Load-Aware Traffic Control in Software-Defined Enterprise Wireless Local Area Networks

Yunong Han

School of Computer Science and Electronic Engineering

University of Essex

This thesis is submitted for the degree of

Doctor of Philosophy

January 2018

Abstract

With the growing popularity of Bring Your Own Device (BYOD), modern enterprise Wireless Local Area Networks (WLANs) deployments always consist of multiple Access Points (APs) to meet the fast-increasing demand for wireless access. In order to avoid network congestion which leads to issues such as suboptimal Quality of Service (QoS) and degraded user Quality of Experience (QoE), intelligent network traffic control is needed. Software Defined Networking (SDN) is an emerging architecture and intensively discussed as one of the most promising technologies to simplify network management and service development. In the SDN architecture, network management is directly programmable because it is decoupled from forwarding layer. Leveraging SDN to the existing enterprise WLANs framework, network services can be flexibly implemented to support intelligent network traffic control.

This thesis studies the architecture of software-defined enterprise WLANs and how to improve network traffic control from a client-side and an AP-side perspective. By extending an existing software-defined enterprise WLANs framework, two adaptive algorithms are proposed to provide client-based mobility management and load balancing. Custom protocol messages and AP load metric are introduced to enable the proposed adaptive algorithms. Moreover, a software-defined enterprise WLAN system is designed and implemented on a testbed. A load-aware automatic channel switching algorithm and a QoS-aware bandwidth control algorithm are proposed to achieve AP-based network traffic control.

Experimental results from the testbed show that the designed system and algorithms significantly improve the performance of traffic control in enterprise WLANs in terms of network throughput, packet loss rate, transmission delay and jitter.

Acknowledgements

This thesis would not exist without the help of many people that supported me throughout the years of research. First of all, I would like to express my deepest gratitude to my supervisor Prof. Kun Yang, who helped me with his advice and inspirational ideas. Without his continuous support, all this would not have been possible. Also, I would like to take the opportunity to thank my colleagues and friends from the School of Computer Science and Electronic Engineering at the University of Essex, in particular, the Network Convergence Laboratory group. Their support made University of Essex to a great place to study and live. Finally, I would like to send my huge thanks to my parents and my lovely wife as they always have loved me, trusted me and encouraged me throughout all the long years of study.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Background

Over the past decades, IEEE 802.11-based Wireless Local Area Networks (WLANs) [1] have become the global trend in wireless communication technologies because of their characteristics such as easy deployment, great mobility and high data throughput. WLANs can be deployed in homes, campus, companies and more recently even on aeroplanes, in cars or on trains as wired network extensions to provide Internet access that enables users to roam freely anywhere within the coverage. Because of the random additions of APs, unmanaged WLANs suffers interference and unbalanced traffic, which lead to significant performance degradation. To solve the issue caused by unmanaged WLANs, enterprise WLANs are introduced. Enterprise WLANs are different from typical WLAN infrastructure mode by introducing a special entity called the wireless controller. Fig.1.1 shows an example of the traditional architecture of enterprise WLANs. An enterprise WLAN normally consists of one wireless controller, several LAN switches and multiple APs. The wireless controller runs on the server cannot only control various parameters of the AP such as operating channel and transmission power but also monitor the whole network by collecting wireless medium information and all APs' status information. Furthermore, centralised network services such

