A Study of Dynamic Access-Point Configuration and Power Minimization in Elastic Wireless Local-Area Network System

September, 2019

Md. Manowarul Islam

Graduate School of
Natural Science and Technology

(Doctor's Course)
Okayama University

Dissertation submitted to
Graduate School of Natural Science and Technology
of
Okayama University
for
partial fulfillment of the requirements
for the degree of
Doctor of Philosophy.

Written under the supervision of

Professor Nobuo Funabiki

and co-supervised by
Professor Satoshi Denno
and
Professor Yasuyuki Nogami

Okayama University, September 2019.

To Whom It May Concern

We hereby certify that this is a typical copy of the original doctor thesis of
Md. Manowarul Islam

Signature of

the Supervisor

Prof. Nobuo Funabiki

Seal of

Graduate School of

Natural Science and Technology

Abstract

Recently, *Wireless Local Area Networks (WLANs)* have increased popularity around the world and have been commonly deployed in various places, including airports, shopping malls, stations, and hotels, due to the characteristics of easy installations, flexible coverages areas, and low costs. The WLAN users can easily access to the Internet through associations with *access points (APs)* using mobile devices like smart phones, tablets, and laptops. In WLAN, the number of users is always changing by day and time, and users are not evenly distributed in the field. To optimize the active APs and their host associations in WLAN depending on traffic demands from users, we have proposed the *elastic WLAN system* and the *active AP configuration algorithm*.

In the *elastic WLAN system*, Linux environments have been adopted to implement the testbed because various software tools are available to realize the functions of the system. Linux PC is used for the management server and the host, and *Raspberry Pi* is for the AP.

The *active AP configuration algorithm* selects the minimum number of active APs in the network while satisfying the constraints of the throughputs for the hosts, and assigns one channel to every active AP from the limited number of the non-interfered channels. We assume that each of the active AP uses the default maximum transmission power for the stronger signal during the communication. The *throughput estimation model* has been used to evaluate the link throughput between an AP and a host, where the model parameter values are obtained from extensive measurements by applying the *parameter optimization tool*. Unfortunately, the throughput measurements need high labor costs and the corresponding measurement can be done manually.

However, in the previous studies of the elastic WLAN system and the active AP configuration algorithm, there are four drawbacks to be solved. The first drawback is that the previous studies do not handle the dynamic nature of WLAN, assuming the fixed user state in the network. Even if only one host joins or leaves the network, it can generate the totally new configuration without considering the currently communicating hosts and APs and their associations. The second drawback is that the elastic WLAN system testbed does not have the security function. To enhance the safety of the system, any system software must be updated for the newest version, and any host user must be authenticated before joining the system. The third drawback is that the transmission power of the AP is fixed at the default maximum one, although the strong power consumes more energy and increases interferences to adjacent wireless nodes. It should be minimized to ensure the required throughput against the associated host. The fourth drawback is that the active AP configuration algorithm needs the accurate throughput estimation model. It requires a lot of throughput measurements in the network field, which should be minimized as best as possible.

To solve the above drawbacks, firstly, in this thesis, we proposed the extension of the *active AP configuration algorithm* to deal with the dynamic nature of the WLAN and implemented the elastic WLAN testbed using Raspberry Pi. This extension considers that any communicating AP cannot be suspended and any communicating host cannot change its associated AP. Through numerical experiments in simulations using the WIMNET simulator and throughput measurements in real

environments using the testbed, the effectiveness of our proposal is demonstrated.

Secondly, in this thesis, we implement the *system software update function* and the *user authentication function*. The former function periodically downloads the latest system software from the website and stores them to a repository server for the APs and the hosts. The latter function authenticates any joining host via an AP using the *RADIUS* server. The correctness of the implemented functions is confirmed through experiments using the testbed.

Thirdly, in this thesis, we propose two approaches for the *transmission power minimization* in the *elastic WLAN system*. The first approach estimates the minimum transmission power of each active AP that satisfies the *minimum throughput constraint*, by using the *throughput estimation model*. The effectiveness of our proposal is verified through simulations using the WIMNET simulator and experiments using the testbed in several network scenarios. The second approach is the use of the PI feedback control, which can avoid the requirement of the accurate model. First, the initial power is examined from the difference between the measured *received signal strength (RSS)* at the AP from the host and the estimated RSS necessary for the target throughput. Then, the power is minimized by using the *PI feedback control*, such that the measured throughput achieves the target one. For evaluations, the proposal is implemented in the elastic WLAN system testbed. The experiment results in various fields confirm that this approach can reduce the transmission power significantly while keeping the target throughput performance.

Fourthly, in this thesis, we propose the method of minimizing the number of host locations for measurements. To reduce the number of locations while keeping the model accuracy, we consider the conditions to select the measurement locations. The effectiveness of the methods are confirmed through experiments, where the number of locations is reduced by 60% and 86% by the previous and proposed method respectively, while the throughput estimation error is not degraded by any method.

In future studies, we will consider further enhancements of the active AP configuration algorithm and the elastic WLAN system implementation, and their evaluations in various network fields.

To My Beloved Wife, Parents and Brother

Acknowledgements

It is my great pleasure to express my heartiest thankfulness to those who gave me the valuable time and supported me in making this dissertation possible. I believe that you are the greatest blessing in my life. Thanks all of you to make my dream successful.

I owe my deepest sense of gratitude to my honorable supervisor, Professor Nobuo Funabiki for his excellent supervision, meaningful suggestions, persistent encouragements, and other fruitful help during each stage of my Ph.D. study. His thoughtful comments and guidance helped me to complete my research papers and present them in productive ways. Besides, he was always patient and helpful whenever his guidance and assistance were needed in both of my academic and daily life in Japan. I have really been lucky in working with a person like him. Needless to say, it would not have been possible to complete this thesis without his guidance and active support.

I am indebted to my Ph.D. co-supervisors, Professor Satoshi Denno and Professor Yasuyuki Nogami, for taking the valuable time to give me advice, guidance, insightful comments, and proof-reading of this thesis.

I want to express my profound gratitude to Associate Professor Minoru Kuribayashi in Okayama University for his valuable discussions during my research. I would like to take this opportunity to thank and convey my respect to all of my course teachers during my Ph.D. study for sharing their great ideas and knowledge with me.

I would like to acknowledge the Ministry of Education, Culture, Sports, Science, and Technology of Japan (MEXT) for financially supporting my Ph.D. study, and the Begum Rokeya University, Rangpur, Bangladesh for giving me the study leave permission for this study.

I would like to thank for the fruitful discussions and cooperations with many people including Dr. Sritrusta Sukaridhoto, Dr. I-Wei Lai, Dr. Nobuya Ishihara, Dr. Md. Ezharul Islam, Dr. Md. Selim Al Mamun, Dr. Khin Khin Zaw, Dr. Kyaw Soe Lwin, Dr. Sumon Kumar Debnath, Ms. Mousumi Saha, Mr. Kwenga Ismael Munene, Mr. Rahardhita Widyatra Sudibyo and Mr. Hendy Briantoro. I would like to convey my respect to all the members of FUNABIKI Lab for their support during the period of this study.

I especially want to thank my beloved wife Murshid Jahan Ferdous, who always comforts, consoles, and encourages me. Thank you for being with me in all the difficult time in Japan.

Last but not least, I am grateful to my family, my mother, father, brother, and all of my friends. For your unconditional loves, supports, patience, and confidence in me are the biggest rewards as well as the driving forces of my life.

Md. Manowarul Islam
Okayama University, Japan
September 2019

List of Publications

Journal Papers

1. **M. M. Islam**, N. Funabiki, M. Kuribayashi, S. K. Debnath, I. M. Kwenga, K. S. Lwin, , R. W. Sudibyo and M. S. A. Mamun, "Dynamic access-point configuration approach for elastic wireless local-area network system and its implementation using Raspberry Pi," Int. J. Netw. Comput., vol. 8, no. 2, pp. 254-281, July 2018.

2. **M. M. Islam**, N. Funabiki, M. Kuribayashi, M. Saha, I. M. Kwenga, R. W. Sudibyo, and W-C. Kao, "A proposal of transmission power minimization extension in active access-point configuration algorithm for elastic wireless local-area network system ," Int J. Comput. Soft. Eng., vol. 4, no. 140, pp. 1-9, Jan. 2019.

3. **M. M. Islam**, N. Funabiki, R. W. Sudibyo, I. M. Kwenga, and W-C. Kao, " A dynamic access-point transmission power minimization method using PI feedback control in elastic WLAN system for IoT applications," Internet of Things, vol. 8, pp. 1-15, Aug. 2019.

International Conference Papers

4. **M. M. Islam**, M. S. A. Mamun, N. Funabiki, and M. Kuribayashi, " Dynamic access-point configuration approach for elastic wireless local-area network system," Proc. Int. Symp. Comput. Netw., pp. 216-222, Nov. 2017.

5. **M. M. Islam**, N. Funabiki, M. Kuribayashi, M. Saha, I. M. Kwenga, R. W. Sudibyo, and W-C. Kao, "An improvement of throughput measurement minimization method for access-point transmission power minimization in wireless local-area network," Proc. IEEE Int. Conf. Consum. Elect. Taiwan (ICCE-TW), May 2019.

Other Papers

6. **M. M. Islam**, M. S. A. Mamun, N. Funabiki, and M. Kuribayashi, "A dynamic host behavior extension of active access-point configuration algorithm for elastic WLAN system," Chugoku-branch Joint Conv. Inst. Elec. Info. Eng., Oct. 2017.

7. **M. M. Islam**, N. Funabiki, M. Kuribayashi, R. W. Sudibyo and I. M. Kwenga, "Implementations of system software update and user authentication functions in elastic wireless local-area network system," IEICE Tech. Rep., SRW2018-14, pp. 31-36, Aug. 2018.

8. **M. M. Islam**, N. Funabiki, M. Kuribayashi, and W-C. Kao, "A proposal of transmission power minimization extension in active access-point configuration algorithm for elastic wireless local-area network system," IEICE Tech. Report, NS2018-173, pp. 83-88, Dec. 2018.

9. **M. M. Islam**, N. Funabiki, M. Saha, I. M. Kwenga, and R. W. Sudibyo, "Throughput measurement location minimization method for access-point transmission power minimization in wireless local-area network," IEICE General Conf., pp. S-76-77, March 2019.

10. **M. M. Islam**, N. Funabiki, R. W. Sudibyo, I. M. Kwenga, and H. Briantoro, "An access-point transmission power minimization approach using PI feedback control in wireless local-area network," IEICE Society Conf., pp. S-29-30, Sept. 2019.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Background

Nowadays, *Wireless Local Area Networks (WLANs)* have become increasingly popular, and have been commonly deployed around the world. Due to the characteristics of easy installations, flexible coverages areas, and low costs, WLANs provide the Internet access in various places, including airports, shopping malls, stations, and hotels [1, 2]. Wireless communications between *Access Points (APs)* and hosts make WLANs more flexible, scalable, and accessible than wired LANs. WLANs have become the common ways for to the Internet access in governments, companies, and educational institutes.

In a WLAN, APs are often installed in the service field randomly, which can cause poor network performances due to overlapping of the same frequency signals [3]. Actually, the configuration of these APs should be properly arranged according to the traffic demands in the field, while redundant APs should be turned off for energy saving and interference preventions.

In a WLAN, the distribution of users is non-uniform [4], and the number of users or traffics fluctuates depending on time and day of the week [2], which is often unpredictable [5]. For example, in a university, a great number of students access to the network in the afternoon on weekdays, while much fewer students do so in the morning/evening and on weekends.

In addition, the conditions of network devices and communication links may be suffered from various factors, such as power shortages, device failures, bandwidth controls by authorities, and even weather changes [6]. Actually, many developing countries including Bangladesh and Myanmar often suffer from the unreliable and slow Internet access due to discontinuities of electricity supplies for the time being [7, 8].

To solve the above-mentioned problems, we have studied the *elastic WLAN system* using heterogeneous AP devices [9, 10]. In our studies, three types of APs, namely *dedicated APs (DAPs)*, *virtual APs (VAPs)*, and *mobile APs (MAPs)*, have been considered. Figure 1.1 shows a simple topology of the elastic WLAN system. It dynamically controls the number of active APs according to traffic demands and device conditions. For this control, we have proposed the *active AP configuration algorithm* that selects the minimum number of active APs in the network, satisfying the constraints of the throughputs for the hosts, and assigns one channel to every active AP from the limited number of the non-interfered channels in a way to minimize the overall interference in the network. Each of the active AP and host in the network uses the default maximum transmission power during the communication.

A *throughput estimation model* has been studied to estimate the link throughput between an AP and a host in a WLAN, which is essential in the active AP configuration algorithm. The

parameter values of the model are optimized by throughput measurement results with different host locations in the network field. To improve the estimation accuracy of the model, the *parameter optimization tool* [11, 12] has been proposed to find the accurate parameter values such that the average difference between the measured throughput and the estimated one becomes minimal.

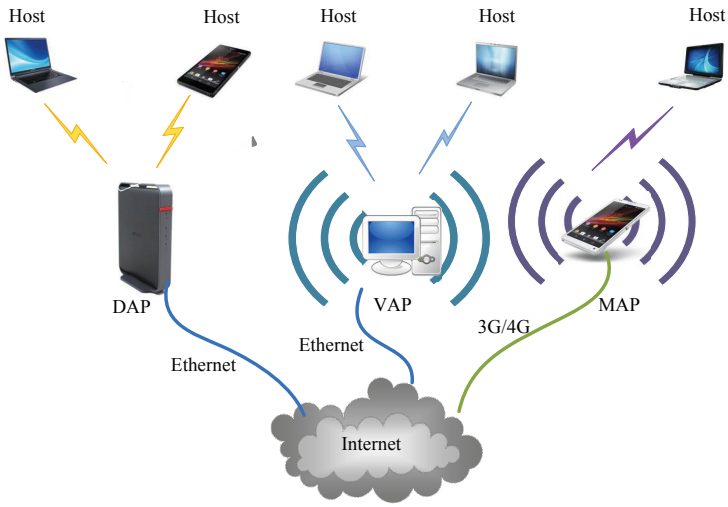


Figure 1.1: Overview of Elastic WLAN.

However, in the previous studies of the elastic WLAN system and the active AP configuration algorithm, the following drawbacks need to be solved for reducing the power consumption and better performance of the current research:

- Hosts often repeat joining and leaving in the WLAN. Thus, the network configuration should be adaptive with this dynamic host behavior. Unfortunately, the current active AP configuration algorithm can only find the whole solution of all the active APs at the same time, with their associated hosts and assigned channels for a given state of the traffic demands and the device conditions in the network. Even if only one host joins or leaves the network, the current algorithm generates the totally new configuration without considering the current state. If some hosts are currently communicating with the Internet, they cannot be stopped to change the associated APs. Thus, the new configuration must avoid this discontinuity of communicating services.

- In the current implementation of the elastic WLAN system, a Linux PC including *Raspberry Pi* is used for a host and an AP. Therefore, to increase the security and performance of the system, it is important to keep updating the system and application software to the latest version. The user authentication is also inevitable to enhance the security of the system.

- In the current algorithm, each of active APs is running with the maximum transmission power. Usually, a significant number of APs is deployed in a WLAN, which may cause a large power consumption of the network and increase the interference to other APs. In such environment, reducing the transmission power of APs is important to reduce the power consumption and interference. Hence, the efficient transmission power management of the AP while maintaining the network performance is a challenging task.

- Finally, the extensive measurements for the accurate parameter tuning in the *throughput estimation model* cause high labor costs. The host must be moved to many locations, and the measurements can be done manually. Thus, the number of measurement locations should be minimized.

1.2 Contributions

In this thesis, we have carried out the following research contributions.

As the first contribution, we propose the extension of the *active AP configuration algorithm* to overcome the discontinuity problem in the current algorithm [13–15]. In this extension, the currently communicating APs and hosts are designated as *communicating APs* and *communicating hosts* for convenience. Any *communicating AP* is not deactivated and any *communicating host* does not change the associated AP by the algorithm. Through numerical experiments using the *WIMNET simulator* [16] in two network instances, the effectiveness of the proposed extension is demonstrated.

Furthermore, the proposed algorithm extension is implemented on the elastic WLAN system testbed to verify the practicality in the real system. In this testbed implementation, *Raspberry Pi* [17] is adopted for the AP and Linux PC is for the host. The performance of this elastic WLAN system testbed is evaluated through experiments in four scenarios, where the measured throughputs by the proposal are always higher than those by the comparison methods.

As the second contribution, we implement the *system software update function* for APs and hosts and *user authentication function* in the elastic WLAN system, to enhance the system performance and security [18]. The system software update function periodically downloads the latest system software to the local repository servers, and installs it into them. Every AP and user host in the system accesses to this server to update the software. Then, the *user authentication function* is implemented to authenticate a newly joining host using the *RADIUS* server. These functions are evaluated with the elastic WLAN system testbed using *Linux* PCs for hosts and *Raspberry Pi* for APs.

As the third contribution, we propose two approaches for the *transmission power minimization* to reduce the power consumption in the *elastic WLAN system*. The first approach is the extension of the *active AP configuration algorithm* to reduce the energy consumption at each of the active APs [19, 20]. In this extension, the previous active AP configuration finds the active APs, AP-host associations, and AP channel assignments, assuming the use of the maximum transmission power. Then, it minimizes the transmission power of each active AP such that it satisfies the *minimum throughput constraint*. The *throughput estimation model* has been used to evaluate the throughput for each transmission power, where the model parameter values are obtained from extensive measurements by applying the *parameter optimization tool* [11, 12].

The effectiveness of the proposal was validated through simulations with the WIMNET simulator and experiments using the testbed in several network scenarios. Unfortunately, the throughput measurement of different transmission powers require a lot of labor costs for accurate model parameters that needs to be minimized.

In the second approach, we propose the PI feedback control for the transmission power minimization [21, 22]. The benefit of this approach is to avoid extensive measurements. The initial power is examined from the difference between the measured *received signal strength (RSS)* at the AP from the host and the estimated RSS necessary for the target throughput. Then, the power is dynamically minimized by using the *PI feedback control* [23], such that the measured throughput

by *iperf* [24], achieves the target one. The proposal was implemented in the *elastic WLAN system* testbed. The experiment results in different network topologies confirmed the effectiveness of the proposal.

Finally, as the fourth contribution, we propose two measurement location minimization methods for selecting host locations that can keep the original accuracy of the *throughput estimation model* [25, 26]. To reduce the number of locations while keeping the model accuracy, we consider the conditions to select the measurement locations. We confirm the effectiveness of our proposal through experiments for IEEE 802.11n WLAN, where the throughput estimation error is not degraded after the location minimization and the number of original measurements is reduced by 60% and 86% by first method and second method respectively.

The four contributions in the thesis are summarized as follows:

- the proposal of the dynamic extension of the active AP configuration algorithm to deal with the dynamic host behavior in the elastic WLAN system.
- the implementation of the system software update and user authentication functions for system security of the elastic WLAN system.
- the proposal of AP transmission power minimization methods using the throughput estimation model and the PI feedback control.
- the proposal of measurement location minimization methods to reduce the labor cost and time for the throughput estimation model.

1.3 Thesis Outline

The remaining part of this thesis is organized as follows.

In Chapter 2, we review IEEE 802.11 wireless network technologies related to this thesis, including the IEEE 802.11n protocols, features of IEEE 802.11n protocols, and software tools in the Linux operating system.

In Chapter 3, we review our previous related studies.

In Chapter 4, we describe the proposal of the dynamic extension of the active AP configuration algorithm, the implementation in the elastic WLAN system testbed, and the evaluation through simulations using the WIMNET simulator and experiments using the testbed.

In Chapter 5, we describe the implementation of the system software update and user authentication functions for system security of the elastic WLAN system.

In Chapter 6, we describe the proposal of the static AP transmission power minimization approach using the throughput estimation model and the evaluation of the proposal through simulations and testbed experiments.

In Chapter 7, we describe the proposal of the dynamic AP transmission power minimization approach using the PI feedback control, the implementation in the elastic WLAN system testbed, and the evaluation in various network fields.

In Chapter 8, we describe the proposal of two measurement location minimization methods for throughput estimation model.

Finally, in Chapter 9, we conclude this thesis with some future works.

Chapter 2

IEEE 802.11 Wireless Network Technologies

In this chapter, we briefly introduce wireless network technologies for backgrounds of this dissertation. First, we review *IEEE 802.11 protocols*. Then, we discuss the *IEEE 802.11n protocol*, especially, the key features of the protocol. After that, we introduce the three different types of APs assumed in the active AP configuration algorithm and discuss the characteristics and the speed difference between them. Finally, we outline some Linux tools and commands for WLANs that are used for measurements, and the implementation of elastic WLAN system.

2.1 802.11 WLAN Overview

IEEE 802.11 standards determine *physical (PHY)* and *media access control (MAC)* layer specifications for implementing high-speed *wireless local area network (WLAN)* technologies. WLAN is an extension to a wired LAN that enables the user mobility by the wireless connectivity and supports the flexibility in data communications [27]. It can reduce the cabling costs in the home or office environments by transmitting and receiving data over the air using *radio frequency (RF)* technology.

2.1.1 Advantages of WLAN

WLAN offers several benefits over the traditional wired networks. Specific benefits are included in the following [27]:

- *User mobility:*
  In a wired network, users need to use wired lines to stay connected to the network. On the other hand, WLAN allows user mobility within the coverage area of the network.

- *Easy and rapid deployment:*
  WLAN can exclude the requirement of network cables between hosts and connection hubs or APs. Thus, the installation of WLAN can be much easier and quicker than wired LAN.

- *Cost:*
  The initial installation cost can be higher than the wired LAN, but the life-cycle cost can be significantly lower. In the environment requiring frequent movements or reconfigurations of the network, WLAN can provide the long-term cost profit.

- *Increased flexibility:*
  The network coverage area by WLAN can be easily expanded because the network medium is already everywhere.

- *Scalability:*
  WLAN can be configured in a variety of topologies suitable to applications. WLAN can support both peer-to-peer networks suitable for a small number of users and full infrastructure networks of thousands of users.

2.1.2 IEEE 802.11 WLAN Components

IEEE 802.11 WLAN consists of four primary components as shown in Figure 2.1 [27]:

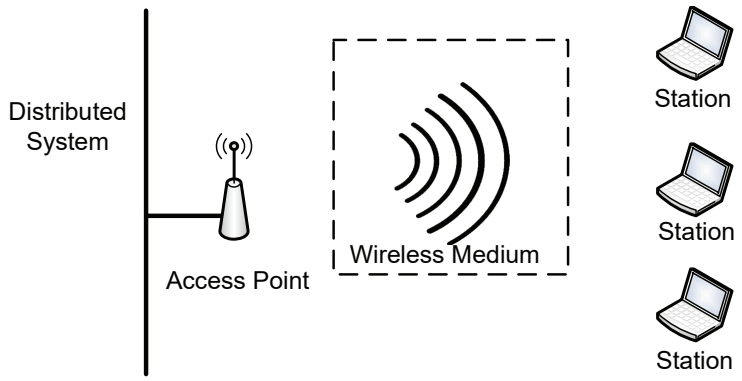


Figure 2.1: Components of IEEE 802.11 WLANs.

- *Stations or hosts:*
  WLAN transfers data between *stations or hosts*. A station in WLAN indicates an electronic device such as a desktop/laptop PC, a smartphone, or a tablet that has the capability of accessing the network over the wireless network interface card (NIC).

- *Access points (APs):*
  An AP acts as the main radio transceiver or a generic base station for WLAN that plays the similar role as a hub/switch in a wired Ethernet LAN. It also performs the bridging function between the wireless and the wired networks with some other tasks.

- *Wireless medium:*
  The IEEE 802.11 standard uses the wireless medium to flow the information from one host to another host in a network.

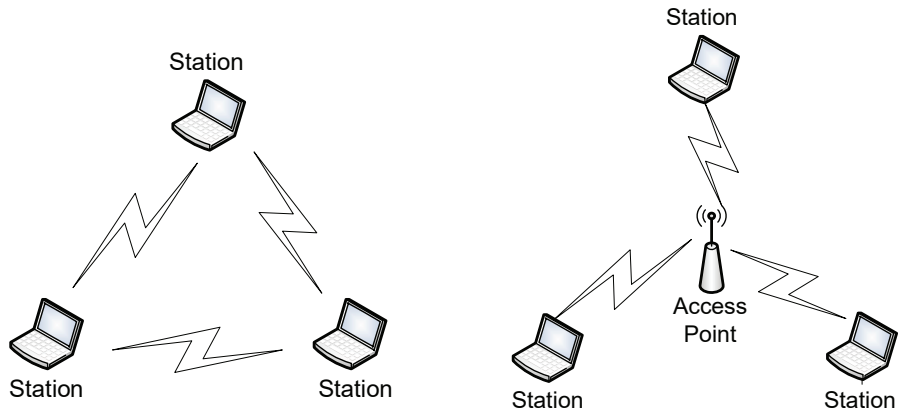
- *Distribution system:*
  When several APs are connected together to form a large coverage area, they must communicate with each other to trace the movements of the hosts. The distribution system is the logical component of WLAN which serves as the backbone connections among APs. It is often called as the *backbone network* used to relay frames between APs. In most cases, *Ethernet* is commonly used as the backbone network technology.

2.1.3 Operating Modes for IEEE 802.11 WLAN

The elementary unit of IEEE 802.11WLAN is simply a set of hosts that can communicate with each other known as the *basic service set (BSS)*. Based on the types of BSS, the IEEE 802.11 supports two operating modes as illustrated in Figure 2.2.

- *Independent or ad hoc mode:*
  In this mode, a collection of stations or hosts can send frames directly to each other without an AP. It is also called as an *independent BSS (IBSS) mode* as shown in Figure 2.2(a). This *ad hoc network* is rarely used for permanent networks due to the lack of required performances and security issues.

- *Infrastructure mode:*
  In this mode, the stations exchange their information through an AP as shown in Figure 2.2(b). A single AP acts as the main controller to all the hosts within its BSS, known as *infrastructure BSS*. In this mode, a host must be associated with an AP to obtain network services [28].



(a) Adhoc network. (b) Infrastructure network.

Figure 2.2: Operating modes of IEEE 802.11 networks.

In addition, multiple BSSes can be connected together with a backbone network to form *extended service set (ESS)* as shown in Figure 2.3. ESS can form a large size WLAN. Each AP in ESS is given an ID called the *service set identifier (SSID)*, which serves as a "network name" for the users. Hosts within the same ESS can communicate with each other, even if they are in different basic service areas.

2.1.4 Overview of IEEE 802.11 Protocols

The IEEE 802.11 working group has improved the existing PHY and MAC layer specifications to realize WLAN at the 2.4-2.5 GHz, 3.6 GHz and 5.725-5.825 GHz unlicensed ISM *(Industrial, Scientific and Medical)* frequency bands defined by the ITU-R. In this working group, several types of IEEE Standard Association Standards are available, where each of them comes with a letter suffix, covers from wireless standards, to standards for security aspects, Quality of Service (QoS) and others, shown in Table 2.1 [28–33].

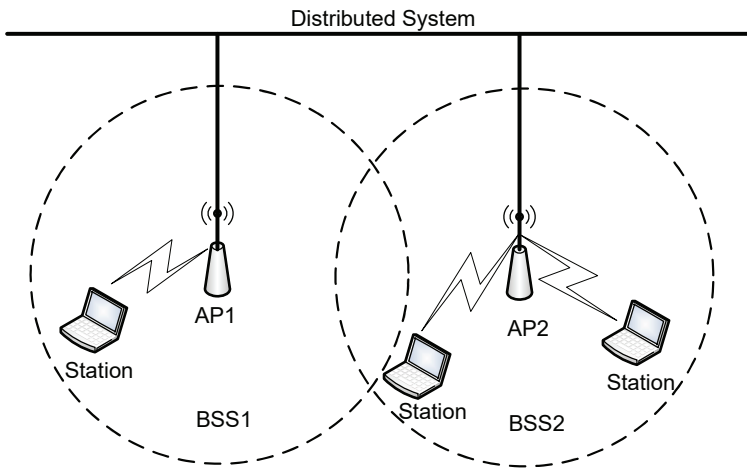


Figure 2.3: Extended service set (ESS).

Table 2.1: IEEE 802.11 Standards.

Standard	**Purpose**
802.11a	Wireless network bearer operating in the 5 GHz ISM band, data rate up to 54$Mbps$
802.11b	Operate in the 2.4 GHz ISM band, data rates up to 11$Mbps$
802.11c	Covers bridge operation that links to LANs with a similar or identical MAC protocol
802.11d	Support for additional regulatory differences in various countries
802.11e	QoS and prioritization, an enhancement to the 802.11a and 802.11b WLAN specifications
802.11f	Inter-Access Point Protocol for handover, this standard was withdrawn
802.11g	Operate in 2.4 GHz ISM band, data rates up to 54$Mbps$
802.11h	Dynamic Frequency Selection (DFS) and Transmit Power Control (TPC)
802.11i	Authentication and encryption
802.11j	Standard of WLAN operation in the 4.9 to 5 GHz band to conform to the Japan's rules
802.11k	Measurement reporting and management of the air interface between several APs
802.11l	Reserved standard, to avoid confusion
802.11m	Provides a unified view of the 802.11 base standard through continuous monitoring, management and maintenance
802.11n	Operate in the 2.4 and 5 GHz ISM bands, data rates up to 600$Mbps$
802.11o	Reserved standard, to avoid confusion
802.11p	To provide for wireless access in vehicular environments (WAVE)
802.11r	Fast BSS Transition, supports VoWiFi handoff between access points to enable VoIP roaming on a WiFi network with 802.1X authentication
802.11s	Wireless mesh networking
802.11t	Wireless Performance Prediction (WPP), this standard was cancelled

Table 2.1: IEEE 802.11 Standards.

Standard	Purpose
802.11u	Improvements related to "hotspots" and 3rd party authorization of clients
802.11v	To enable configuring clients while they are connected to the network
802.11w	Protected Management Frames
802.11x	Reserved standard, to avoid confusion
802.11y	Introduction of the new frequency band, 3.65-3.7GHz in US besides 2.4 and 5 GHz
802.11z	Extensions for Direct Link Setup (DLS)
802.11aa	Specifies enhancements to the IEEE 802.11 MAC for robust audio video (AV) streaming
802.11ac	Wireless network bearer operating below 6 GHz to provide data rates of at least $1Gbps$ for multi-station operation and $500Mbps$ on a single link
802.11ad	Wireless Gigabit Alliance (WiGig), providing very high throughput at frequencies up to 60GHz
802.11ae	Prioritization of management frames
802.11af	WiFi in TV spectrum white spaces (often called White-Fi)
802.11ah	WiFi uses unlicensed spectrum below 1GHz, smart metering
802.11ai	Fast initial link setup (FILS)
802.11aj	Operation in the Chinese Milli-Meter Wave (CMMW) frequency bands
802.11ak	General links
802.11aq	Pre-association discovery
802.11ax	High efficiency WLAN, providing 4x the throughput of 802.11ac
802.11ay	Enhancements for Ultra High Throughput in and around the 60GHz Band
802.11az	Next generation positioning
802.11mc	Maintenance of the IEEE 802.11m standard

Figure 2.4 shows the current and future WiFi standards. Among these standards, the common and popular ones are IEEE 802.11a, 11b, 11g, 11n, and 11ac. For the physical layer, the 11a/n/ac use *Orthogonal Frequency Division Multiplexing (OFDM)* modulation scheme while the 11b uses the *Direct Sequence Spread Spectrum (DSSS)* technology. 11g supports both technologies. Table 2.2 summarizes the features of these common WiFi standards [27, 34, 35].

- *IEEE 802.11b:* IEEE 802.11b operates at 2.4 GHz band with the maximum data rate up to 11 Mbps. 11b is considered to be a robust system and has a capacity to compensate the same IEEE 802.11 protocols. Because of the interoperability feature between products from different vendors, this standard has not only boosted the manufacturing of the products but also motivated the competitions between WLAN vendors. The limitation of this standard is the interference among the products using *industrial, scientific and medical* (ISM) band that uses the same 2.4 GHz band of frequency [36, 37].

- *IEEE 802.11a:* IEEE 802.11a operates at 5 GHz ISM band. It adopts on orthogonal frequency division multiplexing (OFDM) coding scheme that offers a high data rates up to 6,

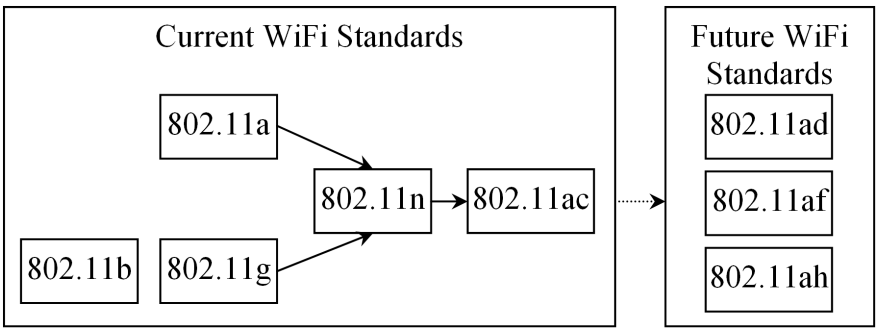


Figure 2.4: Current and future WiFi Standards.

Table 2.2: Characteristics of common IEEE 802.11 standards.

	IEEE 802.11b	IEEE 802.11a	IEEE 802.11g	IEEE 802.11n	IEEE 802.11ac
Release	Sep 1999	Sep 1999	Jun 2003	Oct 2009	Dec 2013
Frequency Band	2.4 GHz	5 GHz	2.4 GHz	2.4/5 GHz	5 GHz
Max. Data Rate	11 Mbps	54 Mbps	54 Mbps	600 Mbps	1300 Mbps
Modulation	CCK[1]modulated with PSK	OFDM	DSSS[2], CCK, OFDM	OFDM	OFDM
Channel Width	20 MHz	20 MHz	20 MHz	20/40 MHz	20/40/80/160 MHz
# of Antennas	1	1	1	4	8
security	Medium	Medium	Medium	High	High

[1] CCK: Complementary Code Keying

[2] DSSS: Direct Sequence Spread Spectrum

12, 24, 54 Mbps, and sometimes beyond this speed in comparison to 11b. Two main limitations of 11a are the compatibility issue of the 11a products with 11b products and the unavailability of 5 GHz band with free of cost for all the countries in the world [36, 37].

- *IEEE 802.11g:* IEEE suggested 11g standard over 11a to improve the 2.4 GHz 11b technology. 11g introduces two different modulation techniques including the *packet binary convolution code (PBCC)* that supports the data rate up to 33Mbps and the *orthogonal frequency division multiplexing (OFDM)* that supports up to 54Mbps data rate. Compatibility issues are also resolved in 11g products with 11b products [36, 37].

- *IEEE 802.11n:* The primary purpose of initiating the 11n standard to improve the usable range and data rate up to 600Mbps. 11n supports both of 2.4 GHz and 5 GHz ISM band *unlicensed national information infrastructure* (UNII) band, and is backward compatible with earlier standards. It introduces new technology features including the use of *channel bonding* and *multiple antennas* to get the better reception of the RF signals to enhance the throughput and coverage range [36, 38].

- *IEEE 802.11ac:* The aim of the 11ac standard to improve individual link performance and total network throughput of more than 1Gbps. Many of the specifications like static and dynamic channel bonding and simultaneous data streams of 11n have been kept and further enhanced for 11ac to reach the gigabit transmission rate. It supports static and dynamic channel bonding up to $160MHz$ and *Multi-User Multiple-Input-Multiple-Output (MU-MIMO)*. 11ac operates only on the 5 GHz band [39–41].

2.2 IEEE 802.11n Protocol

In this section, we overview the IEEE 802.11n protocol that has been used for our throughput measurements and implementations. IEEE 802.11n is an amendment to the IEEE 802.11 2007 wireless networking standard. This standard was introduced with 40 MHz bandwidth channels, *Multiple-Input-Multiple-Output* (MIMO), frame aggregation, and security improvements over the previous 11a, 11b, and 11g standards. Table 2.3 shows a brief summary of this standard.

Table 2.3: IEEE 802.11n specification.

Specification	**IEEE 802.11n**	
Frequency Band	2.4 GHz	5 GHz
Simultaneous Uninterrupted Channel	2 ch	9 ch
Available Channel	13 ch	19 ch
Max. Speed	$600Mbps$	
Max. Bandwidth	40 MHz	
Max. Spatial Streams	4	
Subcarrier Modulation Scheme	64 QAM	
Release Date	Sept 2009	

The IEEE 802.11n is available both on 2.4 GHz and 5 GHz bands. Nowadays, 2.4 GHz is very popular as it was inherited from the IEEE 802.11g. This frequency band has become crowded with lots of WiFi signals using the same channel. As a result, these WiFi signals with adjacent channels will suffer from interferences between them, and end up with throughput performance degrades [32, 34]. For 2.4 GHz band, there is a limited number of non-interfered channels, which are Channel 3 and Channel 11 in 40 MHz bandwidth. While for 20 MHz bandwidth, Channel 1, Channel 6, and Channel 11 are free from interference. In overall, the wider bandwidth will reduce the number of free channels. Figure 2.5 [33] shows the WiFi channels for IEEE 802.11n 2.4 GHz band. In the 5 GHz band of IEEE 802.11n protocol, it has 19 uninterrupted channels available with the $20MHz$ bandwidth. In the $40MHz$ bandwidth, which doubles the channel width from the $20MHz$, there are nine channels. For the $80MHz$ bandwidth, there are four of them. Figure 2.6 shows these WiFi channels for the IEEE 802.11n 5 GHz band [42].

