be introduced in §6.2. The underlying instruments are RA and air-time scheduling. RA is the popular mechanism to adjust the per-packet transmission rate dependent on channel conditions. Together with the air-time scheduling, it is directly responsible for packet-level performance in terms of throughput. Early work [LPL12b, KDC13] also demonstrates it as an effective means for energy savings.

6.2 New Solution Approach via Wireless Service Curve

6.2.1 Adaptive Two-Tier Service

We propose adaptive service to meet the diverse requirements for flows and mobile devices. Each is served with a two-tier service model. The basic tier provides service assurance that honors the minimum throughput of a flow. The additional tier fairly allocates available, extra air-time to improve energy efficiency or other goals upon request.

Wireless Service Curve (WSC) We propose WSC as the novel approach to offering the above two-tier adaptive service. Service curve is a abstraction for the Internet QoS model. It is proposed by Cruz [Cru92, Cru95] as the building block to specify the minimum amount of service (both bandwidth and delay) for a given Internet flow. In this work, we adapt it to the wireless 802.11n domain and extend it to further manage energy efficiency. Our proposed WSC thus defines a generic service model, which takes into account the requirements of throughput, as well as energy efficiency for mobile devices. Note that we leave the cases which involve delay requirements to future work.

Specifically, a backlogged flow f is said to be ensured a service curve $S(t_1, t_2)$ if the amount of service received by flow f during the interval, denoted by $W(t_1, t_2)$, is no less than $S(t_1, t_2)$. That is, we have $W(t_1, t_2) \geq S(t_1, t_2)$. Linear service curve is the popular choice to specify the minimum service for a flow.

Figure 6.1 illustrates different forms of service curves. Broadly speaking, there are two categories of linear and nonlinear service curves. Linear service curve (shown in Figure 6.1(a)) is the popular choice to specify the minimum service for a flow in the Internet [Cru95]. Note that a linear curve is characterized by only one parameter, i.e., the slope. It thus ensures the minimum throughput for the flow. The delay requirement can also be met since it is inverse proportion to the bandwidth. However, there is a limitation with the linear service curve. Even though delay bounds can be provided by solutions to linear service curves, there is coupling between the guaranteed delay bound and bandwidth.

Realizing WSC through Rate Adaptation In our 802.11n context, the above linear service curve can be readily realized using two mechanisms, the airtime scheduler and rate adaptation algorithm for each client. The scheduler allocates the fraction of air time T_f to be used for data transmission to the client. It thus decides how much airtime is available for a flow. Assume stable channel condition for a while[1]. The RA selects the proper rate R and yields the goodput $G(R)$ when transmitting each packet. Therefore, the slope (i.e., the allocated bandwidth) of the linear service curve is given by $T_f \cdot G(R)$ within each time unit.

6.2.2 Piecewise Linear WSC

Based on the above WSC model, our key innovation is to propose piecewise linear service curve (PL-WSC) in our solution. Using PL-WSC, we can realize the two-tier adaptive service model. It

[1]We use the credit/debit mechanism to handle dynamics due to channel variations, mobility and time-varying sources. It will be elaborated in Section 6.3.

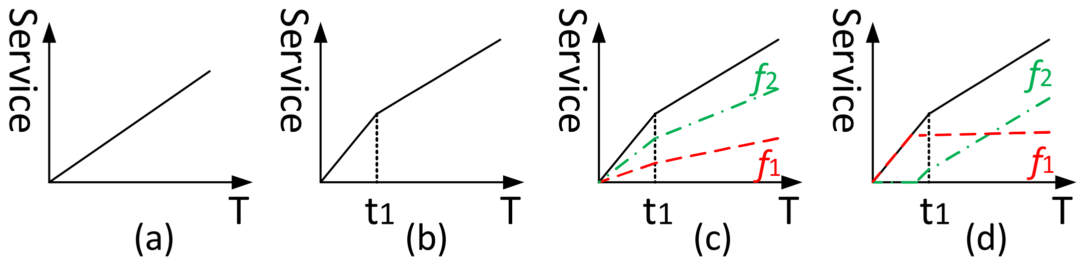


Figure 6.1: Illustration of service curves. (a) linear wireless service curve (L-WSC), (b) piecewise linear WSC (PL-WSC), (c) round-robin scheduling for two flows in PL-WSC, (d) two-flow priority scheduling in PL-WSC.

balances between the basic requirement (in terms of minimum throughput) at the flow level and the energy efficiency at the device level. It thus opens a new venue for service customization among diverse requirements. We further show that, we can achieve higher energy efficiency (30% gain) via PL-WSC compared with the recent EERA scheme [LPL12b]. Note that EERA effectively realizes the linear wireless service curve (L-WSC).

Broadly speaking, our PL-WSC belongs to the nonlinear service curve category. While in theory any non-decreasing functions can be chosen as service curves, in practice piecewise linear functions are used for simplicity. In general, a concave service curve will result in a lower average and worst-case delay for a flow than a linear or convex service curve with the same minimum throughput. Note that, however, it is impossible to have concave service curves for all flows and still reach high average utilization. Intuitively, this is easy to understand as priority is relative and it is impossible to give all flows high priority and result in low delay. It should be noted that, our PL-WSC is different from its counterpart in the Internet case. In the wired Internet, researchers have used it to decouple delay and bandwidth allocation, thus improving both resource utilization and management flexibility [SZN97]. In our case, we use it to tradeoff between throughput and energy savings.

We next use example cases to illustrate how our PL-WSC outperforms the linear curve. Using real experiments, we demonstrate how different linear curves (via single-rate RA) behave in this case and how PL-WSC (via multi-rate rate adaptation) functions and improves energy efficiency. For simplicity, we assume that the channel condition is stable and the maximum channel rate remains constant. We start with the single-flow, one-client case, and then extend it to the multi-flow, multi-client cases. We look at both flows with minimum throughput requirement. We seek to gain design insights from these illustrative examples.

6.2.2.1 Case I: Single Flow at One Client

Problem Scenario in Case I-A Alice watches high-definition video on her iPad. The video transmits 30 frames per second with each frame being 208KB, thus requiring 50Mbps. With stable channel conditions, given each rate setting R, we obtain its achieved goodput $G(R)$, its consumed active power $P_a(R)$ (during packet delivery), idle power $P_i(R)$ (without data transmission), as well as its packet error rate (PER) $PER(R)$.

Alice uses RA to minimize the energy consumption while accommodating the video streaming. It seeks a RA scheme, which maximizes energy savings while providing affordable goodput beyond the minimum requirement θ (e.g., the traffic source rate). The solution thus solves the following optimization problem:

$$RA^{*} = \arg\min_{\{RA\}} \mathbf{E}(RA), \quad \text{s.t. } G(RA) \geq \theta. \tag{6.1}$$

Under stable channel conditions, the resulting RA searches for an optimal rate (for the linear service curve) or multiple rates (for PL-WSC).

Experimental Results We conduct experiments to demonstrate how various rate settings perform in this example. We show that (1) the conventional rate for highest goodput is not most energy-efficient; and (2) the linear-curve rate targeted for energy efficiency is still not optimal compared with the PL-WSC scheme.

To this end, we run fixed-rate experiments in a controlled and static office environment (See Figure 6.4 for the testbed). Both the AP and clients are programmable 802.11n devices. They use Atheros AR9380 dual-band (2.4/5 GHz) chipset and supports three antennas. Each test has

Rate [2]	$G(R)$ (Mbps)	$P_a(R)$ (mW)	$P_i(R)$ (mW)	PER
...	...	...	...	...
3/108DS	75	985	765	3%
3/162DS (HG)	**90**[3]	1000	765	14%
...	...	...	...	
2/81.5SS	65	700	650	0%
2/108SS (EE)	**75**	**705**	**650**	4%
2/121.5SS	50	710	650	42%
...	...	...	...	...
1/40.5SS	35	577	541	0%
1/54SS ($\star$)	45	580	541	0%
1/81.5SS	36	586	541	45%
...	...	...	...	...

Table 6.1: Goodput, active power and idle power of some used rates in the case study.

10 two-minute runs. In each run, we exhaust all the possible rate settings and collect the achieved goodput, active power and idle power for each rate. We use power meter Agilent 34401A to measure the consumed power. Table 6.1 lists the results of chosen rate settings in this case study. For comparison and channel hints, we also list the results for the rates nearby.

We make two observations. First, the HG rate, which is the ultimate goal of most RA algorithms, performs poorly in terms of energy consumption. In this example, the HG rate is 3/162DS, which is definitely able to afford the carried traffic (90Mbps ¿ 50Mbps). However, its consumed energy is larger than the one by other rates (say, 2/108SS). Table 6.2 compares their energy consumption in one cycle.

$$E(R) = P_a(R) \cdot t_a(R) + P_i \cdot t_i(R), \tag{6.2}$$

where $t_a(R)$ and $t_i(R)$ represent the active and idle durations in one cycle (here, 1/30 second) when the rate R is used. Through an exhaustive search, we locate the rate of 2/108SS, which is eligible and consumes the minimal energy in this example. It uses 22.9 mJ, saving 23.2% energy

RA Scheme	**E**nergy-per-cycle (mJ per 1/30 second)	
HG: 3/162DS only	$1000 \cdot \frac{1}{30} \cdot \frac{50}{90} + 765 \cdot (\frac{1}{30} - \frac{1}{30} \cdot \frac{50}{90}) = 29.9$	149.0%
EE: 2/108SS only	$705 \cdot \frac{1}{30} \cdot \frac{50}{75} + 650 \cdot (\frac{1}{30} - \frac{1}{30} \cdot \frac{50}{75}) = 22.9$	113.9%
PL-WSC EE: 2/108SS for 5.6ms; 1/54SS for 27.8ms	$\boxed{\mathbf{705 \cdot 0.0056 + 580 \cdot 0.0278 = 20.1}}$	100%

Table 6.2: Comparison of three rate schemes in Case I-A.

compared with the HG rate. Clearly, the HG rate is not equivalent to the EE rate. This is consistent with recent findings in [LPL12b].