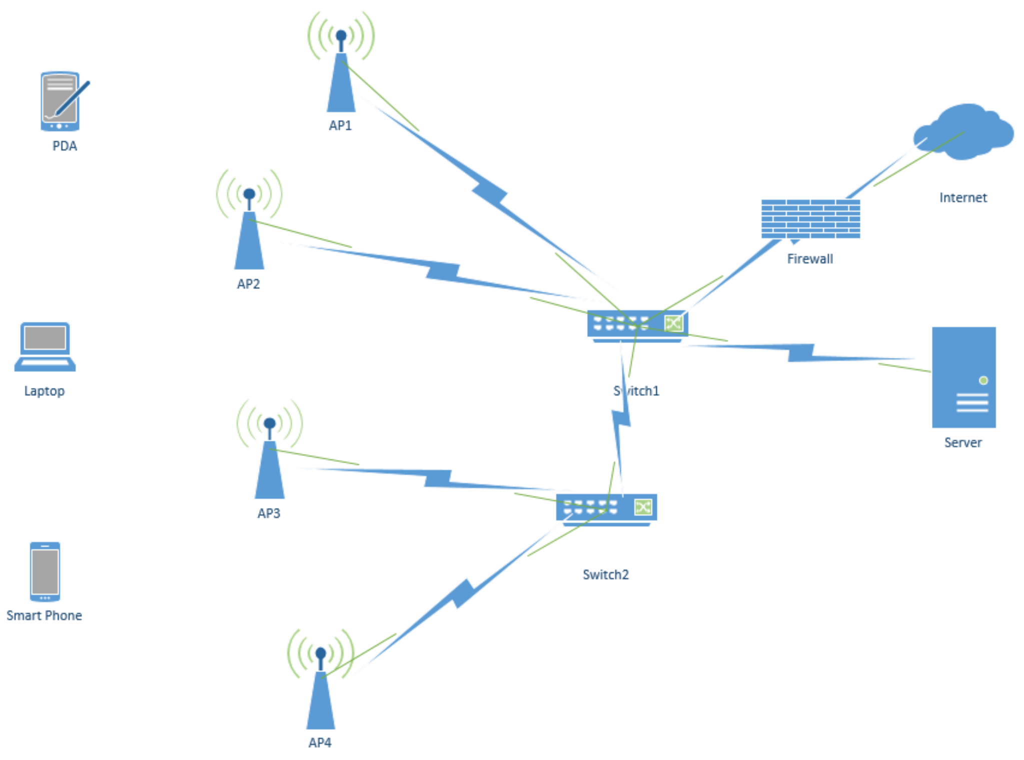


Fig. 1.1 Traditional architecture of an enterprise WLAN

as mobility management, interference management and load balancing can be implemented on the wireless controller to guarantee the network Quality of Service (QoS) and user Quality of Experience (QoE). In order to achieve those control functions and network services, the wireless controller needs information such as Received Signal Strength Indication (RSSI) of APs and clients, traffic load of APs and so on.

Software-Defined Networking (SDN) is an emerging architecture and intensively discussed as one of the most promising technologies to simplify network management and service deployment. The main idea of SDN is to decouple the forwarding of data flow from network control that enables the underlying infrastructure to be highly abstracted for applications and network services. In the traditional network architecture, both control and data plane are managed by network devices, and the vendor-proprietary software that is installed on those devices are extremely hard to be modified. However, with SDN, the network intelligence is logically centralised in the software-based SDN controller to maintain a global