2.3 Features of IEEE 802.11n Protocol

IEEE 802.11n protocol incorporates several new technologies to boost up its performance. The standard uses the multiple antennas technology, channel bonding, frame aggregation, and security

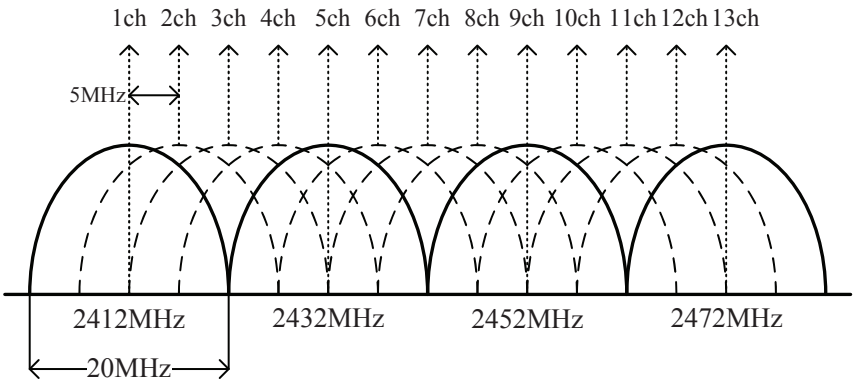


Figure 2.5: WiFi channels in 2.4 GHz band.

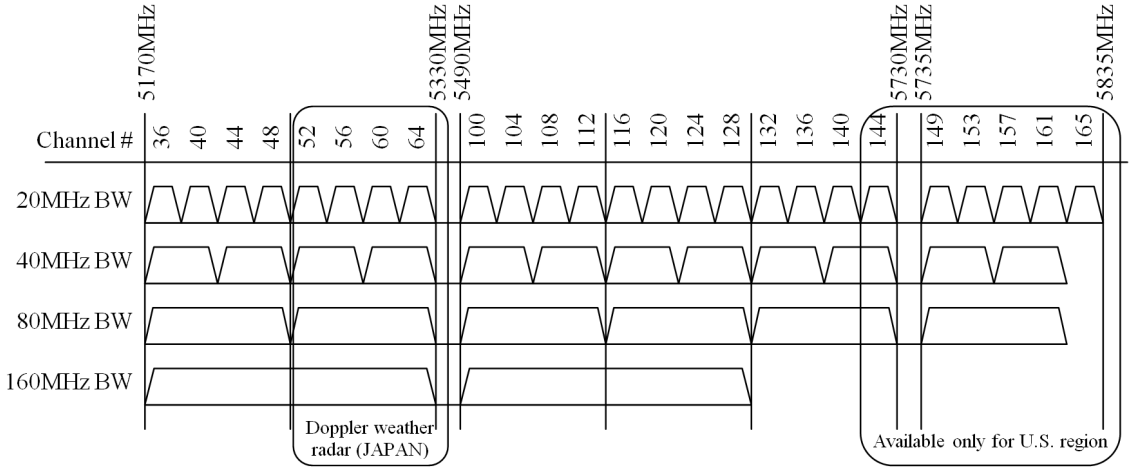


Figure 2.6: WiFi channels in 5 GHz band.

improvements mechanism to improve the throughput. In this section, we describe these features of IEEE 802.11n Protocol.

- *Channel Bonding:*
  In the channel bonding, each channel can operate with the $40MHz$ bandwidth by using two adjacent $20MHz$ channels together to double its physical data rate [43] as shown in Figure 2.7. However, the usage of the channel bonding will reduce the available non-interfered channels for other devices as there are only two non-interfered bonded channels available for IEEE 802.11n protocol at 2.4 GHz band. Table 2.4 shows the usage of different channel bandwidths and spatial streams towards the throughput of IEEE 802.11n.

- *MIMO (Multiple-Input-Multiple-Output):*
  In MIMO, the throughput can be linearly increased to the number of transmitting (T_X) and receiving (R_X) antennas up to four times, without the additional bandwidth or transmission power. The coverage area can be enhanced over the single antenna technology in *Single-Input Single-Output* (SISO). The multiple antenna configurations in *MIMO* can overcome the detrimental effects of multi-path and fading, trying to achieve high data throughput in limited bandwidth channels. For example, in the 4×4 MIMO, four independent data streams can be multiplexed and transmitted simultaneously with the *spatial division multiplexing* (*SDM*), to

Table 2.4: Effects of channel bandwidth and spatial stream's selection towards IEEE 802.11n's throughput.

Stream number	Bandwidth	
	20 MHz	40 MHz
1 Stream	72.2$Mbps$	150$Mbps$
2 Streams	144.4$Mbps$	300$Mbps$
3 Streams	216.7$Mbps$	450$Mbps$
4 Streams	288.9$Mbps$	600$Mbps$

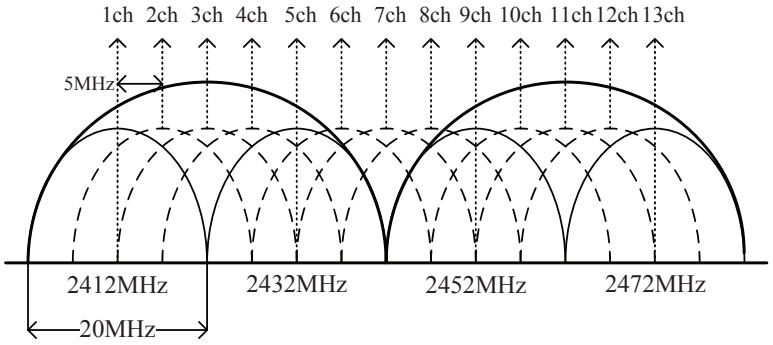


Figure 2.7: IEEE 802.11n channel bonding concept.

speed up the transmission capacity by quadruple as shown in Figure 2.8. When the *space-*

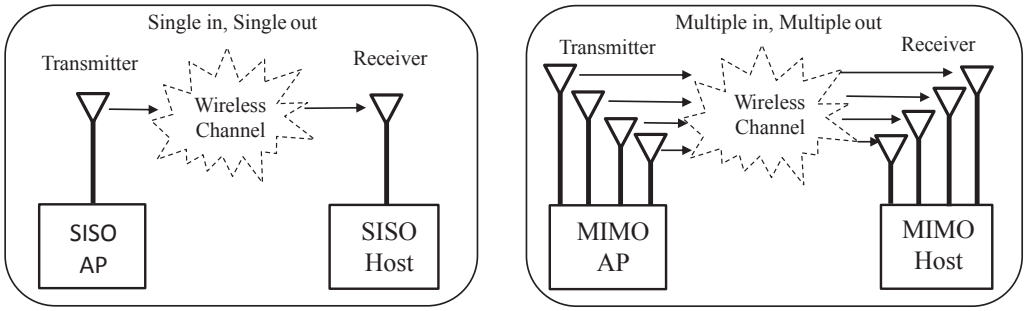


Figure 2.8: Comparison between SISO and 4×4 MIMO technology.

time block coding (*STBC*) is adopted in the 4×4 MIMO link, the sender can transmit four copies of the data stream over four antennas to improve the reliability and the effective range of data transmissions.

- *Frame Aggregation:*
  IEEE 802.11n also provides the performance improvement through the frame aggregation in the MAC layer, besides MIMO. The frame aggregation can transmit multiple frames by one big frame with a single pre-ample and header information to reduce the overhead by them. IEEE 802.11n introduces the *Aggregation of MAC Service Data Units (A-MSDUs)* and *Aggregation of MAC Protocol Data Units (A-MPDUs)*. Frame aggregation is a process of packing multiple A-MSDUs and A-MPDUs together to reduce the overheads and average them over multiple frames, thereby increasing the user level data rate [44].

- *Modulation and Coding Scheme:*
  Various modulation, error-correcting codes are used in IEEE 802.11n, represented by a Modulation and Coding Scheme (MCS) index value, or *mode*. IEEE 802.11n defines 31 different modes and provides the greater immunity against selective fading by using the Orthogonal Frequency Division Multiplexing (OFDM). This standard increases the number of OFDM sub-carriers of 56 (52 usable) in *High Throughput (HT)* with $20MHz$ channel width and 114 (108 usable) in HT with $40MHz$. Each of these sub-carriers is modulated with BPSK, QPSK, 16-QAM or 64-QAM, and Forward Error Correction (FEC) coding rate of 1/2, 2/3, 3/4 or 5/6 [44].

2.4 Heterogeneous Access Points

In this section, we introduce three different types of APs that we use in our active AP configuration algorithm. We also discuss the characteristics and the speed difference of these APs.

A DAP is a wireless AP that adopts the IEEE 802.11n wireless protocol and connects PCs to the Internet. A commercial DAP using IEEE 802.11n has the coverage radius of around $110m$ and the transmission speed around $120Mbps$. However, the transmission speed varies significantly depending on the environment including obstacles, channel interferences, number of antennas and placement heights of APs.

A VAP is a software-based router using a personal computer (PC) with either Windows or Linux for the operating system. Several Internet connection mediums including wired, wireless, or cellular communications are available for the VAP. A network device can connect to a VAP the same way as it does to a conventional DAP. Most VAPs support the IEEE 802.11n protocol [45] with a maximum of $54Mbps$ transmissions.

While, DAP and VAP use wired Ethernet to access the Internet, the MAP is a device that connects to the Internet through 3G/4G wireless technology, e.g., smart phone. For such portable devices, the power supply is unnecessary because of the built-in battery. With the rapid development of the cellular technology, most MAPs support the IEEE 802.11n protocol, in which the transmission speed is capped to around $30Mbps$ due to the bottleneck in the cellular network[1].

2.5 Linux Tools for Wireless Networking

As an open-source operating system, Linux has been used as a platform to implement new algorithms, protocols, methods, and devices for advancements of wireless networks [46]. In this section, we give the overview of the Linux tools and software used for the measurement performed throughout the thesis and the implementation of the *elastic WLAN system*.

2.5.1 'arp-scan' - to Explore Currently Active Devices

arp-scan [47] is a command-line tool using the ARP protocol to discover and fingerprint the IP hosts in the local area network. This tool can discover all the devices using IP addresses in the network. *arp-scan* works on an IEEE 802.11 wireless (WiFi) network as well as a wired Ethernet,

[1]We should note that the DAPs and the VAPs share the gateway to the Internet. The bandwidth of this gateway becomes the total available bandwidth for the whole WLAN.

where the wireless network uses the same data-link protocol. In the Linux operating system, *arp-scan* can be installed by downloading the source code from [48] or using the following command:

```
$ sudo apt-get install arp-scan
```

The simplest command to scan the network using *arp-scan* is given by:

```
$ arp-scan --interface=eth0 --localnet
```

–interface=eth0 represents the interface to be used in scanning devices. The use of *–localnet* makes *arp-scan* scan all the possible IP addresses in the network that are connected to this interface, which is defined by the interface IP address and net mask. The name of the network interface depends on the operating system, the network type (Ethernet, wireless etc.), and the interface card type. Here, the interface name *eth0* is used as an example.

2.5.2 'nm-tool' - to Collect Host Information

nm-tool provides the information of the devices in the local area network including the wireless network [49, 50]. In our design, we use *nm-tool* to get the information for the host such as the currently associated AP, the associable APs, and the receiving *signal strength* from each AP. *nm-tool* is installed as a part of the *NetworkManager* package [51] in the Linux operating system, which is usually installed by default on the Ubuntu distribution. It can be installed using the following command manually:

```
$ sudo apt-get install network-manager
```

The simple way to run *nm-tool* is:

```
$ nm-tool
```

2.5.3 'hostapd' - to Make AP-mode Linux-PC

hostapd is a Linux tool for the AP and the authentication server. It implements IEEE 802.11 AP managements, along with other IEEE 802.1X protocols and security applications. In the Linux operating system, *hostapd* can be installed by downloading the source code from [52] or using the following command:

```
$ sudo apt-get install hostapd
```

After installing this tool in a Linux PC that contains WLAN driver that supports the AP mode, it can be configured to create a command-line based AP in the Linux-PC. The *hostapd* can be started or stopped by the following commands:

```
$ sudo /etc/init.d/hostapd start
$ sudo /etc/init.d/hostapd stop
```

2.5.4 'ssh' - to Remotely Execute Command

ssh is an abbreviation of *Secure Shell* that is a cryptographic network protocol to securely initiate a shell session on a remote machine [53, 54]. It is operated in two parts: *SSH client* and *SSH server*, and establishes a secure channel between them over an insecure network. The open source version of *ssh* is *OpenSSH* [55] that can be installed using the following command [56]:

```
$ sudo apt-get install openssh-server openssh-client
```

The following list shows an example to remotely execute *nm-tool* on a remote host through the network using *ssh* [53, 54, 57]:

```
$ ssh username@192.168.1.31 'nm-tool'
username@192.168.1.31's password:
```

Here, 192.168.1.31 represents the IP address of the remote host.

2.5.5 'nmcli' - to Change Associated AP

nmcli [58, 59] is a command-line Linux tool to manage, configure, and control the powerful *NetworkManager* package. *nmcli* is pre-included in the *NetworkManager* package. *nmcli* is used to associate a host with an AP through the following command line:

```
$ sudo -s nmcli dev wifi connect NewSSID password PASSWORD
```

The above command connects the host to the AP specified with *NewSSID* using the corresponding password *PASSWORD* of the AP.

2.5.6 'iwconfig' - to Collect Information of Active Network Interface

iwconfig [60] is a command-line Linux tool to display and change the parameters of the active network interface for wireless operations. It can also be used to display the wireless network parameters and statistics. *iwconfig* is usually installed by default in the Ubuntu distribution. It can also be installed manually using the following command:

```
$ sudo apt-get install wireless-tools
```

The following list shows the use of *iwconfig* to display the information of the currently associated AP using the network interface *wlan0*:

```
$ iwconfig wlan0
```

2.5.7 'iperf' - to Measure Link Speed

iperf [24] is a software to measure the available throughput or bandwidth on IP networks. It supports both TCP and UDP protocols along with tuning various parameters related to timing and buffers, and reports the bandwidth, the loss, and other parameters. *iperf* is usually installed by default in the Ubuntu distribution. It can also be installed manually using the following command:

```
$ sudo apt-get install iperf
```

To measure the TCP throughput between two devices using *iperf*, one of them uses the server-mode and the other one uses the client-mode, where packets are transmitted from the client to the server. The *iperf* output contains the time-stamped report of the transmitted data amount and the measured throughput. The following list shows the typical use of *iperf* on the server and client side for throughput measurement:

```
$ iperf -s  //server side
$ iperf -c 172.24.4.1 //client side
```

Here, 172.24.4.1 represents the IP address of the server. In this thesis, we use *iperf* to measure the throughput between an AP and a host through the *IEEE802.11n* protocol.

2.6 Summary

In this chapter, we presented various wireless network technologies, key features of IEEE 802.11n protocols, heterogeneous WLAN AP devices and Linux tools which are adopted in this thesis for our experiments as well as simulations. In the next chapter, we will review our previous related studies.

Chapter 3

Review of Previous Studies

In this chapter, we briefly overview our previous studies related to this thesis. Firstly, we review the elastic WLAN system. Secondly, we review the study of the throughput estimation model for the IEEE 802.11n link in WLAN. Thirdly, we review the study of the active AP configuration algorithm for the elastic WLAN system. Finally, we review the implementation details of the elastic WLAN system testbed using Raspberry Pi APs.

3.1 Elastic WLAN System

In this section, we review the elastic WLAN system, which can dynamically optimize the network configuration by activating/deactivating APs, according to traffic demands and network conditions.

3.1.1 Overview

Currently, WLANs have been increasingly installed in business organizations, educational institutions, and public places like buses, airplanes, or trains. In these cases, unplanned or independently controlled APs can lead to problems resulting in performance degradations and/or wastages of energy. In one hand, WLANs can suffer from over-allocation problems with redundant APs that have overlapped coverage areas. On another hand, WLANs can be overloaded with hosts suffering from low performances. Therefore, WLANs should be adaptive by changing the allocations of APs and the associations of hosts to the APs, according to the network traffic demands and conditions. To realize this goal, we have studied the elastic WLAN system.

3.1.2 Related Works in Literature

In this section, we show our brief surveys to this work.

In [61], Lei et al. proposed a campus WLAN framework based on the software defined network (SDN) technology.

In [62], Luengo et al. also proposed a design and implementation of a testbed for integrated wireless networks based on SDN. Although this framework is flexible to design and manage, it requires SDN-enabled devices and network virtualizations.

In [63][64], Sukaridhoto et al. proposed a Linux implementation design using OpenFlow of the fixed backoff-time switching (FBS) method for the wireless mesh network. Their implementation requires Linux driver kernel modifications and specific WLAN hardware interfaces.

In [65], Ahmed et al. describe the important design issues in preparing a large-scale WLAN testbed for evaluation of centralized control algorithms and presented experimental results. They did not analyzed the power-saving and adaptive control mechanism of centralized WLANs which is one of the the main purposes of our research.

In [66], Debele et al. proposed a Resource-on-Demand (RoD) strategy for energy-saving in dense WLANs where they analyzed user behavior in the network and formulated the stochastic characteristics. Our system also adapts with the user demand and frequency, and moreover we present a implementation of our system in real networks.

3.1.3 Design and Operational Flow

Figure 3.1 illustrates an example topology of the elastic WLAN system. The elastic WLAN system dynamically controls the number of active APs in the network by activating or deactivating the installed APs according to traffic demands and network conditions. The implementation of the

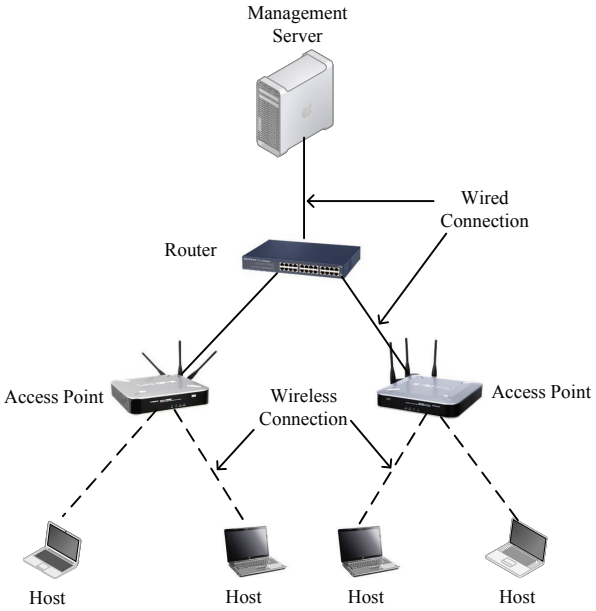


Figure 3.1: Design of Elastic WLAN system.

elastic WLAN system adopts the *management server* to manage the necessary information for the system and control the APs and the hosts. This server not only has the administrative access rights to all the devices in the network, but controls the whole system through the following three steps:

1. The server explores all the devices in the network and collects the requisite information for the active AP configuration algorithm.

2. The server executes the active AP configuration algorithm using the inputs derived in the previous step. The output of the algorithm contains the list of the active APs, the host associations, and the assigned channels.

3. The server applies this output to the network by activating or deactivating the specified APs, changing the specified host associations, and assigning the channels.

3.2 Throughput Estimation Model for IEEE 802.11n Protocol

The link speed or throughput between any AP and host depends on a variety of factors such as the modulation and coding scheme, the transmission power, the transmission distance, and obstacles. The theoretical computation of the accurate link speed is challenging [9][67]. In this section, we review the throughput estimation model for IEEE 802.11n link in WLAN.

3.2.1 Overview of Model

This model estimates the link speed or data throughput of an IEEE 802.11 link in WLAN. It has two main steps to estimate the throughput between a source and a destination node. In the first step, it estimates the *receiving signal strength* (*RSS*) at the host by using the *log distance path loss model*. In the second step, it converts the estimated *RSS* into the corresponding throughput by the *sigmoid function* [11]. Both functions have several configuration parameters that can affect the estimation accuracy, which depends on link specifications and network field environments.

3.2.2 Receiving Signal Strength Estimation by Log-Distance Path Model

The signal strength is estimated by the log-distance path loss model and it considers the multi-path effects by considering both the direct signal along the line-of-sight (*LoS*) and the indirect signal through the non line-of-sight (*NLoS*) path in indoor environments. First, the Euclidean distance *d (m)* is calculated for each link (AP/host pair) by:

$$d = \sqrt{(AP_x - H_x)^2 + (AP_y - H_y)^2} \tag{3.1}$$

where AP_x, AP_y and H_x, H_y does the x and y coordinates for the AP and the host respectively. Then, the *RSS*, P_d (dBm), at the host is estimated using the *log distance path loss model* by considering the distance and the obstacles loss between end nodes[67]:

$$P_d = P_1 - 10\alpha \log_{10} d - \sum_k n_k W_k \tag{3.2}$$

where P_1 represents the signal strength at $1m$ from the AP (source), α is the path loss exponent, d (m) does the link distance from the AP, n_k does the number of the type-k walls along the path between the AP and the host, and W_k does the signal attenuation factor (dBm) for the type-k wall in the environment. The estimation accuracy of *RSS* relies on the parameter values, which depend on the propagation environment [67].

3.2.3 Throughput Estimation by Sigmoid Function

From the *RSS*, the throughput S $(Mbps)$ of the link is derived using following the *sigmoid function*:

$$S = \frac{a}{1 + e^{-(\frac{(P_d+120)-b}{c})}} \tag{3.3}$$

where a, b, and c are the constant parameters of the sigmoid function that should be optimized by parameter optimization tool [12]. The assumption of this sigmoid function for the throughput estimation is based on our real-world measurement results which clearly reflects the relationship between the RSSs and the estimated throughput.

3.2.3.1 Multi-Path Effect Consideration

Due to the *multipath effect* in indoor environments, the receiver may receive the *indirect signal* through an *NLoS* path that arrives after reflected/diffracted at some points on walls or obstacles, in addition of the *direct signal* along the *LOS* path. When the direct signal passes through multiple walls, the *RSS* at the receiver becomes much weaker than that of the indirect signal if it passes through much fewer walls. The throughput estimation model considers this multi-path effect by selecting a *diffraction point* for each AP/host pair such that it is located on the wall in the same room as the host and *RSS* of the AP is largest. The indirect signal reaches the host through this diffracting point after the signal is attenuated there. The *RSS* through indirect path, P_{ind} is calculated by:

$$P_{dif} = P_1 - 10\alpha \log_{10} r - \sum_k n_k W_k \tag{3.4}$$

$$P_{ind} = P_{dif} - 10\alpha \log_{10} t - W_{dif} \tag{3.5}$$

where P_{dif} represents the *RSS* at the diffraction point, r (m) is the distance between the AP and the diffraction point, t (m) does the distance between the diffraction point and the host, and W_{dif} (dBm) does the attenuation factor at the diffraction point. For the estimated RSS at the receiver, the larger one between the direct or indirect signals is selected.

3.3 Active AP Configuration Algorithm

In this section, we review the active AP configuration algorithm for the elastic WLAN system that optimizes the number of active APs and the host associations [9, 10].

3.3.1 Problem Formulation

The active AP configuration problem for this algorithm is formulated as follows:

1. Inputs:
    - Number of hosts: H
    - Number of APs: $N = N^D + N^V + N^M$ where N^D, N^V, and N^M represent the number of DAPs, VAPs, and MAPs respectively.
    - Link speed between AP_i and $host_j$ for $i = 1$ to N, $j = 1$ to H: s_{ij}, where the link speed can be estimated by the model in [11].
    - Minimum link speed for association: S
    - Number of non-interfered channels: C
2. Outputs:
    - Set of active APs
    - Set of hosts associated with each active AP
    - Channel assigned to each active AP
3. Objectives:

- To minimize E_1.
- Holding the first objective, to maximize E_2.
- Holding the two objectives, to minimize E_3 for channel assignments.

Let, E_1 represents the number of active APs (DAPs, VAPs, and MAPs) in the network:

$$E_1 = E_1^D + E_1^V + E_1^M \tag{3.6}$$

where E_1^D represents the number of active DAPs, E_1^V does the number of active VAPs, and E_1^M does the number of active MAPs respectively.
The transmission delay of the jth AP can be defined by:

$$T_j = \sum_{k \in \mathcal{P}_j} \frac{D_k}{s_{jk}} \tag{3.7}$$

where D_k represents the traffic of the kth host, s_{jk} does the link speed between the jth AP to the kth host, and $\mathcal{P}_j$ does the set of the hosts associated with the jth AP. Then, the average throughput TH_{ij} of the ith host associated with the jth AP can be estimated by:

$$TH_{ij} = \frac{D_i}{T_j} = \frac{D_i}{\sum\limits_{k \in \mathcal{P}_j} \frac{D_k}{s_{jk}}} \tag{3.8}$$

Since the real traffic of each host is unpredictable, we assume the identical traffic for every host, which can be represented by the unit traffic for the sake of simplicity. Then, the average host throughput for AP_j is given by:

$$TH_j = \frac{1}{\sum\limits_{k \in \mathcal{P}_j} \frac{1}{s_{jk}}} \tag{3.9}$$

If $TH_j \geq G$, the minimum host throughput constraint is satisfied, where G represents the threshold for this constraint. Then, the second objective function E_2 is defined to maximize the *minimum average host throughput* for the bottleneck, AP which is given by:

$$E_2 = \min_j \left[TH_j \right] \tag{3.10}$$

E_3 signifies the total interfered communication time:

$$E_3 = \sum_{i=1}^{N} [IT_i] = \sum_{i=1}^{N} \left[\sum_{\substack{k \in I_i \\ c_k = c_i}} T_k \right] \tag{3.11}$$

where IT_i represents the *interfered communication time* for AP_i, T_i does the *total communication time* for AP_i, I_i does the *set of the interfered APs* with AP_i, and c_i does the *assigned channel* to AP_i. They are given by follows:

- T_k is given by the sum of the time that is required to transmit one bit data between AP_k and its each associated host.

- I_i represents the set of the indices of the APs that are interfered with AP_i if they are assigned the same channel.
- c_k signifies the channel assigned to AP_k by the active AP configuration algorithm.
- IT_i is given by the sum of the total communication time for the APs that are interfered with AP_i.

4. Constraints:

   - Minimum host throughput constraint: each host in the network must enjoy the given threshold G on average when all the hosts are communicating simultaneously.
   - Bandwidth limit constraint: the bandwidth of the wired network to the Internet must be less than or equal to the total available bandwidth of the network B^a.
   - Channel assignment constraint: each AP must be assigned one channel between 1 and C.

3.3.2 Algorithm Procedure

The active AP configuration algorithm consists of three phases: the *active AP and associated host selection* phase, the *channel assignment* phase, and the *channel load averaging* phase.

3.3.2.1 Active AP and Associated Host Selection Phase

In this phase, the set of the active APs and their associated hosts are selected. This phase comprises following eight steps:

1. **Preprocessing**
   The link speed for each possible pair of an AP and a host is estimated with the measurement or the throughput estimation model [11]. Then, this step initializes the variables for the following steps:

   (a) For each AP, make a list of hosts that can be associated with this AP, where the throughput of the link between a host and an AP is S or larger, it can be associated. This list is called the *associable host list for APs*.

   (b) Initialize each AP as the *non-active* AP. Initially, only the DAPs are selected as *candidate APs*.

2. **Initial Solution Generation**
   An initial solution to the cost function E_1 is derived using a greedy algorithm [68], which repeats the following procedures:

   (a) Select the AP that can be associated with the maximum number of non-associated hosts.

   (b) Activate this AP and increment E_1 by one.

   (c) Update the number of non-associated hosts in the host list for APs.

3. **Host Association Improvement**
   The cost function E_2 is calculated for the greedy solution using Eq. (3.10). Then, this solution is improved by repeating the following procedure:

(a) Find the AP that gives the lowest host throughput in Eq. (3.10).

(b) Select one host randomly from the associated hosts with this AP, and associate it with another active AP that is selected randomly. Then, calculate E_2.

(c) Keep the new association only if this new E_2 is higher than the previous E_2. Otherwise, return to the previous association.

4. **AP Selection Optimization**
The cost functions E_1 and E_2 are further jointly improved in this step under the constraints mentioned before by the *local search* [69]. This local search repeats the following three procedures:

   (a) If the current solution satisfies the *minimum host throughput constraint*, it seeks to reduce the number of active APs E_1 by deactivating an active AP. In the implementation, it repeats to 1) randomly select an active AP and deactivate it, 2) apply *Host Association Improvement*, and 3) check the feasibility of this deactivation.

   (b) Otherwise, it seeks to improve the *minimum average host throughput* E_2 with the same number of active APs by changing the active AP. In the implementation, it repeats to 1) randomly select a non-active AP and activate it, 2) apply *Host Association Improvement*, and 3) check the possibility of deactivating another active AP.

   (c) If (b). is not achieved, it seeks to satisfy the *minimum host throughput constraint* by increasing the number of active APs while improving the *minimum average host throughput*. In the implementation, it 1) randomly selects a non-active AP and 2) applies *Host Association Improvement*.

5. **Link Speed Normalization**
The fairness criterion will be applied when the total expected bandwidth exceeds B^a. Generally, the link speed is normalized as:

   (a) Calculate the expected total bandwidth B^e by the summation of the throughputs of all the APs.

   (b) If $B^e > B^a$, adjust each AP-host link speed as:

$$\hat{s}_{ij} = s_{ij} \times \frac{B^a}{B^e} \tag{3.12}$$

   where $\hat{s}_{ij}$ is the normalized link speed.

6. **Termination Check**
The algorithm is terminated when either of the following conditions is satisfied:

   (a) The *minimum host throughput constraint* is satisfied.

   (b) All the APs in the network have been activated.

7. **Additional VAP Activation**
If all the DAPs become active but the *minimum host throughput constraint* is still not satisfied, VAPs are newly selected as candidate APs. A VAP is slower than a DAP, but faster than a MAP. The locations of hosts are considered as the locations of the candidate VAPs, because user hosts may be used for VAPs. Then, it returns to *AP Selection Optimization* step.

8. **Additional MAP Activation**
   If all the DAPs and VAPs become active but the *minimum host throughput constraint* is still not satisfied, MAPs are newly selected as candidate APs. A MAP is the slowest among the three AP types. The locations of hosts are considered as the locations of the candidate MAPs, because users may have MAPs. Then, it returns to *AP Selection Optimization* step.

3.3.2.2 Channel Assignment Phase

In this phase, the assigned channels to the active APs are selected and it has the following four steps:

1. **Preprocessing**
   The interference and delay conditions of the network are represented by a graph.

   (a) Construct the *interference graph*, $G = (V, E)$, from the APs and the hosts, where the vertex V represents the set of APs and the edge E presents the existence of the interference between two APs. $e(i, j) \in E$ if AP_i is interfered with AP_j in the network.

   (b) Calculate the *communication time* for each AP. The communication time T_i for AP_i is defined as the total time when the AP transmits 1-bit to all the associated hosts. It is given by:

   $$T_i = \sum_j \frac{1}{sp_{ij}} \tag{3.13}$$

   where sp_{ij} represents the link speed between AP_i and $host_j$.

   (c) Calculate the *neighbor interfered communication time* for each AP. The neighbor interfered communication time NT_i for AP_i is given by:

   $$NT_i = \sum_{e(i,k)=1} T_k \tag{3.14}$$

2. **Interfered AP Set Generation**
   The set of APs that are interfering with each other is found for each AP.

   (a) Sort the APs in descending order of NT_i, where the tie-break is resolved by T_i.

   (b) Find the interfered AP set for each AP by repeating the following steps:

      i. Initialize the interfered AP set by $I_i = \{i\}$ for AP_i.

      ii. Expand I_i by examining the APs in sorted order in a) whether the AP is interfered with each AP in I_i. If so, include this AP, AP_j, into I_i, i.e. $I_i = I_i \cup \{j\}$.

   (c) Calculate the total interfered communication time AT_i for AP_i, which is given by:

   $$AT_i = \sum_{k \in I_i} T_k \tag{3.15}$$

3. **Initial Solution Construction**
   Then, an initial solution is derived with a greedy algorithm.

   (a) Sort the APs in descending order of the total interfered communication time AT_i, where the tie-break is resolved by NT_i.

(b) Assign a channel c to AP_i such that the interfered communication time IT_i is minimized if it is assigned. IT_i is given by:

$$IT_i = \sum_{\substack{k \in I_i \\ c_k = c_i}} T_k \tag{3.16}$$

where c_k represents the assigned channel to AP_k.

(c) Repeat 2) until each active AP is assigned to one channel.

(d) Calculate the cost function E_3 using Eq. (3.11) and save this initial solution as the best solution E_3^{best}.

4. **Solution Improvement by Simulated Annealing**
Finally, the initial solution is improved by repeating the following *simulated annealing (SA)* procedure with the constant *SA temperature* T^{SA} at the *SA repeating times* R^{SA}, where T^{SA} and R^{SA} are given algorithm parameters:

(a) Randomly select one AP and one new channel for the channel change trial.

(b) Calculate the interfered communication time IT_i after assigning this new channel by:

(c) Calculate E_3^{new} using Eq. (3.11) for the new channel assignment, and $\Delta E_3 = E_3^{new} - E_3$.

(d) If $\Delta E_3 \leq 0$, accept the new channel assignment, and address this new solution as the best one.

(e) Otherwise, generate a 0-1 random number, *rand*, and if $rand \leq \frac{-\Delta E_3}{T^{SA}}$, then accept the new channel assignment.

3.3.2.3 Channel Load Averaging Phase

After the channel assignment using the limited number of channels, the total loads may be imbalanced depending on different channels that are assigned to the APs. In this phase, the channel load is averaged by changing associated APs for hosts. It has four steps as follows:

1. **Preprocessing**
The *AP flag* for each AP is initialized with OFF to avoid processing the same AP.

2. **AP Selection**
One AP is selected to move its associated host to a different AP that is assigned a different channel.

(a) Terminate the procedure if each AP has ON AP flag.

(b) Initialize the host flag by OFF for each host.

(c) Select one AP, say AP_i, that satisfies the two conditions:

i. The AP flag is OFF.

ii. The interfered communication time IT_i is largest among the OFF APs.

(d) Set the AP flag ON.

3. **Host Selection**
Then, one host associated with AP_i is selected for the associated AP movement.

(a) Select one host, say H_j, that satisfies the four conditions:

i. The host flag is OFF.

ii. The host is associated with AP_i.

iii. The host can be associated with another AP assigned a different channel from AP_i, or is located out of the interference range of AP_i.

iv. The link speed with AP_i is the smallest among the hosts satisfying (a)–(c).

(b) If one host is selected, set the host flag ON.

(c) Otherwise, return to *AP Selection* for the new AP selection.

4. **Association Change Application**
Finally, the new associated AP is selected for H_j.

(a) Select the AP that has the largest link speed among the APs that are assigned to a different channel from AP_i and can be associated with H_j.

(b) Calculate the new cost function E_3^{new} with Eq. (3.11) if H_j is associated with this AP.

(c) If E_3^{new} is equal to or smaller than the previous E_3, accept the new association, and return to *Host Selection*.

(d) Otherwise, select another AP that has the next largest link speed, and return to 3).

(e) If no such AP exists, return to *Host Selection* for the new host selection.

3.4 Testbed Implementation Using Raspberry Pi

In this section, we describe the testbed implementation of the elastic WLAN system using Raspberry Pi and Linux PCs. *Raspberry Pi* is a small-size low-cost computer, and has become popular in academics and industries around the world. Therefore, the use of Raspberry Pi in the elastic WLAN system is significant for its disseminations in developing countries.

3.4.1 Implementation Environment/Platform

As the initial implementation platform of our elastic wireless LAN system, we choose the Linux environment that has been used as a platform to implement new algorithms, protocols, methods, and devices for advancements of wireless networks, because of being an open-source operating system. Linux environment has a lot of open source tools to use. Most of them are easily configurable and have flexibility to use and integrate with other tools [46]. On the other hand, while searching for the network configuration and management tools in Windows operating system, we found most of them are less flexible and less configurable, and not open source. Currently, we are using *Ubuntu* for our implementation platform as the popular distribution of Linux environment for general-purpose users. Implementations of the elastic WLAN system on various platforms will be in our future studies. The device environments and software in Table 3.1 are used for the testbed implementation of the system. The IEEE 802.11n protocol is used for any communication link with the channel bonding.

Table 3.1: Device environment and software in testbed.

devices and software		
server PC	OS	Ubuntu LTS 14.04
	model	Lesance W255HU
	Processor	Intel(R), Core(TM)-i3
client PC	OS	Ubuntu LTS 14.04
	Model	Fujitsu Lifebook S761/C/SSD
	Processor	Intel(R), Core-i5
access point	OS	Raspbian
	Model	Raspberry Pi 3
	Processor	1.2 GHz
software/tools	openssh	to access remote PC and AP
	hostapd	to prepare and configure AP
	nmcli	for association change
	nm-tool	to measure signal strength
	arp-scan	to discover active network devices

3.4.2 System Topology

Figure 3.2 shows the simple network topology of the elastic WLAN system. *Raspberry Pi* is used for the AP and a *Linux laptop PC* is for the server and the host. The server can manage and control all the APs and the hosts by using the administrative access to them. The APs are connected to the server through wired connections. The hosts and the APs are connected through wireless connections.

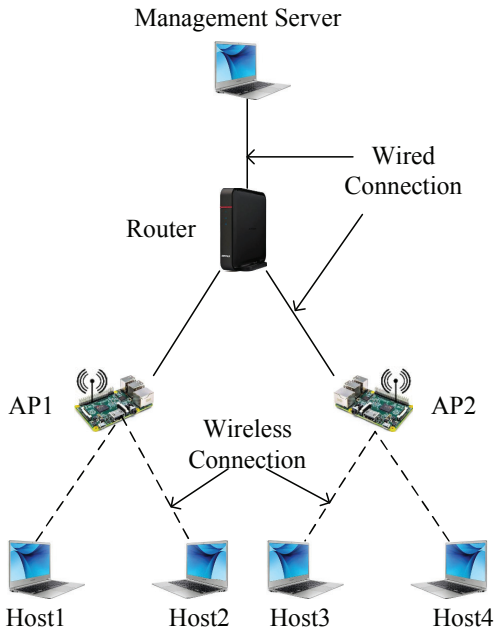


Figure 3.2: Elastic WLAN system topology.

3.4.3 AP Configuration of Raspberry Pi

This section explains how to configure *Raspberry Pi* for AP using *hostapd* daemon [70, 71].