However, we demonstrate that there still exists room to improve energy efficiency compared with the "optimal" EE rate based on the linear curve. It sounds contradictory. But the point is that the conventional RA seeks one rate (assuming the channel is stable) to optimize the performance goal (say, EE in form of Equ.(6.2)); It intrinsically limits the performance bound since it lacks of flexibility to seek hybrid rates for further energy saving. We propose another rate scheme. We use 2/108SS for 5.6 ms (=1/150s) and then use 1/54SS for 27.8 ms (4/150s). This is definitely able to afford the video streaming. It can deliver at most $(75 \cdot 5.6 + 45 \cdot 27.8)/(1/30) = 50.1$ Mbps throughput. Consequently, it only consumes 20.1 mJ, which further reduces 12% energy consumption than the EE setting. Figure 6.2 plots the ideal rate choices and service curves by both schemes.

Insights This example demonstrates that the conventional RA that seeks one single optimal rate option is unable to make full use of its potential. Hybrid rates might perform better. This is because the optimal rates for different goals or requirements might be different (e.g., the HG $\neq$ the EE). In this example, 1/54SS outperforms 2/108SS in terms of active power and idle power. Clearly, it would be more energy efficient if it is used. However, purely using 1/54SS fails to meet the minimal goodput requirement. By using both rates, PL-WSC RA thus fits well to chase for energy efficiency while meeting the minimum goodput requirement.

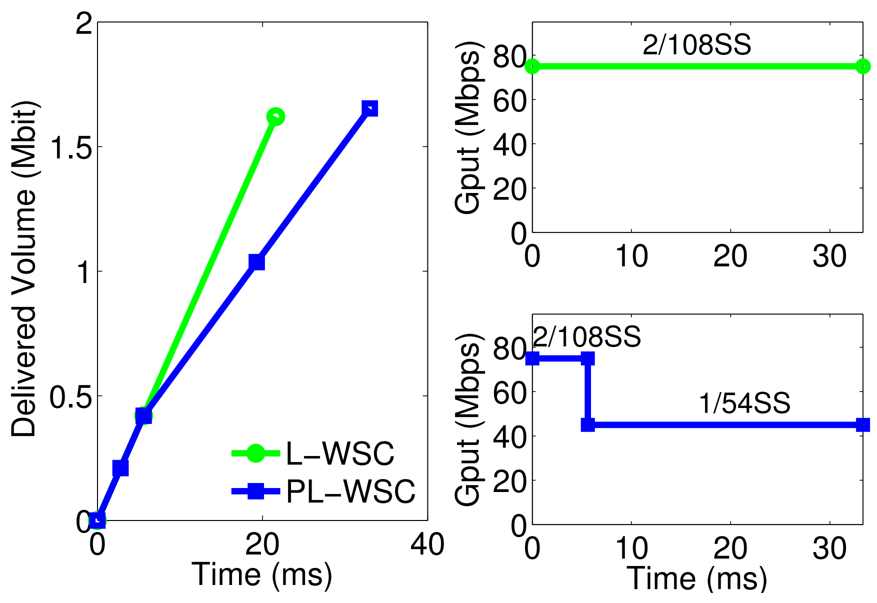


Figure 6.2: Rate choices and service curves in Case I. Left: service curves (delivered volume) via single-EE and hybrid-EE rate schemes; Right-top: the ideal rate choice in single-EE RA; Right-bottom: the ideal rate settings in two-rate hybrid-EE RA.

6.2.2.2 Case II: Multi-Flow at One/Multi-Client

We now extend Case I to the multi-flow case at single or multiple clients.

Case II-A: Multi-flow at one client We extend Case I-A into the two-flow case at one client, still with the minimum goodput constraints. Flow 1 is a video streaming, requiring 10 Mbps of bandwidth. Flow 2 is a FTP session, requiring 40 Mbps. The total data traffic is still 50 Mbps. The channel is the same as Case I. The goal is to minimize its energy consumption. In addition, we want to reduce the latency for Flow 1. By treating all the flows as one virtual flow, we can arrive at the same RA decision as Case I-A. That is, we sum up the minimum goodput requirements and obtain the lower bound of PL-WSC. Once the rates are determined, there are two scheduling schemes to allocate the resource among two flows. First, we run round-robin for packets belonging to these two flows (see Figure 6.1(c)). Both flows are served proportionally. That is, Flow 1 uses around 20% (10/50) of transmission time and the delivery for one video frame completes in 1/30 second. Second, Flow 1 is served with high priority since it is sensitive to latency (See Figure 6.1(d)).

We serve Flow 1 first (using 2/108SS) and then Flow 2 (using 2/108SS + 1/54SS). By this way, the video frame is delivered within 4.4 ms ($= 10 * 1/30/75$), while the energy efficiency remains invariant. As a result, multiple flows with common performance goals (here, energy efficiency) can be integrated into one "virtual" flow. In terms of diverse latency requirement, it can be handled by a priority scheduling independently.

Case II-B: Multi-flow at multiple clients The clients compete with each other to occupy wireless airtime. So in a time slot, the client will have $\beta\%$ ([0,100]) airtime, instead of occupying the channel all the time. The only change to PL-WSC is to obtain the effective goodput as $\beta \cdot G(R)$. Let us assume the PL-WSC's constraint at time t is $G(R)[t] > \theta[t]$ in single-client case. It turns into $G(R)[t] > \theta[t]/\beta$ in the multi-client case. Later, we will address how to allocate β among multiple clients §6.3.2.

The key design insight we have learned from the above examples is that, PL-WSC outperforms the linear service curve realized by conventional RA algorithms. Moreover, PL-WSC can be realized through multi-rate RA schemes. The WSC based approach offers a new solution alternative to meet diverse requirements in an 802.11n WLAN.

The above examples also shed lights on how to devise such a PL-WSC based RA algorithm. Initially, a rate with higher goodput is preferred due to the constraints on the deadline and goodput. The queue is typically longer when the traffic burst arrives. As time proceeds, the initial rate, usually larger than the lower bound of PL-WSC, will gradually drain the queue. It thus implies that the demand on the delivered goodput abates. Therefore, the remaining PL-WSC has a smaller slope. As a consequence, a more energy-efficient rate setting (here, 1/54SS) becomes more likely to be eligible and chosen by the RA algorithm. Moreover, we can prove that this scheme is almost optimal. Nevertheless, devising such a RA algorithm remains quite challenging. It must address many practical issues, such as time-varying channel, scheduling among flows, conflicts among clients, and traffic dynamics. We address them in the next section.

6.3 PL-WSC Based Rate Adaptation

Our goal is to meet minimum requirements (in terms of throughput) for flows, while improving energy efficiency. Our solution has the following three components:

- HetRA based on PL-WSC: This novel RA algorithm realizes PL-WSC for a client. It improves energy efficiency while meeting the per-flow requirement of minimum throughput.
- Air-time scheduler: It schedules air-time among clients. For each client, it allocates air-time sufficient to ensure minimum assurance in throughput. When extra air-time is available, it is fairly distributed among those battery-constrained clients.
- The credit/debit mechanism: It determines which flows are leading or lagging their reference service, and by how much. This mechanism further compensates lagging flows at the expense of leading flows, thus addressing dynamics due to channel variation, traffic source fluctuation, and mobility.

The overall solution works as follows. Each client obtains a fraction of air-time for all its application flows. It further activates HetRA, a PL-WSC based RA to select piecewise linear rates. It thus ensures minimum throughput bounds,while saving energy. The air-time is allocated to each client via the air-time scheduler. It first sets the fraction based on the minimum throughput bounds. Extra air-time beyond the minimum throughput bounds is fairly distributed among clients. Debit is logged for those who use the extra air-time that another client cannot use due to the lack of packet arrivals. When channel changes and the assigned air-time is not sufficient to meet the minimum throughput bounds, more airtime is allocated for the flow. The leading flows with debits will yield their extra air-time to the such a flow. This way, the leading flows compensate such flows using the extra air-time debit beyond their minimum assurance.

6.3.1 HetRA: 802.11n RA Based on PL-WSC

We now describe our new RA algorithm, HetRA, which uses PL-WSC to improve energy savings while ensuring minimum throughput. HetRA works with each client and handles all flows at the client. It needs to address four issues: (1) How to meet the basic requirement in throughput and delay? (2) How to save energy subject to the delay and throughput bounds? (3) How to arbitrate between different flows on the same client? (4) How to address practical issues in RA, including probing for highest-goodput (HG) rate, most energy-efficient rate, and estimate of traffic source?