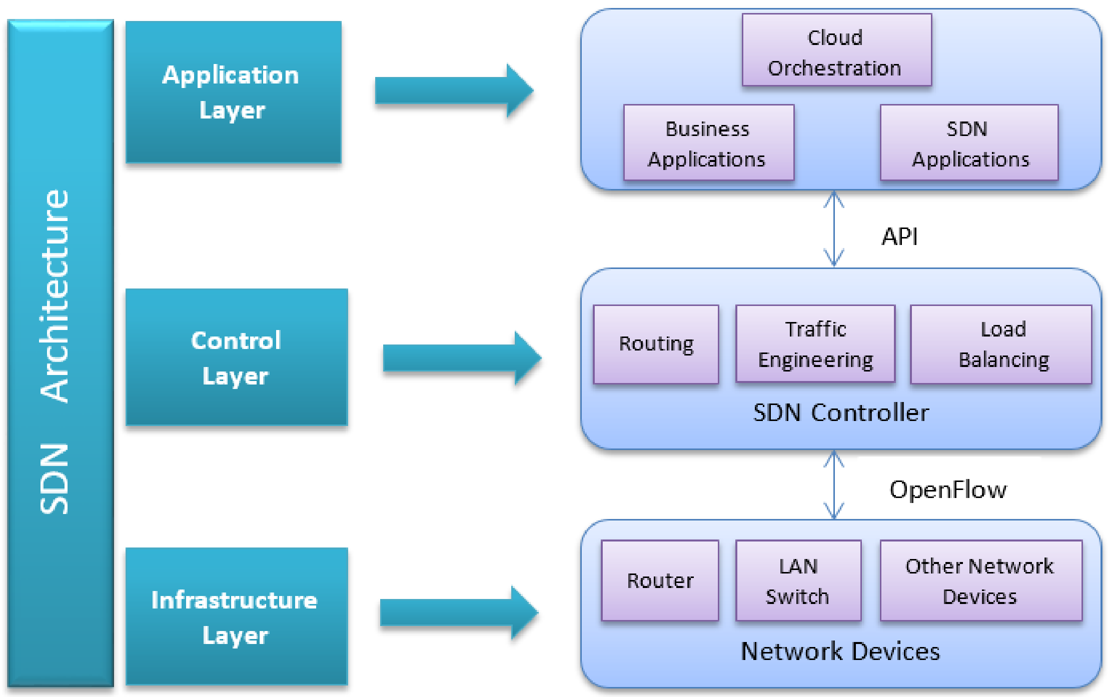


Fig. 1.2 Architecture of SDN

view of the network. As a result, the network appears to the applications and network services as a single, logical switch, which greatly simplifies the network design and operation. Driven by the architecture presented in [2], the *Open Networking Foundation* (ONF) proposed the SDN logical architecture [3], which is illustrated in Fig.1.2. In this architecture, an SDN can be divided into an application layer, a control layer and an infrastructure layer. Network devices in the infrastructure layer provide access to the data plane and communicate with the SDN controller, which acts as the control plane for the network. The control plane then exposes open Application Programming Interfaces (APIs) to applications and network services that are implemented in the application layer. In recent years, SDN has been implemented into several areas such as Internet research [4], rural connectivity [5], updating data centre networks [6], mobile application offloading [7] and mobile virtual machine [8].

OpenFlow [9] is the first standard communications interface defined between the control and forwarding layers of the SDN architecture. It is a key enabler for SDN and currently is the only standardised SDN protocol that allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual. The OpenFlow protocol is implemented on both sides of the interface between network infrastructure devices and the SDN control software. OpenFlow uses the concept of flows to identify network traffic based on pre-defined match rules that can be statically or dynamically programmed by the SDN control software. Since OpenFlow allows the network to be programmed on a per-flow basis, an OpenFlow-based SDN architecture provides extremely granular control, enabling the network to respond to real-time changes at the application, user, and session levels. Fig.1.3 illustrates the architecture of an OpenFlow switch. An OpenFlow logical switch consists of one or more Flow-Tables, a Group-Table and one or more OpenFlow Channel. Each Flow-Table maintains multiple flow entries, and every flow entry is associated with an action to tell the switch how to process the flow respectively. The Group-Table performs packet lookups and forwarding, and the OpenFlow Channel connects the switch to a remote control process which allows commands and packets to be sent between the controller and switch. By using the OpenFlow protocol, the controller can add, update and delete flow entries in Flow-Tables, both reactively and proactively. Each flow entry consists of three main fields which are match fields, counters and a set of instructions to apply to matching packets. The match fields contain a set of packet headers which specifies the flow while the counters update the statistics such as the number of forwarded and received packets for each flow. Instructions associated with each flow entry either contain simple actions (e.g. forward packets to a given port (or ports), encapsulate and send packets to the controller, drop packets) or modify pipeline processing which allows packets to be transferred to subsequent Flow-Tables for future processing.