1. Install the *hostapd* using the following command:

   ```
   $ sudo apt-get install hostapd
   ```

2. Modify the configuration file */etc/hostapd/hostapd.conf* with the desired SSID and PASSWORD. A simple example of the configuration file is given below:

   ```
   interface=wlan0
   ssid=SSID
   channel=1
   wpa_passphrase=PASSWORD
   ```

3. Uncomment and set *DAEMON_CONF* to the absolute path of a hostapd configuration file to start hostapd during system boot:

   ```
   DAEMON_CONF="/etc/hostapd/hostapd.conf"
   ```

4. Setup the *wlan0* interface to have a static IP address in the network interface configuration file */etc/network/interfaces*. An example of the interface file is given below:

   ```
   auto wlan0
   iface wlan0 inet static
   address 192.168.1.11
   netmask 255.255.255.0
   network 192.168.1.0
   ```

5. Finally, install the DHCP server for assigning the dynamic IP addresses to the hosts.

3.4.4 Execution Flow of Elastic WLAN

Figure 3.3 shows the execution flow of the elastic WLAN system testbed implementation.

3.4.4.1 Generation of input for Active AP Configuration Algorithm

In this step, the server explores all the connected device to the network and generates the input for the active AP configuration algorithm using the following procedure:

1. The server explores all the connected device to the network using *arp-scan* [48]. The command is given below:

   ```
   $ sudoarp-scan --interface=eth0 192.168.11.0/24
   ```

   Here, *–interface=eth0* represents the interface and 192.168.11.0/24 is the network IP range to scan. The output consists of the IP and MAC addresses of the hosts and the APs that are available in the network. A simple C program is developed to identify the hosts and APs in this system using the MAC addresses of the devices. After this, the server generates the list of permitted APs and the list of permitted hosts.

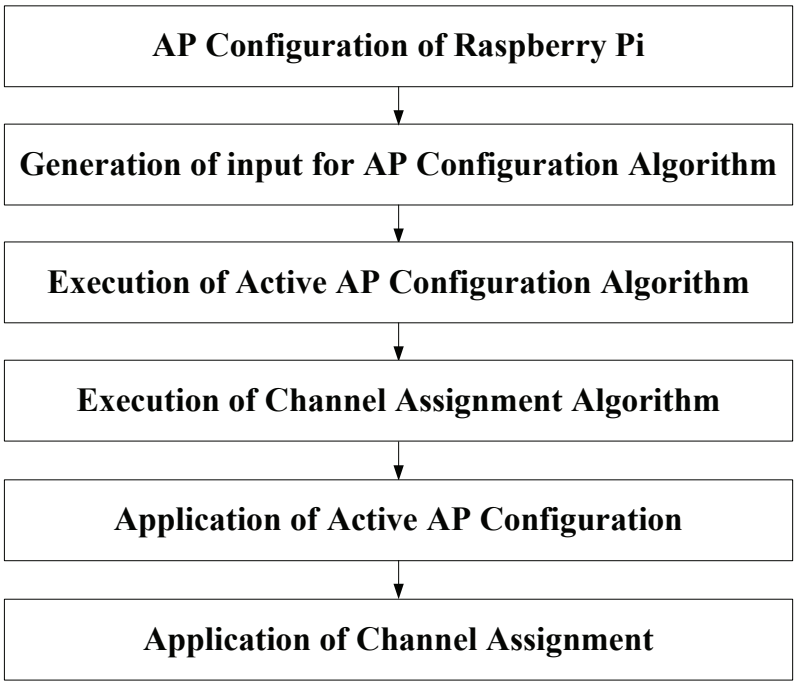


Figure 3.3: Execution Flow of Elastic WLAN system.

2. The following command finds the receiving signal strength of each host from each AP using *nm-tool* [49, 50]. *ssh* protocol is used to execute the command remotely in each host [53, 54, 57].

   ```
   $ sudo nm-tool
   ```

3. After this, the server converts the receiving signal strength to the estimated link speed using the sigmoid function in [72], and generates the input for the active AP configuration algorithm.

3.4.4.2 Execution of Active AP Configuration Algorithm

The active AP configuration algorithm is executed in this step. The following commands compile the program for the active AP configuration algorithm and execute it respectively. The *minimum host throughput constraint* and the *bandwidth limitation constraint* are specified by the user.

```
$ g++ -o apc APConfigurationAlgorithm.cpp
$ ./apc input.txt min_host_throughput bw_limit
```

Here, *input.txt* presents the input file generated in the previous step, *min_host_throughput* does the minimum host throughput constraint, and *bw_limit* does the bandwidth limitation constraint. After this, the list of active APs and their associations with the hosts are obtained.

3.4.4.3 Execution of Channel Assignment Algorithm

The following commands compile the program for the channel assignment extension and execute it respectively.

```
$ g++ -o ca ChannelAssignment.cpp
$ ./ca HostAPassociation.txt num_of_channels
```

HostAPassociation.txt presents the input file to the channel assignment extension that contains the list of active APs and their associations with the hosts, and *num_of_channels* does the number of available channels.

3.4.4.4 Application of Active AP Configuration

The management server applies the output of the two algorithm.

1. The server adjusts the number of active APs according to the algorithm output by activating or deactivating APs in the network. The two commands given below is used to activate and deactivate the Raspberry Pi AP respectively.

   ```
   $ sudo /etc/init.d/hostapd start
   $ sudo /etc/init.d/hostapd stop
   ```

2. The following command connects a host to a new AP using *nmcli* [58, 59]. *NewSSID* represents the new AP for the host and *PASSWORD* does the security key of the AP. The server modifies the AP-host association according to the algorithm output using this command.

   ```
   $ sudo -s nmcli dev wifi connect NewSSID password PASSWORD
   ```

3.4.4.5 Application of Channel Assignment

The server uses the following commands to assign the new channel to the Raspberry Pi AP using *sed* [73]. For this, the server modifies the configuration file */etc/hostapd/hostapd.conf* with the channel number.

```
$ sed -i -e 's/.*channel.*/channel='$NewChannel'/'
/etc/hostapd/ hostapd.conf
$ sudo /etc/init.d/hostapd restart
```

Here, *'s'* represents the substitution command and *NewChannel* does the channel to be assigned in the *hostapd.conf* file of the AP. After the assignment of the new channel, the server restarts it to make the change take effect.

3.5 Summary

In this chapter, firstly, we reviewed the elastic WLAN system. Secondly, we reviewed the previous works of the throughput estimation model in WLAN. Thirdly, we reviewed the active AP configuration algorithm. Finally, we described the implementation details of the elastic WLAN system. In the next chapter, we will present the dynamic AP configuration algorithm for dynamic host associations considering the communicating APs and hosts.

Chapter 4

Dynamic Active AP Configuration

In this chapter, we present the extension of the active AP configuration algorithm for dynamic behaviors of joining and leaving hosts. Firstly, we describe the motivation of our proposal. Secondly, we describe some related works. Thirdly, we present the algorithm extension for join and leave operations considering the currently communicating APs and hosts. Fourthly, we evaluate the proposal through simulations using two networks topologies. Fifthly, we present the implementation of the proposal in the elastic WLAN system testbed. Finally, we evaluate our proposal using testbed experiments.

4.1 Introduction

In Chapter 3, we reviewed the previous studies of the active AP configuration algorithm for the elastic WLAN system. This algorithm can optimize the number of active APs depending on traffic demands and network situations [9, 10]. Unfortunately, this algorithm can find the solution for the fixed user state of the network, although user hosts often repeat joining and leaving the network.

To solve this issue, we propose the extension of the *active AP configuration algorithm* in order to deal with this dynamic nature. By considering the currently communicating hosts and APs, the algorithm can find the new configuration for host joining and leaving operations in the network.

4.2 Motivation

The *active AP configuration algorithm* consists of three phases, namely, the *active AP and associated host selection* phase, the *channel assignment* phase, and the *channel load averaging* phase. In the first phase, the algorithm selects the active APs and the host associations based on the static host information in the system. In this phase, the algorithm selects the minimum number of active APs that can satisfy the predefined throughput constraints on the hosts. In the second phase, it assigns one channel to each active AP from the limited number of the non-interfered channels in order to minimize the overall interference of the network. Finally, in the third phase, the channel loads are averaged by changing associated APs for hosts when their loads are high.

In a real WLAN system, hosts often join and leave from the network and the number of hosts usually fluctuate depending on the time and days of the week. Thus, the network configuration should be adaptive for the dynamic nature of host joining and leaving. Also, some hosts may currently send or receive data to/from the Internet. Thus, their services should not be stopped when a new configuration is composed.

4.3 Drawbacks of the Previous Algorithm

Unfortunately, the previous *active AP configuration algorithm* can find a solution for the fixed user state in the network, although it is usual that users repeat joining and leaving the network. Even for any single join or leave operation, the algorithm may generate a totally new configuration without considering the current state. Thus, it cannot provide continuous services to the currently communicating hosts. The new network configuration must avoid this discontinuity of communication services.

4.4 Contributions

In this chapter, we first propose the extension of the *active AP configuration algorithm* to overcome the discontinuity problem of the algorithm. The algorithm can handle the dynamic host behavior by considering the rapid host join and leave operation. Figure 4.1 shows the dynamic extension of the previous algorithm. It takes the current network association as the input and finds the currently communicating APs and hosts those are exchanging the data using Internet. Then, it finds the joining or leaving hosts in the network. Finally, depending on the join or leave operation, the algorithm finds the new configuration. In this extension, the currently communicating APs and hosts are designated as *communicating APs* and *communicating hosts* for convenience. Then, as the new constraints, any *communicating AP* must not be deactivated and any *communicating host* must not change the associated AP by the algorithm.

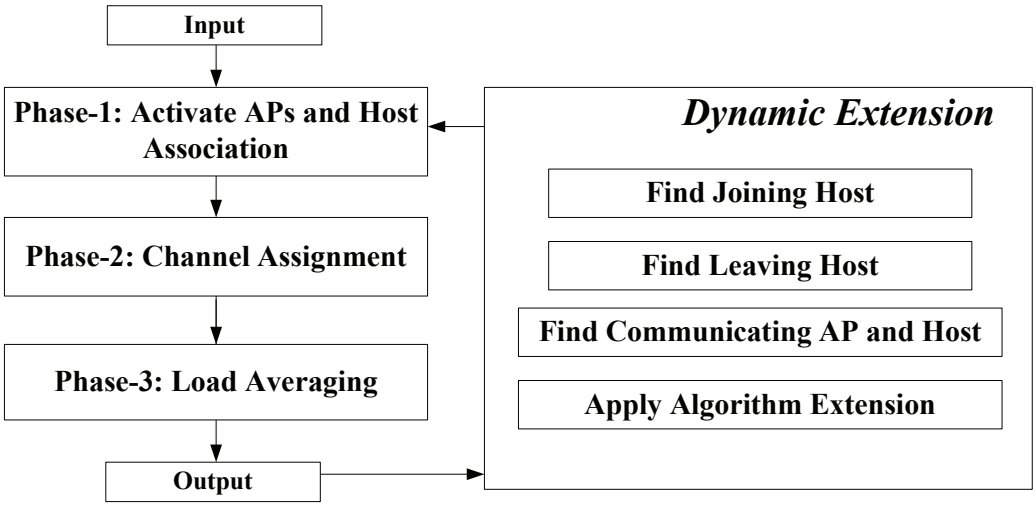


Figure 4.1: Dynamic active AP Configuration Algorithm extension operation flow.

Furthermore, the proposed algorithm extension is implemented on the elastic WLAN system testbed to verify the practicality in the real system. In this testbed implementation, *Raspberry Pi* is adopted for the AP and Linux PC is for the host.

4.5 Related Works

In this section, we briefly introduce related works in literature to this work.

[4] adopts a game theoretic approach to balance the loads among the APs. That is, the users associated with highly congested APs will be moved to less loaded APs for better throughputs. Although this approach can balance the loads among APs, the user movement from one AP to another AP makes it impractical in real applications, because a host has to discover another suitable AP and move to the place, if it doesn't succeed in receiving the required data rate or service from the associated AP.

[74, 75] propose a load balancing approach among APs to mitigate congestions and fair distributions of users. Each host monitors the wireless channel qualities that it experiences from nearby APs and reports them to the network control center that determines the host and AP associations. Since the objective function of the proportional fairness is non-linear, its implementation is much more challenging, where detecting the bottleneck users and finding their normalized bandwidth are NP-hard.

[76] provides an algorithm to place several *extension points* for cellular networks such that the throughput capacity of WLAN is maximized. These extension points act as relay points between the APs and the hosts located at longer distances. Each extension point can receive the packets from the APs and deliver them to the hosts. In a conventional WLAN, the extension point is not used, because the multi-hop communications degrade the performance due to strong interferences among multi-hop links.

[77] presents an AP association approach to improve the network throughput while balancing the loads among APs. The authors investigate and evaluate a measurement driven framework with three objective functions in a dense WLAN; (i) the frequency/channel selection across the APs to minimize the noise or interference between neighboring APs, (ii) the user association by considering the load of the AP and the receiving signal strength, and (iii) the power control for each AP. While each of the three objective functions achieves its optimization objective individually, in certain cases, to apply all of them (channel selection, user association, and power control) will lead to the suboptimal performance. Moreover, the authors found that if all of them are randomly applied, it will degrade the performance. A host is initially associated with the AP that provides the highest *received signal strength indicator (RSSI)*. Then, any host from an overloaded AP is migrated to a lightly loaded AP for balancing the load in the network.

[78] introduces a heuristic algorithm for the AP placement based on the user distribution on an indoor WLAN system. The authors have proposed a fuzzy C-clustering-based AP deployment strategy to maximize the user coverage and balance the load among the APs.

[79] examines an AP selection algorithm to maximize the throughput for a newly joining host in the multi-rate WLAN by moving the host towards the location of the desired AP.

[80–82] focus on the highest RSSI to select the AP association. However, the AP with the highest RSSI can be overloaded by a number of users, and thus, may not ensure the proper load balancing among APs.

[83] presents a smart AP solution to balance the loads across multiple Wi-Fi interfaces. Still, the implementation of this approach is difficult, since each AP must be equipped with multiple interfaces operating on different channels simultaneously.

4.6 Extension of Active AP Configuration Algorithm

In this section, we present the extension of the active AP configuration algorithm to consider the currently communicating APs and hosts.

4.6.1 Definitions of Terms

In the extension, any *communicating host* must not change the associated AP to ensure the continuous Internet service, while any *communicating AP* must not be deactivated. For the algorithm extension, we introduce the following terms to describe the states of the network and simulations:

- **joining host** is a host that newly joins the network to be associated with an active AP for the Internet access. The algorithm must find the associated active AP for the joining host.
- **leaving host** is a host that leaves the network after completing the Internet service.
- **communicating host** is a host that is currently communicating with the Internet. The algorithm cannot change the associated AP to avoid the communication suspension.
- **communicating AP** is an AP that is currently associated with a communicating host. The algorithm cannot stop this AP to avoid the communication suspension.
- **active host** is a host that is currently associated with an active AP.
- **inactive host** is a host that is not associated with an active AP.

4.6.2 Joining Host Operation

First, we present the algorithm extension for one *joining host*. When a new host joins the network, the associated active AP must be selected for the *joining host* such that the constraints in subsection 3.3.1 are satisfied. Here, the selected active AP must satisfy the *minimum host throughput constraint* when the following *normalized link speed* is considered if $B^a < B^e$:

$$\hat{s}_{jk} = s_{jk} \times \frac{B^a}{B^e} \tag{4.1}$$

where B^a and B^e represent the available and expected total bandwidth respectively, s_{jk} represents the link speed between the jth AP and the kth host, and $\hat{s}_{jk}$ does its normalized link speed. Then, the *average host throughput* (TH_j) using the normalized link speeds is calculated for each active AP (jth AP) when the *joining host* is associated with it:

$$TH_j = \frac{1}{\sum_k \frac{1}{\hat{s}_{jk}}} \tag{4.2}$$

where TH_j represents the *average host throughput* for the jth AP. If $TH_j \geq G$, the *minimum host throughput constraint* is satisfied, where G represents the threshold for this constraint. If multiple active APs can satisfy it, the AP that maximizes the E_2 defined in Eq. (3.10) is selected. On the other hand, if any active AP cannot satisfy the constraints for the *joining host*, this host is first associated with the active AP that maximizes the E_2 defined in Eq. (3.10), and the active AP configuration algorithm in the previous chapter is applied from *AP Selection Optimization* (Step (4) in Section 3.3.2.1).

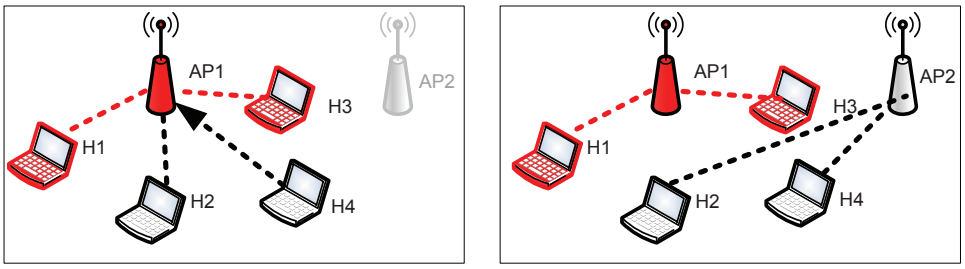
Here, to avoid the suspensions of the currently communicating hosts, the two modifications are incorporated: 1) any *communicating AP* is not deactivated in *AP Selection Optimization*, and 2) any *communicating host* does not change the associated active AP in *Host Association Improvement* (Step (3) in Section 3.3.2.1) and in *AP Selection Optimization* (Step (4) in Section 3.3.2.1).

4.6.3 Leaving Host Operation

Next, we present the algorithm extension for one *leaving host*. When an existing host leaves the network, it is only necessary to remove this *leaving host* from the associated active AP. Then, if the AP is not associated with any host, this AP is deactivated. Otherwise, to minimize the number of active APs and improve the host associations, the active AP configuration algorithm is applied from *AP Selection Optimization* (Step (4) in Section 3.3.2.1) with the two modifications for the *joining host*.

4.6.4 Examples of Host Join Operation

In this paper, it is assumed that at most one host may join or leave the network at each time. First, the host join operation in Figure 4.2 is discussed. When H4 joins the network, it newly activates AP2 to satisfy the minimum host throughput constraint. Then, to balance the loads between AP1 and AP2, H2 changes the associated AP from AP1 to AP2, because H3 cannot change the associated AP as the communicating host.

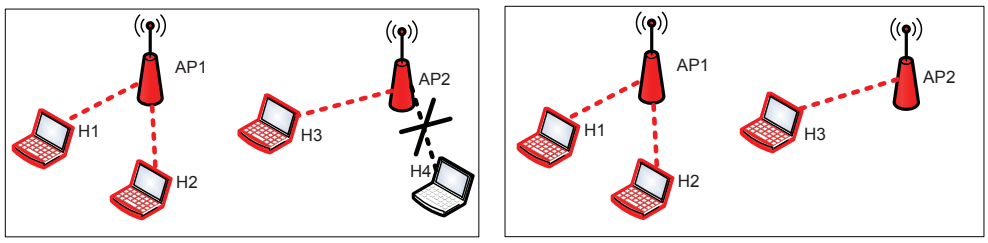


(a) Before H4 join (b) After H4 join

Figure 4.2: Host join operation.

4.6.5 Examples of Host Leave Operation

Then, the host leave operation in Figure 4.3 is discussed. When H4 leaves the network, if H3 changes the associated AP to AP1, it can deactivate AP2 while satisfying the minimum host throughput constraint. However, H3 cannot change the associated AP as the communicating host, and AP2 cannot be deactivated.



(a) Before H4 leave (b) After H4 leave

Figure 4.3: Host leave operation.

4.7 Evaluations by Simulations

In this section, we evaluate the algorithm extension through simulations using the *WIMNET simulator* [16].

4.7.1 WIMNET Simulator

The WIMNET simulator simulates least functions for wireless communications of hosts and *access-points* (*APs*) that are required to estimate the data throughputs and delays of packet transmissions in WLAN. It has originally been developed to evaluate a large-scale *Wireless Internet-access Mesh NETwork (WIMNET)* [16] with reasonable CPU time on a conventional PC. It consists of multiple APs that can form large WLANs and communicate through wireless links.

The WIMNET simulator needs to select several network field parameters such as the AP and host locations, their associations, the existences of different types of wall obstacles, and the communication channels, in order to model various networking scenarios. A sequence of functions such as host movements, communication request arrivals, and wireless link activations are synchronized by a single global clock called a *time slot*. Within an integral multiple of time slots, a host or an AP can complete one-frame transmission and the acknowledge reception. A different transmission rate can be realized by manipulating the time slot length and the number of time slots for one link activation.

4.7.2 Network Environments

In this thesis, we consider the *indoor network environment* for evaluations of our proposal. This environment consists of several rooms that can be used for offices, laboratories, or classrooms in a building. It is noted that WLANs are usually deployed in a building. In such an environment, WLAN users normally use personal computers (PCs) to access the Internet at fixed positions where chairs and tables are available. They access the Internet while sitting on chairs and putting their PCs on tables, because PCs are much larger and heavier than smartphones, and often require the use of both hands. As a result, the mobility of WLAN users is much lower than that of the cellular system users. In such cases, the possible locations of the hosts can be determined by the locations of desks and chairs and can be fixed.

4.7.3 Simulation Platform

In simulations, we adopt the hardware and software in Table 4.1. For the performance compari-

Table 4.1: Simulation environment.

simulator	WIMNET Simulator
interface	IEEE 802.11n
CPU	Intel Core i7
memory	4GB
OS	Ubuntu LTS 14.04

son, the *nearest AP association approach (NAP)* is considered as the simple method for AP-host associations since NAP can offer the fastest link speed between an AP and a host when other hosts

are not associated with the AP, where the active APs are selected by our algorithm. Table 4.2 summarizes the simulation parameters that are used for the simulation using WIMNET simulator.

Table 4.2: Simulation parameters for WIMNET simulator.

parameter	values
packet size	2360 bytes
max. transmission rate	150 Mbit/s
propagation model	log distance path loss model
rate adaptation algorithm	link speed estimation model [9]
carrier sense threshold	−85 dBm
collision threshold	10
RTS/CTS	yes

4.7.4 Host Behavior Model

In our simulations, it is assumed that a host randomly joins and leaves the network by following the *Poission* distribution, and a host continues the communication with the Internet during the time derived by the *Exponential* distribution. In each simulation in this paper, at the first, $N1$ hosts are randomly selected from all the N hosts in the network for $N1 < N$ as *active hosts*, and the original active AP configuration algorithm is applied to the $N1$ active hosts so that the active APs and the associated active hosts are obtained. For each active host, the communication duration time is calculated by the following equation:

$$F(p, \mu) = \frac{-ln(1-p)}{\mu} \tag{4.3}$$

where p is $0-1$ random number and μ is the parameter for the *Exponential* distribution. Then, every time the constant time Td is elapsed, the *communicating host* is maintained if the communication duration time is not over since the arrival to the network. Here, $Td = 10sec$ is used in this paper. Besides, the number of newly joining hosts from *inactive hosts* and the number of leaving hosts from *active hosts* are calculated here by the following procedure:

1. Calculate the probability to select each number k by the *Poission* distribution:

$$F(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}, \tag{4.4}$$

   where λ is the parameter for the *Poission* distribution.

2. Calculate the cumulative probability $C(k, \lambda)$ for each number k by:

$$C(k, \lambda) = \sum_{i=0}^{k} \frac{\lambda^i \cdot e^{-\lambda}}{i!} \tag{4.5}$$

3. Generate a $0 - 1$ random number p.

4. Find k such that $C(k, \lambda) \leq p < C(k+1, \lambda)$.

Then, the corresponding number of joining hosts are randomly selected from *inactive hosts*. For each newly joining host, the communication duration time is calculated by Eq. (4.3).

4.7.5 Evaluation of Small Topology

Firstly, we evaluate our proposal using a small-size network topology.

4.7.5.1 Small Topology

The *small topology* in Figure 4.4 is first considered for simulations, where 30 hosts, 4 DAPs are distributed in the $100m \times 50m$ rectangular area. The circles and squares represent the APs and hosts respectively. The minimum host throughput threshold $G = 10Mbps$ and the bandwidth limit constraint $B^a = \infty$ are examined. As the host behavior model in this topology, $N1 = 18$ hosts are randomly selected as initial *active hosts* from $N = 30$ hosts, and $\lambda = 1$ and $\mu = .02$ are adopted.

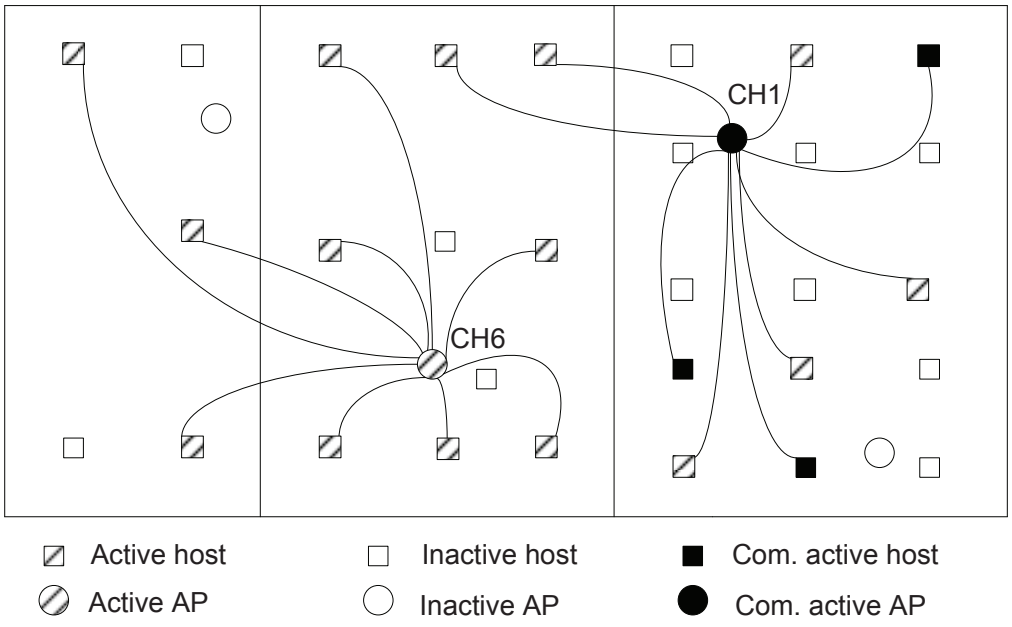
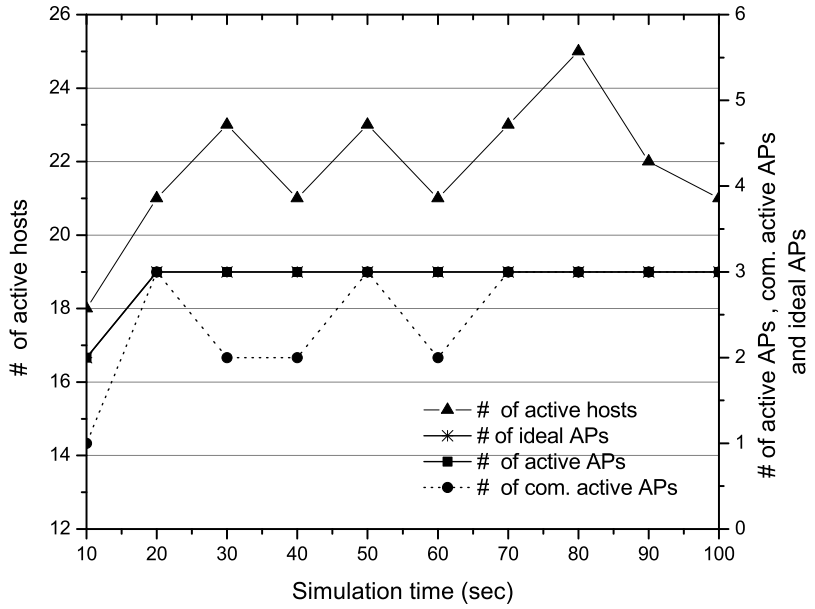


Figure 4.4: Solution for small topology with 30 hosts and 4 APs.

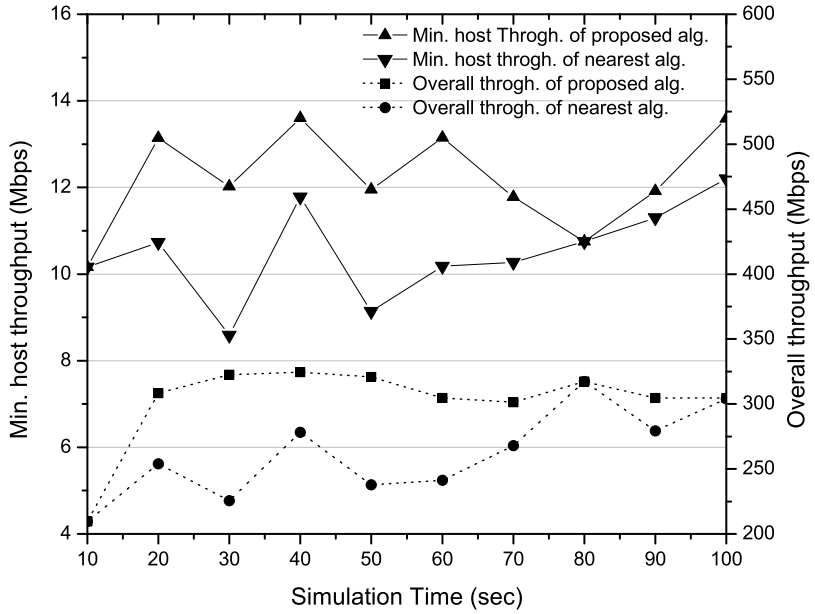
4.7.5.2 Results

Figure 4.5 (a) shows the changes of the number of active hosts and the number of communicating active APs by the host behavior model, the number of active APs by the proposed algorithm, and the least number of APs required to satisfy the *minimum host throughput constraint* (ideal APs), when the simulation time elapsed. The number of ideal APs is calculated by $G \times N1/AT$. AT represents the average total throughput for one AP that is given by the throughput at the average least distance with each host in the network. This graph indicates that the number of active APs is dynamically controlled by the proposed algorithm depending on the number of active hosts.

Figure 4.5 (b) shows the changes of the minimum host throughput and the overall throughput by the WIMNET simulator for the AP-host association by the proposal and for that by the compared NAP. This graph indicates that the minimum host throughput by the proposal is always larger than the threshold $G = 10Mbps$ whereas that by the comparison is sometimes smaller than the threshold. Besides, the overall throughput by the proposal is larger than that by the comparison at most of the simulation time.



(a) Change of active hosts and APs.



(b) Change of min. host and overall throughputs.

Figure 4.5: Performance graph for small topology.

Table 4.3 compares the average minimum host throughput and overall throughput by the two methods. This table also indicates that the performance by the proposal is better than that by the comparison.

4.7.6 Evaluation of Large Topology

Secondly, we evaluate our proposal using a large-size network topology.

Table 4.3: Throughput comparison between two methods for both topology.

instance	small topology		large topology	
method	proposed	compare	proposed	compare
ave. min. host throughput (Mbps)	12.21	10.51	11.53	10.31
ave. overall throughput (Mbps)	301.81	261.47	450.70	407.69

4.7.6.1 Large Topology

The *large topology* in Figure 4.6 is then considered for simulations, which basically models the third floor of Engineering Building-2 in Okayama University. 70 hosts and 10 DAPs are regularly distributed in the six rooms with two different sizes, $7m \times 6m$ and $3.5m \times 6m$. The same minimum host throughput constraint $G = 10Mbps$ and bandwidth limit constraint $B^a = \infty$ are examined. As the host behavior model in this topology, $N1 = 35$ hosts are randomly selected as initial *active hosts* from $N = 70$ hosts, and $\lambda = 1.5$ and $\mu = .02$ are adopted.

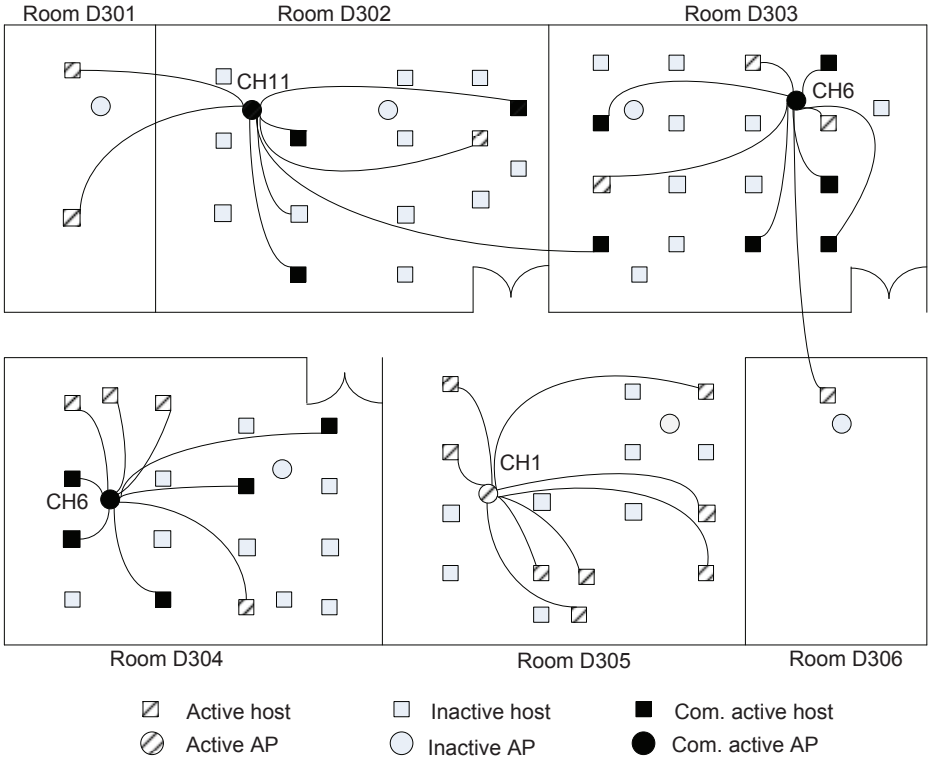
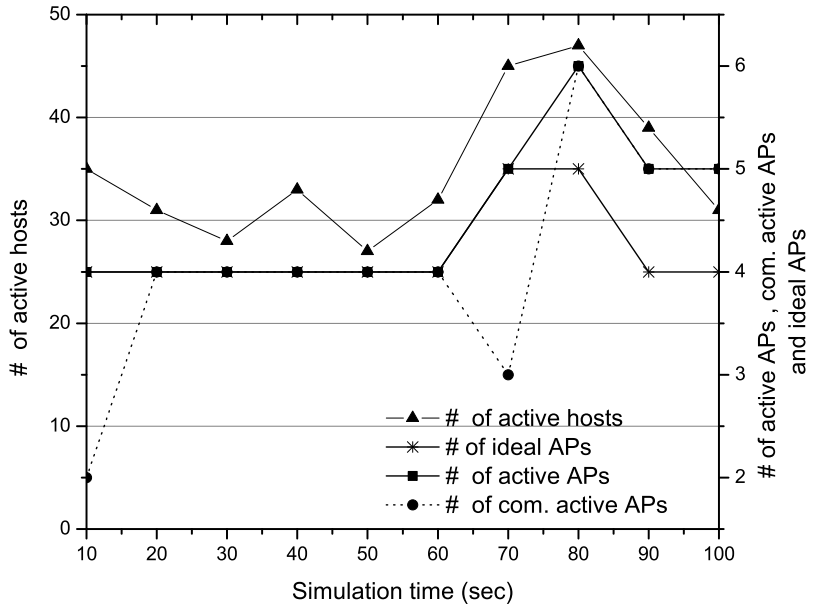


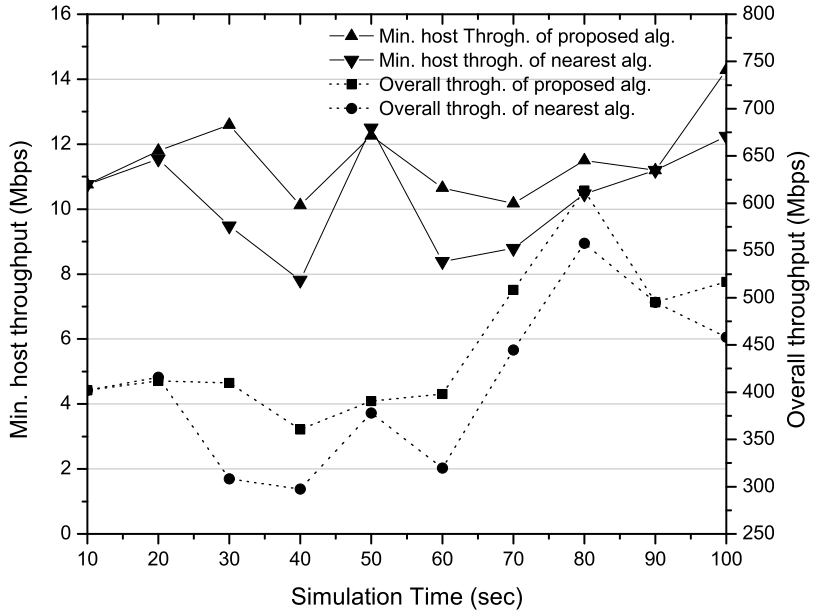
Figure 4.6: Solution for large topology with 70 hosts and 10 APs.

4.7.6.2 Results

Figure 4.7 (a) shows the changes of the number of active hosts and the number of communicating active APs by the host behavior model, and the number of active APs by the proposed algorithm, when the simulation time elapsed. Figure 4.7 (b) shows the changes of the minimum host throughput and the overall throughput by the WIMNET simulator for the AP-host association by the proposal and for that by the compared NAP. From them, the same performance results can be observed for the large topology as for the small topology.



(a) Change of active hosts and APs.