Main Idea We first illustrate the main idea, as well as the analytical results using a simple one-flow case on a single client. The flow has a minimum throughput requirement. To stay focused, we derive the best rate settings in case of two rate phrases for time-invariant channels. Given the traffic source S and traffic interval T, we need to satisfy two constraints: (1) the achieved goodput in period T is larger than one threshold θ, and/or (2) the total transmission time is smaller than certain deadline D. The performance goal is to minimize the total energy consumption, including the one during the active and idle time. $\theta \leq S$ because the total carried traffic is constrained by the traffic source. By default, $\theta = S$. It means that the traffic load is just affordable in each interval. Otherwise, given a fluid flow, the queue overflow will eventually happen and it is not stable. When $\theta < S$, it implies that the application flow can tolerate certain packet loss. Therefore, the problem with minimum goodput constraint turns into

$$\{R_1^*, R_2^*, t_1^*\} = \arg \min_{\{R_1, R_2, t_1\}} \mathbf{E}(R_1, R_2, t_1)$$
$$\text{subject to} \begin{cases} \frac{G(R1)t_1 + G(R2)t_2}{T} \geq \theta \\ t_1 + t_2 \leq D \leq T, \quad t_1, t_2 \geq 0 \end{cases} \tag{6.3}$$

Theorem 7. *Given identical idle power at different rates. The optimal choice $\{R_1^*, R_2^*, t_1^*\}$ to Equation (6.3) belongs to three categories. First, no feasible solution exists when $\max_R(G(R)) < \theta \cdot T/D$. This implies the highest goodput-rate of the client under the given channel condition fails to meet the minimum QoS in throughput and delay. Optimal solutions exist for two other cases:*

$$R_2^* = \arg\min \frac{P_a(R) - P_i(R)}{G(R)}. \tag{6.4}$$

Case (I): The two piecewise linear scheme reverts to the linear highest-goodput rate. Specifically, when $G(R_2^*) \geq \theta \cdot T/D$, we have

$$R_1^* = R_2^*, t_1^* = 0, t_2^* = \theta \cdot T/G(R_2^*). \tag{6.5}$$

Case (II): The two piecewise linear rates exist. Specifically, when $G(R_2^*) < \theta \cdot T/D$, the two rates and the transition point can be obtained via

$$R_1^* = \arg \min_{G(R) \geq \theta \cdot T/D} \frac{P_a(R) - P_i(R_2^*)}{G(R) - G(R_2^*)}, \tag{6.6}$$

$$t_1^* = \frac{\theta \cdot T - G(R_2^*) \cdot D}{G(R_1^*) - G(R_2^*)} > 0, \tag{6.7}$$

$$t_2^* = \frac{G(R_1^*) \cdot D - \theta \cdot T}{G(R_1^*) - G(R_2^*)} > 0. \tag{6.8}$$

Corollary 1. *The solution in Theorem 7 always performs no worse than the optimal linear service curve-based solution (via a single rate).*

Intuitively, R_2^* tends to optimize the EE goal, while R_1^* is used to satisfy the goodput or deadline requirement when R_2^* is unable to afford it. When a flow specifies a deadline, we convert this deadline constraint into a minimum throughput requirement.

Basic Algorithm The above analytical results set the basis for the algorithm design of HetRA. The algorithm for one client is described at Algorithm 1. Let β be the assigned airtime portion for the client. Our algorithm performs every δ_t. To handle traffic source dynamics, we set the goodput threshold $\theta[t]$ dynamically based on the queue change. Let $Q[t]$ be the queue size at time t.

$$\theta[0] = \sum_i \theta_i[0], \theta_i[0] = \theta_i \tag{6.9}$$

where θ_i is the goodput requirement for flow i.If θ_i is not given, it can be the estimated traffic source rate through $\theta_i = \hat{S}_i \approx \Delta_{Qi}/\Delta_T$, where Δ_{Qi} and Δ_T are the arrival of new data and the used interval for flow i. It can be smoothed via a moving average.

We now start our loop every δ_t. We first locate the second phrase rate $R_2^* = \arg\min \frac{P_a(R)-P_i(SS)}{G(R)}$, based on the rate probing results. Here, we uses $P_i(SS)$ since it is common to all the rates. If $G(R_2^*) > \theta[t]$, it implies that R_2^* satisfies the requirement. Otherwise, we choose another rate $R = \arg\min_{\beta G(R) \geq \theta[t]} \frac{P_a(R)-P_i(R_2^*)}{G(R)-G(R_2^*)}$. Here, $\theta[t]$ is a generic expression of the throughput constraint. We use Rule*, first-come-first-serve, to decide which packets should be sent out from the

Algorithm 1 HetRAat Single-Client

```
1: init: $\beta =$ time portion allocated to the client, $\theta[0]$ via Equ. (6.9),
2: while Q[t] is not empty do
3:     RateProbe(), update the rate table        ▷ channel probing module
4:     Update $R_2^* = \arg\min \frac{P_a(R)-P_i(SS)}{G(R)}$ if channel change
5:     if $\beta G(R_2^*) \geq \theta[t]$ then
6:         $R = R_2^*$
7:     else
8:         $R = \arg\min_{\beta G(R) \geq \theta[t]} \frac{P_a(R)-P_i(R_2^*)}{G(R)-G(R_2^*)}$
9:     end if
10:    select packets via Rule* and send at rate R for $\delta_t$ unless $Q[t] = 0$;
11:    update the rate table based on the receiving status
12:    update $\theta[t+\delta_t]$ via Equ. (6.12)
13:    $t = t + \delta_t$
14: end while
```

queue.:We then update $\theta[t+\delta_t]$ based on the arrival of new packets and the delivery of old packets. Let $\Delta_{Qi}[t]$ be the number of new arrival packets for flow i. The increase $\Delta\theta[t+\delta_t]$ is incurred by the requirements of new flows. Similar to Equ (6.9), it counts for all the goodput (θ_i') requirement for new packets. The decrease part due to using the rate R can be calculated by $\triangledown\theta[t+\delta_t]$. It is based on a simple equation $\frac{x}{G(R)} + \frac{y}{\theta[t]-\triangledown\theta[t+\delta_t]} = \frac{x+y}{\theta[t]}$. We use $G(R)$ for x packets and $\theta[t] - \triangledown\theta[t+\delta_t]$ for the remaining y packets, which should be equivalent to sending $(x+y)$ packets with goodput $\theta[t]$. Clearly, x represents the number of packets sent $\triangledown Q[t+\delta_t]$ in $(t, t+\delta_t)$ and $x+y = Q[t]$.

$$\Delta\theta[t+\delta_t] = \sum x_i, \quad x_i = \theta_i \tag{6.10}$$

$$\triangledown\theta[t+\delta_t] = \frac{(G(R)-\theta[t])\cdot\triangledown Q[t+\delta_t]}{Q[t]G(R)-\theta[t]\cdot\triangledown Q[t+\delta_t]} \cdot \theta[t], \tag{6.11}$$

$$\theta[t+\delta_t] = \theta[t] + \Delta\theta[t+\delta_t] - \triangledown\theta[t+\delta_t]. \tag{6.12}$$

Note that, R_2^* and R are updated based on the rate table (derived from the rate probing) and $\theta[t]$. In case of no significant change, they will remain invariant.

Supporting Components HetRA also needs a few supporting components: (i) rate probing for a client during each interval and (ii) distributing airtime among flows at the same client. Rate probing is a mandatory procedure to learn the time-varying channel. It has been well studied in the literature. Our design to this component borrows the solution in the existing RA algorithms [PHW10, LPL12b]. The idea is to probe each rate in different MIMO modes (number of streams and antennas varying). The search space is pruned by some optimization techniques (e.g., stop probing higher rates once the PER is too high, or stop probing lower-rate once the PER is very small.). For (ii), the above algorithm briefly describes Rule*. We are also open to other options, for example, a simple weighted round robin [SV95] among flows. The flow weight is set in proportion to the converted minimum goodput requirement. The scheduling is an independent topic.

6.3.2 Air-time Scheduling

To make HetRA work with multiple clients, the AP has to activate air-time scheduling. Without prudent coordination, some clients might get hurt.

Figure 6.3 compares the packet delivery ratio in multi-client scenarios using various RAs. Packet delivery ratio measures how many packets have been delivered (including retransmitted ones) per client. In both scenarios, four clients carry the same traffic load. In the left plot, all four chase for HG, whereas in the right plot, two chase for HG and two for EE. We see that the total affordable traffic load decreases as the RA for EE is adopted. It aims to support four clients with 25 Mbps traffic each. However, all the clients get penalized (20%-35% loss) when two clients adopt the EE settings. The good news is that, abundant airtime is typically available in practical 802.11n networks. We measure the free airtime of wireless channels in three typical usage settings: office, cafe (public), home (with 10+ WLAN nearby) as [RPM09] does. The free airtime is affected by the transmission activity in its own and neighboring WLANs, as well as nearby interference. It is computed by using low-level information from the 802.11n NIC [RPM09], which monitors the energy level on a specific channel to determine whether it is idle or busy. Due to space limit, we briefly summarize our findings without plotting the details. We make two observations. First, free airtime varies at locations. Wireless channel is much busier in the office than at home/cafe because the number of clients is typically higher. Nevertheless, we still observe 80% free airtime