1.2 Problem Statement and Motivation

Today's modern enterprise WLANs typically consist of dozens to hundreds of APs providing WLAN services for multiple user devices such as smartphones, laptops and tablet PCs and

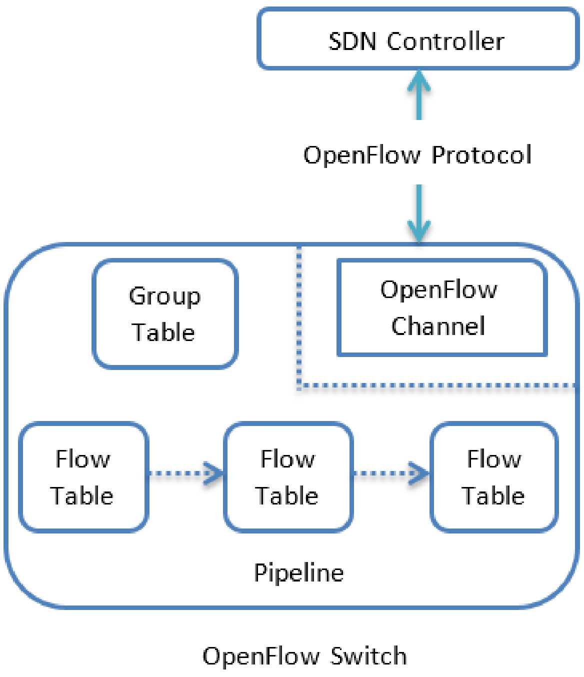


Fig. 1.3 Architecture of an OpenFlow Switch

if the traffic load is concentrated to specific APs, clients associated to those APs suffer from significant performance degradation. To avoid issues caused by network congestion, enterprise WLANs always require network services such as mobility management, load balancing, and bandwidth control to run on top of the fundamental management functionality of the individual APs to achieve efficient network traffic control. However, different devices offered by different vendors may not provide the needed functions because of the difference of devices' firmware. Additionally, with the growing popularity of Bring Your Own Device (BYOD) that is extensively adopted in today's enterprise WLANs, it is more difficult for the network to accommodate an even more diverse set of client device types. Moreover, the complexity of setting up, management and maintenance of current enterprise WLAN is heavily increased due to the explosion of novel technologies such as Network Function Virtualization (NFV) and cloud computing. Thus, the author argues that the traditional static network architecture is ill-suited to orchestrate the requirements of modern enterprise

WLANs and novel network paradigms as well as improved network services, are needed for network operators to address the growing complexity.

In IEEE 802.11 WLANs, the handover function refers to the sequence of messages exchanged by a mobile station and APs which then results in a transfer of state information and physical layer connectivity of this wireless station from one AP to another. The whole handover process can be divided into three steps which are scanning, authentication and re-association. During the initial stage of a handover, the mobile station first scans the wireless channels either passively or actively. In the passive scan mode, the mobile station listens to the beacon frame which is sent out by APs periodically and decides the best target AP to associate with based on the information from the beacon frame. In the active scan mode, the mobile station not only waits for beacon frame but also broadcasts additional probe request frame and receives probe responses from APs. Once the scanning step is finished, the mobile station attempts to authenticate and re-association with the newly selected target AP. The station credentials and other state information will be transferred from the old-AP to the new-AP during these two steps, and after that, the handover process is finished. Also, the 802.11 standard lets clients make AP association decisions locally, and the infrastructure has no control over these decisions. With the trend of fast adaption to implement real-time multimedia services, such as video conferencing over enterprise WLANs, seamless and infrastructure controlled handover of client devices has become one of the crucial criteria to guarantee satisfied QoS and user QoE. To enable efficient control on client handover in current enterprise WLANs, intelligent mobility management service is required. With conventional mobility management algorithm, clients always prefer to connect to the AP with the best signal strength. However, this mechanism can cause unbalanced traffic load among APs as it does not consider the AP load condition during handover.