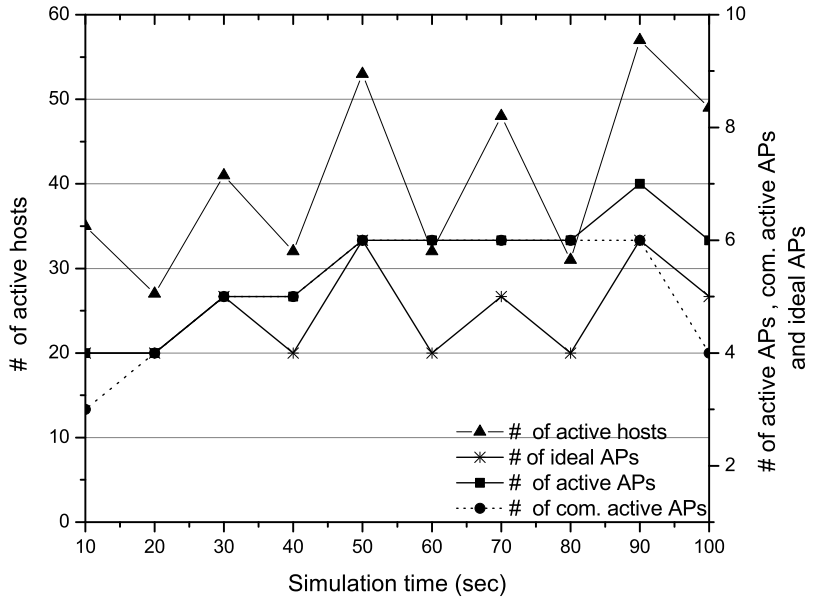


(b) Change of min. host and overall throughputs.

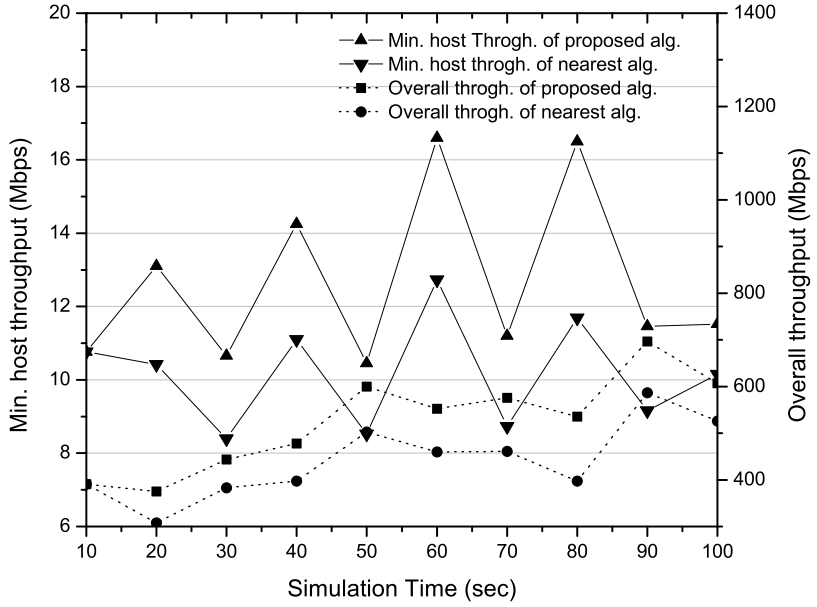
Figure 4.7: Performance graph for small topology.

4.7.6.3 Results for Larger λ

In some networks, the number of users can be more widely fluctuated by time. For example, at public WLANs in a station or an airport, a lot of users may arrive there by a train or an airplane and join the network at the same time. They often stay there for a short time. To simulate such cases, the larger value of $\lambda = 4$, 5 and $\mu = .05$ are adopted. Figures 4.8 (a) and 4.9 (a) show the changes of the number of active hosts and the number of communicating active APs and the number of active APs for each λ respectively. Figures 4.8 (b) and 4.9 (b) indicate that by the proposal, the minimum host throughput is always larger than the threshold $G = 10Mbps$ and the



(a) Change of active hosts and APs.



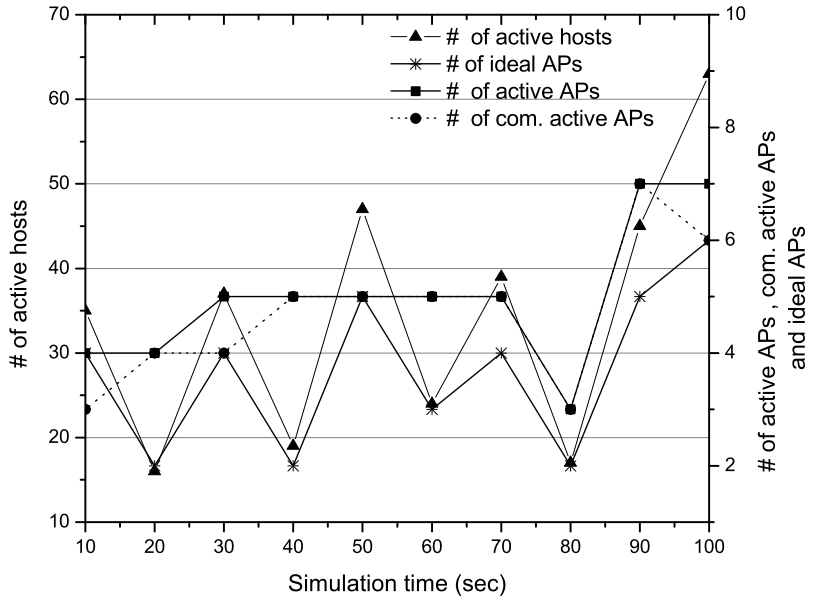
(b) Change of min. host and overall throughputs.

Figure 4.8: Performance graph for large topology with $\lambda = 4$.

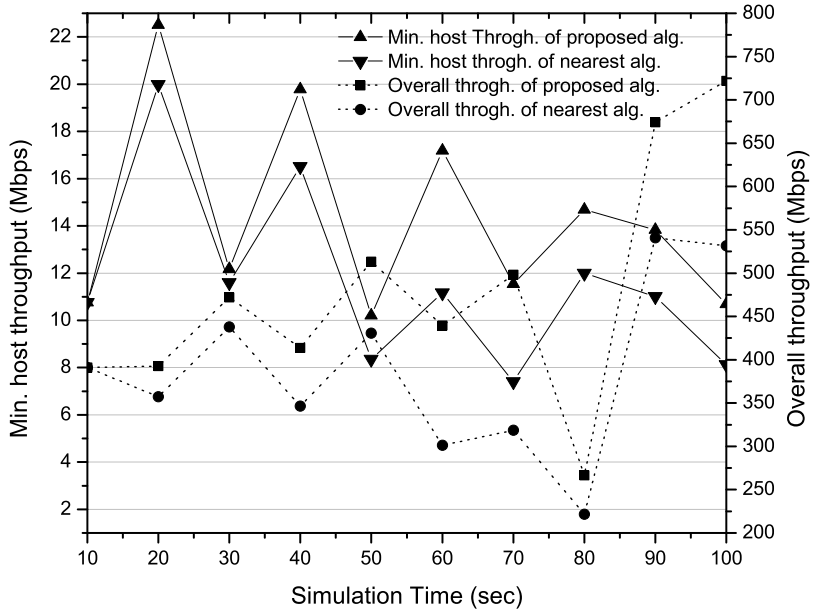
overall throughput is larger than that of the comparison. Table 4.4 compares the average minimum host throughput and overall throughput between the two methods. This table indicates that the performance by the proposal is better than that by the comparison.

4.7.6.4 Results for Moving Hosts

As described before, this study assumes that WLAN users usually use PCs on fixed positions where desks or chairs are available. However, users may move to different positions in the network field if the new positions are better for users. This move of a host to a different position can be represented by a pair of the leave operation from the current position and the join operation at the new position.



(a) Change of active hosts and APs.



(b) Change of min. host and overall throughputs.

Figure 4.9: Performance graph for large topology with $\lambda = 5$.

Table 4.4: Throughput comparison between two methods for large λ.

instance	$\lambda = 4$		$\lambda = 5$	
method	proposed	compare	proposed	compare
ave. min. host throughput (Mbps)	12.65	10.16	14.33	11.70
ave. overall throughput (Mbps)	525.39	441.21	478.29	387.73

Thus, when a host move appears, basically, the algorithm extension for a leaving host and that for a joining host are applied sequentially. However, if the host moves to a near position from the current

one, it can keep the same associated AP if the minimum host throughput constraint is satisfied. In this case, the proposed extension is not applied, where only the link speed is updated.

To consider moving hosts, we simulated the proposal with them for the *large topology*. Here, $\lambda = 1.5$ is used for the parameter of joining and leaving hosts, $\mu = .02$ is for the parameter of the communication duration time, and 10% is for the communicating host selection probability. At the time hosts may join or leave the network, the moving hosts are randomly selected from non-communicating hosts with the fixed probability, where 0%, 5%, 10%, and 15% are used for this probability. The selected moving host is moved to a randomly selected host position instantly.

Table 4.5 summarizes the simulation results. The throughput performance is similar in any case. Therefore, our proposal is effective for the moving hosts in the network field.

Table 4.5: Simulation results for moving hosts.

parameters	results			
moving host selection probability	0%	5%	10%	15%
avg. # of active APs	4.54	4.54	4.58	4.54
avg. # of moving hosts	0	1.86	3.70	5.40
avg. # of communicating hosts	3.38	3.34	3.44	3.36
avg. min. host throughput (Mbps)	11.32	11.16	11.34	11.33
avg. overall throughput (Mbps)	452.44	448.22	446.95	443.50

4.7.6.5 Comparisons with Previous Study

To clarify the effectiveness of the proposed algorithm extensions, we simulate the cases when the rate of the communicating hosts among the hosts is set to 25% and 50%, and compare the number of the active APs, the minimum host throughput, the total throughput, and the number of hosts changing the associated APs between the previous algorithm and the proposed one. It is noted that as the number of hosts changing the associated APs increases, the number of suspending hosts to change the associated APs increases.

Table 4.6: Comparisons of performances between proposal and previous.

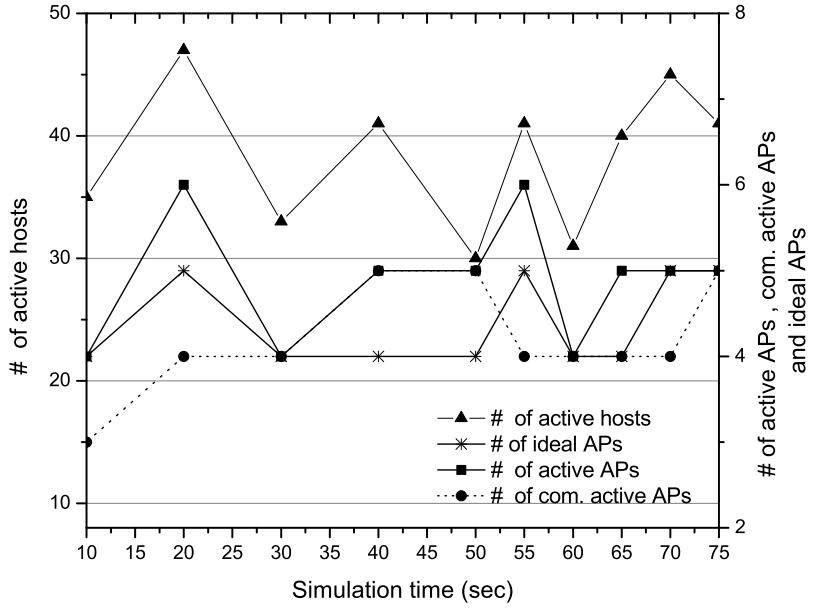
simulation cases		results	
rate of com. host	parameters	proposal	previous
25%	avg. # of active APs	5.00	5.00
	avg. min. host throughput (Mbps)	10.96	10.97
	avg. overall throughput (Mbps)	487.45	480.81
	avg. rate of AP change hosts among com. hosts	0%	37.5 %
50%	avg. # of active APs	6.00	6.00
	avg. min. host throughput (Mbps)	12.31	13.03
	avg. overall throughput (Mbps)	562.74	568.98
	avg. rate of AP change hosts among com. hosts	0%	41.2%

Table 4.6 shows the results when the rate of the communicating hosts is set to 25% and 50%. The network load is changed where the number of active hosts is set to 41 on average, to take the average values of the minimum host throughput and the overall throughput for each case. It is found that the average values of the both throughputs are similar between the two algorithms,

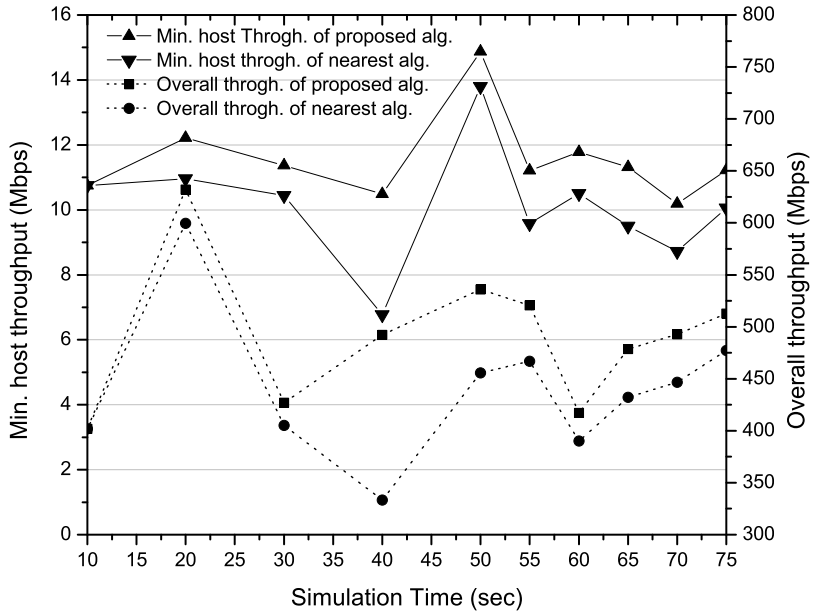
while 37.5% or 41.2% hosts must change the associated APs while communicating, which can increase the network management overheads and suspend the Internet services for them.

4.7.6.6 Discussions

The simulation results show that at any situation, the extended algorithm can minimize the number of active APs while the minimum host throughput is maintained, and the overall throughput is better than the nearest AP association approach. They confirm the effectiveness of the proposal.



(a) Change of active hosts and APs.



(b) Change of min. host and overall throughputs.

Figure 4.10: Performance graph for large topology with network parameter change.

In Figure 4.5 (a), the number of active APs and the number of ideal APs are always same. In

others, they are sometimes different by 1 or 2 because of the communicating APs that cannot be deactivated. Furthermore, in order to see the performance of the extended algorithm when important parameters in the host behavior model are changed during the simulation, another simulation is conducted, where Td is changed from $10sec$ to $5sec$ and λ is changed from 3 to 2 when $50sec$ is elapsed. As shown in Figure 4.10, our algorithm can adapt to the change of the network parameters immediately.

The results in Section 4.7.6.5 show that our proposal can avoid the suspensions of communicating host due to the change of the associated APs. The host mobility results in Section 4.7.6.4 also show that the proposed algorithm can improve the network performance similar to related work [79]. Furthermore, by considering host leaving from the network, this algorithm deactivates unnecessary APs to save energy and reduce interferences. On the whole, the proposed extension exhibits dynamic feature of real time WLAN system and outperform all the other algorithms.

4.8 Testbed Implementation

In this section, we present the implementation of the elastic WLAN system testbed using *Raspberry Pi* for the proposal. This testbed adopts a server to manage the information and control the system.

4.9 Execution Flow of Elastic WLAN for Dynamic Approach

Figure 4.11 shows the execution flow of the elastic WLAN system testbed. To deal with dynamic behaviors of joining and leaving hosts, the four steps, Steps 7-10, are added to the six steps, Steps 1-6, of the original implementation in [9]. Before starting the execution of Step 7 to Step 10, shown in Figure 4.11, the server compiles all the programs made by C at once to save the CPU time and the load. The following commands perform the compilation of each program.

Linux commands for compilation of each C program

```
#/bin/bash
# to detect the network change and identify joining or leaving host
01: g++ -o chkD CheckDevicesForjoinleave.cpp
02: g++ -o Inputforleavinghost GenAlgmInputforLeaving.cpp
03: g++ -o ckcurap CheckWithCurrentAP.cpp
04. g++ -o dcs Decisioncurjoin.cpp
05. g++ -o CrIn CreateInputforAllHost.cpp
06: g++ -o ckcomap Comaphost.cpp
07: g++ -o DAPC DAPC.cpp
```

After that, the server uses the executable files and the necessary input files generated by each step to perform the whole execution process. The following section describes how the server can detect communicating APs and hosts. Then, all the steps are describes with detail operations.

4.9.1 Detection of Communicating AP and Host

The AP association of any host that is currently communicating with the Internet, must be continued. To detect such communicating APs and hosts, the server inspects the packets that are

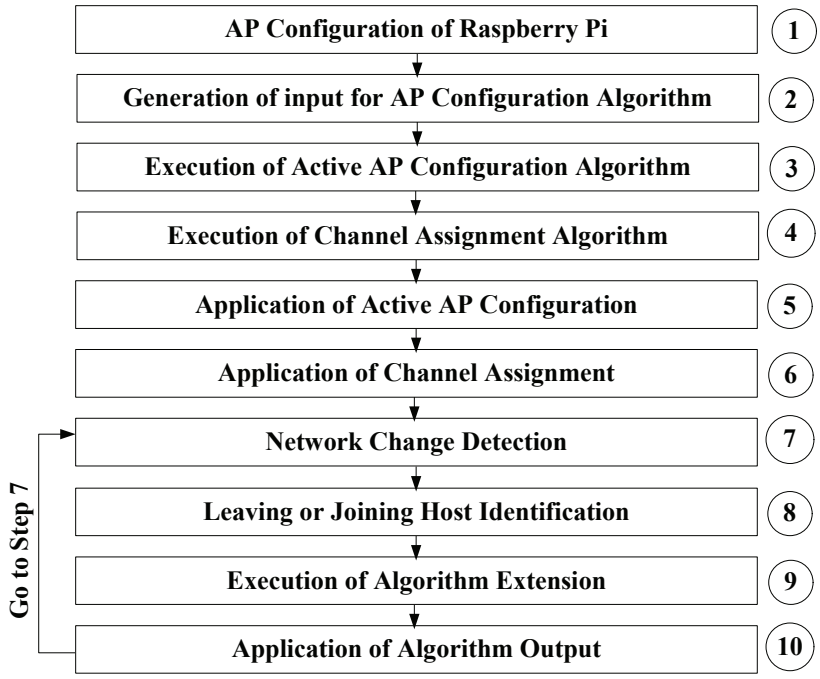


Figure 4.11: Elastic WLAN system execution flow for Dynamic active AP Configuration.

exchanged between the host and the AP, using *tcpdump* [84]. The *tcpdump* is a powerful network debugging tool and command-line packet analyzer on a network interface. The server collects the packets between the active hosts and their associated active APs, and detects the hosts and APs that are currently communicating from them. The following commands are used in this step:

Linux commands for communicating hosts and APs dectection

```
#/bin/bash
# to inspect packets exchange between hosts and APs
01: timeout 50 tcpdump -i wlan0 host ip
# analyze and detect Com. APs and hosts
02: ./ckcomap tcppacket.txt
```

The command in `01` inspects the TCP packets between the APs and their associated hosts for 50*sec*. using *tcpdump*. Here, *host ip* represents the IP address of any host connecting to any active AP. Then, a C program is executed to filter and analyze the packets. The server detects the communicating hosts and APs using the command in `02`.

4.9.2 Network Change Detection

In this step, the server detects whether the network state is being changed due to joining of new hosts or leaving of existing hosts in the network. To detect it, the server uses the following commands:

Linux commands for network change detection

```
#/bin/bash
# to explore the connected devices in network
01: sudoarp-scan --interface=eth0 192.168.11.0/24
# to detect the network change
02: ./chkD netoutpre.txt netoutrecent.txt
```

The command in 01 explores the connected devices in the network using *arp-scan* [47]. The output consists of the IP and MAC addresses of the hosts and the APs that are available in the network. The command in 02 identifies network changes. Here, *netoutpre.txt* and *netoutrecent.txt* contain the device information of the network according to previous and current status. The C program will check the devices and can detect the network changes. After detecting the network state change, the server takes the following steps to control the number of active APs.

4.9.3 Leaving or Joining Host Identification

After detecting the network state change, the server identifies the hosts that newly joined or left the network and finds their IP addresses by running the same C program.

Linux commands for leaving or joining host identification

```
#/bin/bash
# identify joining or leaving hosts/hosts
01: ./chkD netoutpre.txt netoutrecent.txt
```

The command in 01 takes the two files, *netoutpre.txt* and *netoutrecent.txt*, as the inputs. After analyzing the IP and MAC addresses of the devices, the server identifies and prepares the list of the changed hosts. If there is no change in the network, the server goes back to the previous step in Figure 4.11.

4.9.4 Execution of Algorithm

According the host identifications, the active AP configuration algorithm is applied for joining hosts or leaving hosts. The output of the algorithm contains the information of the network changes, including the associations between the hosts and APs and the list of APs for activation or deactivation.

1. **Steps in Leaving Operation**
   Before executing the algorithm for the leaving host, the server prepares the input by the following commands:

   Linux commands for leaving host operation

   ```
   #/bin/bash
   # generate input for the Modified APC Alg.
   01: ./Inputforleavinghost leavinghosts.txt
   alg_out.txt previousinput.txt group_info.txt newinput.txt
   #run the DAPC
   02: ./DAPC newinput.txt min_host_throughput com.txt
   ```

   The command in 01 takes the input from *previousinput.txt* and prepares the new input for the

leaving host extension. Before applying the algorithm, the server detects the communicating hosts and APs, as described in Section 4.9.1. After that, the extended active AP configuration algorithm is executed using the command in `02` to finds the new configuration by considering the communicating APs and the hosts in *com.txt* detected in Section 4.9.1.

2. **Steps in Joining Operation**
The server collects the signal strengths of the newly joining hosts and the active APs in the network. Then, it applies the following commands:

Linux commands for joining host operation

```
#/bin/bash
# to inspect signal strength for joining host
01: sudo nm-tool
02: sh ./nm_scriptjoinhost.sh
03: ./ckcurap JoinHosts.txt
# decision on current APs
04: ./dcs ch_association.sh ap_activation.sh
# generate input for the Modified APC Alg
# if we need to activate new AP
05: ./CrIn CurHostsall.txt CurAPall.txt
#run the DAPC
06: ./DAPC newinput.txt min_host_throughput com.txt
```

The commands in `01` and `02` find the receiving signal strength of each joining host from every AP using *nm-tool* [49, 50]. The *ssh* [53, 54] protocol is used to execute the command in `01` remotely in each host. The output consists of the currently associated APs, the list of associable APs, and the receiving signal strength of each host from all the associable APs. After this, the server converts the receiving signal strength to the estimated link speed using the sigmoid function in [72], and generates the input for the active AP configuration algorithm. By using the commands in `03` and `04`, the server knows whether the currently activated APs can satisfy the minimum host throughput constraint for the hosts or not. It also detects the changes of their AP associations, if necessary. If they cannot satisfy the constraint, the server applies the algorithm to activate some deactivated APs, after preparing the algorithm input using the command in `05`. Finally, the dynamic active AP configuration algorithm is executed to find the association after the join operation using the command in `06`. Again, the communicating hosts and APs are detected using the procedure in Section 4.9.1. Here, *com.txt* is the list of currently communicating hosts and APs.

4.9.5 Application of Algorithm Output

Then, the server changes the corresponding associations and deactivates all the unused APs by following the algorithm output. For the joining hosts, some additional APs are activated and are assigned the channels, if necessary. Besides, the channels of some active APs are also changed, if necessary. The following commands are used in this step. The commands in `01` and `02` activate or deactivate the *Raspberry Pi* AP respectively. The server adjusts the number of active APs according to the algorithm output by activating or deactivating APs in the network. The command in `03` connects a host to a new AP using *nmcli* [58, 59]. Here, *NewSSID* represents the new AP

for the host and *PASSWORD* does the security key of the AP. The server modifies the AP-host association according to the algorithm output using this command.

Linux commands for application of algorithm output

```
#/bin/bash
# for activation of a Raspberry Pi AP
01: sudo /etc/init.d/hostapd start
# for deactivation of a Raspberry Pi AP
02: sudo /etc/init.d/hostapd stop
# to change the association of a host to a new AP
03: sudo -s nmcli dev wifi connect NewSSID password PASSWORD
# to change the channel of a raspberry Pi AP
04: sed -i -e 's/.*channel.*/channel='$NewChannel'/'/etc/hostapd/
hostapd.conf
# to restart the service of hostapd daemon
05: sudo /etc/init.d/hostapd restart
```

The command in `04` assigns the new channel to the AP using *sed* [73]. For this, the server modifies the configuration file */etc/hostapd/hostapd.conf* with the channel number. Here, *'s'* represents the substitution command and *NewChannel* does the channel to be assigned in the *hostapd.conf* file of the AP. The command in `05` restarts the *hostapd* daemon [70, 71]. After the assignment of the new channel, the server restarts it to make the change take effect. It takes $20 \sim 30sec$. to stop the *hostapd* service, and takes $40 \sim 60sec$. to change the channel of an active AP. To restart the *hostapd* daemon, it takes $20 \sim 30sec$ on average. The server changes the channel of an active AP, only if (i) the AP is not the *communicating AP* and (ii) the algorithm changes the channel because of the joining or leaving hosts.

4.10 Evaluations by Testbed Experiments

In this section, we evaluate the implemented elastic WLAN system testbed using four network scenarios in Okayama University.

4.10.1 Devices and Software

The devices and software in Table 4.7 are used for throughput measurements using the testbed in real network environments. The IEEE 802.11n protocol is used for any communication link with the *channel bonding*.

4.10.2 Two Comparison Methods

To evaluate the throughput performance by our proposal, the two simple methods for selecting active APs and host associations are considered to be compared.

4.10.2.1 Comparison Method 1 (COMP-1)

To compare the throughput results in various network scenarios, a simple comparison method (COMP-1) is adopted. In this method, the same number of APs is activated in the network by

Table 4.7: Devices and software in the testbed.

Devices and software		
server PC	OS	Ubuntu LTS 14.04
	model	Lesance W255HU
	Processor	Intel(R), Core(TM)-i3
client PC (type-1)	OS	Ubuntu LTS 14.04
	Model	Toshiba Dynabook R731/B
	Processor	Intel(R), Core-i5
client PC (type-2)	OS	Ubuntu LTS 14.04
	Model	Fujitsu Lifebook S761/C/SSD
	Processor	Intel(R), Core-i5
access point	Raspberry Pi 3	
	OS	Raspbian
	Processor	1.2 GHz
software/tools	openssh	to access remote PC and AP
	hostapd	to prepare and configure AP
	nmcli	for association change
	nm-tool	to measure signal strength
	tcpdump	to analyze packets
	arp-scan	to discover active network devices

randomly selecting them from available APs, where the channel is assigned by our algorithm. For any newly joining host, the host is associated with the AP that provides the highest RSSI from the active APs.

4.10.2.2 Comparison Method 2 (COMP-2)

As another comparison method (COMP-2), the active APs are also selected by the RSSI to the joining host. For any joining host, if the number of active APs is smaller than the algorithm, the AP that provides the highest RSSI is newly activated, and the host is associated with it. Again, for any AP, the channel is assigned by the algorithm.

4.10.3 Network Scenarios

For evaluations, four network scenarios are prepared for the elastic WLAN system testbed. For each AP, one of the three orthogonal channels, 1, 6, and 11, is assigned by the proposed algorithm.

4.10.3.1 3×4 Scenario in One Room

In the first scenario, three *Raspberry Pi* devices for APs and four Linux PCs for hosts are prepared in a room of size $7m \times 6m$. Figure 4.12 shows the distance between the hosts and APs. Any access point is connected to the server using the wired connection.

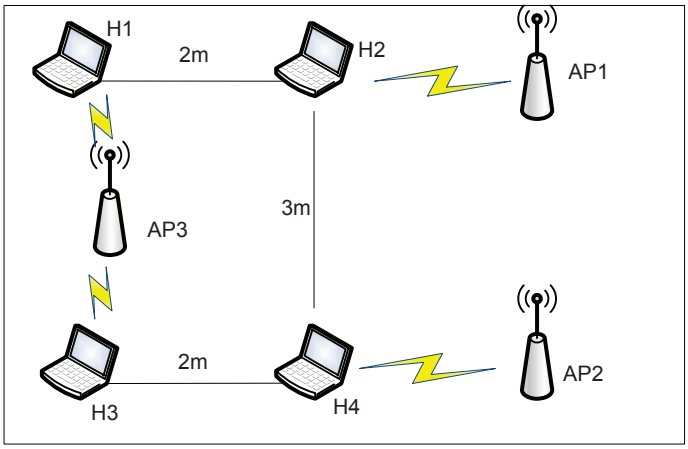


Figure 4.12: Testbed for 3×4 scenario in one room.

4.10.3.2 3×4 Scenario in Different Rooms

In the second scenario, three *Raspberry Pi* devices for APs and four Linux PCs for hosts are prepared in two rooms with the size of $7m \times 6m$ separated by the wall and one corridor at the third floor of Engineering Building-2 in Okayama University. As shown in Figure 4.13, any AP is $5m - 6m$ away from another AP in the different room and corridor to reduce the interference.

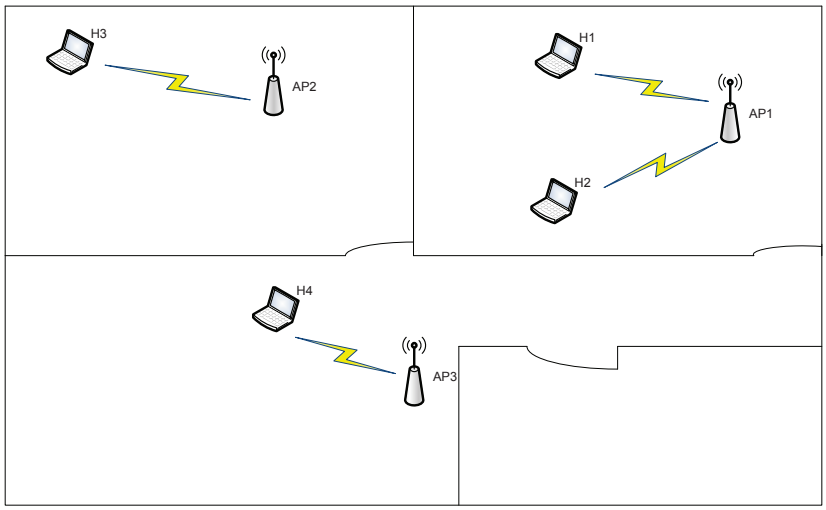


Figure 4.13: Testbed for 3×4 scenario in different rooms.

4.10.3.3 3×6 Scenario

In the third scenario, three *Raspberry Pi* devices for APs and six Linux PCs for hosts are placed in the same field, as shown in Figure 4.14.

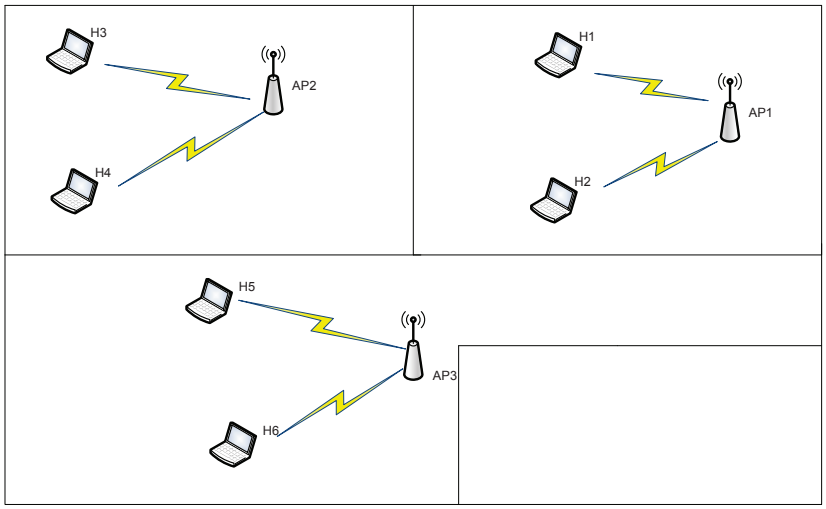


Figure 4.14: Testbed for 3×6 scenario.

4.10.3.4 4×8 Scenario

In the fourth scenario, four *Raspberry Pi* devices for APs and eight Linux PCs for hosts are distributed in the three rooms and the corridor as shown in Figure 4.15 at the second floor of Graduate School Building in Okayama University. The size of each room is $9m \times 5.5m$, $3.5m \times 5.5m$, and $7m \times 5.5m$ respectively. Any AP is $4m$ away from another AP in the same room.

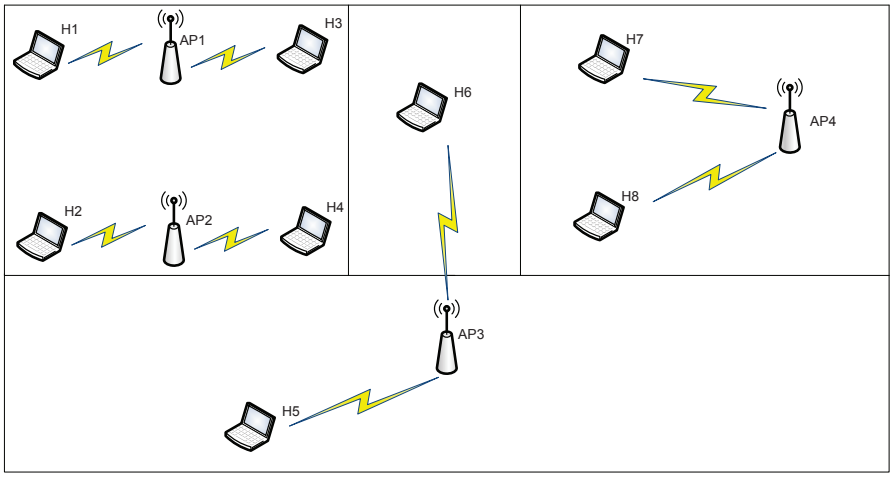


Figure 4.15: Testbed for 4×8 scenario.

4.10.4 Host Join/Leave Dynamics

In each scenario, the host join/leave dynamics in the network are represented by a sequence of stages. At each stage, 1) one host joins or leaves the network, 2) Steps 7-10 in Section 4.9 are executed, and 3) the throughputs of all the active hosts are measured when they are concurrently

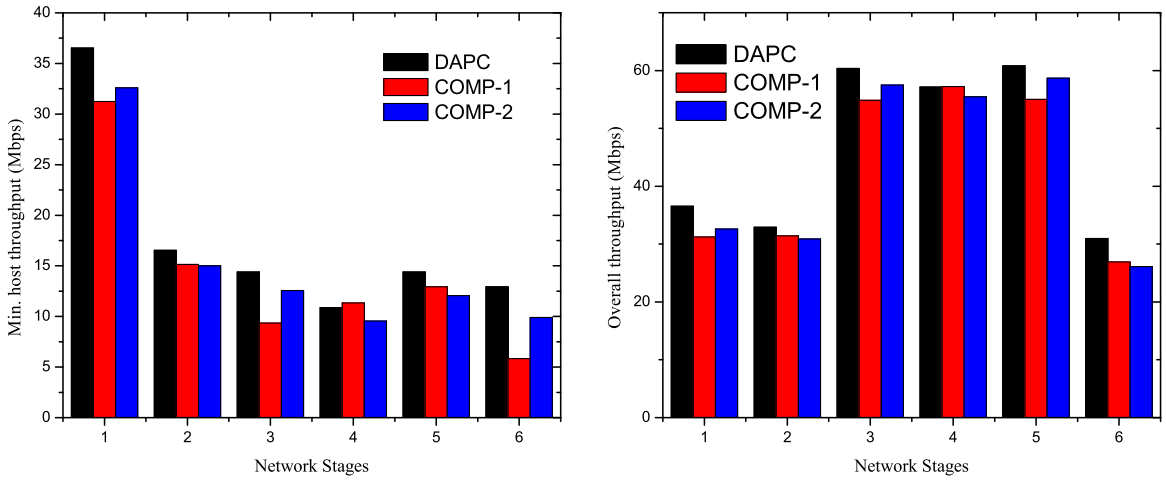
communicating with the *iperf* [24] server through the associated APs. By following the host behavior model in 4.7.4, the joining and leaving hosts are randomly selected for each network stage with $\lambda = 1$ and $\mu = .02$.

4.10.5 Throughput Measurement Results

For each scenario, the throughputs at each stage are measured and compared.

4.10.5.1 3×4 Scenario in One room

Figure 4.16 (a) and (b) show the minimum host throughput results and the overall throughput results in the testbed for the 3×4 scenario in one room, by the proposal, by COMP-1, and by COMP-2 at each stage, where the number of active hosts is changed from 1 to 4. Except for the minimum host throughput at stage 4, our proposal always provides the better performance than COMP-1.



(a) Minimum host throughput.

(b) Overall throughput.