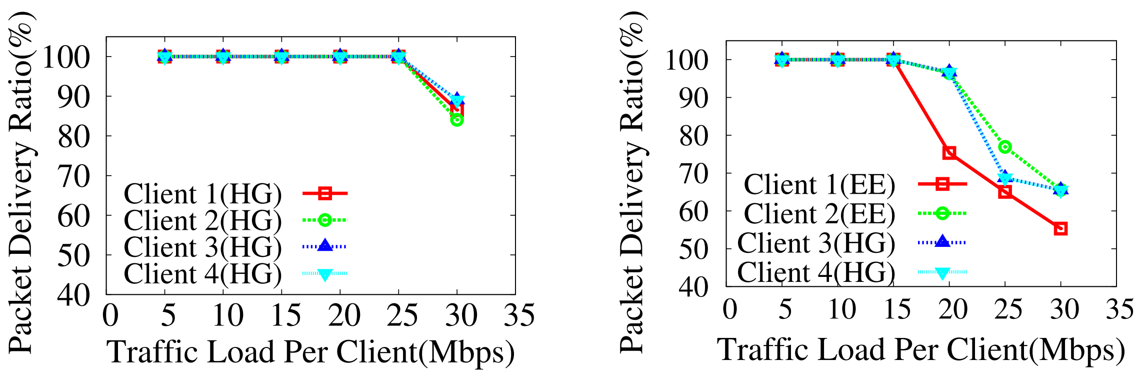


Figure 6.3: Packet delivery ratio in case of all clients for HG (left) and for HG+EE (right).

in 99% samples over the 5GHz channel in the office. Second, free airtime is relatively stable over the long term, but dramatically fluctuates in small time granularity. During busy hours, the ratio of free airtime decreases from almost 100% to 80-90% (average case) and 30-50% (worst case). The difference between two adjacent samples reaches up to 40-60% in three tested settings. Such findings motivate the following air-time scheduler and credit/debit mechanisms.

The air-time scheduling module works at two levels. It first allocates airtime among different clients. When a client has multiple ongoing flows, it further distributes the per-client air-time among flows.

At the higher level, the AP periodically allocates airtime for each client over a chosen time interval T_{AP}, called an epoch. The airtime is first assigned to each client to meet the minimum throughput bounds. The aggregate sum of the required minimum throughput of each flow at the client is computed. Given the highest-goodput (HG) rate at the client, we can determine whether it is feasible for the AP to serve the client by meeting the minimum throughput requirements of all its flows. Given the aggregate minimum throughput for a client, the AP computes the assigned airtime T_k for client k. After this step, if extra airtime $T_{AP} - \sum_{k=1}^{n} T_k$ is still available, the AP distributes such extra portion fairly among clients that are able to consume. Various fairness notions, e.g., max-min fairness, weighted fairness, and proportional fairness, can be adopted. Standard procedures to ensure such fairness are available in the literature [Hah91]. The total airtime available to each client includes the basic portion (that ensures the minimum throughput for each of its flows) and the extra portion (that comes from the fair share of extra free airtime).

At runtime, the AP keeps track of the usage of allocated airtime for each client. To this end, a virtual timer and a status monitor are kept for each client. The virtual timer records how much airtime has been consumed by each client in the current epoch. The status monitor interacts with the HetRA algorithm for each client. It oversees whether the allocated airtime is sufficient to meet the client requirement. Note that if the channel condition degrades and results in the reduced HG rate, more airtime is needed to deliver the packets to the receive client. In this case, the AP uses the credit/debit mechanism to handle it.

6.3.3 Credit/Debit Mechanism

Our design uses a credit/debit model in air time scheduler to handle various forms of dynamics. Such dynamics include deteriorated channel condition over time, user mobility, and dynamic packet arrivals. Such factors have two impacts on our design. First, when the channel condition worsens, the allocated airtime may not be sufficient since the client highest-goodput rate decreases. We thus need to allocate more airtime to still possibly achieve the minimum throughput requirements. Second, due to dynamic traffic arrivals, some portion of the allocated airtime over each epoch may remain unused. In the former case, we need to allocate extra airtime to such clients. In the latter case, we can let others use it first to improve channel utilization. Both are implemented via the credit/debit model.

The credit/debit model is defined for the extra airtime used by a client. The extra airtime is the portion of free air-time beyond that used to ensure minimum throughput requirements. The model keeps three counters for credit, debit, and overall usage. The credit counter logs the amount of airtime that is allocated yet unused by the client. This unused airtime can be consumed by other flows at other clients, e.g., a greedy TCP flow without minimum throughput requirement. A client with a nonzero credit is lagging behind its allocated extra airtime. The debit counter records the amount of airtime that is used by this client but pre-allocated to another client. A client with a nonzero debit is leading ahead of its allocated extra airtime. The usage counter keeps track of the total extra airtime used by a client over a large time window (say, tens or hundreds of epoches). It can be reset once the time window is reached.

The above model is used for two scenarios. In the first setting, when more airtime is needed to the client with worse channel condition, the client with a nonzero debit is the first to relinquish its extra airtime during the current epoch. Its debit counter is updated accordingly. If this is still insufficient after collecting the extra airtime from all leading clients (with nonzero debits), the flow with largest usage counter gives up its extra airtime. This way, we use the extra airtime to compensate the client which needs more to meet its minimum throughput requirements. The credit/debit/usage counters are updated over each epoch. In the second scenario, no client needs more airtime to ensure its minimum throughput bounds. A client with nonzero debit yields to a client with nonzero credit. This effectively allows for the leading clients to compensate the lagging clients in the extra free airtime. Long-term fair share of the extra free airtime is still preserved among clients.

6.3.4 Other Issues

We now discuss several important and related issues.

Evolving Requirements Our design can be readily extended to clients with time-varying goals (e.g., minimum throughput at the start, and solely energy saving as its battery power drains). When goals switch, we do not need to start HetRA for the client from scratch, but leverage the recent information before the switch.

Specifying Minimum Requirement A key issue is on how to specify the required minimum throughput for flows. There are two approaches. One is to let each client notify the AP on its per-flow requirement. However, this cannot prevent cheating behaviors by certain clients. An alternative way is to let the AP profile the most popular applications. The AP decides on such minimum throughput bounds on a per-application basis. This application QoS profile can be gradually built over time.

More Piecewise Linear Segments in PL-WSC Our HetRA effectively explores two piecewise linear curves in its design. It can be easily extended to more piecewise linear segments. We set the choice of linear segments to two because we seek to limit the rate-switching frequency and increase the time spent at each rate. This also offers more room for more accurate rate probing in the RA algorithm. We configure the number of piecewise linear segments as a design parameter for further improvement and flexibility.

Worst-case Scenario The work so far has not addressed certain worst-case scenarios. In one setting, the channel condition of a client becomes so bad that no feasible solution exists for its flows. In this case, we simply revert some or all flows on this client as the best-effort ones without minimum guarantees. This option shields the rest of clients and flows from being affected by this unfortunate client.

Uplink Flows For simplicity of presentation, we consider downlink flows only. For uplink flows, the design of HetRA still applies but at the sender client, not at the AP. However, the air-time scheduler at the AP needs to be extended. Each uplink flow needs to notify the AP on its minimum throughput requirement, so that the AP can still schedule air-time in the centralized manner for both uplink and downlink flows over each epoch.

Coexistence with 802.11a/b/g Clients Different from the MIMO-based 802.11n/ac, the legacy clients do not need to design a new RA. In most cases, the HG rate is also the EE rate in most cases. Overall solution still works.

6.4 Implementation and Evaluation

We implement the PL-WSC design in the ath9k driver, which is for all Atheros 802.11n Wi-Fi chipsets. We then present the performance evaluation in the real testbed, deployed in the office environment (see Figure 6.4).

6.4.1 Implementation

We implement HetRA, air-time scheduler and the credit/debit mechanism. HetRA is in the rate control module, whereas the last two items are realized in the transmission module. We tackle three issues at the AP and one issue at the client. First, we enable HetRA to retrieve the queue status information from the transmission module. It is updated every rate query. Second, the air-time scheduler takes control of queue management and determines which flow has higher priority to be served. Its scheduling is triggered once any new packet comes or any transmission has been done. After each transmission, the counters of credit, debit, overall usage are updated. Third, each flow's packets is tagged with the deadline and then scheduled based on its requirement. We also slightly revise the client in order to work with HetRA. We use an 802.11n action frame to notify the client to switch the number of antennas upon its request. The AP will be notified of this successful switch from the the action frame's ACK.

6.4.2 Evaluation

We evaluate our system in both single-client and multi-client scenarios. In the single-client case, we mainly focus on assessing HetRA by comparing it with other two 802.11n RA algorithms: MiRA [PHW10] and EERA [LPL12b]. They are proposed to pursue high-goodput and energy efficiency, respectively. In the multi-client case, we evaluate a whole system, covering the air-time scheduler, the credit/debit mechanism, as well as HetRA. In the experiments, we use the following default settings unless explicitly specified. Both the AP and the clients support up to three antennas (TS mode), and use 40 MHz channel width over 5GHz band.

In summary, our HetRA can not only satisfy the requirements of minimum throughput and maximum deadline, but also has energy improvement up to 35.4% and 20.1%, compared with MiRA and EERA, respectively. In the multi-client case, our results show that the clients' minimum requirements can be ensured while the others require extra air-time. In case of no sufficient airtime which is required by all the clients, the extra airtime is fairly shared among these clients under 0.5% difference. We further show that the credit/debit mechanism is able to assure the client's minimum requirements in the mobility case.