Load balancing is one of the major challenges in current enterprise WLANs. Traffic load on APs is often unevenly distributed as wireless stations (WS) independently select the AP to associate with, where APs with a large number of associated users have much heavier traffic load than those with less associated users. The unbalanced traffic distribution can cause issues such as constant packet loss and sub-optimal network throughput, which dramatically degrades network QoS and user QoE. Although AP load balancing problems have been extensively studied for years, existing solutions continue to exhibit inefficiencies. Because without knowing future traffic demands on individual APs, it is very hard to make an optimal assignment or adjustment of WLAN users among a set of APs. Additionally, it can cause link disruption when dynamically migrate users from heavy-duty APs to those with light traffic load. In order to achieve efficient AP load balancing and maintain satisfied QoS and user QoE, seamless user mobility over multiple wireless channels is required. Moreover, not only the AP signal strength but also the AP traffic load need to be considered when making load balancing decisions.

Due to the explosive increase in demand for wireless connectivity, modern enterprise WLANs always deploy multiple APs to provide large signal coverage. APs based on IEEE 802.11 standards can operate on the 2.4GHz and the 5GHz unlicensed industrial, scientific and medical (ISM) band. The 5GHz band offers a large number of non-overlapping channels, but higher frequency bands have a larger propagation loss, which reduces the effective coverage of the signal. In the 2.4GHz band, there are only three non-overlapping channels, which are clearly not enough for dense deployments. When two or more APs operate on the same channel frequency, their associated users will have to contend for the same portion of the spectrum. In case that two users transmit at the same time, their radio signals collide which results in data corruption and packet loss. In order to avoid collisions, 802.11 users perform the Clear Channel Assessment (CCA) to detect if other users are transmitting on the channel before starting its own transmission. If another transmission is detected, it will wait for a random short period after which it would perform another check before attempting to transmit again. With the increasing number of contenders on the same channel, users have to wait longer to send data. This transmission delay is caused by the so-called co-channel interference. On the other hand, an interfering transmission on a partially-overlapping channel is called adjacent channel interference. Depending on its intensity and on the channel distance, adjacent channel interference can cause data corruption and packet loss as the CCA check may not detect the collision due to channel overlapping. With the number of interfering users increasing, the potential of re-transmission is raised, and it takes longer for users to successfully send or receive a data packet. The outcome of both types of channel interference is the degradation of network performance, such as low network throughput, high packet loss rate and transmission delay, which can significantly affect QoS and user QoE. To minimise the channel interference issue, optimised channel allocation mechanism is critical to modern enterprise WLANs in order to make intelligent network traffic control, especially in the 2.4GHz band. In traditional enterprise WLANs deployment, network administrators manually assign the operating channel for APs based on detailed site surveys. However, with the increasing traffic variability and user mobility, the performance of existing static channel assignment approach is bound to suffer; furthermore, it is insufficient for modern enterprise WLANs as the airtime demand in enterprise WLANs can vary significantly both across APs and across time. To be effective for usage patterns of modern enterprise WLANs, a centralised automatic channel switching approach that adapts to the rapidly changing wireless environment and supports seamless user channel switching is needed.

Bandwidth control in another key service to enable efficient traffic control in enterprise WLANs. In the past few years, the introduction of BYOD attracts significant attention on improving enterprise productivity and efficiency. According to the prediction from Gartner [10], 50% of employers will require employees to supply their own devices for work purposes by 2017. With the fast increasing trend of BYOD in modern enterprise WLANs, bandwidth control becomes a key challenge for network administrators. In general, wireless bandwidth resource should be allocated according to user priorities. For instance, staff and BYOD users should be assigned with higher priorities than visitors as the data traffic generated by staff and BYOD users are more concerned about the productivity of the company. To manage the bandwidth allocation, network administrators always configure relative policies on APs to distribute a larger amount of bandwidth to users with high priorities. However, users in an enterprise WLAN may roam between the coverage of APs, resulting in changing users'association state. Thus, the conventional bandwidth control solution based on fixed bandwidth allocation policy is inefficient for modern enterprise WLANs. Because network traffic varies with the fast-changing users'association state, it's vital for the bandwidth control algorithm to be adaptive to the AP traffic load in order to perform efficient dynamic bandwidth control on users with different priorities.