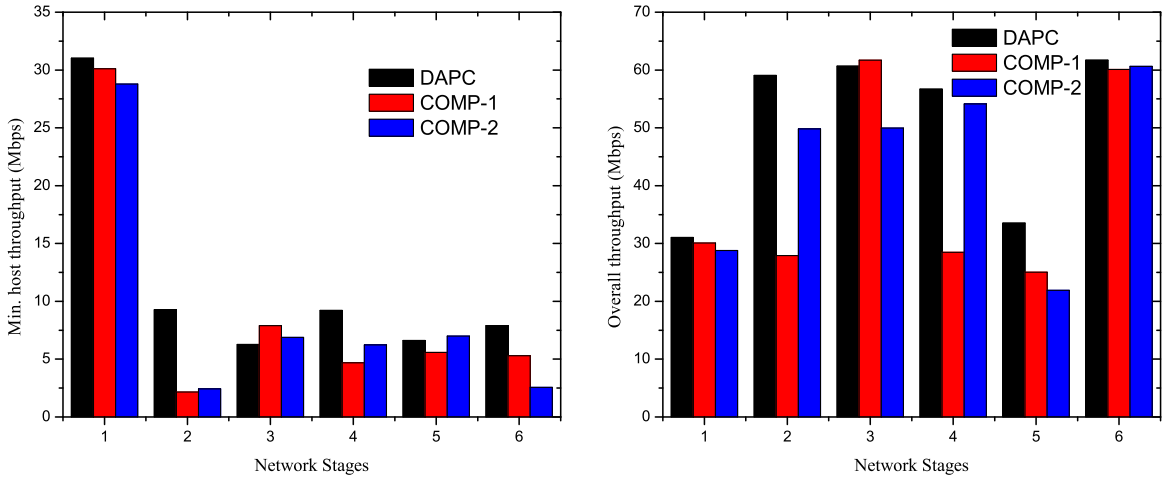
Figure 4.16: Throughput results for 3×4 scenario in one room.

4.10.5.2 3×4 Scenario in Different Rooms

Figure 4.17 (a) and (b) show the minimum host throughput results and the overall throughput results in the testbed for the 3×4 scenario in different rooms, by the proposal, by COMP-1, and by COMP-2 at each stage, where the number of active hosts is changed from 1 to 4. Except for the minimum host throughput at stages 3, 5 and the overall throughput at stage 3, our proposal always provides the better performance than COMP-1 and COMP-2. It can be observed that the minimum host throughput becomes lower at any stage than that in one room case. Here, since a host is connected to an AP in a different room, the RSS at such a host from the AP becomes smaller due to the wall attenuation, and thus, the throughput becomes lower.

4.10.5.3 3×6 Scenario

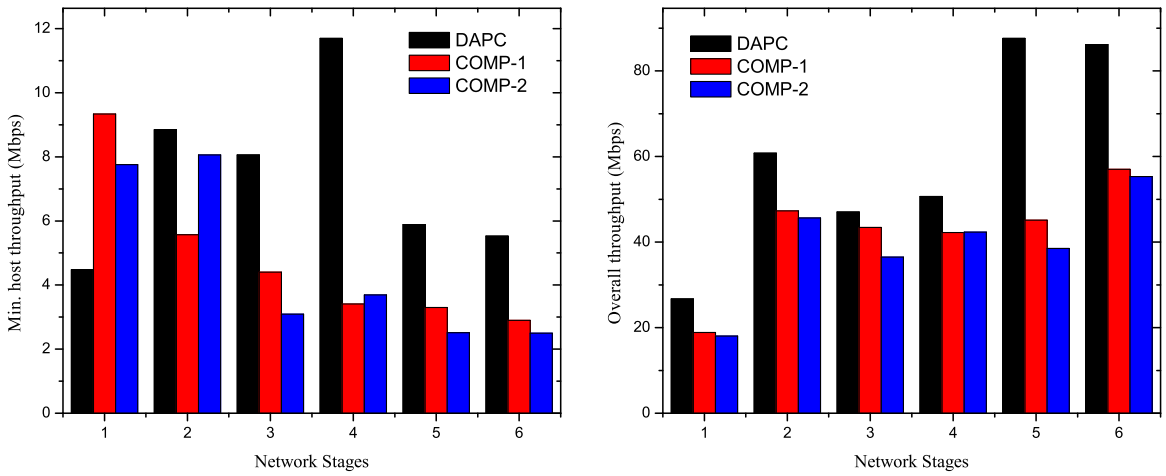
Figure 4.18 (a) and (b) show the minimum host throughput results and the overall throughput results in the testbed for the 3×6 scenario, by the proposal, by COMP-1, and by COMP-2 at



(a) Minimum host throughput. (b) Overall throughput.

Figure 4.17: Throughput results for 3×4 scenario in different rooms.

each stage, where the number of active hosts is changed from 2 to 6. Except for the minimum host throughput at stage 1, our proposal always provides the better performance than COMP-1 and COMP-2.

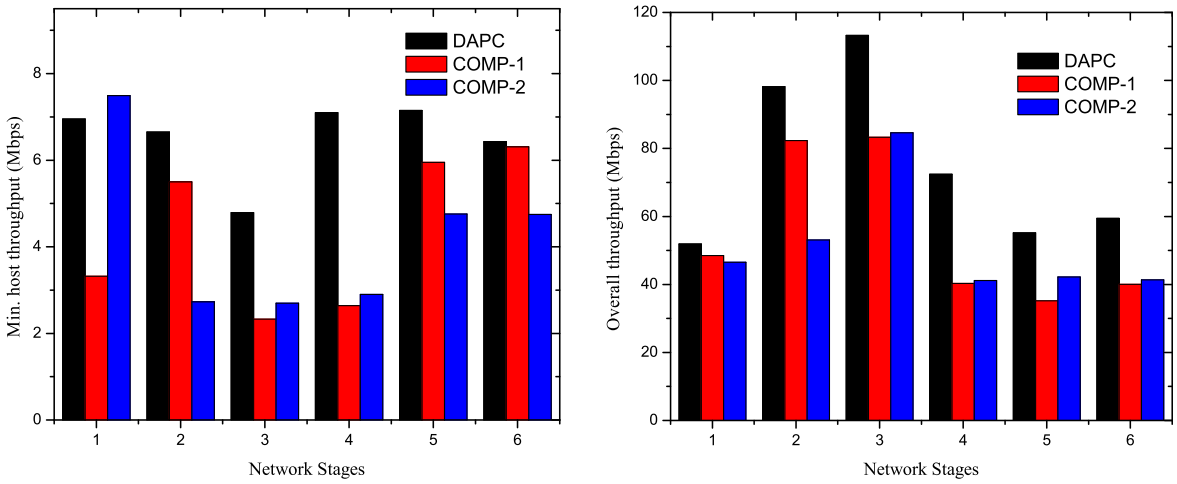


(a) Minimum host throughput. (b) Overall throughput.

Figure 4.18: Throughput results for 3×6 scenario.

4.10.5.4 4×8 Scenario

As the largest topology, Figure 4.19 (a) and (b) show the minimum host throughput results and the overall throughput results in the testbed for the 3×6 scenario, by the proposal, by COMP-1, and by COMP-2 at each stage, where the number of active hosts is changed from 3 to 8. Except for the minimum host throughput at stage 1, our proposal always provides the better performance than COMP-1 and COMP-2.



(a) Minimum host throughput.

(b) Overall throughput.

Figure 4.19: Throughput results for 4×8 scenario.

4.11 Summary

In this chapter, firstly, we described the motivation of our proposal. Secondly, we described some related works. Thirdly, we presented the extension of the active AP configuration algorithm for dynamic host joining and leaving. Fourthly, we evaluated the proposal through simulations using the WIMNET simulator. Fifthly, we described the implementation of the proposal in the elastic WLAN system testbed. Finally, we evaluated our proposal through experiments using the testbed. In the next chapter, we will present the security update functions for the elastic WLAN system.

Chapter 5

System Security Implementation in elastic WLAN Testbed

In this chapter, we describe the implementations of the two security functions for the elastic WLAN system testbed: the *system software update* function and the *user authentication function*. We also confirm the correctness of them through experiments.

5.1 Motivation

In the current implementation of the elastic WLAN system, we limit a *Linux PC* for a host and the server, and choose *Raspberry Pi* for an AP. Since all the host PCs and the APs are using the same platform, it is important to keep the system software updated for the latest version for maintaining the operational availability, confidentiality and integrity. Basically the majority of cyber-attacks target known vulnerabilities in the system software. Any software vulnerability is one of the security hole found in a software program or operating system. Thus, the management of the system software updates has become a serious challenge for modern organizations.

To increase the security and performance of the system, it is important to keep updating the system and application software to the latest version. Here, to reduce the overhead in downloading the software files from the Internet, a local *repository server* is prepared to download and store all the updated files and packages periodically from the original repository and to distribute them to the APs and the hosts. The user authentication is also inevitable to enhance the security of the system.

In this study, we implement the *system software update function* and the *user authentication function* in the elastic WLAN system. The system software update function identifies the newest version of the system and application software that have been installed into the AP and the host in the system. This function periodically checks the version for each software in the original repository in the Internet. Then, if the newer version exists, it downloads the necessary files to update the software, and stores them in the prepared local repository server. Every AP and user host in the system accesses to this server to update the system software.

The user authentication function is implemented using the *RADIUS* server. Any newly joining user to the system must register the information for authentication at the server. Using this information, the server authenticates the joining host user by sending the authentication message via one AP in the system to the server. Through experiments, we confirm the correctness of these implemented functions.

5.2 Importance of System Software Update

The system software update function is implemented based on the following observations:

- **Security:**
  Hackers may be always finding new ways to attack a system by finding security holes of the software in the system. The regular update of the software can remove the known security holes.

- **Functionality:**
  The latest software often provides new features and running speed enhancements.

- **Load:**
  The use of the local repository server can reduce the bandwidth required to download the files from the original repository in the Internet. It can be very helpful in developing countries including Bangladesh and Myanmar suffering from the unreliable and slow Internet access.

5.3 Implementation of System Software Update Function

In this section, we present the implementation of the system software update function in the elastic WLAN system testbed.

5.3.1 Configuration of Local Repository Server

The configuration of the local repository server for the system software update function is described in details here. This server stores the latest files and packages for the system software locally inside the elastic WLAN system. A Linux PC is used for this server. The *apt-mirror* tool is used to mirror or copy any part (or even all) of Ubuntu GNU/Linux distributions and any other *apt* source provided by an open source developer [85]. The following commands are used to configure the server with *apt-mirror*:

1. sudo apt-get install apt-mirror
2. cd /var/spool/apt-mirror
3. mkdir var mirror skel
4. sudo nano /etc/apt/mirror.list
5. sudo apt-mirror

The command in (1) is used to install *apt-mirror* to the local server. The commands in (2) and (3) create the necessary folder to store the files, so that this repository server works correctly. Then, *apt-mirror* is configured to collect the necessary files from the original repository. The command in (4) is used to edit the source list of *mirror.list* file. Here, we put the source address of the original mirror site for the Raspbian OS. From this site, *apt-mirror* downloads all the packages and files. Figure 5.1 shows an example *mirror.list* file. Finally, the command in (5) allows the server to download the system files and packages to be stored in this server.

```
############# config ##################
#
# set base_path    /var/spool/apt-mirror
#
# set mirror_path  $base_path/mirror
# set skel_path    $base_path/skel
# set var_path     $base_path/var
# set cleanscript $var_path/clean.sh
# set defaultarch  <running host architecture>
# set postmirror_script $var_path/postmirror.sh
# set run_postmirror 0
set nthreads     20
set _tilde 0
#
############## end config ##############

deb-armhf http://mirrordirector.raspbian.org/
raspbian/ jessie main contrib non-free rpi

deb-src  http://mirrordirector.raspbian.org/
raspbian/ jessie main contrib non-free
clean http://mirrordirector.raspbian.org/raspbian
```

Figure 5.1: Example of *mirror.list* in *apt-mirror* in server.

5.3.2 Installation of Web Server Function

For the access to the local repository server from the APs and hosts, this server should adopt the Web server function and offer the link to the packages and binary files stored in the server. Here, we choose and configure *Apache* for the Web server system [86]. The following commands are used to install and configure the Web server function:

1. sudo apt-get install apache2
2. ln -s /var/spool/apt-mirror/mirror/mirrordirector. raspbian.org/raspbian/ /var/www/html/raspbian
3. sudo service apache2 start

The command in (1) is used to install *Apache* in the server machine. The command in (2) creates *symlink* to allow the access to the stored files and the system administrator to manage and browse the stored packages and directories in the server. The command in (3) is used to start the installed server after configuration.

5.3.3 Configuration of AP

Every AP or client of the local repository server is configured so that it can download the system software from the server. In this configuration, it is necessary to edit the */etc/apt/source.list* file in each client. The original repository address is changed to the IP address of the server. The following command is used to change the source list by this file:

- sudo nano /etc/apt/sources.list

The sample *sources.list* file after modification is shown in Figure 5.2. Then, any client uses the following commands to update all the system software or install a specific package from the local server:

```
deb http://192.168.11.30/raspbian/ jessie main
contrib non-free rpi

# Uncomment line below then 'apt-get update' to
enable 'apt-get source'

#deb-src http://192.168.11.30/raspbian/ jessie main
contrib non-free rpi
```

Figure 5.2: Example of source.list in Raspberry Pi.

1. sudo apt-get update
2. sudo apt-get install XXXX

where *XXXX* represents the application or system software package.

5.3.4 Periodic update function in server

The latest version of any system software and package is collected from the original mirror repository provided by the open source developer, and is stored periodically at the server. This can be done at a fixed time in a day, where the system schedule job is assigned at 2 AM every day in our implementation. This process runs automatically in the background using the *cron* daemon in Linux [87]. The following commands are used for the periodic update of files and packages:

1. 0 2 * * * /usr/bin/apt-mirror
2. 0 5 * * * /var/spool/apt-mirror/var/clean.sh

The command in (1) automatically and periodically updates the system software files and packages. The command in (2) deletes the unnecessary files at 5AM every day, so that the server can update files and packages without interruptions.

5.3.5 Software Update for Host

In the current implementation of the elastic WLAN system testbed, any host must use Linux. To enhance the security, it is essential to keep updating the system software on time using the local repository server for hosts. To create this server, the same steps in Sections 5.3.1 to 5.3.4 are used, including *apt-mirror* and *Apache*. During the configuration of *apt-mirror*, the default *mirror.list* file is changed to the list in Figure 5.3. After configuring *apt-mirror*, the procedure in Section 5.3.3 is applied where the *source.list* file of each host is changed to get the necessary files and packages from the server for hosts. Fig. 5.4 shows this *source.list* file for each host.

5.3.6 System Topology with Update Servers for APs and Hosts

Figure 5.5 illustrates the elastic WLAN system topology with the system software update servers for APs and hosts. Each server finds and downloads the updated files and packages from the

```
############# config ##################
##set base_path    /var/spool/apt-mirror
## set mirror_path  $base_path/mirror
# set skel_path    $base_path/skel
# set var_path     $base_path/var
# set cleanscript $var_path/clean.sh
# set defaultarch  <running host architecture>
# set postmirror_script $var_path/postmirror.sh
# set run_postmirror 0
set nthreads     20
set _tilde 0
############## end config ##############

deb-amd64 http://jp.archive.ubuntu.com/ubuntu trusty main restricted
universe multiverse
deb-amd64 http://jp.archive.ubuntu.com/ubuntu trusty-security main
restricted universe multiverse
deb-amd64 http://jp.archive.ubuntu.com/ubuntu trusty-updates main restricted
universe multiverse
#deb http://archive.ubuntu.com/ubuntu trusty-proposed main restricted
universe multiverse
#deb http://archive.ubuntu.com/ubuntu trusty-backports main restricted
universe multiverse


deb-i386 http://jp.archive.ubuntu.com/ubuntu trusty main restricted universe
multiverse
deb-i386 http://jp.archive.ubuntu.com/ubuntu trusty-security main restricted
universe multiverse
deb-i386 http://jp.archive.ubuntu.com/ubuntu trusty-updates main restricted
universe multiverse

deb-src http://jp.archive.ubuntu.com/ubuntu trusty main restricted universe
multiverse
deb-src http://jp.archive.ubuntu.com/ubuntu trusty-security main restricted
universe multiverse
deb-src http://jp.archive.ubuntu.com/ubuntu trusty-updates main restricted
universe multiverse
#deb-src http://archive.ubuntu.com/ubuntu trusty-proposed main restricted
universe multiverse
#deb-src http://archive.ubuntu.com/ubuntu trusty-backports main restricted
universe multiverse

clean http://jp.archive.ubuntu.com/ubuntu
```

Figure 5.3: Example of *mirror.list* in *apt-mirror* in server for host.

Internet. Each AP and host updates the system software using the server respectively. As a result, even if the Internet is not available for APs or hosts, they can still update the system software using the local network or the elastic WLAN system.

```
# newer versions of the distribution.
deb http://192.168.11.30/ubuntu/ trusty main restricted
deb-src http://192.168.11.30/ubuntu/ trusty main restricted

deb http://192.168.11.30/ubuntu/ trusty-updates main restricted
deb-src http://192.168.11.30/ubuntu/ trusty-updates main restricted

deb http://192.168.11.30/ubuntu/ trusty universe
deb-src http://192.168.11.30/ubuntu/ trusty universe
deb http://192.168.11.30/ubuntu/ trusty-updates universe
deb-src http://192.168.11.30/ubuntu/ trusty-updates universe

deb http://192.168.11.30/ubuntu/ trusty multiverse
deb-src http://192.168.11.30/ubuntu/ trusty multiverse
deb http://192.168.11.30/ubuntu/ trusty-updates multiverse
deb-src http://192.168.11.30/ubuntu/ trusty-updates multiverse

deb http://192.168.11.30/ubuntu/ trusty-backports main restricted universe
multiverse
deb-src http://192.168.11.30/ubuntu/ trusty-backports main restricted
universe multiverse

deb http://192.168.11.30/ubuntu trusty-security main restricted
deb-src http://192.168.11.30/ubuntu trusty-security main restricted
deb http://192.168.11.30/ubuntu trusty-security universe
deb-src http://192.168.11.30/ubuntu trusty-security universe
deb http://192.168.11.30/ubuntu trusty-security multiverse
deb-src http://192.168.11.30/ubuntu trusty-security multiverse
```

Figure 5.4: Example of *source.list* for host.

5.3.7 Correctness of System Software Update

As we mentioned, each of the hosts and APs in our currently implemented elastic WLAN system uses Linux based operating system. Most of the Linux based system uses the public key cryptography based concept called *SecureApt* to verify the integrity and the correctness of the downloaded packages from *Apt* package repositories [88–90]. *GNU Privacy Guard* (GPG) is a tool used in *SecureApt* to sign files and check their signatures. The open source developer and the package maintainers generate and publish a list of checksums calculated by secure hash functions from their packages and sign that list with their private GPG key. Each *sources.list* entry points *APT* to a release file *Release.gpg* signature that contains *md5* checksums of other files in the archive. *SecureApt* maintains a key ring with public GPG keys for the outside world. During the update process, upon package download and installation, it can verify the integrity of the checksum and the software package based on those verified the keys and checksums.

On the other hand, if it cannot find the corresponding *Release.gpg* or if the signature is incorrect, it will complain, and will make note that the packages files that the release file points to, and all the packages listed therein, are from an untrusted source.

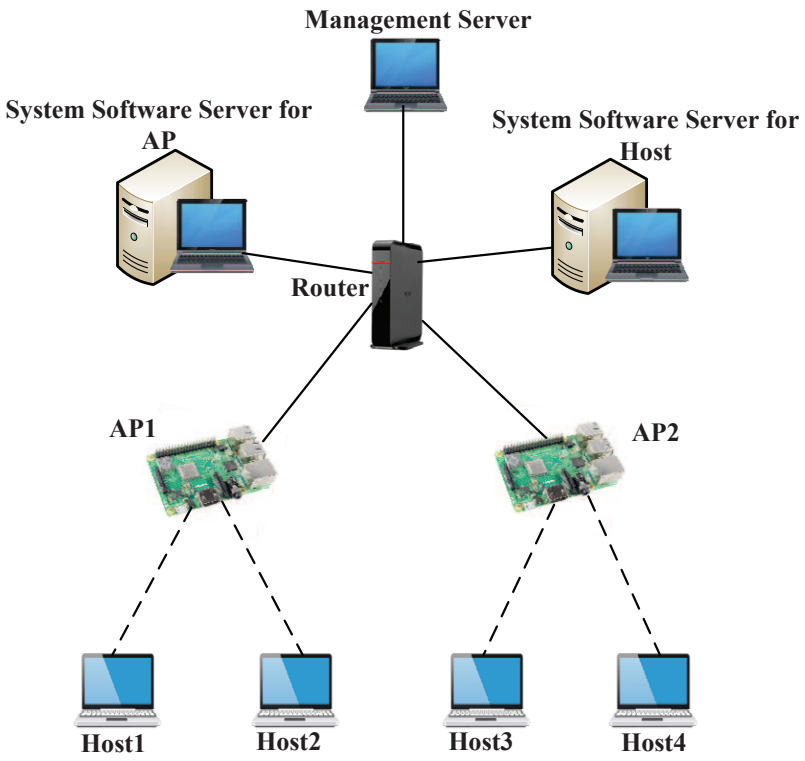


Figure 5.5: Elastic WLAN system topology with update servers.

5.4 Implementation of User Authentication System Function

In this section, we present the implementation of the user authentication function in the elastic WLAN system testbed.

5.4.1 System Configuration

When any user host wants to join the elastic WLAN system, it goes through the authentication process using the *RADIUS* server. *RADIUS* can authenticate a Wi-Fi user with the *802.1X* protocol [91]. In our current implementation, *802.1X* has the following three components:

1. **Host or supplicant**: a user that wants to access to the Internet.
2. **RADIUS client**: the AP in the elastic WLAN system.
3. **Authentication server**: the server that manages the information of the authenticators and the hosts.

Figure 5.6 shows the overview of the authentication process for a new user host. Whenever a user host tries to join the network, the supplicant or host requests it with the identity (SSID and password) for the authentication. Upon receiving the identity, the *RADIUS* client forwards the request to the authentication server. The authentication server verifies the information and replies back with success or deny.

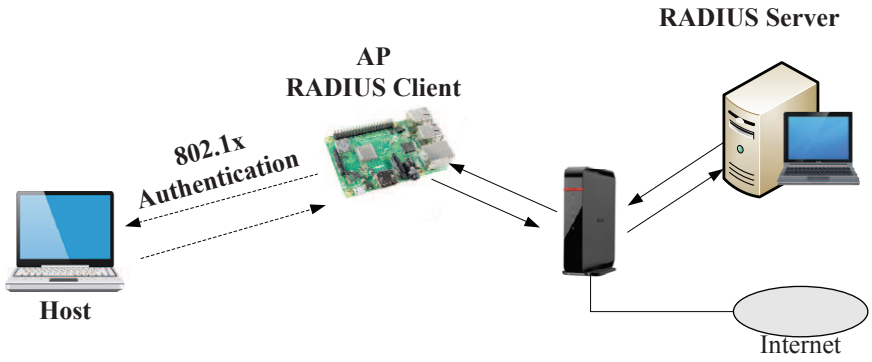


Figure 5.6: Elastic WLAN system authentication process.

5.4.2 Installation of FreeRADIUS

We use the *FreeRADIUS* to prevent any unauthorized access to the elastic WLAN system. To authenticate any newly joining host *FreeRADIUS* is adopted as the open source popular *RADIUS* server software that has been used in many organizations, due to the following features:

- open and scalable solution
- easy modification
- separation of security and communication processes
- adaptable to most security systems.

The following command is used to install the *FreeRADIUS* server with the necessary packages:

- sudo apt-get install freeradius

5.4.3 Configuration of Server

The AP in the network are termed as the *RADIUS* client. Any AP is added as a client in the server, so that the server can accept any request to process it. The AP and the authentication server use the shared secret key to verify each other. In order to add the AP, the client configuration file, namely *client.conf* in the */etc/freeradius/* directory in the RADIUS server, must be modified. The individual AP can be added with each name and IP address. To modify the file, the following command are used:

- sudo nano /etc/freeradius/client.conf

Figure 5.7 shows the example client configuration file for the AP. Here, $SoftAP$ represents the name of the AP, *ipaddr* does the IP address of the AP, and *secret* does the shared secret key between the AP and the server. This key is used to authenticate the AP in the *RADIUS* server.

5.4.4 Registration of System User

The next step is to register the user in the server by modifying the client configuration file, namely *user* in the */etc/freeradius/* directory of the server:

```
client SoftAP77
{
        ipaddr  = 192.168.11.17
        netmask = 24
        secret = strong-passphrase

}
```

Figure 5.7: Adding client (AP) information.

- sudo nano /etc/freeradius/users

The username and password can be added using the following line:

- userid Cleartext-Password := password

Here, *userid* is the user name of the host and *password* describes the password for the user. Finally, the *RADIUS* server must be restarted to apply the changes using the following two commands:

- sudo service freeradius stop
- sudo service freeradius start

5.4.5 Configuration of hostapd for RADIUS Server

To enable the authentication process using the *RADIUS* server, the *hostapd* deamon [70, 71] in each AP must be configured properly. The *hostapd.conf* file is modified with the corresponding IP address of the server along with the port number and other related fields. Figure 5.8 shows the necessary configuration to enable the *RADIUS* server. Here, *auth_server_addr* represents the *RADIUS* server IP address, *own_ip_addr* does the IP address of the AP, and *auth_server_ shared_secret* does the shared secret key that is compared with the corresponding one of each AP in the *client.conf* file in the *RADIUS* server. The elastic WLAN system controls the number of active APs and the host associations in the network running the active AP configuration algorithm in the management server. Figure 5.9 shows the commands with the necessary parameters to connect an existing host to a new active AP. Here, 802.1*x.identy* and 802.1*x.password* represents the ID and password of the user, and *con-name SoftAP77* does the name of the AP where the host will be associated. Using the *nmcli* tool, the association is changed.

5.5 Evaluations

In this section, we evaluate the correctness of the two implemented functions using the elastic WLAN system testbed.

5.5.1 Hardware and Software Specification

Table 5.1 shows the hardware and software specifications adopted in this elastic WLAN system testbed.

```
##### hostapd configuration file ###
###Radius Setup
own_ip_addr=192.168.11.17
auth_server_addr=192.168.11.32
auth_server_port=1812
auth_server_shared_secret=password
acct_server_addr=192.168.11.32
acct_server_port=1813
acct_server_shared_secret=password
###WPA Setup
wpa=1
wpa_key_mgmt=WPA-EAP
```

Figure 5.8: Configuration of *hostapd* deamon in AP for *RADIUS*.

```
sudo -S nmcli connection add type wifi
con-name SoftAP77 ifname wlp3s0 ssid SoftAP77
-- wifi-sec.key-mgmt wpa-eap 802-1x.eap ttls
802-1x.phase2-auth mschapv2 802-1x.identity
userid 802-1x.password password &&
sudo nmcli con up SoftAP77
```

Figure 5.9: Remote host change association.

Table 5.1: Hardware and software in testbed.

hardware/software	specification
server PC	OS: Ubuntu LTS 14.04 model: Lesance W255HU
client PC	OS: Ubuntu LTS 16.04 model: Toshiba Dynabook R731/B
AP	OS: Raspbian Jessie 4.4.13-v7+ model: Raspberry Pi 3
nmcli	version: 0.9.8.8
hostapd	version: 2.3
FreeRADIUS	version: 2.1.12
apt-mirror	version: 0.5.1-1
Apache2	version: 2.4.10

5.5.2 Evaluation of System Software Update Function

First, we evaluate the system software update function for APs and hosts in the testbed. The local repository servers periodically download the latest versions of the system files and packages, and update them. Then, any AP and host can download them from the server and update them. It is found that the update time can be reduced from the direct Internet access. Table 5.2 compares the time required to download and install the necessary packages and files when the size of the package file is changed in three cases. Clearly, this local server can speed up the system update than the direct access to the Internet. Since all the APs and hosts are using the updated server,

it can reduce the unnecessary bandwidth cost. Table 5.3 compares the time for installing various

Table 5.2: Time comparison for system update.

software size	from update server (sec.)	from Internet (sec.)
Case-1 (5.3Mb)	6	7
Case-2 (7.2Mb)	9	12
Case-3 (20.9Mb)	17	22

open source application software from our server and the Internet. Again, this server can update the necessary packages with less time.

Table 5.3: Time comparison for open source application software/tool installation.

application Software/tools	from update server (sec.)	from Internet (sec.)
apt-mirror	4	4
Apache2	12	18
db5.3-sql-util	8	10
VLC media player	25	39

5.5.3 Evaluation of User Authentication Function

Then, we evaluate the user authentication function. For any new host, this function authenticates the user by comparing the information of the user with the information stored in the *RADIUS* server. Any unauthorized user is denied from the access to the elastic WLAN system.

5.6 Summary

In this chapter, first, we implemented the *system software update function* for APs and hosts in the elastic WLAN system, to enhance the system performance and security. This function periodically downloads the latest system software to the local repository servers, and installs it into the APs and hosts. Then, we implemented the *user authentication function* to authenticate a newly joining host user using the *RADIUS* server. Finally, these functions are evaluated with the elastic WLAN system testbed. In the next chapter, we will present AP power transmission minimization approach.

Chapter 6

Static AP Transmission Power Minimization

In this chapter, we present the *AP transmission power minimization approach* as an extension of the *active AP configuration algorithm* for the *elastic WLAN system*. First, we describe the background of this proposal followed by some related works. Then, we present the AP transmission power minimization approach. Finally, we evaluate the proposal through simulations using the WIMNET simulator and testbed experiments.

6.1 Introduction

In the *active AP configuration algorithm* in Chapters 3 and 4, we assume that any AP uses the default maximum transmission power to provide the strongest signal to hosts during the communication. When a large number of APs is deployed in a network field, this default maximum power may suffer from large power consumptions, large radio interferences, and short lifetimes of wireless nodes. Instead, the transmission power of the APs on a large scale should be properly controlled to minimize the power consumption and increase the lifetime of the APs [92–94], while ensuring the required data transmission speed or throughput against the associated host.

In this chapter, we propose the *transmission power minimization* extension in the active AP configuration algorithm to further reduce the energy consumption. After the active AP configuration algorithm finds the active APs, the AP-host associations, and the AP channel assignments, assuming the use of the maximum transmission power, this extension minimizes the transmission power of each active AP such that it satisfies the *minimum throughput constraint*. The throughput estimation model [11] has been used to estimate the throughput for each transmission power, where the model parameter values for different transmission powers are obtained from extensive measurements.

6.2 Related Works

In this section, we briefly introduce related works in literature to this paper. A substantial amount of research works has been found in literature that focus on the power consumption at each active AP. For instance, [93] explores a genetic algorithm to control the power of each active APs in industrial wireless local area network (IWLAN).

[94] analyses the disadvantages of using the highest transmission power of the AP and explains that it can reduce the lifetime of the device, increases the interference with the nearby APs and thus, reduce the performance.

[95] proposes a channel assignment and transmission power control algorithm for multi-rate WLAN. The authors consider both overlapping and non-overlapping channel assignment. In the first step, the channel is assigned to the AP while the other parameters such as power and positions were constant. Then, the second step improves the network throughput by optimizing the transmission power of the APs.

[96] proposes a joint approach of power tuning and partially overlapping channel (POC) assignment. Based on the assigned POC, the algorithm performs an effective power tuning to improve the network performance.

[97] provide a heuristic approach to control the power and channel allocation process iteratively. The authors select a set of level of transmission power for the APs and the available channel in order to determine the best set-up for each AP. Then, based on the initial channel and power assignment, the algorithm try to assign the best channel and power level by error and trial method. The algorithm finds the best solution after a predefined number of trial so that the selected power and assigned channel can improve the overall network throughput.

6.3 Static AP Transmission Power Minimization Approach

In this section, we present the proposed *static transmission power minimization* approach.

6.3.1 Overview

To reduce the power consumption of the active APs, the *transmission power minimization* phase is proposed as the additional phase of the active AP configuration algorithm. This phase is realized through the following three stages. In the *first stage*, extensive throughput measurements are conducted in the target network field under different transmission powers for the *Raspberry Pi* AP. In the *second stage*, the parameter values of the throughput estimation model for different transmission powers are found by applying the *parameter optimization tool* [12] using the measurement results with different transmission powers. In the *third stage*, the minimum transmission power is selected to satisfy the *minimum host throughput constraint* for any AP through estimating the throughput for each transmission power using this model. Then, the transmission power of any *Raspberry Pi* AP is controlled by a Linux command.

6.3.2 Throughput Measurements under Different Transmission Powers

To observe the effect of the different transmission power in the throughput and estimate the value of P_1 in the model, extensive throughput measurements are conducted by using different transmission powers, namely, $5dBm$, $10dBm$, $20dBm$, and $30dBm$, for *Raspberry Pi* AP on the 3rd floor of Engineering Building-2 of Okayama University. The locations of the AP and the host in measurements are illustrated in Figure 6.1. Table 6.1 shows the adopted hardware and software in measurements.

The value of P_1 in the throughput estimation model is found for each transmission power based on measurements, where P_1 is related to the transmission power. The other parameters can be fixed at the ones with the full transmission power in Section 6.4.1. *iperf 2.05* is used to generate TCP packets during the measurement [24]. Figure 6.2 offers the average throughput results among 10 measurements for each host location with the four different transmission powers. This figure indicates that 1) the host throughput becomes smaller when the transmission power is reduced

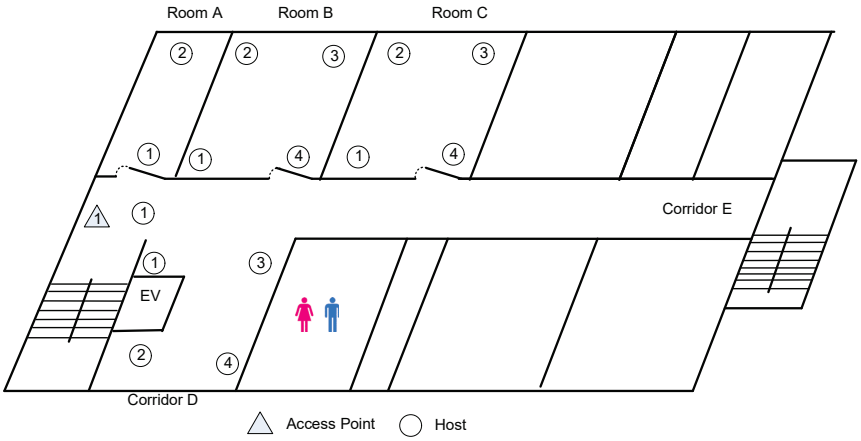


Figure 6.1: Measurement field.

Table 6.1: Hardware and software in measurements.

hardware/software	specification
server PC	OS: Ubuntu LTS 14.04 model: Lesance W255HU
client PC	OS: Ubuntu LTS 16.04 model: Toshiba Dynabook R731/B
AP	OS: Raspbian Jessie 4.4.13-v7+ model: Raspberry Pi 3
iperf	Version: 2.05

except the host position $E1$, and 2) the throughput is not always proportional to the transmission power, because the *modulation and coding scheme (MCS)* [98] is changed stepwise. We found that, the throughput is decreased as the transmission power and the distance between the AP and the host is decreased. The host position $E1$ is very near from the AP. Even if the transmission power of the AP is reduced to $5dBm$, the measured throughput is not decreased, where it is just fluctuated. At $E1$, the RSS at the AP and the host with the smallest transmission power is sufficiently large to offer the highest throughput using the fastest MCS.

6.3.3 Throughput Measurements at Near Locations for Two Hosts

The throughput is not reduced even if the transmission power at the AP is reduced at the host location near from the AP. To confirm it using two hosts, we conduct experiments in the similar field in Figure 6.3. The transmission power of the AP is changed from $30dBm$ to $0dBm$ and the throughput is measured using *iperf*. Figure 6.4 shows the measured RSS and throughput results for the different transmission powers. Even when the power becomes $0dBm$, the reduction of RSS is very small and the throughput is not reduced.

6.3.4 P_1 for Different Transmission Powers

Then, the value of P_1 in the throughput estimation model is estimated by applying the measurement results in Figure 6.2 to the parameter optimization tool [11]. Table 6.2 shows the value of P_1 for each transmission power. It is compared with the measured value of P_1. This table reveals that

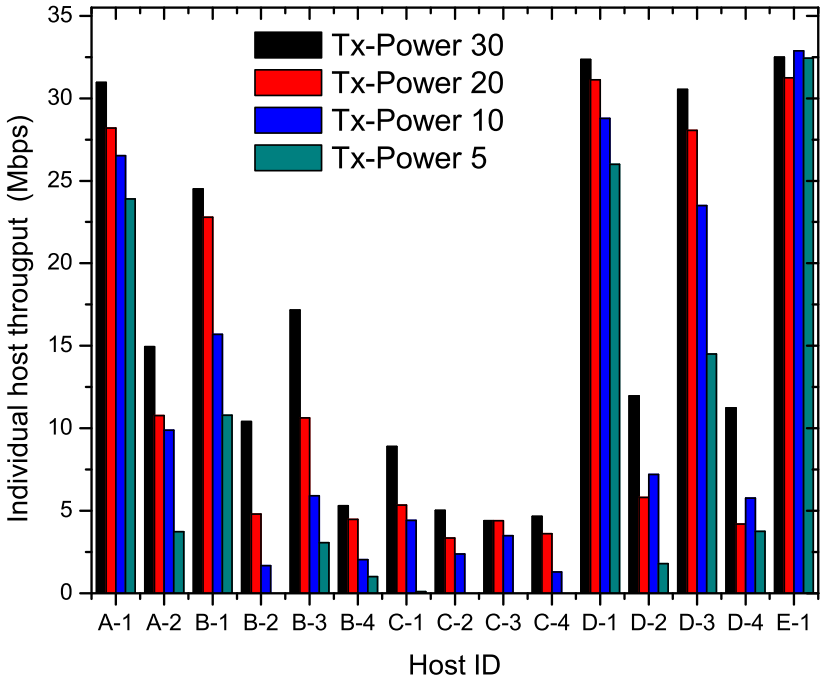


Figure 6.2: Throughput results at different transmission powers.