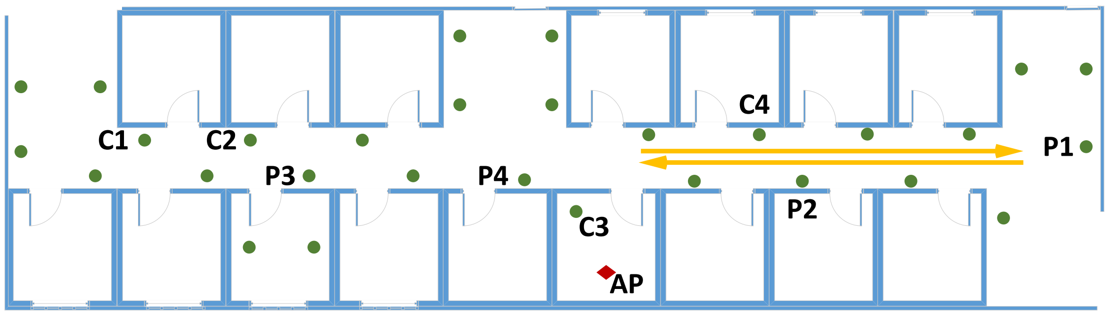
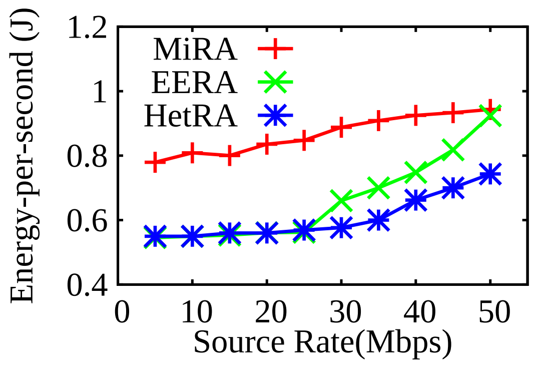


Figure 6.4: Testbed. Red diamond marks the AP location and green circles mark the client locations.

6.4.2.1 Single-client Case

We evaluate the flow with minimum throughput requirement in both single-flow and multi-flow cases. The multi-flow case has similar results as the single-flow case, since multiple flows are treated as a virtual flow. Thus, we present the results in the single-flow case.

Figure 6.5 shows the per-second energy consumption by different RA algorithms under various minimum throughput requirement and at different locations. HetRA consistently outperforms MiRA and EERA with energy saving up to 35.4% and 20.1%, respectively. The left plot in Figure 6.5 shows the energy consumption varies with different source rates at P3. The minimum goodput requirement is set as the traffic source rate. HetRA performs similarly as EERA when the traffic source is low (here < 25 Mbps) and then outperforms EERA when the source rate increases from 30 Mbps to 50 Mbps. It is easy to infer from the HetRA design. When the goodput requirement is low, HetRA reverts to EERA. As the goodput requirement increases, EERA reverts to the HG rate setting to afford the traffic whereas HetRA seeks each possibility for energy efficiency. Despite its energy consumption also increases, it is still much more efficient than other schemes. This energy saving is also observed at different locations, as shown in Figure 6.5 (right). We test four locations with the following minimum throughput requirement, P1(50 Mbps), P2(100 Mbps), P3 (50 Mbps), and P4 (150 Mbps). It is observed that the HetRA outperforms MiRA and EERA with energy saving, 22.6-26.4% and 15.1-20.0% respectively.



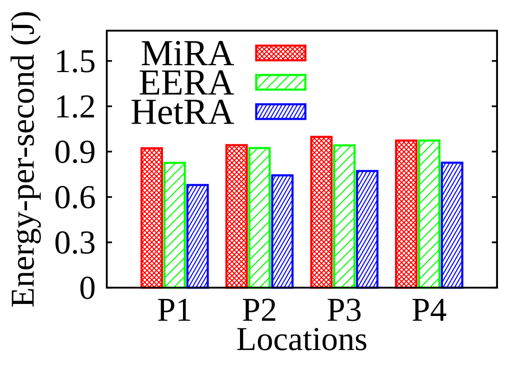
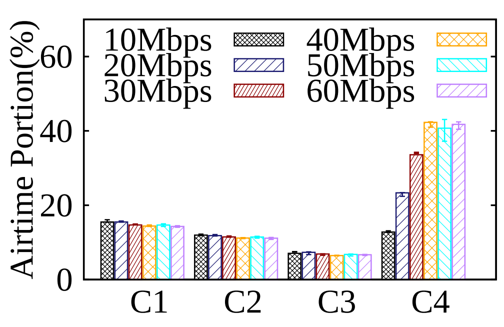


Figure 6.5: The energy consumption varies with source rates (Left) and locations (Right).



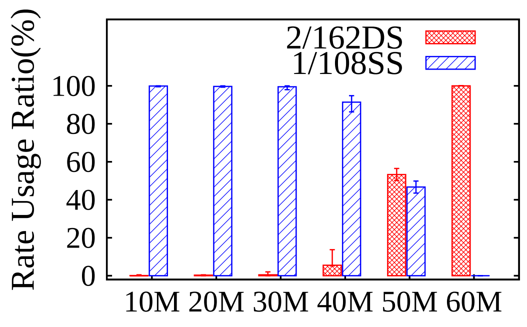


Figure 6.6: The airtime usage of 4 clients (Left) varies with different C4's source rates, and the rate usage portion is used by C4 (Right).

6.4.2.2 Multi-client Case

We first evaluate the air-time scheduler by examining minimal requirement assurance and the fairness on extra air-time share in different scenarios. We further access the performance of credit/debit mechanism by considering the mobility case. In the experiments, we deploy four clients to associate with the AP. They are noted by C1-C4 as shown in Figure 6.4. The default setting is that each clients has a 10 Mbps UDP downlink traffic, and they do not require extra air-time unless explicitly specified.

Minimal Requirement Assurance We show that the air-time scheduler is able to assure the minimum throughput requirements while the extra air-time is allocated. We vary the traffic source

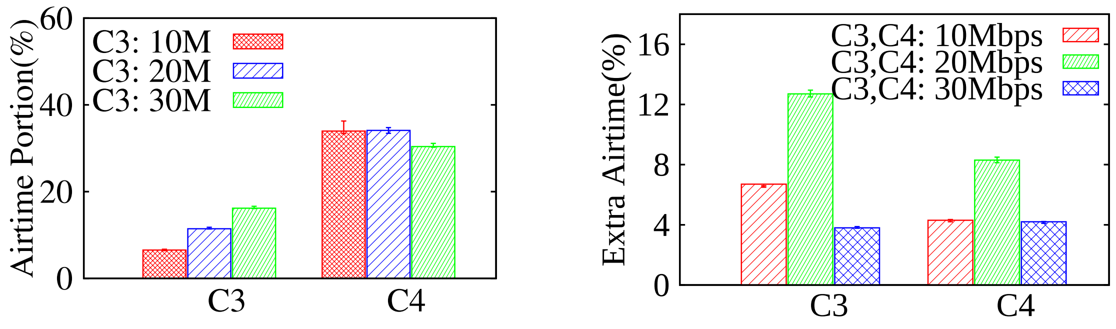


Figure 6.7: Left: the minimum requirement is assured at C3. Right: the extra airtime fairness among C3 and C4.

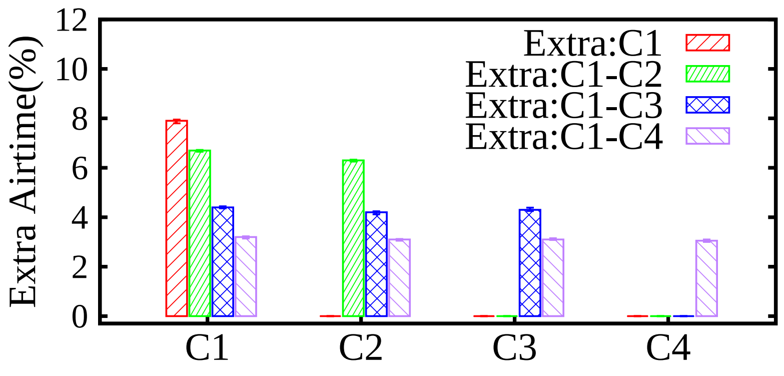


Figure 6.8: The extra air-time assignment of 4 clients varies with different numbers of clients who require extra air-time.

rate of C4, which requires extra air-time for energy efficiency. The left plot of Figure 6.6 shows that the minimum requirements of C1, C2, and C3 are assured and not affected, while the extra air-time allocated to C4 increases. After the source rate reaches 40 Mbps, the available airtime becomes insufficient for C4's extra air-time demand.At the 60 Mbps source rate, C4 gets assigned only 3.8% out of total for the extra on top of its basic airtime assignment. The right plot of Figure 6.6 shows the transition point of PL-WSC at C4 changes with the amount of extra air-time. The transition point of PL-WSC moves away from the more energy-efficient setting, 1/108SS, when the amount of extra air-time decreases. The usage percentage of this setting decreases from 91.4%, to 46.7%

and 0.1% while the extra air-time decreases from 10.0% out of total, to 6.2% and 1.8%.

We further show that the minimum throughput requirement at other clients are still assured while it increases. As shown in Figure 6.7 (Left), the increasing throughput requirement of C3 can be assured while C4 requires extra air-airtime. It is because the extra air-time is allocated to C4 after considering the minimum throughput requirements. When the source rate of C3 increases from 20 Mbps to 30 Mbps, the amount of extra air-time allocated to C4 changes from 4.9% out of total to 1.8%. The reason is that the C3's throughput requirement reduces the total available extra air-time.