Because of the low-cost equipments and ease of setup, WLAN deployments used to be unplanned, which lead to a large number of uncoordinated and interfering deployments. To solve this issue, novel WLAN deployments have emerged, especially in the large enterprise, where network management is performed by centralised network controllers. In order to achieve intelligent network traffic control in modern enterprise WLANs, commercial vendors offer solutions [11] [12] [13] through vendor proprietary software and hardware. However, these solutions do not have open programmable interface and are unable to be achieved with the low-cost commodity AP hardware that is used by provider networks. Recently, SDN has gained increasing popularity as a novel network architecture, which introduces programmability to provide more flexibility in network management. SDN aims to separate control policies from forwarding mechanisms and enable network administrators to easily implement control policies through a set of high-level open APIs. Leveraging SDN to the existing enterprise WLAN architecture, network intelligence can be logically centralised and infrastructure controlled network traffic management can be achieved with the low-cost OpenWrt-based [14] AP hardware. By utilising the high-level open APIs, network services such as mobility management, load balancing, automatic channel switching and bandwidth control can be simply added on top of the central controller, and by monitoring real-time network traffic load, the performance of these network services can be improved significantly.

1.3 Contributions

This thesis merges SDN into the existing architecture of enterprise WLANs, and investigates how to improve network traffic control from a client-side and an AP-side perspective. Moreover, five novel algorithms for WLAN-specific network services are proposed to achieve efficient traffic control in software-defined enterprise WLANs. A brief description of each contribution is listed below.

Adaptive Mobility Management and Load Balancing

Mobility management and load balancing are two key network services to achieve efficient traffic control in enterprise WLANs. As the first contribution of this thesis, the author proposes two novel adaptive algorithms and implements both algorithms on an existing software-defined enterprise WLAN framework to improve the performance of client mobility management and AP load balancing respectively. AP load is defined as a novel metric together with the RSSI to enable the proposed adaptive algorithms, and a custom protocol message called Forward_AP_Load is introduced to enable the central controller to have the knowledge of each AP's load information. Channel Switch Announcement (CSA) is added to custom beacon frame to achieve seamless client over multiple wireless channels. The results have been published in paper 1 and 2.

Load-Aware Automatic Channel Switching

Channel interference is one of the major problems that can significantly affect the network performance. To mitigate the influence of channel interference, intelligent channel selection mechanism is required in modern enterprise WLANs. As the second contribution of this thesis, the author presents a software-defined enterprise WLAN system and a load-aware automatic channel switching solution. Two channel switching algorithms are proposed to improve the overall user QoE and the QoE of users with highest traffic load respectively. AP load which is defined in Chapter 3 is used together with a novel metric called Channel Interference Factor (CIF) to enable the proposed algorithms. In order to make the central controller aware of the channel information of each AP, a new custom protocol message called Forward-AP-Chan is defined. Beacon frame with CSA is used to achieve seamless channel switching. The system and the results have been published in paper 3.

QoS-Aware Bandwidth Control

With the growing trend of BYOD, bandwidth control becomes an indispensable network service in enterprise WLANs that enable AP-based traffic control on clients with different priorities. As the final contribution of this thesis, the author proposes a QoS-aware bandwidth control algorithm and implements it on the software-defined enterprise WLAN system which is presented in Chapter 4. The proposed algorithm utilises AP load that is defined in Chapter 3 as the bandwidth control metric to achieve adaptive uplink bandwidth control on users with different priorities. Virtual OpenFlow switch Open vSwitch [15] is used to achieve queue-based traffic shaping, and OpenFlow protocol is used to direct user traffic flows. The main contents have been submitted for publication as paper 4.