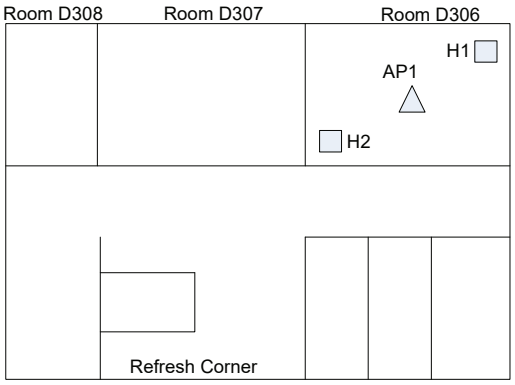
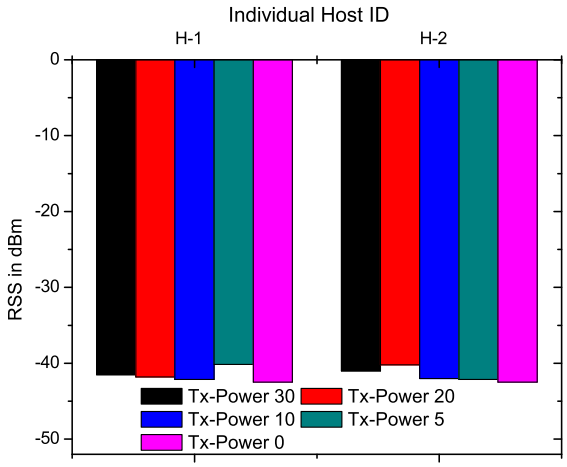


Figure 6.3: Network topology for near two-host experiments.

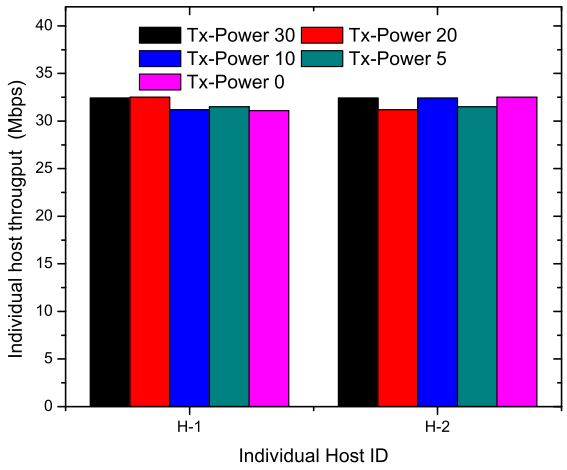
both values are nearly the same, which confirms the accuracy of the throughput estimation model. It should be pointed that, for practical use of the proposal, it is necessary to reduce the number of throughput measurements required to tune the model parameters, which will be explored in Chapter 8.

Table 6.2: P_1 values by estimation and measurement for each transmission power.

P_1 for 1 meter distance.			
transmission power (dBm)	P_1 by est. (dBm)	P_1 by meas. (dBm)	difference (dBm)
30	-34.00	-34.20	.20
20	-38.20	-38.40	.20
10	-44.50	-44.60	.10
5	-52.60	-52.20	.40



(a) RSS results.



(b) Throughput results.

Figure 6.4: Measurement results with different transmission powers for near two-host experiments.

6.3.5 Static Transmission Power Minimization

Using the throughput estimation model, the following procedure is proposed to statically minimize the transmission power of each active AP (let *AP i* here) that satisfies the *minimum host throughput constraint*, which is added to the active AP configuration algorithm as the fourth phase. To improve the accuracy while reducing the computation time, the throughput is first estimated at an equal interval of the transmission power using the throughput estimation model. Then, the throughput at an arbitrary transmission power is obtained by the interpolation of the results at the two adjacent powers. We calculate the *average host throughput* that should be greater or equal to the throughput constraint G as follows:

$$aveS_i = \frac{1}{\sum_j \frac{1}{s_{ij}}} \geq G \tag{6.1}$$

Here, the *minimum host throughput constraint* is defined in this algorithm so that the throughput of every host exceeds the given threshold G on average when all the hosts are communicating simultaneously.

1. Calculate the link throughput between AP i and its associated host j, namely S_{ij}^{5}, S_{ij}^{10}, S_{ij}^{20}, and S_{ij}^{30}, using the throughput estimation model, when the transmission power of *AP i* is reduced to $5dBm$, $10dBm$, $20dBm$, and $30dBm$, respectively.

2. Calculate the corresponding average host throughput for *AP i*, $aveS_i^{5}$, $aveS_i^{10}$, $aveS_i^{20}$, and $aveS_i^{30}$, respectively, using Eq. (6.1).

3. Calculate the average host throughput for *AP i*, $aveS_i^{p}$, at the arbitrary transmission power p, from them by the following procedure:

   - if $p \leq 5 \rightarrow aveS_i^{p} = aveS_i^{5}$,
   - else if $p \leq 10 \rightarrow aveS_i^{p} = \{(10-p) \times aveS_i^{5} + (p-5) \times aveS_i^{10}\}/5$,
   - else if $p \leq 20 \rightarrow aveS_i^{p} = \{(20-p) \times aveS_i^{10} + (p-10) \times aveS_i^{20}\}/10$,
   - else $aveS_i^{p} = \{(30-p) \times aveS_i^{20} + (p-20) \times aveS_i^{30}\}/10$.

4. Determine the minimum transmission power TX_i that satisfies the average host throughput constraint by:

   - if $aveS_i^{5} \geq G \rightarrow TX_i = 5$,
   - else if $aveS_i^{p} \geq G$ for $p = 6, 7, 8..., 30 \rightarrow TX_i = p$.

6.4 Evaluations by Simulations

In this section, we evaluate the static transmission power minimization approach through simulations in two network topologies using the *WIMNET simulator* [16]. The same parameter values of Chapter 4 are used here. Table 4.1 shows the adopted hardware and software in simulations, and Table 4.2 provides the parameters in the WIMNET simulator.

6.4.1 Parameters of Throughput Estimation Model

The parameter values of the throughput estimation model for different transmission powers at an AP were derived based on measurement results using the *parameter optimization tool* [11]. Table 6.3 reveals the parameter values of the throughput estimation model used in simulations.

Table 6.3: Parameter values in throughput estimation model.

parameter	value
α	3.00
W_1	0
W_2	7
W_3	6
W_4	7
W_5	2.30
W_6	3.40
W_7	5
a	34
b	57
c	8

6.4.2 Simulation in Small Network Topology

First, the *small network topology* in Figure 6.5 is considered in simulations. It basically models the third floor of Engineering Building-2 in Okayama University. 15 hosts and six APs are regularly distributed in the corridor and rooms with $7m \times 6m$ and $3.5m \times 6m$. $G = 10Mbps$ and $B^a = \infty$ are adopted for the minimum host throughput constraint and the bandwidth limit constraint, respectively. Table 6.4 shows the simulation results where the average minimum host throughput,

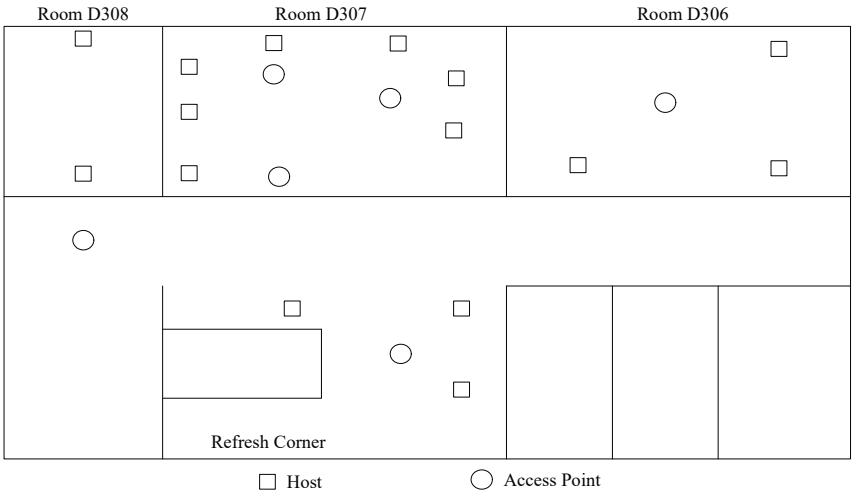


Figure 6.5: Small network topology for simulations.

the overall throughput, and the average transmission power is summarized. The results indicate that the proposal can reduce the transmission power by 26.13% while satisfying the minimum host

throughput constraint and keeping the overall throughput before applying the proposal. Figure 6.6 compares the transmission power in each AP before and after the proposal. This figure implies that it can reduce the transmission power significantly while maintaining the performance, which confirms the effectiveness of the proposal.

Table 6.4: Simulation results in small network topology.

instance	Results	
proposal	before	after
ave. min. throughput (Mbps)	10.01	10.10
overall throughput (Mbps)	151.08	150.31
ave. trans. power (dBm)	30.00	22.16

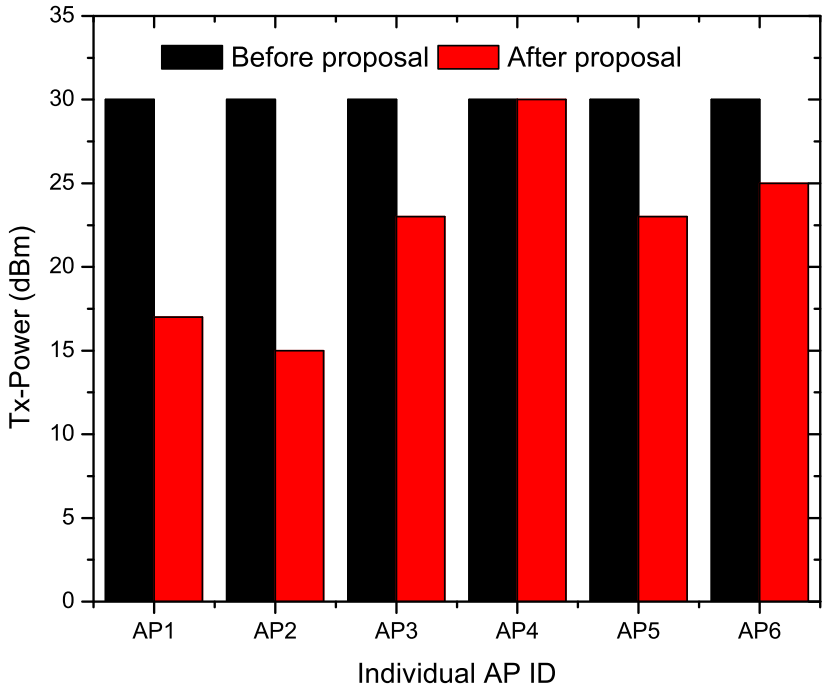


Figure 6.6: Transmission powers in small network topology.

6.4.3 Simulation in Large Network Topology

Next, the *large network topology* in Figure 6.7 is considered. The network field has the same size as the *small network topology*, but 40 hosts and 18 APs are allocated. For the minimum host throughput constraint, $G = 10Mbps$ is adopted. Table 6.5 reveals the simulation results, which indicate that the proposal reduces the transmitting power by 51.20% in the large topology. Figure 6.8 shows the transmission power reduction in each AP by the proposal. Once more, it was observed that the proposal has reduced the transmission power prominently while keeping the throughput performance.

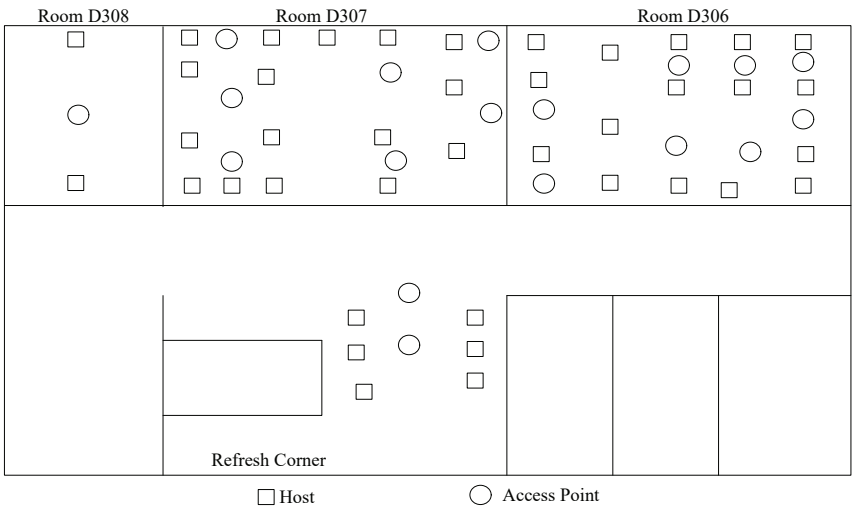


Figure 6.7: Large network topology for simulations.

Table 6.5: Simulation results in large network topology.

instance	Results	
proposal	before	after
ave. min. host throughput (Mbps)	10.40	10.01
overall throughput (Mbps)	413.83	400.04
ave. trans. power (dBm)	30.00	14.64

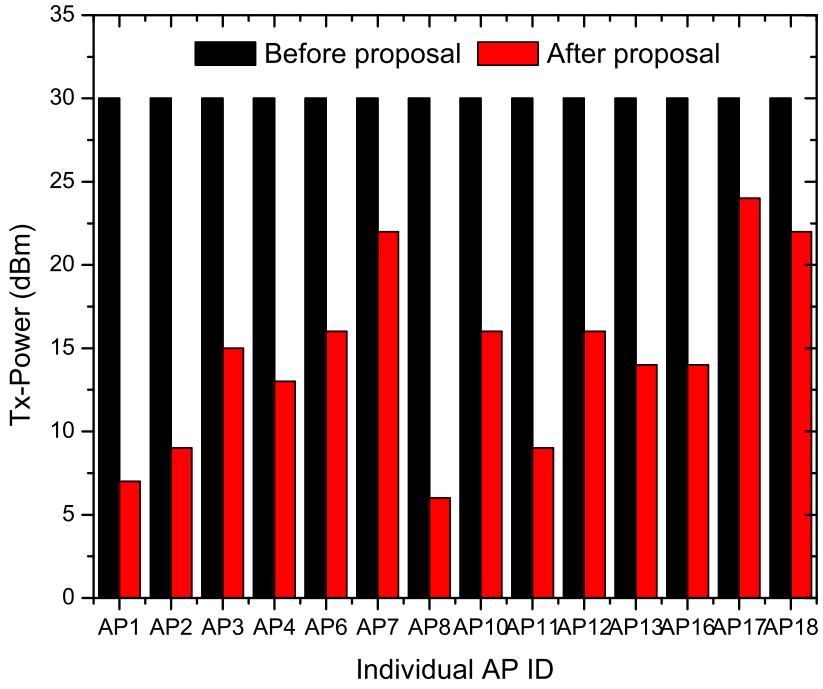


Figure 6.8: Transmission powers in large network topology.

6.5 Evaluations by Testbed Experiments

In this section, we evaluate our proposal through experiments using the elastic WLAN system testbed [15] in two buildings at Okayama University.

6.5.1 Experiments in Engineering Building-2

First, the third floor in *Engineering Building-2* is adopted in experiments. Two scenarios are considered using a different number of rooms. Each room has $7m \times 6m$ size, where one *Raspberry Pi* for the AP and two Linux PCs for hosts are allocated. Then, two rooms are used in 2×4 scenario in Figure 6.9, and three rooms are in 3×6 scenario in Figure 6.10.

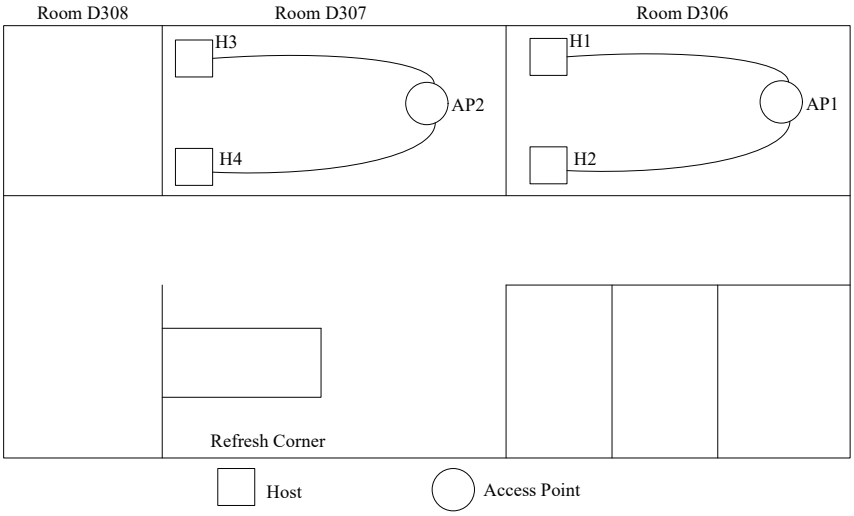


Figure 6.9: Testbed topology for 2×4 scenario in Engineering Building-2.

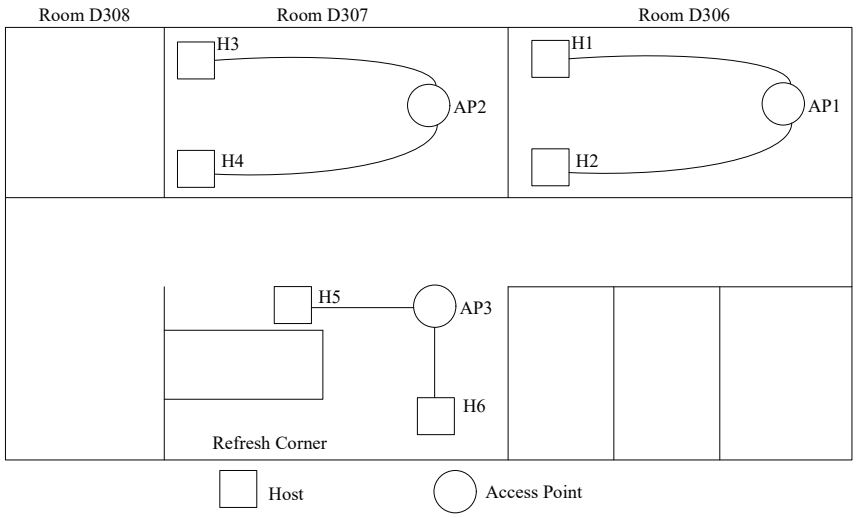


Figure 6.10: Testbed topology for 3×6 scenario in Engineering Building-2.

Table 6.6 compares the overall throughput and the average transmission power before and after applying the proposal in two scenarios. The overall throughput is similar between them, whereas the transmission power is reduced greatly by the proposal. Figures 6.11 and 6.12 show

Table 6.6: Experiment results in Engineering Building-2.

scenario	2×4		3×6	
proposal	before	after	before	after
overall throughput (Mbps)	60.22	58.78	89.26	84.75
ave. trans. power (dBm)	30.00	11.00	30.00	10.67

the transmission power change at each AP.

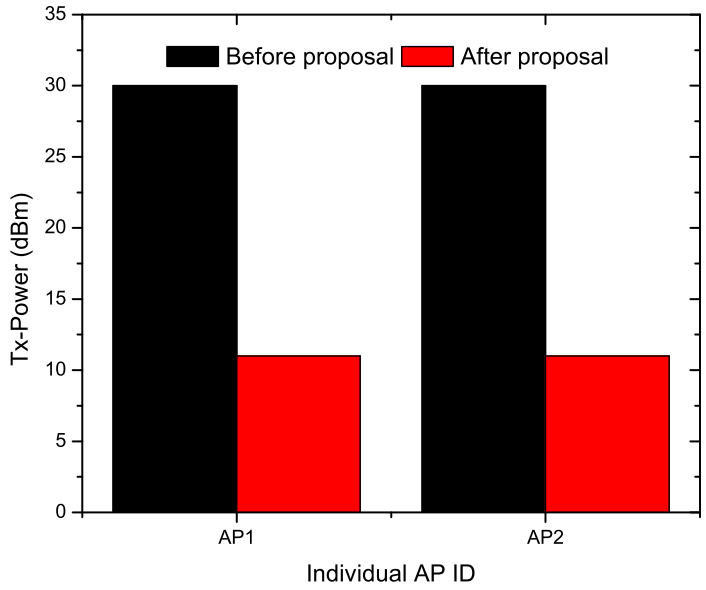


Figure 6.11: Transmission powers for 2×4 scenario in Engineering Building-2.

6.5.2 Experiments in Graduate School Building

Next, the second floor in *Graduate School Building* is adopted in experiments with two scenarios. In 2×4 scenario, one room with $9m \times 5.5m$ size is used, where two *Raspberry Pi* APs and four hosts are allocated, as indicated in Figure 6.13. In 3×6 scenario, additionally, one AP and two hosts are allocated in another room with $3.5m \times 5.5m$ size, as shown in Figure 6.14.

Table 6.7 compares the overall throughput and the average transmission power before and after applying the proposal. Figure 6.15 and 6.16 show the transmission power at each AP namely, the overall throughput is similar between them, whereas the transmission power is reduced significantly.

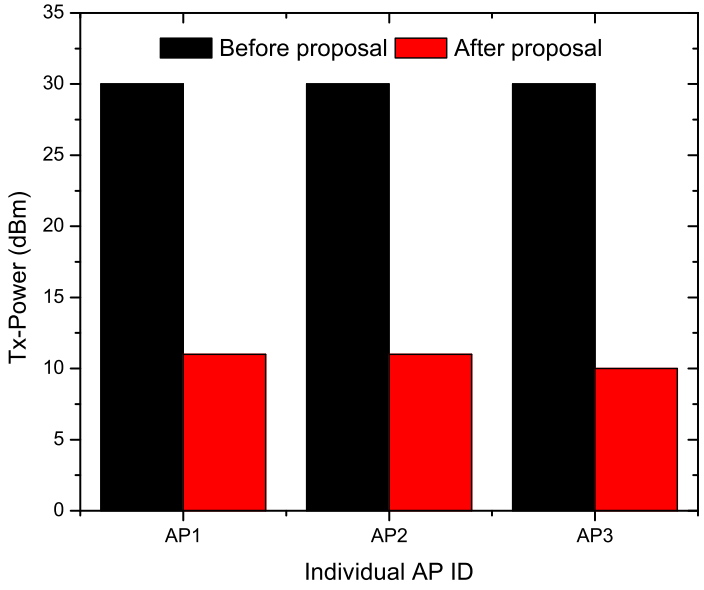


Figure 6.12: Transmission powers for 3×6 scenario in Engineering Building-2.

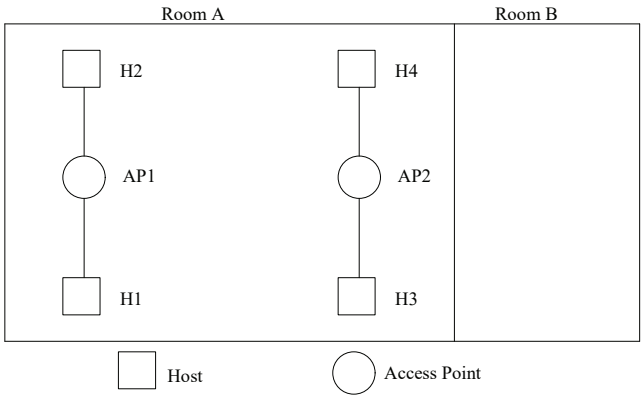


Figure 6.13: Testbed topology for 2×4 scenario in Graduate School Building.

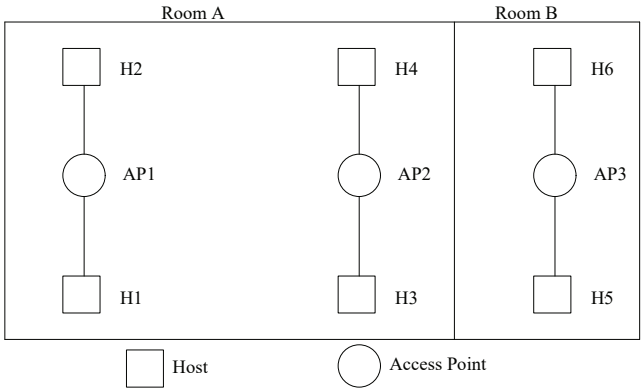


Figure 6.14: Testbed topology for 3×6 scenario in Graduate School Building.

Table 6.7: Experiment results in Graduate School Building.

scenario	2×4		3×6	
proposal	before	after	before	after
overall through. (Mbps)	68.99	62.19	97.50	92.07
ave. trans. power (dBm)	30.00	8.00	30.00	8.00

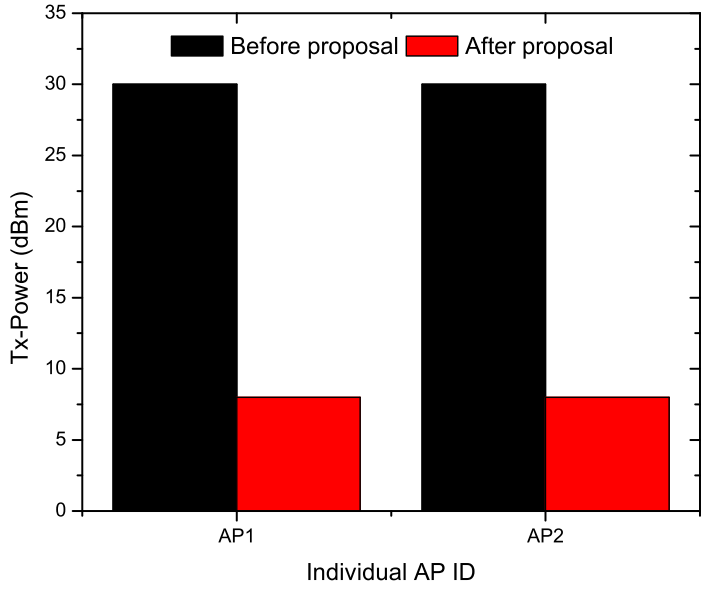


Figure 6.15: Transmission powers for 2×4 scenario in Graduate School Building.

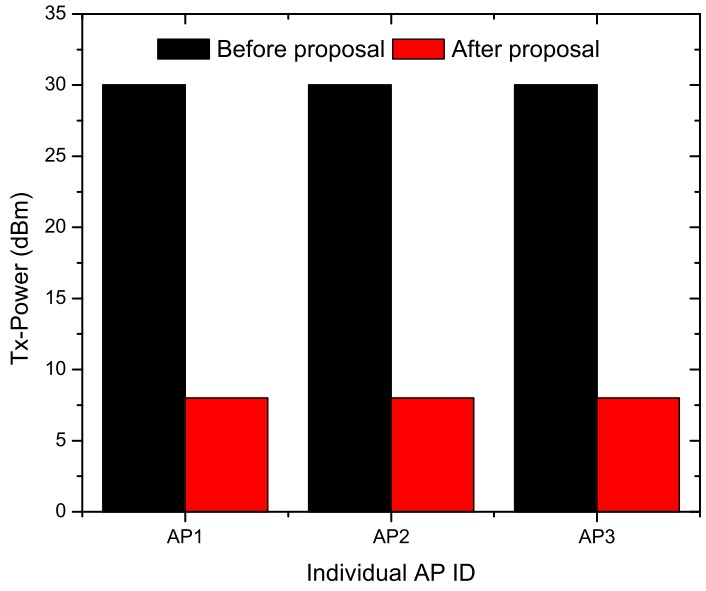


Figure 6.16: Transmission powers for 3×6 scenario in Graduate School Building.

6.6 Summary

In this chapter, we proposed the *static transmission power minimization approach* as the extension of the active AP configuration algorithm to further minimize the power consumption. Then, we evaluated our proposal using simulations and testbed experiments. The results confirmed that the proposal can reduce the transmission power significantly while keeping the throughput performance. In the next chapter, we will present the dynamic transmission power minimization approach using the PI feedback control.

Chapter 7

Dynamic AP Transmission Power Minimization

In this chapter, we propose an *AP transmission power minimization approach* using the *PI feedback control* in the *elastic WLAN system*. Firstly, we describe the drawback of the previous static approach. Secondly, we describe the procedure for this dynamic approach. Thirdly, we describe the implementation of the proposal in the *elastic WLAN system testbed*. Finally, we evaluate the proposal through testbed experiments.

7.1 Introduction

In Chapter 6, we proposed the *static AP transmission power minimization approach*. This approach estimates the minimum transmission power to achieve the required throughput using the *throughput estimation model* [11]. Unfortunately, this approach needs a lot of throughput measurements to obtain the accurate model parameters under various transmission powers. In this chapter, we propose an *dynamic AP transmission power minimization approach* using the *PI feedback control* [23] to avoid extensive measurements. The initial transmission power is examined from the difference between the measured *received signal strength (RSS)* at the AP from the host and the estimated RSS necessary for the target throughput. Then, this power is dynamically minimized by using the *PI feedback control*, such that the measured throughput using *iperf* [24] achieves the target one.

For evaluations, we implement the proposal in the *elastic WLAN system testbed* using *Raspberry Pi* APs, and conduct experiments with five topologies. The results confirm that the proposal has significantly reduced the transmission power while achieving the target throughput.

7.2 Drawbacks of Static Approach

In the previous approach, the AP transmission power is statically selected using the *throughput estimation model*, [11], and is fixed during communications. Thus, the accuracy of the model determines the performance of this method. For the accurate model, the parameters are tuned through the two steps: 1) throughputs are measured at different host locations with various transmission powers, 2) the model parameter values are optimized by applying the *parameter optimization tool* with the throughput results. The throughput at an arbitrary transmission power is obtained by the interpolation of the estimated results at the two adjacent measured powers. Then, the least transmission power is selected to satisfy the required throughput for each AP.

Unfortunately, this model-based static method has two drawbacks. One drawback is the high load of throughput measurements that are required to obtain the accurate model. That is, the host and the AP must be located at different designate positions. Several transmission powers must be selected at each position, then the throughput must be measured several times for each condition of the location and the power. Nevertheless, due to frequent fluctuations of throughputs, one throughput measurement requires several minutes using *iperf* and even the measurement for a single link could take a whole day. Another drawback is the throughput difference from the model result, due to the model accuracy and the environmental changes such as the interference from other Wi-Fi devices, the weather, and the congestion. Hence, the transmission power needs to be dynamically adjusted during communications.

7.3 Dynamic AP Transmission Power Minimization Approach

In the section, we propose the AP transmission power minimization method using the PI feedback control.

7.3.1 Overview

Figure 7.1 shows the overview of the proposal, which consists of two phases.

The *initial power selection* examines the initial transmission power of the AP from the difference between the measured RSS at the AP from the host and the required RSS for the target throughput estimated by the model, assuming that the RSS is proportional to the transmission power. Here, the transmission power of the host is set $30dBm$.

The *dynamic power minimization* continues changing the transmission power by the PI feedback control. The actual throughput of the link between the server and the host is measured using *iperf* periodically, assuming that the *iperf* client is installed on the host. Then, the measured throughput is used to update the power by the PI control so that it could achieve the target throughput. In the implemented testbed, the throughput measurement and the power update are applied iteratively every $20sec$.

7.3.2 Initial Power Selection by Model

The initial transmission power of the AP, $TP(0)$ (dBm), is examined through the following procedure:

1. The required RSS, $Pd(Th_{tar})$ (dBm), evaluated with the throughput estimation model to achieve the target throughput, Th_{tar} $(Mbps)$:

$$Pd(Th_{tar}) = b - M - c \times ln(\frac{a}{Th_{tar}} - 1) \tag{7.1}$$

   where a, b, c, and M represent the constant coefficients. In this paper, $a = 34$, $b = 57$, $c = 8$, and $M = 120$, in [20], are adopted. This equation is derived from Eq. (3.3).

2. The RSS at the AP from the host, RSS_m, is measured when the maximum transmission power is used for both the AP and host.

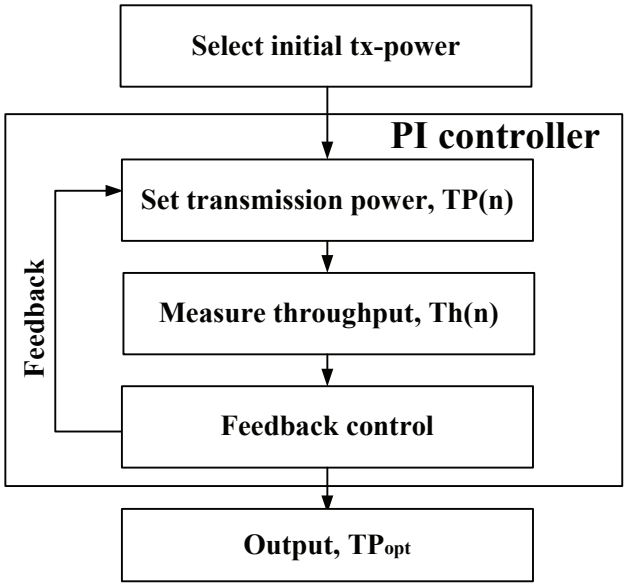


Figure 7.1: Overview of AP transmission power minimization method.

3. The difference between RSS_m and $Pd(Th_{tar})$, ΔPd, is calculated:

$$\Delta Pd = RSS_m - Pd(Th_{tar}) \tag{7.2}$$

4. The initial transmission power, $TP(0)$ (dBm), is calculated:

$$TP(0) = MaxTx - \Delta Pd \tag{7.3}$$

where $MaxTx$ represents the maximum transmission power of the AP. Here, $TP(0)$ must be in the feasible range:

$$TP(0) = \begin{cases} MaxTx & \text{if } TP(0) > MaxTx \\ MinTx & \text{if } TP(0) < MinTx \end{cases} \tag{7.4}$$

where $MinTx$ represents the minimum transmission power of the AP. In this paper, $MaxTx = 30dBm$ and $MinTx = 0dBm$ are adopted for the *Raspberry Pi* AP.

7.3.3 Dynamic Power Minimization by PI Control

The transmission power of the AP, $TP(n)$ (dBm), is updated iteratively for its minimization by the *PI feedback control*:

$$TP(n) = K_P \times (Th_{tar} - Th(n)) + K_I \times \sum_{i=0}^{n} (Th_{tar} - Th(i)) \tag{7.5}$$

where $TP(n)$ and $Th(n)$ indicate the transmission power and the measured throughput at the n-th iteration, Th_{tar} does the target throughput, and K_P and K_I do the P and I control gains, respectively. In this paper, $K_P = 0.4$, and $K_I = 0.0015$ are adopted.

In the implementation, the difference equation of the *PI control* is used:

$$TP(n) = TP(n-1) + K_P \times (Th(n-1) - Th(n)) + K_I \times (Th_{tar} - Th(n)) \tag{7.6}$$

7.3.4 Testbed Implementation

The proposed method is implemented on the elastic WLAN system testbed using *Raspberry Pi* APs.

7.3.4.1 Initial Power Selection

For the initial transmission power selection, the following procedure is implemented on the server:

1. The MAC and IP addresses of the AP and the associated hosts are collected.
2. The AP is requested to measure the RSS from each host for $30sec$ at $1sec$ interval using the maximum transmission power ($30dBm$), and send it to the server.
3. The average of the received RSS is calculated and the host (bottleneck host) with the smallest RSS will be identified.
4. The initial transmission power of the AP is calculated by the procedure in Section 7.3.2 using the smallest RSS.

7.3.4.2 Dynamic Power Minimization

For the dynamic transmission power minimization, the following procedure is implemented on the server:

1. The throughput to each host is measured for $20sec$ using *iperf*.
2. The average AP-host throughput is calculated.
3. $TP(n)$ is updated by Eq. (7.6).
4. $TP(n)$ is set as the transmission power of the AP.
5. Go back to 1.

7.4 Evaluations

In this section, we evaluate the proposed transmission power minimization method through experiments using the *elastic WLAN system testbed*.

7.4.1 Network Fields

In experiments, we consider five network topologies in two buildings in Okayama University. The two building have different structures where they were built at different years. The rooms in them are different in their sizes and are separated by different types of wall structure that has the different wall attenuation factors for the signal propagation. These two different network environments are selected to demonstrate that the proposal can be adopted in different kinds of environments and can save the power significantly.

7.4.2 Three Methods for Comparison

In evaluations, the throughput and transmission power results in three cases are compared to verify the effectiveness of the proposal, namely, *method-1: no minimization*, *method-2: dynamic power minimization only*, and *proposal: initial power selection and dynamic power minimization*. In *method-1)*, the transmission power is always set at the largest one ($30dBm$). In *method-2)*, the power is initialized at the largest one.

7.4.3 Throughput Constraint Setup

The *minimum average host throughput threshold (G)* is basically selected in three ways so that the final transmission power will become either under saturated (=0dBm), within the dynamic range (=0-30dBm), or over saturated (=30dBm). It is noted that G is selected in two ways when the power cannot be controlled within the range.

7.4.4 Experiments in Engineering Building-2

First, the third floor in Engineering Building-2 is adopted for *Topology 1 - Topology 3*.

7.4.4.1 Topology 1: One Host in Same Room

In *Topology 1*, one *Raspberry Pi* AP and one *Linux PC* host are allocated in the same room of the size of $7m \times 6m$, as shown in Figure 7.2. $G = 10Mbps$ and $G = 40Mbps$ are adopted for the throughput constraint. Figures 7.3 and 7.4 present the changes of the throughput and transmission

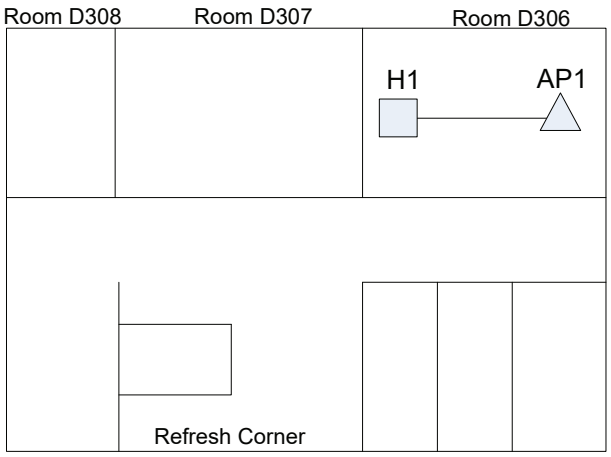
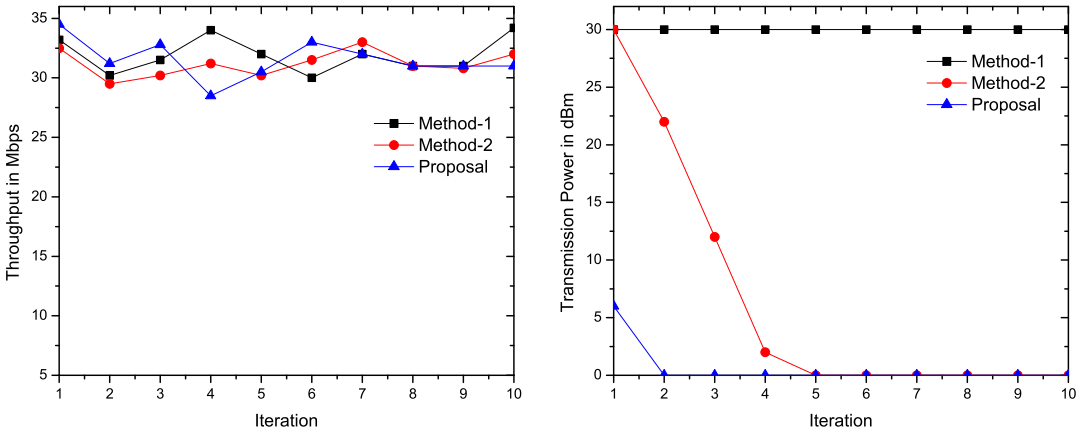


Figure 7.2: *Topology 1*: one host in one room.

power results for $G = 10Mbps$ and $G = 40Mbps$, respectively. Table 7.1 shows the average throughout. The experiment was executed three times and their average results were used here.