Fairness on Extra Airtime Share We show that the air-time scheduler achieves fairness on extra airtime share. We vary the source rates of C3 and C4, both of which require extra air-time. Figure 6.7 (right) shows that they get the fair share of extra air-time for different requirements. With 10 Mbps and 20 Mbps source rates, they can get their required air-time as much as possible. It is because there is still abundant air-time. However, at the 30 Mbps source rate, the airtime shares allocated to them are equal since the available air-time is not sufficient. The air-time shares range from 3.8% to 4.2% and their gap is observed below 0.5%.

We further consider different number of the clients who require extra air-time, as shown in Figure 6.8. In this experiment, C1 is loaded with 10 Mbps source rate, whereas C2, C3, and C4 are loaded with 20 Mbps. It is observed that when more than one client require extra air-time, they fairly share the available amount. When only C1 is able to use more air-time, it gets assigned 7.9% out of total for the extra. When both C1 and C2 require extra air-time, each gets a portion between 6.4-6.8%. Each portion of fair share becomes 3.9-4.3% when C1, C2 and C3 require the extra. It reduces to 2.7-3.1% when all the four clients require it. The gap between different portions is observed below 0.5%.

Mobility Case We also test with the mobility case. We seek to examine whether our HetRA, airtime-scheduler and credit-debit mechanism work well to satisfy the minimum requirement in case of mobility. We consider one mobile client C5 and one static client C4, which requires extra airtime. They are loaded with the source rates 20 Mbps and 30 Mbps, respectively. The mobility

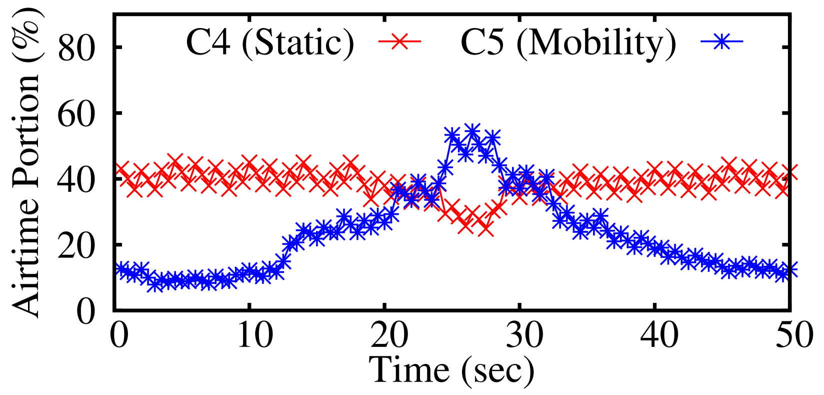


Figure 6.9: The airtime portions of one static client and one mobile client varies with time.

path is shown by the arrows in Figure 6.4. C5 first moves far away from the AP and then moves back to the room where the AP is. Figure 6.9 shows that the used airtime by C5 and C4 varies with time. Initially, C4 can get the extra air-time as much as possible. When C5 moves away from the AP (0-26s), its required airtime increases since the channel condition is dynamic and the available rate setting used by C5 becomes slower. Between 24-29s, the available airtime becomes insufficient so that the C4's extra airtime is released to satisfy C5's minimum requirement. After the channel of C5 becomes better, C4 can then get assigned more extra air-time.

CHAPTER 7

Conclusion and Future Work

We present the summary of our work, as well as share the insights and lessons we have learned. We finally point out the future work.

7.1 Summary of Results

A key direction for the wireless networking innovation has been the customizable Wi-Fi technology. With the proliferation of various WiFi devices and multimedia applications in ever-increasing usage scenarios, the operation requirements are becoming increasingly diversified. In addition to the conventional quest for high throughput, there exist other important ones, such as energy efficiency and perceived delay. The former is mainly driven by the reality that the battery life is a main bottleneck for mobile device operations. The latter stems from most multimedia applications, such as VoIP and gaming, which require timely data delivery. To satisfy such diverse performance goals, we re-instrument the popular mechanism of rate adaptation (RA), which does not require any hardware modification or firmware upgrade on the commodity Wi-Fi devices. Moreover, it can be utilized to chase for high throughput, energy efficiency, and latency.

Along the path toward the customizable Wi-Fi, we have obtained three main results in our three concrete design efforts.

Energy-saving for 802.11n Wi-Fi NIC In this work, we have shown that the high-goodput designs [WGB08, WYL06, PHW10, HHS10, VBJ09, ASB10, CQY07, GK11] may not be energy efficient for Wi-Fi NICs, since the high-goodput is possibly achieved at higher energy cost. Energy efficiency is deemed to be equally important as high speed, so the Wi-Fi technology has to balance between energy and speed.

We thus devise EERA to improve NIC energy efficiency by adapting RA. The core idea is to trade off goodput for energy saving while satisfying the traffic source from the applications. However, there exists a scaling issue when locating the most energy-efficient setting in the current 802.11n/ac Wi-Fi. The 802.11n/ac Wi-Fi speed is boosted by some new features (e.g., MIMO), so RA has to adapt over multi-dimensional parameter space. In the EERA design, we apply two major techniques to accelerating the search for energy-efficient rate setting. It adopts ternary search over each setting branch, and also utilizes simultaneous pruning mechanisms to narrow the search space across different branches. In the multi-client case, the slowdown of the EERA clients are constrained to prevent the others' performance from being hurt. The major idea is to do isolation among multiple clients via fair share of airtime. The EERA clients slow down for energy-saving by equally sharing the spare airtime. Our early experience shows that this software-only design readily yields 24% to 34% gains in energy savings over existing approaches.

Low-latency for 802.11n Home Wi-Fi Network Supporting latency-sensitive applications is important for 802.11n/ac home networks. As 802.11n/ac is increasing its speed, the common-sense perception "*higher speed suffices to ensure low latency for such applications*" is plain wrong. The fundamental problem lies in the long-tailed latency distribution at the link layer. In this work, we have shown that current RA for 802.11n devices works well for throughput, but it cannot meet the millisecond latency demand.

We then propose LLRA to achieve millisecond-level latency in a typical, multi-client home 802.11n network. The LLRA design consists of three components: rate control, frame aggregation scheduling, and retransmission dispatching. The key idea of the rate control is to locate a LL rate setting, which balances between initial queue delay and service time, thereby achieving the lowest latency at the given α percentile. It dynamically balances between various settings ranging from *"higher goodput yet higher loss"* to *"slightly lower goodput but much lower loss"*. The two mechanisms of aggressive aggregation scheduling and prioritized retransmission dispatching are further used to reduce the initial queue delay and service time, respectively. Our early experience shows that the LLRA design reduces tail latency by 21.8% to 66.2% over existing approaches.

Heterogeneous Goals for 802.11n Wi-Fi The requirement diversity for mobile applications and mobile devices is typically characterized by the concurrent goals of minimum throughput and energy efficiency, etc.. With the explosive growth of mobile applications and accelerating shipments of devices with the built-in 802.11n interfaces, the requirement diversity is likely to become the norm rather than exception. However, the deployed 802.11n solutions cannot do it, since they always pursue a single goal.

We further propose HetRA to pursue concurrent goals for 802.11n Wi-Fi. The key innovation of HetRA is the novel concept of wireless service curve (WSC). WSC offers a conceptual framework to specify the diverse requirements by mobile applications and devices. As we have demonstrated in this study, nonlinear WSC opens new space for performance improvement, thus outperforming its linear counterpart. Furthermore, the credit/debit mechanism enables us to address network dynamics including dynamic traffic source, time-varying channel conditions, and mobility. Our results validate that HetRA can meet the throughput requirements by application flows while simultaneously saving energy. It outperforms existing algorithms MiRA and EERA by up to 35.4% and 20.1%, respectively.

7.2 Insights and Lessons

We here present the insights and lessons learned from our study.

From conventional single-goal, high-goodput driven designs to customized-goal driven At first glance, the conventional high-goodput driven designs can also satisfy other performance goals. The high-goodput designs can put the Wi-Fi NIC to stay idle or sleep longer, thereby experiencing longer low-power duration and achieving energy efficiency. Moreover, it can offer shorter transmission time, thereby lower latency. However, our study has shown that it is not true in general. The highest goodput does not always assure other performance goals, energy efficiency and low latency. In the former, the root cause is that the small goodput gain may be at a high energy cost. In the latter, the high-goodput rate reduces the average or medium latency, but not necessarily the tail latency, due to retransmission.

Needless to say, the conventional single-goal designs cannot either satisfy the demand of a heterogeneous goal (e.g., concurrent goals of throughput and energy efficiency, etc.). In order to move toward customized Wi-Fi, which satisfies nowadays diverse demands, the conventional single-goal, high-goodput driven has to change to the customized-goal driven. It is able to not only achieve the single goal of energy efficiency (e.g., EERA) and low tail latency (e.g., LLRA), but also satisfy a heterogeneous goal (e.g., HetRA).

Concern of fairness for diverse goals Our study has revealed that the clients with non-high-goodput goals may slow down their transmission, thereby occupying longer airtime. It is because they mainly trade off goodput for the other performance goals. For energy efficiency, clients prevent unnecessary power-hungry high-speed components from being activated. To reduce the tail latency, clients choose a slightly lower goodput but much lower loss. However, these slowdown clients may seize other clients' necessary airtime, thereby hurting their performance.