1.4 Structure of Thesis

The rest of this thesis is organised as follows. Chapter 2 presents comprehensive and critical literature reviews of software-defined enterprise WLAN systems and traffic control services which include mobility management, load balancing, automatic channel selection and bandwidth control. The literature review chapter is followed by Chapter 3, where two adaptive algorithms are proposed and implemented on an existing software-defined enterprise

WLAN framework to improve the performance of client mobility management and AP load balancing respectively. Chapter 4 introduces a software-defined enterprise WLAN system and a load-aware automatic channel switching solution. Two channel switching algorithms are proposed to improve the overall user QoE and the QoE of users with highest traffic load respectively. Chapter 5 further extends the system that is presented in Chapter 4 by proposing and implementing a QoS-aware bandwidth control algorithm to achieve dynamic bandwidth control on users with different priorities. Finally, Chapter 6 concludes the thesis with a summary and an outlook to future work.

Chapter 2

Literature Review

2.1 Introduction

SDN is a new paradigm for the programmable network, and it has demonstrated to be an important technology driver for flexible service development. Recent advances in SDN and OpenFlow has led the trend of reformation of wired networks with flexible programmatic control mechanisms. Enabling SDN for enterprise WLAN presents a promising perspective towards more manageable, agile enterprise WLAN deployments that network services can be easily implemented. This chapter provides an insight into this exciting new area starting with the introduction of the existing software-defined enterprise WLAN frameworks. Then moving into a technical review of the network services (i.e., Mobility Management, Load Balancing, Automatic Channel Switching and Bandwidth Control) that are used for traffic control in enterprise WLANs. Finally, the related state-of-the-art network services in software-defined enterprise WLANs are introduced by highlighting their key technical features as well as their drawbacks.

2.2 Software-Defined Enterprise WLANs

In recent years, several research works have been focused on investigating the possibility of implementing SDN into enterprise WLANs. For instance, the authors in [16] [17] have presented a system called FlowVisor which uses OpenFlow as a means to enable wireless network virtualisation. FlowVisor is built on OpenFlow as an abstraction of the underlying hardware, and it switches virtualisation in which the same hardware forwarding plane can be shared among multiple logical networks, each with distinct forwarding logic. The authors clarify that FlowVisor runs on existing or new low-cost hardware, is backwardly compatible with the current legacy network and might be used to virtualise networks in data centres, enterprises, homes, Wide Area Networks (WANs) and so on.

OpenRoads [18] [19] is the first work that implements OpenFlow into mobile wireless networks. The OpenRoads'architecture consists of three layers: flow, slicing and controller. These layers provide flexible control, virtualisation and high-level abstraction which make OpenRoads an open platform to implement wildly different algorithms and run them concurrently in one network. OpenRoads also incorporates multiple wireless technologies, specifically Wi-Fi and Worldwide Interoperability for Microwave Access (WiMAX). Also, FlowVisor [16] [17] is used to create and isolate network slices which enable multiple experiments to run simultaneously in production wireless network, and Simple Network Management Protocol (SNMP) demultiplexer is implemented to mediate device configuration access among experiments.

In [20] [21], the authors present CloudMAC, an OpenFlow-based novel architecture for enterprise WLANs where the MAC layer processing is partially offloaded to virtual machines provided by cloud services. With CloudMAC, APs only perform MAC frames forwarding between virtual APs and mobile stations. All the processing of MAC layer frames and the generation of management frame is done by virtual APs while the binding between the virtual APs and the physical APs is managed using OpenFlow. With the integration of

OpenFlow, CloudMAC achieves similar performance as normal WLANs but a higher level of flexibility and programmability which allows novel services to be easily implemented with high-level programming languages. Moreover, the authors present a case study which shows that dynamically switching off APs to save energy can be performed seamlessly with CloudMAC, while a traditional WLAN architecture causes large interruptions for users.