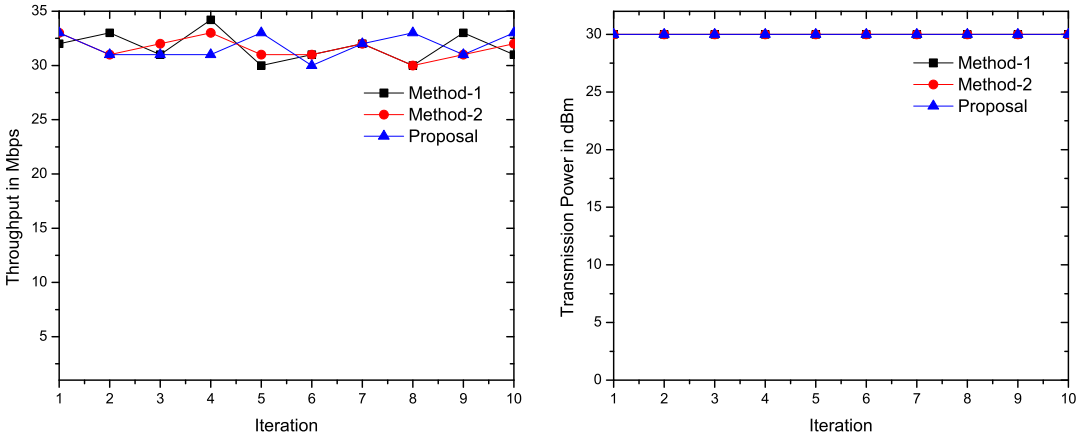
For $G = 10Mbps$, the required throughput is satisfied with the minimum transmission power (=0dBm). However, in *method-1*, the power is fixed at $30dBm$. In *method-2*, it becomes $0dBm$ at the fifth iteration. In the *proposal*, the power becomes $0dBm$ at the second iteration. Thus, the *proposal* has significantly reduced the transmission power while meeting the throughput requirement.

For $G = 40Mbps$, the throughput is not satisfied even with the maximum transmission power. Thus, any method becomes the maximum power.



(a) Throughput results. (b) Transmission power results.

Figure 7.3: Results for *Topology 1* with $G = 10Mbps$.



(a) Throughput results. (b) Transmission power results.

Figure 7.4: Results for *Topology 1* with $G = 40Mbps$.

Table 7.1: Result for *Topology 1*.

G=10 Mbps		
method	avg. thr. (Mbps)	avg. power (dBm)
method-1	31.91	30.00
method-2	31.19	6.60
proposal	31.55	0.60
G=40 Mbps		
method-1	31.72	30.00
method-2	31.60	30.00
proposal	31.80	30.00

7.4.4.2 Topology 2: One Host in Corridor

In *Topology 2*, one AP is allocated in one room of size $7m \times 6m$ and one host is allocated in the corridor, as shown in Figure 7.5.

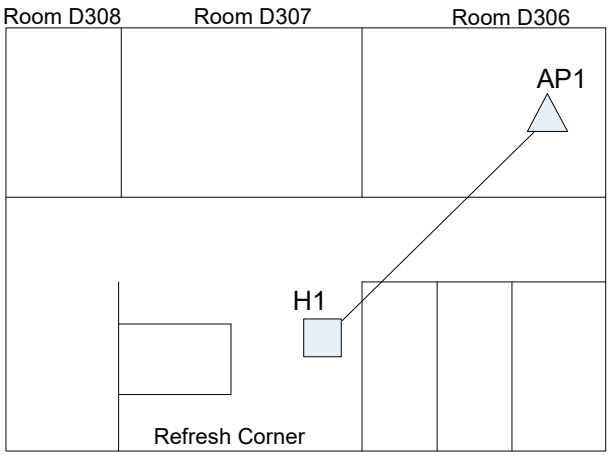
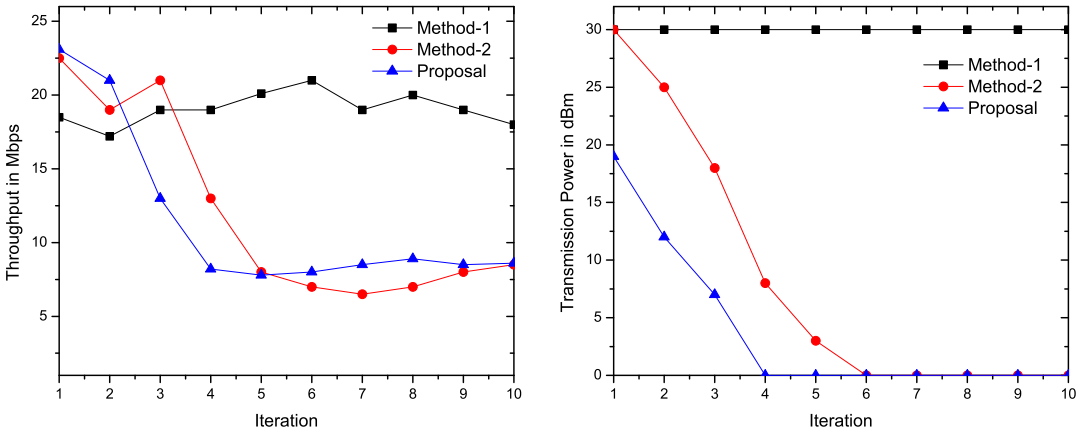


Figure 7.5: *Topology 2*: one host in corridor.

Figures 7.6 - 7.8 illustrate the changes of the throughput and transmission power results for $G = 5Mbps$, $G = 15Mbps$, and $G = 25Mbps$ respectively. Table 7.2 shows the average throughput and transmission power results. The RSS at the AP from the host is $-66.10dBm$ on average. Then,

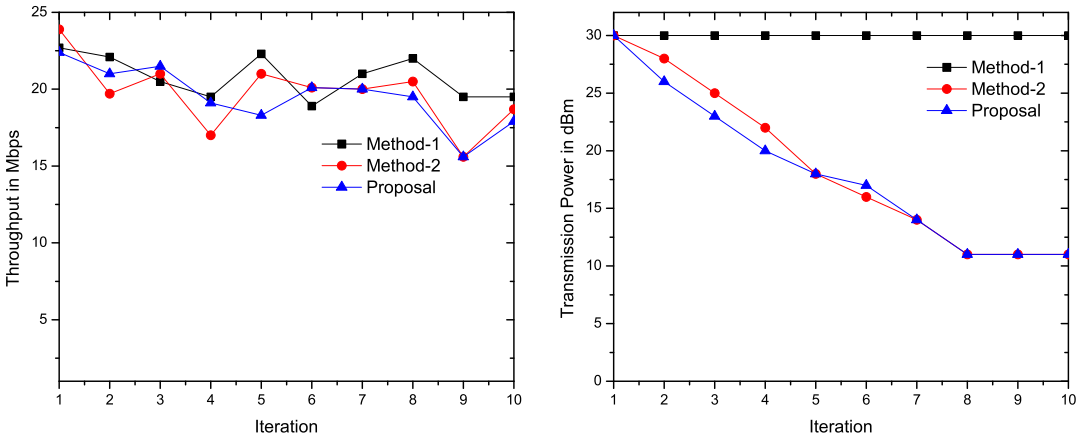


(a) Throughput results.

(b) Transmission power results.

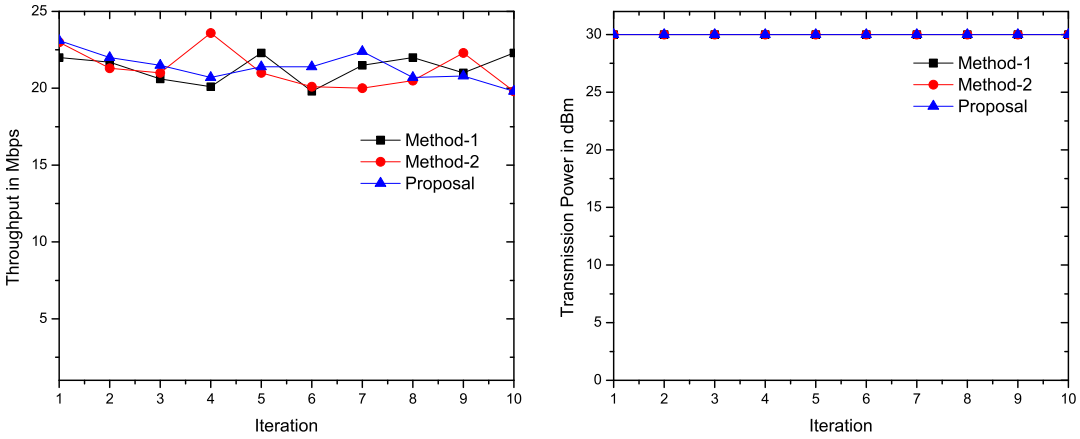
Figure 7.6: Results for *Topology 2* with $G = 5Mbps$.

the *proposal* selects $19dBm$, $30dBm$, and $30dBm$ as the initial power for them. The PI feedback control is applied to minimize the power to $0dBm$ and $11dBm$ for $G = 5Mbps$ and $G = 15Mbps$ respectively. For $G = 30Mbps$, the highest power (=$30dBm$) is selected as the best throughput performance. Thus, the *proposal* has remarkably reduced the power while maintaining the target throughput performance.



(a) Throughput results. (b) Transmission power results.

Figure 7.7: Results for *Topology 2* with $G = 15Mbps$.



(a) Throughput results. (b) Transmission power results.

Figure 7.8: Results for *Topology 2* with $G = 25Mbps$.

7.4.4.3 Topology 3: Two Hosts in Different Room

In *Topology 3*, two hosts in the same room communicate with the same AP in a different room, revealed in Figure 7.9. Also figure 7.10 - 7.12 demonstrate the throughput and power change results. Lastly, table 7.3 provides the summary.

For $G = 5Mbps$ and $10Mbps$, the *proposal* minimizes the power at the fourth or sixth iteration respectively, where *method-2* does it at the eighth or ninth iteration. The results prove that the *proposal* has significantly reduced the transmission power while satisfying the throughput requirement.

7.4.5 Experiments in Graduate School Building

As the second network field, the second floor in Graduate School Building is adopted in experiments for *Topology 4* and *Topology 5*.

Table 7.2: Result for *Topology 2*.

G=5 Mbps		
method	avg. thr. (Mbps)	avg. power (dBm)
method-1	19.08	30.00
method-2	12.05	8.40
proposal	11.56	3.80
G=15 Mbps		
method-1	20.80	30.00
method-2	19.75	18.60
proposal	19.54	18.10
G=25 Mbps		
method-1	21.33	30.00
method-2	21.26	30.00
proposal	21.38	30.00

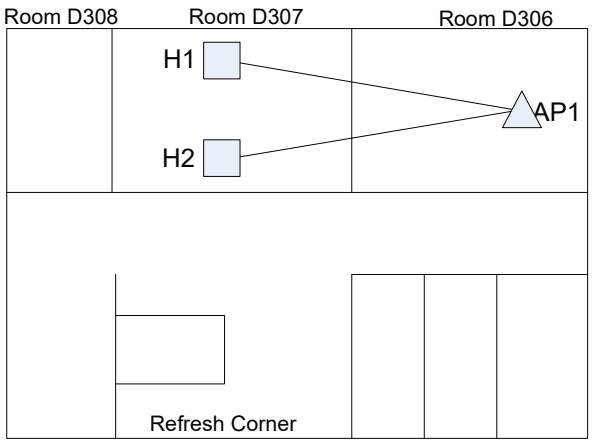
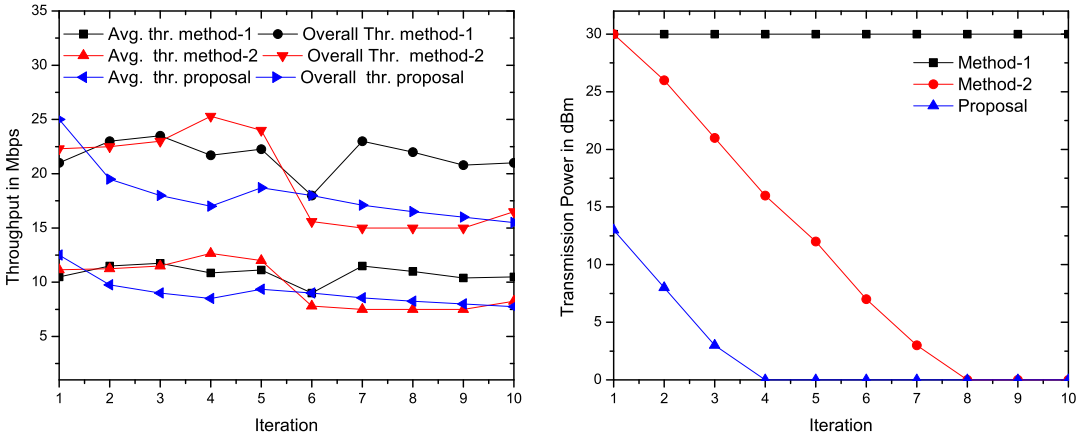


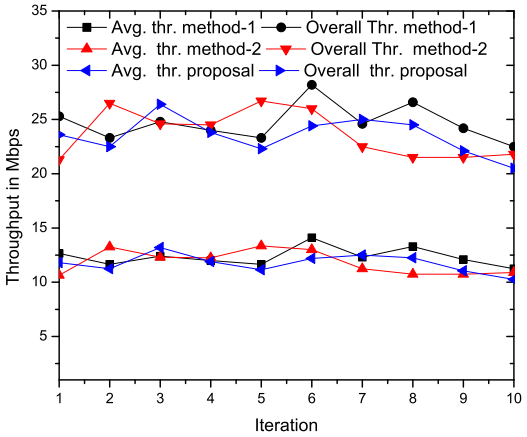
Figure 7.9: *Topology 3*: two hosts in different room.



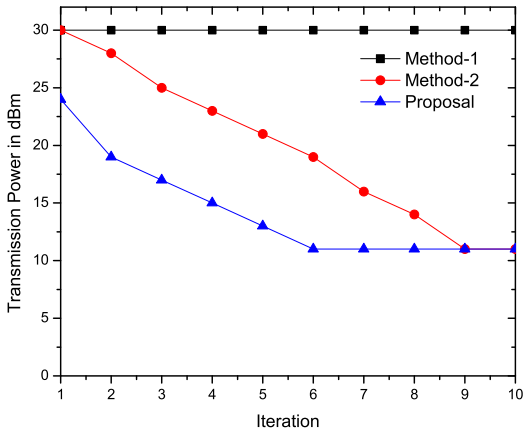
(a) Throughput results.

(b) Transmission power results.

Figure 7.10: Results for *Topology 3* with $G = 5Mbps$.

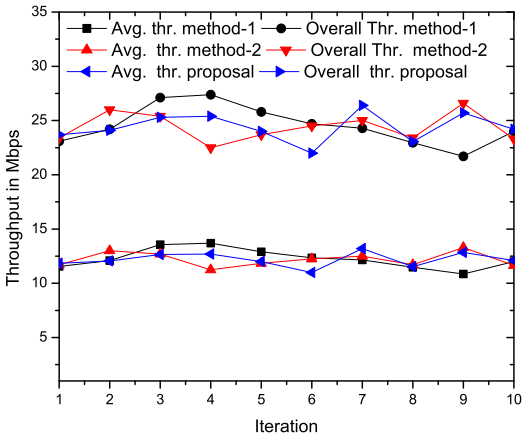


(a) Throughput results.

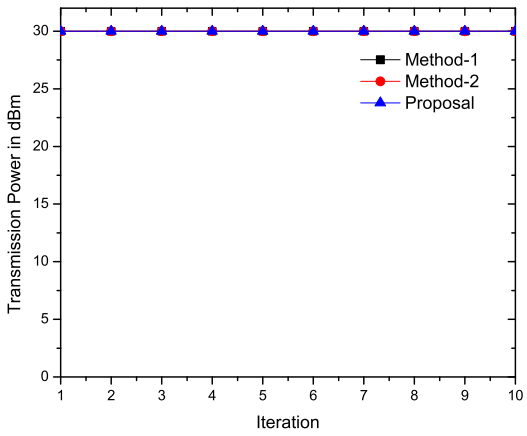


(b) Transmission power results.

Figure 7.11: Results for *Topology 3* with $G = 10Mbps$.



(a) Throughput results.



(b) Transmission power results.

Figure 7.12: Results for *Topology 3* with $G = 15Mbps$.

Table 7.3: Results for *Topology 3*.

G=5 Mbps			
method	ave. thr (Mbps)	avg. total thr. (Mbps)	avg. power (dBm)
method-1	10.81	21.63	30.00
method-2	9.71	19.42	11.50
proposal	9.07	18.14	2.40
G=10 Mbps			
method-1	12.34	24.68	30.00
method-2	11.85	23.69	19.80
proposal	11.76	23.51	14.30
G=15 Mbps			
method-1	12.26	24.53	30.00
method-2	12.19	24.38	30.00
proposal	12.19	24.39	30.00

7.4.5.1 Topology 4: Two Hosts in Same Room

In *Topology 4*, one AP and two hosts are allocated in the same room of size $9m \times 5.5m$, shown in Figure 7.13. The two hosts concurrently communicate with the AP. Figures 7.14 and 7.15 show

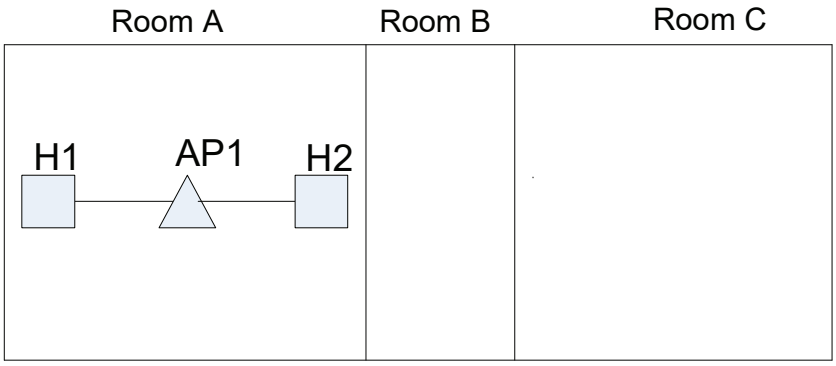
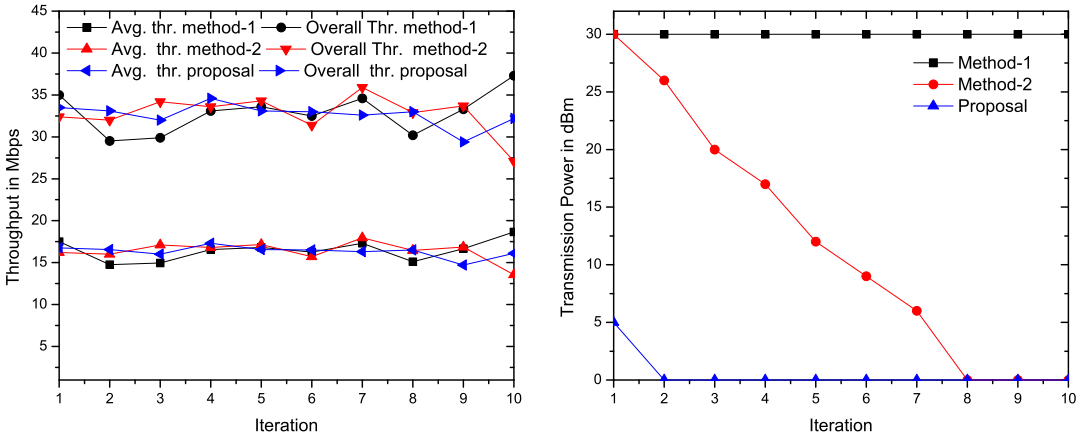


Figure 7.13: *Topology 4*: two hosts in same room.

the changes of the throughput and transmission power results for $G = 10Mbps$ and $G = 40Mbps$ respectively while table 7.4 contains the summary results. *Method-2* minimizes the power at the eighth iteration for $10Mbps$ whereas *proposal* does it at the second iteration. Once more, the approach successfully reduces the transmission power while satisfying the throughput requirement.



(a) Throughput result.

(b) Transmission power results.

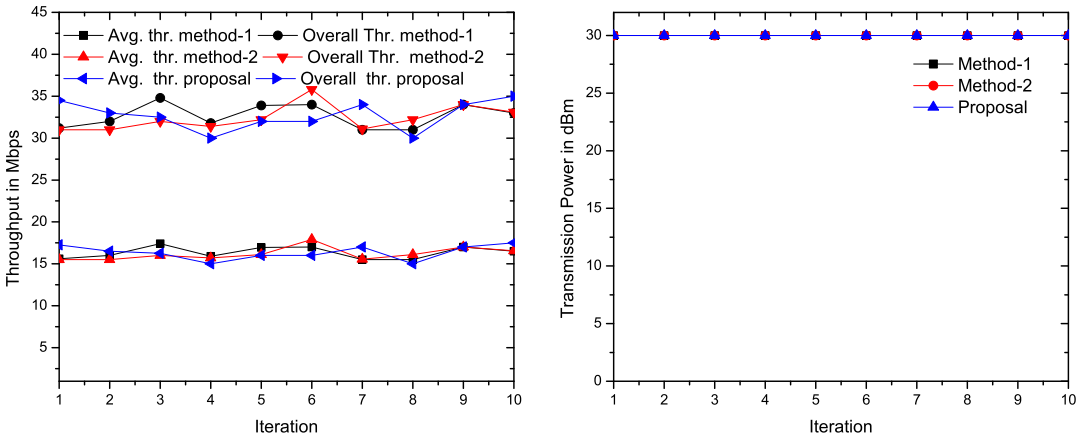
Figure 7.14: Results for *Topology 4* with G=10Mbps.

7.4.5.2 Topology 5: Two Hosts in Different Room

Finally, two hosts are located in a different room from the AP, as shown in Figure 7.16.

Figures 7.17 - 7.19 present the changes of the throughput and transmission power results for $G = 3Mbps$, $G = 10Mbps$, and $G = 15Mbps$ respectively.

Table 7.5 signifies the average throughout. Again, for $G = 15Mbps$, the power is set $30dBm$ all the time. For $G = 3Mbps$ and $G = 10Mbps$, the *proposal* minimizes the power at $0dBm$



(a) Throughput result.

(b) Transmission power results.

Figure 7.15: Results for *Topology 4* with G=40Mbps.

Table 7.4: Results for *Topology 4*.

G=10 Mbps			
method	ave. thr (Mbps)	avg. total thr. (Mbps)	avg. power (dBm)
method-1	16.45	32.90	30.00
method-2	16.38	32.76	12.00
proposal	16.33	32.66	0.50
G=40 Mbps			
method-1	16.34	32.67	30.00
method-2	16.19	32.38	30.00
proposal	16.35	32.70	30.00

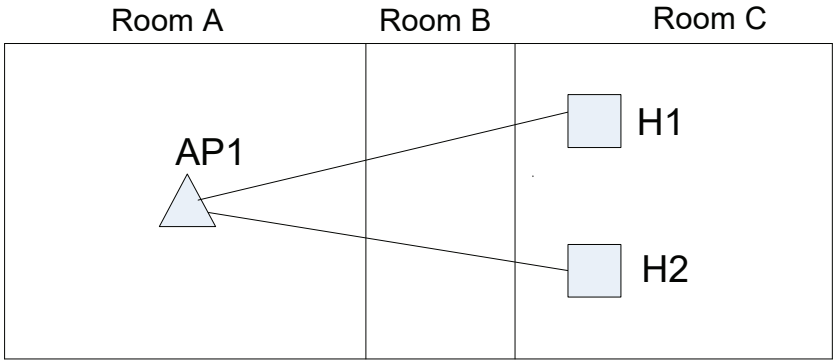
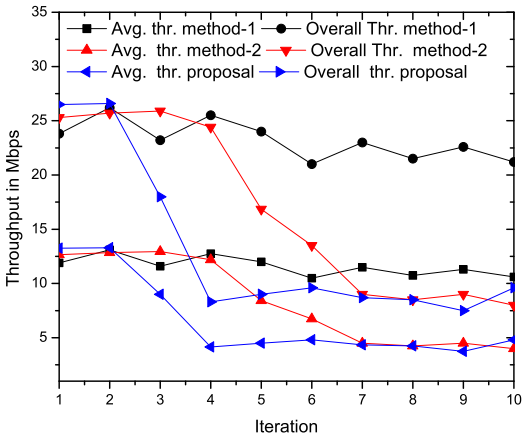
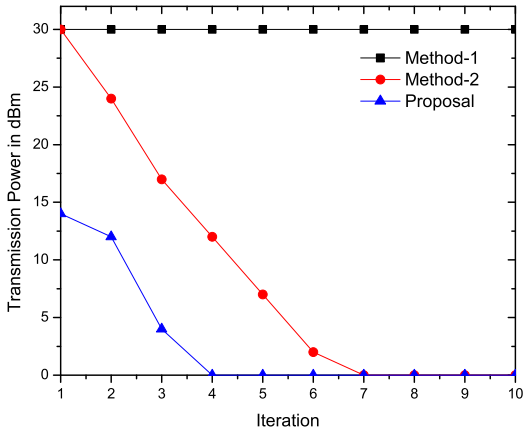


Figure 7.16: *Topology 5*: two hosts in different room.

and 12dBm respectively, with shorter time than the *method-2*. The effectiveness of the *proposal* is confirmed then.

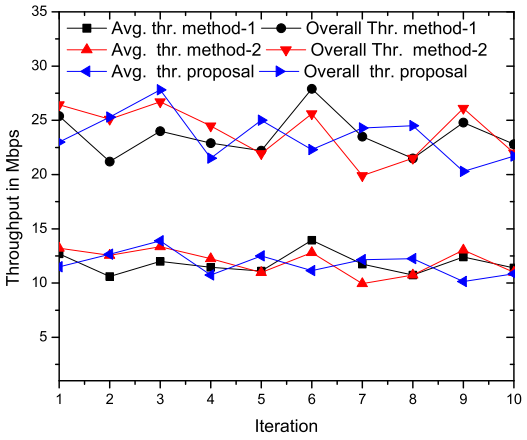


(a) Throughput results.

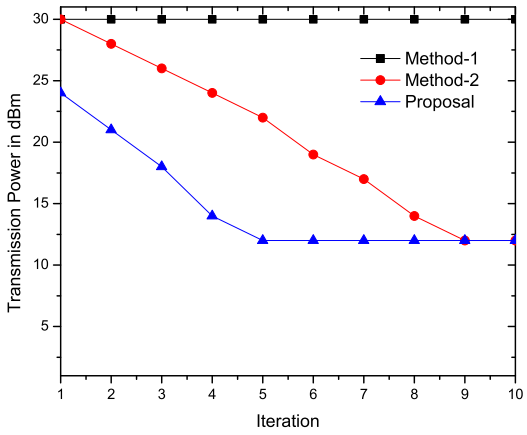


(b) Transmission power results.

Figure 7.17: Results for *Topology 5* with G=3Mbps.

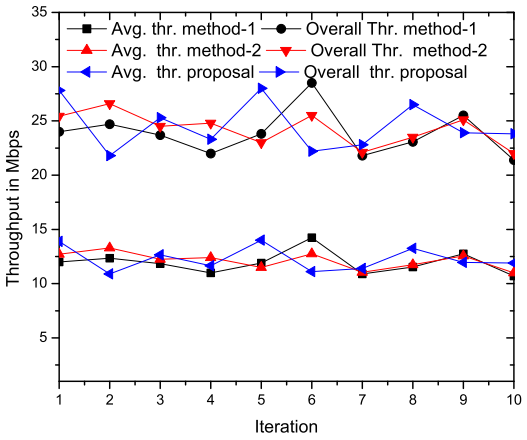


(a) Throughput results.

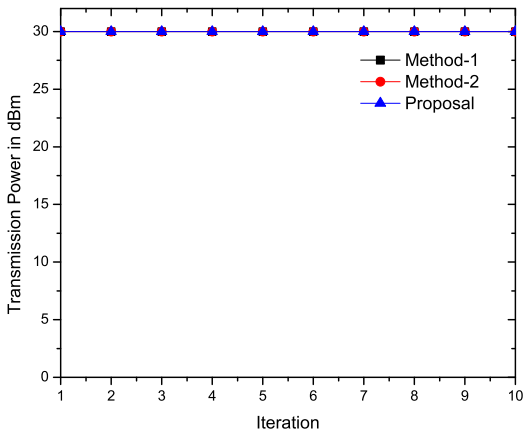


(b) Transmission power results.

Figure 7.18: Results for *Topology 5* with G=10Mbps.



(a) Throughput results.



(b) Transmission power results.

Figure 7.19: Results for *Topology 5* with G=15Mbps.

Table 7.5: Results for *Topology 5*.

G=3 Mbps			
method	ave. thr (Mbps)	avg. total thr. (Mbps)	avg. power (dBm)
method-1	11.60	23.20	30.00
method-2	8.31	16.62	9.20
proposal	6.62	13.24	3.00
G=10 Mbps			
method-1	11.81	23.62	30.00
method-2	11.99	23.98	20.40
proposal	11.79	23.57	14.90
G=15 Mbps			
method-1	11.92	23.85	30.00
method-2	12.13	24.25	30.00
proposal	12.27	24.54	30.00

7.4.6 Overall Discussions

In our current implementation, the initial transmission power is set by collecting the measured RSS at the AP from the stations. Then, the power is dynamically minimized by measuring the throughput with *iperf* using the feedback control during the communication. The transmission power can be changed within the dynamic range of the AP device.

Depending on the network topology and the target *throughput constraint G*, the algorithm can find the appropriate transmission power. It is found that, when the hosts are located in the same room or in very near rooms as/from the AP, such as *Topology 1* and *Topology 4*, the fluctuation of RSS is small even for the minimum transmission power. Thus, in this case, the fluctuation of the transmission power may not affect the current throughput performance of the station.

However, when the hosts are located far from the AP, such as *Topology 2*, *Topology 3*, and *Topology 5*, the RSS will be changed significantly. Thus, in this case, a significant fluctuation in the transmission power may affect the performance of our proposal as shown in Figure 7.6, 7.10 and 7.17. Also, it is observed that the initial power can provide the similar result to the highest power. The selected power still can satisfy the performance because even if the transmission power is decreasing, the reduced power is enough to provide the required RSS. In future, we will investigate the effect of the power fluctuation to the performance and the improvement of the proposal.

Besides the initial transmission power selection process is an empirical method depends on the accurate parameter values. These values are derived using the measurement results setting the different transmission powers at the AP for different host locations. Also, the RSS between the AP and host is always fluctuating even if the distance between them is constant. Furthermore, other factors can affect the throughput performance like the interference from other Wi-Fi devices, the weather change, and the congestion. Thus, this model sometimes over estimates the initial power than the selected transmission power. In future work, we will further investigate to minimize this over estimation of the initial transmission power selection process.

7.5 Summary

In this chapter, we proposed the *dynamic AP transmission power minimization approach* using the *PI feedback control*. First, the initial power is examined from the difference between the measured RSS at the AP from the host and the estimated RSS necessary for the target throughput. Then, the power is minimized by using the *PI feedback control*, such that the measured throughput achieves the target one. The proposal was implemented in the *elastic WLAN system testbed* using *Raspberry Pi* APs. The testbed experiments in different network topologies confirmed the effectiveness of the proposal. In the next chapter, we will present two measurement location minimization methods for reducing the measurement time and cost for the *throughput estimation model*.

Chapter 8

Measurement Location Minimization for Throughput Estimation Model

In this chapter, we present two measurement location minimization methods to reduce the load in obtaining the accurate *throughput estimation model* with various transmission powers, which is essential in the *static AP transmission power minimization approach* in Chapter 6. First, we point out the drawbacks in our previous studies. Then, we present the two methods for the measurement location minimization. Finally, we evaluate the proposed methods through extensive experiments, and confirm that the estimation error is not degraded for any case, while the number of measurement locations is reduced by up to 86%.

8.1 Drawbacks in Previous Approach

To improve the performance and reduce the energy consumption of a *wireless local-area network (WLAN)*, this thesis presented the *transmission power minimization* of *access-points (APs)* using the *throughput estimation model* in Chapter 6. The *throughput estimation model* [11] is adopted to estimate the minimum power satisfying the required throughput performance. The parameters in the model must be tuned accurately by throughput measurement results for different transmission powers of the AP in the target field.

Unfortunately, the measurement process needs a lot of manual works, which takes long time. The host must be moved at every location, and the transmission power must be changed by several values. The corresponding measurements can be done manually. Also, due to fluctuations of throughputs, the measurements must be done several times, where one throughput measurement requires several minutes using *iperf*. As a result, even for a single link, it can take a whole day. Thus, the number of measurement host locations for each AP should be minimized to reduce them.

In this chapter, we present two measurement location minimization methods. The first method can reduce the number of host locations to 60% of the original by considering the conditions presented in [99]. The second method can reduce it to 86% by further limiting the host locations. We evaluate the effectiveness through experiments using the IEEE802.11n WLAN. We confirm that for any transmission power, the estimation accuracy of the model is similar before and after applying each minimization method, which is validated by *T-test* [100].

8.2 Measurement Location Selection Method-1

In this section, we present the first measurement location minimization method, *method-1*, for selecting the host locations in the network field by the following conditions in [99]:

(1) In each room in the field, the host location that is nearest from the AP must be selected.

(2) For each wall type, the host location whose shortest path from the AP is crossing it should be selected to determine the attenuation factor.

(3) If the host location has three or more walls along the shortest path from the AP, it must be selected.

8.2.1 Host Location Selection Result

After applying the conditions in the network field in Figure 8.1, original 14 host locations in four rooms are reduced to the five marked locations.

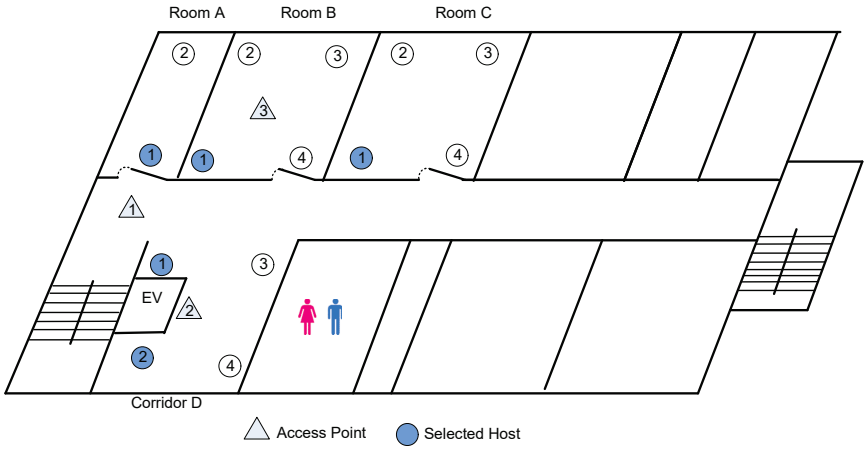


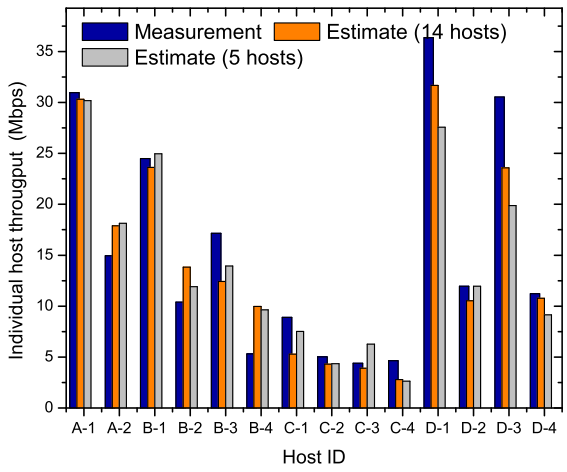
Figure 8.1: Host selection results.

8.2.2 Parameter Optimization Results

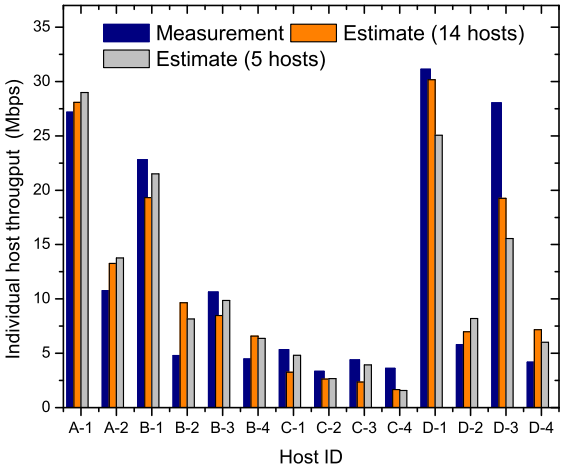
For evaluations, we conducted experiments with $5dBm$, $10dBm$, $20dBm$, and $30dBm$ for the transmission power of the AP on the 3rd floor of Engineering Building-2 in Okayama University in Figure 8.1. Then, the model parameter values is tuned by applying the *parameter optimization tool*. P_1 is changed by the transmission power and the other parameters are fixed at the ones for the full transmission power, shown in Table 8.1.