In order to serve diverse goals in Wi-Fi networks, an airtime scheduler has to be introduced to maintain fairness among clients. The fairness policies may differ, but the scheduler is essential for the support of diverse goals to constrain the slowdown clients. In our design, only the spare airtime can be assigned to the clients with non-high-goodput goals for their slowdown. Each client's necessary airtime can thus always be assured.

Scaling issue arising from speed-boost features The increase of high-speed rate settings is accelerated by the current 802.11n/ac standards, due to the introduction of several speed-boost features. In our 802.11n testbed with three antennas, there are 48 available options. The number can increase to 360 options by considering an 802.11ac platform with eight antennas. When another feature, channel width, is also considered, these numbers can be double, triple, and quadruple for two, three, and four different channel widths, respectively. Thus, the scaling issue has become the common challenge for the link-layer or RA designs.

Such scaling issue makes RA for current 802.11n/ac devices to be more complex than that in the legacy 802.11a/b/g systems. It has to adapt over multi-dimensional parameter space. To address this challenge, we apply ternary search and simultaneous pruning into the search of EERA design, and use association rule-based pruning in the LLRA design. The key idea of them is to reduce the search space as much as possible from each probing result. Therefore, our designs can rapidly locate the best setting over the large search space.

WSC framework for customized goals WSC offers a conceptual framework to achieve customized goals. It adapts the service curve concept in the Internet QoS to the wireless 802.11n domain. It serves as the building block to specify the minimum amount of service (both bandwidth and delay) for a given flow, while managing energy efficiency for mobile devices. Thus, customized goals, which involve minimum throughput, perceived delay, and energy efficiency, can be served based the WSC framework. The framework can convert the requirements of throughput and delay to the minimum amount of service, while the energy efficiency can be achieved by the mechanism of piecewise linear WSC (PL-WSC). The preliminary results in our work have confirmed the viability of this framework. We believe that it is worthwhile to further explore this general approach.

7.3 Future Work

Along the line of this thesis, there are two major pieces of future work.

Achieving customized goals with the WSC framework We plan to support customized goals with the WSC framework. There are two major tasks. First, we extend HetRA to support another dimension of perceived delay, in addition to the current throughput and energy. The delay dimension in the WSC framework requires to be decoupled from the throughput. Specifically, the delay-sensitive flow is represented by the convex curve, whereas the flow with only throughput requirement is represented by the concave one. The initially steeper slope of the convex curve offers the higher priority to the delay-sensitive flow. By considering both delay and throughput (i.e., both concave and convex curves), the service requirements for one client become more dynamic. Thus, the major challenge is how to derive the best rate settings over time to achieve energy efficiency at the client, while satisfying its dynamic service requirements.

Second, we enable users to customize each goal by configuring its scale. For example, they require the most energy efficiency when they have low battery. After the battery is charged, they want to trade off some energy efficiency for higher goodput or lower perceived delay. A customized goal can consist of any combination of throughput, delay and energy goals with different scales. It is then converted to WSC to serve users.

Adding support of diverse goals into upcoming Wi-Fi standards We plan to enhance the fundamental Wi-Fi operations to better serve each performance goal (e.g., energy efficiency and perceived latency), and strive for adding them into the upcoming standards. There are two major threads. First, the Wi-Fi operations in the firmware/hardware should offer more flexibility so that the software (e.g., RA algorithms) can have more fine-grained control over Wi-Fi techniques, thereby achieving better performance. Take energy efficiency as an example. A power-saving mechanism for MIMO, called Spatial Multiplexing Power Save (SMPS), is introduced in the 802.11n/ac standards. It supports only two modes: static and dynamic. The former retains a single antenna for all the time, whereas the latter either activates all the antennas during MIMO transmission or retains one antenna. However, the most energy-efficient setting may belong to those intermediate settings. Specifically, the SMPS mechanism cannot support the activation of partial antennas except one. It cannot either allow a single-stream MCS rate for being used by multiple antennas, since multiple antennas are activated only when multi-stream rates are used. As a result, current power-saving mechanisms do not offer enough flexibility for being leveraged to achieve energy efficiency.

Second, some Wi-Fi operations in the firmware/hardware should be improved to serve performance goals of energy efficiency and perceived latency. For energy efficiency, some operations with energy waste can be enhanced. For example, the SMPS mechanism relies on RTS/CTS to dynamically activate multiple antennas; that is, one RTS/CTS exchange has to precedes each MIMO transmission. However, since the RTS and CTS messages are transmitted via a low, stable MCS rate, their exchange may take longer time than the actual data transmission. The former may thus consume more energy than the latter, thereby resulting in energy waste. We plan to provide a better coordination mechanism so that the RTS/CTS exchange can be avoided while the dynamic of antennas usage is still maintained.

For perceived latency, two major mechanisms for retransmission can be enhanced. First, to avoid the latency incurred by software reschedule, the retransmission of the partial packets in an aggregation frame can start directly from the hardware retry, instead of the software reschedule. However, the issue is that it may reduce the transmission efficiency, since the frame containing only retransmitted packets is relatively smaller than the normal aggregation frame. The smaller frame incurs larger MAC overhead. Thus, the proposed design needs to balance between the latency caused by the retransmission, and the transmission efficiency. Second, in the nowadays multi-client scenarios, the prioritized MAC scheduling is also critical for the retransmission of latency-sensitive traffic. However, the challenge is how to offer this prioritized MAC while still maintaining fairness among clients.

References

[11a11] "IEEE P802.11 Wireless LANs Proposed TGac Draft Amendment.", Jan 2011.

[11n09] "IEEE 802.11n Standard: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput.", 2009.

[ACW07] Yuvraj Agarwal, Ranveer Chandra, Alec Wolman, Paramvir Bahl, Kevin Chin, and Rajesh Gupta. "Wireless Wakeups Revisited: Energy Management for VoIP over Wi-Fi smartphones." In *ACM MOBISYS*, 2007.

[AKE12] Mohammad Alizadeh, Abdul Kabbani, Tom Edsall, and Balaji Prabhakar. "Less is More: Trading a Little Bandwidth for Ultra-Low Latency in the Data Center." In *NSDI*, 2012.

[AL13] Mansoor Alicherry and T. V. Lakshman. "Optimizing Data Access Latencies in Cloud Systems by Intelligent Virtual Machine Placement." In *IEEE INFOCOM*, 2013.

[Ana14] Strategy Analytics. "Embedded WLAN (Wi-Fi) CE Devices: Global Market Forecast.", February 2014.

[ASB10] Prashanth A.K. Acharya, Ashish Sharma, Elizabeth M. Belding, Kevin C. Almeroth, and Konstantina Papagiannaki. "Rate Adaptation in Congested Wireless Networks through Real-time Measurements." *IEEE Transactions on Mobile Computing*, **9**(11), November 2010.

[ath] "ath9k." `http://wireless.kernel.org/en/users/Drivers/ath9k`.

[Bic05] John C. Bicket. "Bit-rate selection in wireless networks." Technical report, Master thesis, MIT, 2005.

[CGB05] Shuguang Cui, Andrea J. Goldsmith, and Ahmad Bahai. "Energy-constrained Modulation Optimization." *IEEE Transactions on Wireless Communications*, **4**(5), September 2005.

[CGL04] Chris Chafe, Michael Gurevich, Grace Leslie, and Sean Tyan. "Effect of Time Delay on Ensemble Accuracy." In *ISMA*, 2004.

[chra] "Chromecast." `http://www.google.com/chrome/devices/chromecast/`.

[chrb] "Chromecast Gaming." `http://www.technologytell.com/gaming/123089/`.

[CK08] Joseph Camp and Edward Knightly. "Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation." In *MOBICOM*, 2008.

[CLS11] An Chan, Henrik Lundgren, and Theodoros Salonidis. "Video-Aware Rate Adaptation for MIMO WLANs." In *IEEE ICNP*, 2011.

[CM03] Hemant M. Chaskar and Upamanyu Madhow. "Fair Scheduling with Tuable Latency: A Round-robin Approach." *IEEE Transactions on Networking*, **11**(4), August 2003.

[con] "Razer Gaming Mouse." `http://www.razerzone.com/gaming-mice`.

[Cor13] IDC (International Data Corporation). "Worldwide Smart Connected Device Market Crossed 1 Billion Shipments in 2012.", March 2013.

[CQY07] Xi Chen, Daji Qiao, Jeonggyun Yu, and Sunghyun Choi. "Probabilistic-Based Rate Adaptation for IEEE 802.11 WLANs." In *IEEE GLOBECOM*, 2007.

[Cru92] Rene L. Cruz. "Service Burstiness and Dynamic Burstiness Measures: A Framework." *Journal of High Speed Networks*, **1**:105–127, 1992.

[Cru95] Rene L. Cruz. "Quality of Service Guarantees in Virtual Circuit Switched Networks." *IEEE Journal on Selected Areas in Communications*, **13**:1048–1056, 1995.

[DSP10] Fahad R. Dogar, Peter Steenkiste, and Konstantina Papagiannaki. "Catnap: Exploiting High Bandwidth Wireless Interfaces to Save Energy for Mobile Devices." In *ACM MOBISYS*, 2010.

[FPK13] Zi Feng, George Papageorgiou, Srikanth V. Krishnamurthy, Ramesh Govindan, and Tom La Porta. "Trading Off Distortion for Delay for Video Transmissions in Wireless Networks." In *IEEE INFOCOM*, 2013.

[GCN01] Javier Gomez, Andrew T. Campbell, Mahmoud Naghshineh, and Chatschik Bisdikian. "Conserving Transmission Power in Wireless Ad Hoc Networks." In *IEEE ICNP*, 2001.

[GD11] Wesam Gabran and Babak Daneshrad. "Hardware and Physical Layer Adaptation for a Power Constrained MIMO OFDM System." In *IEEE ICC*, 2011.

[GK11] Aditya Gudipati and Sachin Katti. "Strider: Automatic Rate Adaptation and Collision Handling." In *SIGCOMM*, 2011.

[Hah91] Ellen L. Hahne. "Round-Robin Scheduling for Max-Min Fairness in Data Networks." *IEEE Journal on Selected Areas in Communications*, **9**:1024–1039, September 1991.

[HGS10] Daniel Halperin, Ben Greenstein, Anmol Sheth, and David Wetherall. "Demystifying 802.11n Power Consumption." In *HotPower*, 2010.

[HHS10] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. "Predictable 802.11 Packet Delivery from Wireless Channel Measurements." In *ACM SIGCOMM*, 2010.

[HVB] Gavin Holland, Nitin Vaidya, and Paramvir Bahl. "A Rate-adaptive MAC protocol for Multi-Hop Wireless Networks.".

[JHS11] Ki-Young Jang, Shuai Hao, Anmol Sheth, and Ramesh Govindan. "Snooze: Energy Management in 802.11n WLANs." In *ACM CoNEXT*, 2011.

[JIP07] Amit P. Jardosh, Gianluca Iannaccone, Konstantina Papagiannaki, and Bapi Vinnakota. "Towards an Energy-Star WLAN Infrastructure." In *IEEE HotMobile*, 2007.

[JPB09] Amit P. Jardosh, Konstantina Papagiannaki, Elizabeth M. Belding, Kevin C. Almeroth, Gianluca Iannaccone, and Bapi Vinnakota. "Green WLANs: On-demand WLAN Infrastructures." *Mobile Networks and Applications*, **14**:798–814, December 2009.

[KCD09] Hongseok Kim, Chan-Byoung Chae, Gustavo De Veciana, and Robert W. Heath. "A Cross-layer Approach to Energy Efficiency for Adaptive MIMO Systems Exploiting Spare Capacity." *IEEE Transactions on Wireless Communication*, **8**:4264–4275, August 2009.

[KDC13] Muhammad Owais Khan, Vacha Dave, Yi-Chao Chen, Oliver Jensen, Lili Qiu, Apurv Bhartia, and Swati Rallapalli. "Model-driven Energy-aware Rate Adaptation." In *ACM MOBIHOC*, 2013.

[KM97] Ad Kamerman and Leo Monteban. "WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed Band." *Bell Labs Technical Journal*, **2**(3):118–133, 1997.

[KMG11] Kyu-Han Kim, Alexander W. Min, Dhruv Gupta, Prasant Mohapatra, and Jatinder Pal Singh. "Improving energy efficiency of Wi-Fi sensing on smartphones." In *IEEE INFOCOM*, 2011.

[LMT04] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. "IEEE 802.11 Rate Adaptation: A Practical Approach." In *ACM MSWiM*, 2004.

[LPL12a] Chi-Yu Li, Chunyi Peng, and Songwu Lu. "Achieving 802.11n NIC Energy Efficiency through Rate Adaptation." Technical report, UCLA Computer Science, 2012.

[LPL12b] Chi-Yu Li, Chunyi Peng, Songwu Lu, and Xinbing Wang. "Energy-based Rate Adaptation for 802.11n." In *ACM MOBICOM*, 2012.

[LZ08] Jiayang Liu and Lin Zhong. "Micro Power Management of Active 802.11 Interfaces." In *ACM MOBISYS*, 2008.

[MAZ11] Justin Manweiler, Sharad Agarwal, Ming Zhang, Romit Roy Choudhury, and Paramvir Bahl. "Switchboard: A Matchmaking System for Multiplayer Mobile Games." In *ACM MOBISYS*, 2011.

[MCR06] Nilesh Mishra, Kameswari Chebrolu, Bhaskaran Raman, and Abhinav Pathak. "Wake-on-WLAN." In *ACM WWW*, 2006.

[NK12] Vinod Namboodiri and Abtin Keshavarzian. "Alert: An Adaptive Low-Latency Event-Driven MAC Protocol for Wireless Sensor Networks." In *IEEE IPSN*, 2012.

[NLB99] Thyagarajan Nandagopal, Songwu Lu, and Vaduvur Bharghavan. “A Unified Architecture for the Design and Evaluation of Wireless Fair Queueing Algorithms.” In *ACM MOBICOM*, 1999.

[PHW10] Ioannis Pefkianakis, Yun Hu, Starsky H.Y. Wong, Hao Yang, and Songwu Lu. “MIMO Rate Adaptation in 802.11n Wireless Networks.” In *ACM MOBICOM*, 2010.

[PJ00] Wouter Pasman and Frederik W. Jansen. “Latency Layered Rendering for Mobile Augmented Reality.” In *ISIT*, 2000.

[PLL11] Ioannis Pefkianakis, Chi-Yu Li, and Songwu Lu. “What is Wrong/Right with IEEE 802.11n Spatial Multiplexing Power Save Feature?” In *IEEE ICNP*, 2011.

[PLP13] Ioannis Pefkianakis, Chi-Yu Li, Chunyi Peng, Suk-Bok Lee, and Songwu Lu. “CMES: Collaborative Energy Save for MIMO 802.11 Wireless Networks.” In *IEEE ICNP*, 2013.

[QCJ03] Daji Qiao, Sunghyun Choi, Admit Jain, and Kang G. Shin. “MiSer: An Optimal Low-Energy Transmission Strategy for IEEE 802.11a/h.” In *ACM MOBICOM*, 2003.

[REK] Hariharan Rahul, Farinaz Edalat, Dina Katabi, and Charles Sodini. “Frequency-Aware Rate Adaptation and MAC Protocols.”.

[RPM09] Ramya Raghavendra, Jitendra Padhye, Ratul Mahajan, and Elizabeth Belding. “Wi-Fi Networks are Underutilized.” Technical report, Microsoft Research, 2009.

[Sch06] Mihaela Van Der Schaar et al. “Optimized Scalable Video Streaming over IEEE 802.11a/e HCCA Wireless Networks under Delay Constraints.” *IEEE Transactions on Mobile Computing*, **5**(6), June 2006.

[SNC08] Dionysios Skordoulis, Qiang Ni, Hsiao-Hwa Chen, Adrian P. Stephens, Changwen Liu, and Abbas Jamalipour. “IEEE 802.11n MAC frame aggregation mechanisms for next-generation high-throughput WLANs.” *IEEE Wireless Communications*, **15**(1):40–47, 2008.

[SV95] M. Shreedhar and George Varghese. “Efficient Fair Queueing Using Deficit Round Robin.” In *ACM SIGCOMM*, 1995.

[SZN97] Ion Stoica, Hui Zhang, and T. S. Eugene Ng. “A Hierarchical Fair Service Curve Algorithm for Link-sharing, Real-time and Priority Services.” In *ACM SIGCOMM*, 1997.

[VBJ09] Mythili Vutukuru, Hari Balakrishnan, and Kyle Jamieson. “Cross-layer Wireless Bit Rate Adaptation.” In *ACM SIGCOMM*, 2009.

[VGM13] Ashish Vulimiri, P. Brighten Godfrey, Radhika Mittal, Justine Sherry, Sylvia Ratnasamy, and Scott Shenker. “Low Latency via Redundancy.” In *ACM CoNEXT*, 2013.

[WGB08] Michael S. Y. Wong, Jeffrey M. Gilbert, and Craig H. Barratt. "Wireless LAN using RSSI and BER Parameters for Transmission Rate Adaptation.", 2008. US patent, 7,369,510.

[wif12] "Study: 61% of U.S. Households Now Have WiFi.", 2012. http://techcrunch.com/2012/04/05/study-61-of-u-s-households-now-have-wifi/.

[WSB13] Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan. "Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks." In *NSDI*, 2013.

[WYL06] Starsky H. Y. Wong, Hao Yang, Songwu Lu, and Vaduvur Bharghavan. "Robust Rate Adaptation for 802.11 Wireless Networks." In *ACM MOBICOM*, 2006.

[Xu10] XiaoHua Xu et al. "A Delay-Efficient Algorithm for Data Aggregation in Multihop Wireless Sensor Networks." *IEEE Transactions on Parallel and Distributed Systems*, **22**(1), April 2010.

[ZGK11] Zheng Zeng, Yan Gao, and P. R. Kumar. "SOFA: A Sleep-Optimal Fair-Attention Scheduler for the Power-Saving Mode of WLANs." In *IEEE ICDCS*, 2011.

[ZLC11] Wenxuan Zhou, Qingxi Li, Matthew Caesar, and P. Brighten Godfrey. "ASAP: A Low-latency Transport Layer." In *ACM CoNEXT*, 2011.

[ZS11] Xinyu Zhang and Kang G. Shin. "E-MiLi: Energy-Minimizing Idle Listening in Wireless Networks." In *ACM MOBICOM*, 2011.