8.2.3 Throughput Estimation Results

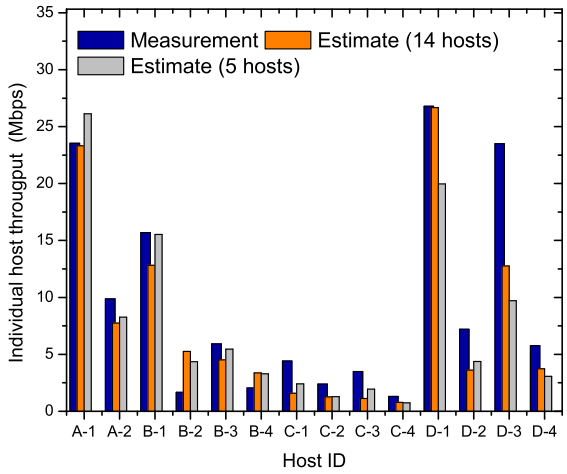
Subsequently, the throughput is estimated by using the throughput estimation model with the parameter values in Table 8.1. Figures 8.2 shows the measured throughput, the estimated one by considering all the host locations data, and the estimated one by the proposal at every host location for AP1 for different transmission power. These results are similar to each other that confirms the correctness of the proposal. Then, we compare the estimation errors of the models by all the



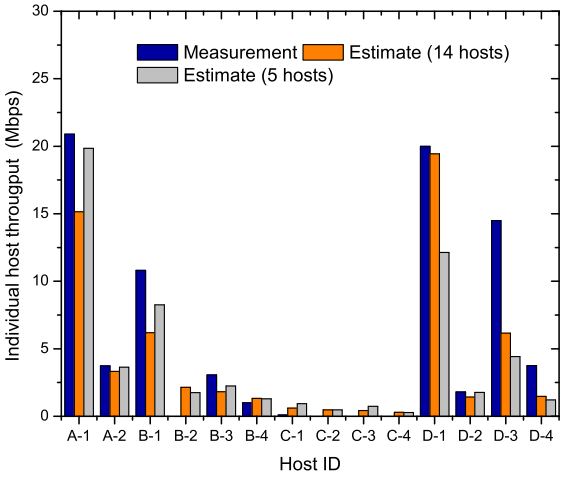
(a) Throughput for tx-power, 30 dBm.



(b) Throughput for tx-power, 20 dBm.



(c) Throughput for tx-power, 10 dBm.



(d) Throughput for tx-power, 5 dBm.

Figure 8.2: Measured and estimated throughput for AP1 for different transmission power.

Table 8.1: Parameters in throughput estimation model.

parameter	method	
	original	method-1
α	3.00	2.73
W_1	0	0
W_2	7	8
W_3	6	6
W_4	7	7
W_5	2.30	2.50
W_6	3.40	3.80
W_7	5.00	4.00
a	34	32
b	57	57
c	8	8
P_1 (dBm) for each transmission power		
trans. power (dBm)	method	
	original	method-1
30	-34.00	-34.00
20	-38.20	-38.40
10	-44.50	-44.60
5	-52.60	-52.60

locations and by the selected locations of Figure 8.1. Tables 8.2 - 8.4 summarize the average, maximum, minimum, and standard deviation (SD) of the throughput estimation errors (Mbps) on the links between the 14 host locations and each AP location respectively. The results show that for any transmission power, the estimation accuracy of the model is similar. Thus, the effectiveness of our proposal in reducing the measurement cost while keeping the model accuracy is confirmed.

Table 8.2: Through. estimation errors (Mbps) for AP1.

tx-power	method	ave.	max.	min.	SD
30	original	2.68	6.96	0.46	2.05
	method-1	2.93	10.67	0.01	3.13
20	original	2.62	8.80	0.72	2.10
	method-1	2.76	12.50	0.46	3.16
10	original	2.49	10.73	0.12	2.63
	method-1	2.86	13.79	0.16	3.54
5	original	1.98	8.34	0.29	2.51
	method-1	2.10	10.08	0.03	3.05

Table 8.3: Through. estimation errors (Mbps) for AP2.

tx-power	method	ave.	max.	min.	SD
30	original	1.81	4.65	0.10	1.30
	method-1	2.10	5.55	0.39	1.47
20	original	1.92	5.00	0.15	1.36
	method-1	1.95	4.91	0.07	1.89
10	original	1.65	4.58	0.14	1.21
	method-1	1.58	4.57	0.27	1.21
5	original	1.20	3.06	0.03	0.84
	method-1	1.08	2.50	0.03	0.78

Table 8.4: Through. estimation errors (Mbps) for AP3.

tx-power	method	ave.	max.	min.	SD
30	original	1.88	5.13	0.03	1.29
	method-1	1.97	5.69	0.03	1.57
20	original	2.22	7.62	0.06	2.07
	method-1	2.25	7.53	0.31	2.01
10	original	3.36	8.17	0.32	2.76
	method-1	3.14	7.76	0.08	2.73
5	original	3.65	8.53	0.21	2.66
	method-1	3.39	7.93	0.02	2.54

8.2.4 Validation by T-test

To validate the results statistically, we apply *T-test* to determine if the two sets of data are significantly similar or different from one another [100]. For any AP with different transmission powers, we take the average estimation error results among them, and apply T-test between 14 host results and 5 hosts results. Table 8.5 summarizes the T-test result. According to the T-test, if the t-statistics (test statistics) hold the following condition, the two datasets are similar:

-Test critical one-tail < test statistics < Test critical one-tail

Table 8.5: T-test results on average throughput estimation results for each APs.

AP id	parameters	Results of T-test
AP1	Test statistics	-0.885596947
	Test critical one-tail	1.943180281
AP2	Test statistics	-0.117365045
	Test critical one-tail	1.943180281
AP3	Test statistics	0.163831679
	Test critical one-tail	1.943180281

From Table 8.5, we find that the test-statistics value holds the above condition. Thus, the results confirm that the both datasets provide similar performances.

8.2.5 Measurement minimization Results

Table 8.6 shows the measurement minimization results of the proposal. The results indicate that the proposed method can reduce the number of measurement host locations by 60% from the original one. Thus, the effectiveness of the proposal is confirmed.

Table 8.6: Measurement minimization results.

method	no. of host	no. of APs	no. of power level	total no. of measurements
original	14	3	4	168
method-1	5	3	4	60

8.3 Measurement Location Selection Method-2

In this section, we present the second measurement location minimization method, *method-2*, as the improvement of *method-1*. It limits the host locations into two nearest ones from the AP by the following procedure:

(1) Two host locations that are closest from each AP among the candidates from *method-1* are selected.

(2) The measurement is conducted at each selected host location in (1) for each transmission power.

(3) All the parameter values of the throughput estimation model are optimized for the $30dBm$ transmission power, using the measurement results with this power at all the selected host locations.

(4) The parameter P_1 value of the model is optimized for each different power from $30dBm$, using the measurement results with the corresponding power at all the selected host locations.

8.3.1 Host Location Selection Result

In Figure 8.3, two hosts locations marked as *A-1* and *B-1* are selected for *AP1* by the proposal. Similarly, *D-1* and *D-2* are for *AP2*, and *B-1* and *B-2* are for *AP3*.

8.3.2 Parameter Optimization Results

The parameter value is found as shown in Table 8.7 by the proposal.

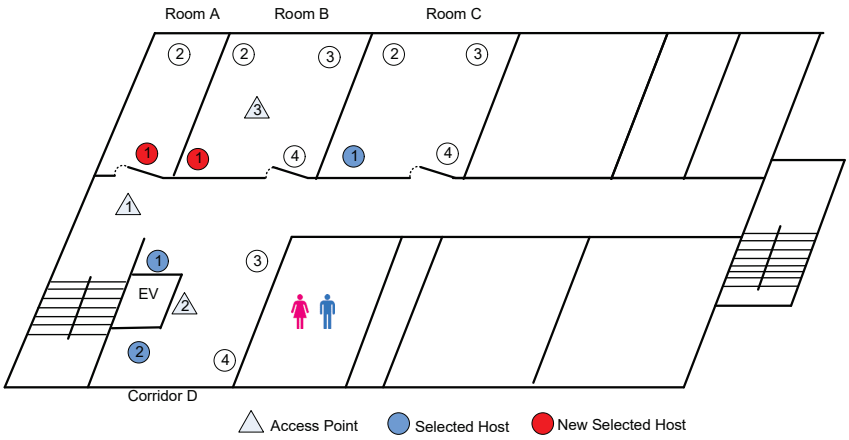


Figure 8.3: Host selection results.

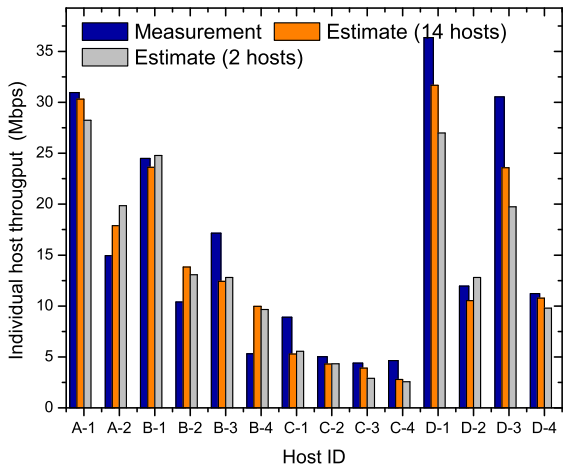
Table 8.7: Parameters in throughput estimation model.

parameter	method	
	original	method-2
α	3.00	3.00
W_1	0	0
W_2	7	8
W_3	6	6
W_4	7	7
W_5	2.30	2.70
W_6	3.40	3.40
W_7	5.00	3.00
a	34	31
b	57	55
c	8	8
P_1 (dBm) for each transmission power		
trans. power (dBm)	method	
	original	method-2
30	-34.00	-34.00
20	-38.20	-38.40
10	-44.50	-44.60
5	-52.60	-52.60

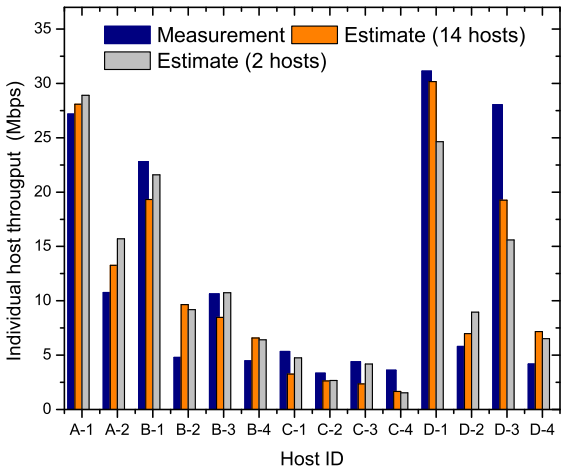
8.3.3 Throughput Estimation Results

Then, the throughput is estimated by using the throughput estimation model with the parameter values in Table 8.7. Figures 8.4 show the measured throughput, the estimated one by all the host locations, and the estimated one by the proposal at every host location for AP1 for different transmission power. These results are similar to each other that confirms the correctness of the proposal.

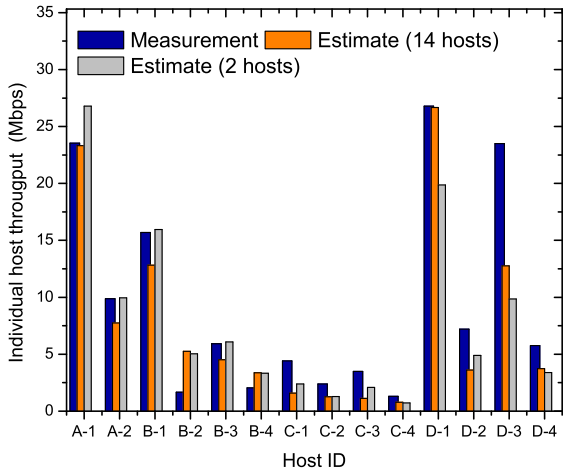
Tables 8.8 - 8.10 summarizes the average, maximum, minimum, and standard deviation (SD) of the throughput estimation errors (Mbps) on the links between the 14 host locations and each AP location, respectively. It has been observed that for any transmission power, the estimation accuracy of the model is similar before and after the minimization. Thus, the effectiveness of our proposal is confirmed.



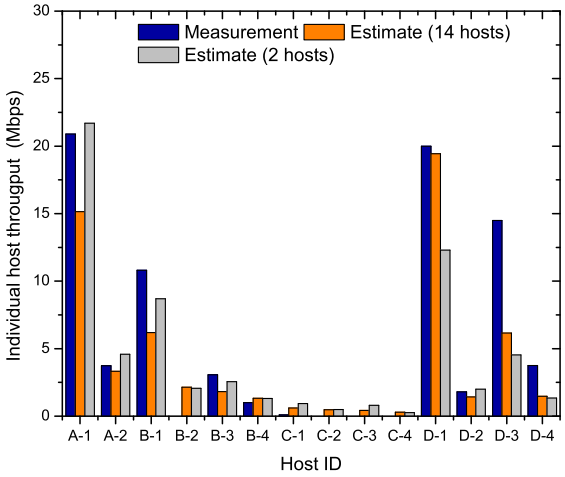
(a) Throughput for tx-power, 30 dBm.



(b) Throughput for tx-power, 20 dBm.



(c) Throughput for tx-power, 10 dBm.



(d) Throughput for tx-power, 5 dBm.

Figure 8.4: Measured and estimated throughput for AP1 for different transmission power.

Table 8.8: Through. estimation errors (Mbps) for AP1.

tx-power	method	ave.	max.	min.	SD
30	original	2.68	6.96	0.46	2.06
	method-2	3.53	10.79	0.28	3.14
20	original	2.62	8.80	0.72	2.10
	method-2	3.02	12.45	0.11	3.30
10	original	2.49	10.73	0.12	2.63
	method-2	2.77	13.64	0.07	3.60
5	original	1.98	8.34	0.29	2.51
	method-2	2.09	9.96	0.20	2.98

Table 8.9: Through. estimation errors (Mbps) for AP2.

tx-power	method	ave.	max.	min.	SD
30	original	1.81	4.65	0.10	1.30
	method-2	2.37	6.50	0.95	1.62
20	original	1.92	5.00	0.15	1.36
	method-2	2.03	4.94	0.06	1.66
10	original	1.65	4.58	0.14	1.21
	method-2	1.60	4.58	0.15	1.24
5	original	1.20	3.06	0.03	0.84
	method-2	1.03	2.30	0.01	0.72

Table 8.10: Through. estimation errors (Mbps) for AP3.

tx-power	method	ave.	max.	min.	SD
30	original	1.88	5.13	0.03	1.29
	method-2	2.12	5.21	0.49	1.39
20	original	2.22	7.62	0.06	2.07
	method-2	2.35	7.66	0.30	2.03
10	original	3.36	8.17	0.32	2.76
	method-2	3.17	7.61	0.16	2.72
5	original	3.65	8.53	0.21	2.66
	method-2	3.33	8.04	0.01	2.54

8.3.4 Validation by T-test

To validate the results statistically, we finally apply *T-test* on the average estimation error results for both 14 host results and two host results. Table 8.11 implies that, for any AP in the network field, test-statistics value holds the condition of similarity. Thus, the both datasets provide similar performance.

Table 8.11: T-test results on average throughput estimation results for each APs.

AP id	parameters	Results of T-test
AP1	Test statistics	-1.209308875
	Test critical one-tail	1.943180281
AP2	Test statistics	-0.341225841
	Test critical one-tail	1.943180281
AP3	Test statistics	0.066880846
	Test critical one-tail	1.943180281

8.3.5 Measurement minimization Results

Table 8.12 shows the measurement minimization results of the proposal for different transmission powers. If four different transmission powers are assigned, the total number of measurements for

Table 8.12: Measurement minimization results.

method	no. of host	no. of APs	no. of power level	total no. of measurements
original	14	3	4	168
method-1	5	3	4	60
method-2	2	3	4	24

the three AP locations is reduced from 168($= 14 \times 4 \times 3$) in the original and 60($= 5 \times 4 \times 3$) in the previous to 24($= 2 \times 4 \times 3$). It means the 86% reduction from the original and the 60% reduction from the previous by the proposal.

8.4 Summary

In this chapter, we presented two measurement host location minimization methods to minimize the labor cost in the parameter optimization of the throughput estimation model. Then, we confirmed the effectiveness of the methods by evaluating the estimation errors for different AP transmission powers. In the next chapter, we will conclude this thesis with some future works.

Chapter 9

Conclusion

In this thesis, we presented our studies of enhancements of the *active AP configuration algorithm* and the *elastic WLAN system*. First, we proposed the extension of the *active AP configuration algorithm* to deal with the dynamic nature of the WLAN network. The number of active APs is minimized such that the minimum host throughput constraint is satisfied, while any communicating AP is not suspended and any communicating host does not change the associated AP. Through simulations and testbed experiments, the effectiveness was confirmed.

Then, we implemented two security functions, namely, the *system software update function* and the *user authentication function*, in the *elastic WLAN system testbed*. The former function periodically downloads the latest system software to the local repository servers, and installs it into them. The latter function authenticates a newly joining host user using the *RADIUS* server. The functions were evaluated with the testbed.

We also proposed two *transmission power minimization* approaches to reduce the transmission power of the AP. The static approach finds the minimum power using the throughput estimation model. The dynamic approach optimizes the power dynamically using the *PI feedback control*. Their effectiveness were validated through simulations and testbed experiments.

Finally, we presented two *measurement host location minimization methods* to minimize the labor cost in the parameter optimization of the throughput estimation model. The effectiveness were examined by comparing the number of measurements and the throughput estimation errors.

In future studies, we will further improve the transmission power minimization approach and the throughput estimation model, particularly for concurrent communications of multiple hosts with the same AP. We will also conduct performance evaluations of our proposals in various network scenarios.

Bibliography

[1] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 wireless local area networks," IEEE Commun. Mag., vol. 35, no. 9, pp. 116-126, Sept. 1997.

[2] M. Balazinska and P. Castro, "Characterizing mobility and network usage in a corporate wireless local-area network," Proc. Int. Conf. Mobile Systems, Appl. Services, pp. 303-316, 2003.

[3] S. Lanzisera, B. Nordman, and R. E. Brown, "Data network equipment energy use and savings potential in buildings," Energy Efficiency, vol. 5, no. 2, pp. 149-162, May 2012.

[4] K. Mittal, E. M. Belding, and S. Suri, "A game-theoretic analysis of wireless access point selection by mobile users," Computer Commun., vol. 31, no. 10, pp. 2049-2062, Jan. 2008.

[5] D. Kotz and K. Essien, "Analysis of a campus-wide wireless network," Wireless Networks, vol. 11, no. 1-2, pp. 115-133, Jan. 2005.

[6] F. Nadeem, E. Leitgeb, M. S. Awan, and S. Chessa, "Comparing the life time of terrestrial wireless sensor networks by employing hybrid FSO/RF and only RF access networks," Proc. Int. Conf. Wireless. Mobile Commun., pp. 134-139, 2009.

[7] "Electricity sector in Bangladesh," Internet: https://en.wikipedia.org/wiki/Electricity_sector_in_Bangladesh, Access Jan. 20, 2019.

[8] "Electricity to transform rural Myanmar," Internet: http://www.worldbank.org/en/news/feature/2015/09/16/electricity-to-transform-rural-myanmar, Access Jan. 20, 2019.

[9] M. S. A. Mamun, M. E. Islam, N. Funabiki, M. Kuribayashi, and I.-W. Lai, "An active access-point configuration algorithm for elastic wireless local-area network system using heterogeneous devices," Int. J. Netw. Comput., vol. 6, no. 2, pp. 395-419, July 2016.

[10] M. S. A. Mamun, N. Funabiki, K. S. Lwin, M. E. Islam, and W.-C. Kao, "A channel assignment extension of active access-point configuration algorithm for elastic WLAN system and its implementation using Raspberry Pi," Int. J. Netw. Comput., vol. 7, no. 2, pp. 248-270, July 2017.

[11] K. S. Lwin, N. Funabiki, C. Taniguchi, K. K. Zaw, M. S. A. Mamun, M. Kuribayashi, and W.-C. Kao, "A minimax approach for access point setup optimization in IEEE 802.11n wireless networks," Int. J. Netw. Comput., vol. 7, no. 2, pp. 187-207, July 2017.

[12] N. Funabiki, C. Taniguchi, K. S. Lwin, K. K. Zaw, and W.-C. Kao, "A parameter optimization tool and its application to throughput estimation model for wireless LAN," Proc. Int. Work. Virtual Environ. Netw.-Orient. Appli. (VENOA), pp. 701-710, July 2017.

[13] M. M. Islam, M. S. A. Mamun, N. Funabiki, and M. Kuribayashi, "A dynamic host behavior extension of active access-point configuration algorithm for elastic WLAN system," Chugoku-branch Joint Conv. Inst. Elec. Info. Eng., Oct. 2017.

[14] M. M. Islam, M. S. A. Mamun, N. Funabiki, and M. Kuribayashi, "Dynamic access-point configuration approach for elastic wireless local-area network system," Proc. Int. Symp. Comput. Netw., pp. 216-222, Nov. 2017.

[15] M. M. Islam, N. Funabiki, M. Kuribayashi, S. K. Debnath, I. M. Kwenga, K. S. Lwin, R. W. Sudibyo and M. S. A. Mamun, "Dynamic access-point configuration approach for elastic wireless local-area network system and its implementation using Raspberry Pi," Int. J. Netw. Comput., vol. 8, no. 2, pp. 254-281, July 2018.

[16] N. Funabiki ed., "Wireless mesh networks," InTech-Open Access Pub., Jan. 2011, Internet: http://www.intechopen.com/books/wireless-mesh-networks.

[17] "Raspberry Pi Teach, Learn, and Make with Raspberry Pi," Internet: https://www.raspberrypi.org/, Access Jan. 20, 2019.

[18] M. M. Islam, N. Funabiki, M. Kuribayashi, R. W. Sudibyo, and I. M. Kwenga, "Implementations of system software update and user authentication functions in elastic wireless local-area network system," IEICE Tech. Rep., SRW2018-14, pp. 31-36, Aug. 2018.

[19] M. M. Islam, N. Funabiki, M. Kuribayashi,and W-C. Kao, "A proposal of transmission power minimization extension in active access-point configuration algorithm for elastic wireless local-area network system," IEICE Tech. Report, NS2018-173, pp. 83-88, Dec. 2018.

[20] M. M. Islam, N. Funabiki, M. Kuribayashi, M. Saha, I. M. Kwenga, R. W. Sudibyo, and W-C. Kao, "A proposal of transmission power minimization extension in active access-point configuration algorithm for elastic wireless local-area network system," Int J. Comput. Soft. Eng., vol. 4, no. 140, pp. 1-9, Jan. 2019.

[21] M. M. Islam, N. Funabiki, R. W. Sudibyo, I. M. Kwenga and H. Briantoro, "An access-point transmission power minimization approach using PI feedback control in wireless local-area network," IEICE Society Conf., pp. S-29-30, Sept. 2019.

[22] M. M. Islam, N. Funabiki, R. W. Sudibyo, I. M. Kwenga, and W-C. Kao, " A dynamic access-point transmission power minimization method using PI feedback control in elastic WLAN system for IoT applications," Internet of Things, vol. 8, pp. 1-15, Aug. 2019.

[23] K. Astrom and T. Hagglund, "PID controller: theory, design, and tuning," 2nd Edition, Instrument Society of America, 1995.

[24] "iPerf - The TCP, UDP and SCTP network bandwidth measurement tool," Internet: https://iperf.fr/, Access Jan. 20, 2019.

[25] M. M. Islam, N. Funabiki, M. Saha, I. M. Kwenga, and R. W. Sudibyo, "Throughput measurement location minimization method for access-point transmission power minimization in wireless local-area network," IEICE General Conf., pp. S-76-77, March 2019.

[26] M. M. Islam, N. Funabiki, M. Saha, I. M. Kwenga, R. W. Sudibyo, and W-C. Kao, "An improvement of throughput measurement minimization method for access-point transmission power minimization in wireless local-area network," Proc. IEEE Int. Conf. Consum. Elect. Taiwan (ICCE-TW), May 2019.

[27] M. S. Gast, "802.11 wireless networks - the definitive guide," 2nd ed., O'Reilly, Sebastopol, Apr. 2005.

[28] S. Banerji, and R. S. Chowdhury, "On IEEE 802.11: wireless LAN technology," Int. J. Mobi. Net. Commun. & Telematics, vol. 3, no. 4, Aug. 2013.

[29] S. Banerji, "Upcoming standards in wireless local area networks," Wireless & Mobile Technol., vol. 1, no. 1, Sept. 2013.

[30] I. Poole, "IEEE 802.11 Wi-Fi standards," Internet: http://www.radio-electronics.com/info/wireless/wi-fi/ieee-802-11-standards-tutorial.php, Access Jan. 20, 2019.

[31] V. Beal, "What is 802.11 wireless LAN standards?," Internet: http://www.webopedia.com/TERM/8/802_11.html, Access Jan. 20, 2019.

[32] C. C. Choon, "A study of active access-point selection algorithm for wireless mesh network under practical conditions," Ph.D. thesis, Grad. Sch. Natural Sci. Technol., Okayama Univ., Japan, Sept. 2015.

[33] "IEEE 802.11 - Wikipedia," Internet: http://en.wikipedia.org/wiki/IEEE_802.11, Access Jan. 20, 2019.

[34] M. E. Islam, "A study of access-point aggregation algorithm for elastic wireless local-area network system and its implementation," Ph.D. thesis, Grad. School of Natural Science and Technology, Okayama Univ., Japan, March 2016.

[35] S. Sendra, M. Garcia, C. Turro, and J. Loret, "WLAN IEEE 802.11 a/b/g/n indoor coverage and interference performance study," Int. J. Adv. Netw. Service., vol. 4, no. 1, pp. 209-222, 2011.

[36] A. S. Tanenbaum, and D. J. Wetherall, "Computer networks," 5th ed., Pearson Prentice Hall, 2011.

[37] J. H. Yeh, J. C. Chen, and C. C. Lee, "WLAN standards," IEEE Potentials, vol. 22, no. 4, 2003.

[38] IEEE, "IEEE Std 802.11-2012, IEEE standard for Information technology," Internet: https://standards.ieee.org/findstds/standard/802.11-2012.html, Access Jan. 20, 2019.

[39] O. Bejarano, E. W. Knightly, and M. Park, "IEEE 802.11ac: from channelization to multi-user MIMO," IEEE Comm. Mag., vol. 51, no. 10, pp. 84-90, Oct. 2013.

[40] L. Verma, M. Fakharzadeh, and S. Choi, "Wifi on steroids: 802.11ac and 802.11ad," IEEE Trans. Wireless Comm., vol. 20, no. 6, pp. 30-35, Dec. 2013.

[41] M. S. Gast, " 802.11ac: a survival guide," O'Reilly Media, Inc., 2013.

[42] "WiMAX page, IEEE802.11ac works," Internet: http://123-info.net/wimax-page/archives/918, Access Jan. 20, 2019.

[43] L. Deek, E. Garcia-Villegas, E. Belding, S. J. Lee, and K. Almeroth, "The impact of channel bonding on 802.11n network management," Proc. Conf. Emer. Netw. Exp. Tech. (CoNEXT), Dec. 2011.

[44] "IEEE 802.11n-2009 - Wikipedia," Internet: http://en.wikipedia.org/wiki/IEEE_802.11n-2009, Access Jan. 20, 2019.

[45] M. S. Gast, "802.11n: A survival guide," O'Reilly Media, Inc., Mar. 2012.

[46] M. Vipin, and S. Srikanth, "Analysis of open source drivers for IEEE 802.11 WLANs," Proc. Int. Conf. Wireless Comm. and Sensor Comput. (ICWCSC 2010), pp. 1-5, IEEE, Jan. 2010.

[47] "arp-scan user guide," Internet: http://www.nta-monitor.com/wiki/index.php/Arp-scan_User_Guide, Access Jan. 20, 2019.

[48] "Arp Scan," Internet: http://www.nta-monitor.com/tools-resources/security-tools/arp-scan, Access Jan. 20, 2019.

[49] "nm-tool," Internet: http://linux.die.net/man/1/nm-tool, Access Jan. 20, 2019.

[50] "View your current network settings with nm-tool," Internet: https://nfolamp.wordpress.com/2010/05/21/view-your-current-network-settings-with-nm-tool/, Access Jan. 20, 2019.

[51] "Network Manager," Internet: https://help.ubuntu.com/community/NetworkManager, Access Jan. 20, 2019.

[52] J. Malinen, "hostapd: IEEE 802.11 AP, IEEE 802.1 X. WPA/WPA2/EAP/RADIUS authenticator," Internet: https://w1.fi/hostapd/, Access Jan. 20, 2019.

[53] D. J. Barrett, R. E. Silverman, and R. G. Byrnes, "SSH, the secure shell: the definitive guide," O'Reilly Media Inc., 2005.

[54] T. Ylonen and C. Lonvick, "The secure shell (SSH) protocol architecture," 2006.

[55] "OpenSSH," Internet: http://www.openssh.com/, Access Jan. 20, 2019.

[56] O. Terpollari, "How to install and configure OpenSSH server in Linux," Internet: http://www.tecmint.com/install-openssh-server-in-linux/, Access Jan. 20, 2019.

[57] "Running commands on a remote Linux server over SSH," Internet: http://www.shellhacks.com/en/Running-Commands-on-a-Remote-Linux-Server-over-SSH, Access Jan. 20, 2019.

[58] "nmcli - command-line tool for controlling NetworkManager," Internet: http://manpages.ubuntu.com/manpages/maverick/man1/nmcli.1.html, Access Jan. 20, 2019.

[59] "Using the NetworkManager command-line tool, nmcli," Internet: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Networking_Guide/sec-Using_the_NetworkManager_Command_Line_Tool_nmcli.html, Access Jan. 20, 2019.

[60] J. Tourrilhes, "iwconfig," Internet: http://www.linuxcommand.org/man_pages/iwconfig8.html, Access Jan. 20, 2019.

[61] T. Lei, Z. Lu, X. Wen, X. Zhao, and L. Wang, "SWAN: An SDN based campus WLAN framework," Proc. Int. Conf. Wireless Commun., Vehi. Technol., Inform. Theory., Aerospace & Electronic Systems (VITAE), pp. 1–5, IEEE, May 2014.

[62] E. Luengo, C. Weber, S. S. Heydari, and K. El-Khatib, "Design and implementation of a software-defined integrated wired-wireless network Testbed," Proc. The 13th ACM Int. Symp. on Mobility Management and Wireless Access, pp 47–52, ACM, Nov. 2015.

[63] S. Sukaridhoto, N. Funabiki, T. Nakanishi, and K. Watanabe, "A design for OpenFlow implementation of fixed backoff-time switching method in wireless mesh networks," Proc. IEICE Gen. Conf., BS-1-26, pp. S-51-52, Mar. 2013.

[64] S. Sukaridhoto, N. Funabiki, D. Pramudihanto, and Z. Arief, "A fixed backoff-time switching method for wireless mesh networks: Design and Linux implementation," Proc. Int. Conf. Inform. Technol. Elec. Eng. (ICITEE), pp. 248–253. IEEE, Oct. 2013.

[65] N. Ahmed and U. Ismail, "Designing a high performance wlan testbed for centralized control," Proc. The 5th Int. Conf. Testbeds and Research Infr. for the Dev. of Net. & Communities and Work., pp. 1–6, IEEE, Apr. 2009.

[66] F. G. Debele, M. Meo, D. Renga, M. Ricca, and Y. Zhang, "Designing resource-on-demand strategies for dense WLANs," IEEE J. on Selected Areas in Commun., vol. 33, no. 12, pp. 2494–2509, Dec. 2015.

[67] D. B. Faria, "Modeling signal attenuation in IEEE 802.11 wireless LANs," Tech. Report, TR-KP06-0118, Stanford Univ., July 2005.

[68] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," Combinatorica, vol.2, no.4, pp. 385-393, Dec. 1982.

[69] D. P. Williamson, and D. B. Shmoys, "The design of approximation algorithms," Cambridge Univ. Press, April 2011.

[70] "Hostapd: the linux way to create virtual Wifi access point," Internet: nims11.wordpress.com/2012/04/27/hostapd-the-linux-way-to-create-virtual-wifi-access-point/, Access Jan. 20, 2019.

[71] "Setting up a Raspberry Pi as a WiFi access point," Internet: https://cdn-learn.adafruit.com/downloads/pdf/setting-up-a-raspberry-pi-as-a-wifi-access-point.pdf, Access Jan. 20, 2019.

[72] M. E. Islam, K. S. Lwin, M. S. A. Mamun, N. Funabiki, and I.-W. Lai, "Measurement results of three indices for IEEE802.11n wireless networks in outdoor environments," The 17th IEEE Hiroshima Section Student Symposium, pp. 410-414, Nov. 2015.

[73] "Learning Linux Commands: sed," Internet: https://linuxconfig.org/learning-linux-commands-sed, Access Jan. 20, 2019.

[74] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," IEEE/ACM Trans. Networking, vol. 15, no. 3, pp. 560-573, June 2007.

[75] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," Proc. Int. Conf. Mobile Comput. Netw., pp. 315-329, 2004.

[76] A. So and B. Liang, "An efficient algorithm for the optimal placement of wireless extension points in rectilineal wireless local area networks," Proc. Int. Conf. Quality. Service. Hetero. Wired/Wireless Netw., 2005.

[77] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. P. Mhatre, "Measurement-driven guidelines for 802.11 WLAN design," IEEE/ACM Trans. Networking, vol. 18, no. 3, pp. 722-735, June 2010.

[78] S. Tang, L. Ma, and Y. Xu, "A novel AP placement algorithm based on user distribution for indoor WLAN system," China Commun., vol. 13, no. 10, pp. 108-118, Oct. 2016.

[79] S. Miyata, T. Murase, and K. Yamaoka, "Novel access-point selection for user QoS and system optimization based on user cooperative moving," IEICE Trans. Commun., vol. 95, no. 6, pp. 1953-1964, 2012.

[80] S. K. Lundsgaard, M. S. Bhally, G. M. Di Prizio, and T. Lee, "Selection of a prepared access point from among a plurality of access points," US Patent 8,654,741, Feb. 18, 2014.

[81] Y. H. Seok, "Method for providing information of access point selection," US Patent 9,008,210, Apr. 14, 2015.

[82] H. Gong and J. Kim, "Dynamic load balancing through association control of mobile users in WiFi networks," IEEE Trans. Consumer Electronics, vol. 54, no. 2, pp. 342-348, May 2008.

[83] X. Chen, Y. Zhao, B. Peck, and D. Qiao, "SAP: Smart access point with seamless load balancing multiple interfaces," Proc. INFOCOM, pp. 1458-1466, 2012.

[84] "Tcpdump and libpcap," Internet: http://www.tcpdump.org, Access Jan. 20, 2019.

[85] "apt-mirror," Internet: https://apt-mirror.github.io/, Access Jan. 20, 2019.

[86] "APACHE HTTP Server Project," Internet: https://httpd.apache.org/, Access Jan. 20, 2019.

[87] "Cron," Internet: https://en.wikipedia.org/wiki/Cron, Access Jan. 20, 2019.

[88] "All about secure apt"Internet: https://wiki.debian.org/SecureApt, Access Jan. 20, 2019.

[89] "Introduction to Apt Authentication"Internet: https://help.ubuntu.com/community/SecureApt, Access Jan. 20, 2019.

[90] "How is the authenticity of Ubuntu updates verified?"Internet: https://askubuntu.com/questions/710158/how-is-the-authenticity-of-ubuntu-updates-verified, Access Jan. 20, 2019.

[91] "The FreeRADIUS Server Project," Internet: https://freeradius.org/about/, Access Jan. 20, 2019.

[92] D. Qiao, S. Choi, and K. G. Shin, "Interference analysis and transmit power control in IEEE 802.11a/h Wireless LANs," IEEE/ACM Trans. Networking, vol. 15, no. 5, pp. 1007-1020, Oct. 2007.

[93] X. Gong, D. Plets, E. Tanghe, T. D. Pessemier, L. Martens, and W. Joseph, "An efficient genetic algorithm for large-scale transmit power control of dense and robust wireless networks in harsh industrial environments," Appl. Soft Comput., vol. 65, no. 5, pp. 243-259, 2018.

[94] "8 reasons to turn down the transmit power of your Wi-Fi," https://metis.fi/en/2017/10/txpower/, Access Jan. 20, 2019.

[95] A. Kachroo, J. Park, and H. Kim, "Channel assignment with transmission power optimization method for high throughput in multi-access point wlan," Int. Wire. Com. and Mob. Comp. Conference, pp. 314-319. IEEE,2015.

[96] B. P. Tewari, and S. C. Ghosh, "Combined power control and partially overlapping channel assignment for interference mitigation in dense WLAN." IEEE Int. Conf. Adv. Inform. Net. App, pp. 646-653. 2017.

[97] G. Giovanna, H. I. D. Monego, M. E. Pellenz, R. D. Souza, A. Munaretto, and M. S. P. Fonseca, "An iterative heuristic approach for channel and power allocation in wireless networks," Annals of Telecommunications, pp. 1-11, 2018.

[98] "Understanding modulation and coding schemes," https://www.controleng.com/single-article/understanding-modulation-and-coding-schemes.html, Access Jan. 20, 2019.

[99] K. S. Lwin, K. K. Zaw, and N. Funabiki, "Throughput measurement minimization for parameter optimization of throughput estimation model," Proc. Chugoku-Branchi J. Conf., Oct. 2017.

[100] "T-Test Definition," Internet: https://www.investopedia.com/terms/t/t-test.asp, Access Jan. 20, 2019.