Kim *et al.* [22] proposes the OpenFlow AP system that is a software-defined enterprise WLAN system based on the OpenFlow-based AP and the OpenFlow controller. By extending the OpenFlow protocol to apply to WLAN control, it applies the SDN framework to Enterprise WLAN and achieves seamless mobility and total network throughput improvement. However, it uses expected data rate based on the RSSI of the client to select the best handover target which cannot be trusted completely as the channel interference can be quite large when in a noisy environment. Also, it handover low data rate client to most crowded AP to alleviate the performance anomaly effect which sacrifices the user experience of low data rate clients.

The authors in [23] [24] present a scalable software-defined wireless network called AeroFlux. AeroFlux uses a 2-tiered control plane: the Global Control (GC) plane and the Near-Sighted Control (NSC) plane. The GC is logically centralised to handle network functions that require global visibility, such as mobility management and load balancing, whereas the NSCs are located close to the wireless APs to control per-client or per-flow transmission settings such as power and rate. By offloading latency-critical or high-load tasks from the GC to the NSCs, AeroFlux supports large enterprise and carrier Wi-Fi deployments with low latency programmatic control of fine-grained Wi-Fi specific transmission settings. Moreover, AeroFlux inherits LVAP abstraction from Odin to enable infrastructure controlled user handovers and application-aware service differentiation.

Zhao *et al.* [25] [26] proposes a flexible architecture of enterprise WLAN called SD-WLAN which introduce two salient features. First, most of 802.11 AP functions are decoupled from forwarding devices and centralised in an SDN controller, leaving some simplified devices (i.e., wireless AP) manipulated by the controller through extended OpenFlow protocol. Second, the control of APs and wired backbone are consolidated to provide a unified network control platform. By reorganising 802.11 AP's functional modules, SDWLAN can achieve remarkable flexibility. Moreover, SDWLAN requires no client-side modification and supports a client-unaware fast AP handoff mechanism which achieves negligible throughput fluctuation of on-going connection compared to traditional architecture with 802.11 standard handoff mechanism.

Odin [27–30] is a novel OpenFlow-based SDN framework for enterprise WLANs. Based on the architecture of SDN, the system separates the control plane from the data plane by having a logically centralised SDN controller, which uses OpenFlow and a custom control plane protocol to manage the wired and wireless networks respectively. Odin provides a virtual AP association for each mobile client by introducing a novel programming abstraction called Light Virtual Access Points (LVAP). Every time a client performs a probe scan, the central controller assigns an LVAP on the physical AP that first receive the probe request. The assigned LVAP is what responds to the client's probe scan with a probe response. After receiving the probe response, the client can then continue the regular 802.11 association handshake with the LVAP. Once the handshake is finished, the association between a client device capable of mobility, and an LVAP is formed. At this point, the client is only concerned with getting acknowledgements (ACKs) for the data frames it generates and receiving beacons from its LVAP in a fixed time interval. With LVAP abstraction, clients no longer need to re-associate with AP during a handoff process as the LVAP can be migrated between physical APs and the migration is fast enough such that it retains all the association state of the client, who continues to receive ACKs and beacon frames, and transmit data frames without noticing any period of disconnectivity. This method works because the client only cares about the responses it gets from an AP which is identified by a Basic Service Set Identifier (BSSID) and is oblivious to the actual physical radio that is generating these responses. Therefore, LVAP abstraction decouples the association state of a Wi-Fi link from the physical APs in the network and makes clients see the same consistent virtual AP regardless of the actual physical AP the client is connected.

Riggio *et al.* propose EmPOWER [31–34], a set of high-level programming abstractions modelling the fundamental aspects of a wireless network, namely state management, resource provisioning, network monitoring, and network reconfiguration. The proposed abstractions hide away the implementation details of the underlying wireless technology which enable a flexible management of the network. Moreover, a Software-Defined Radio Access Network (SDRAN) controller for Enterprise WLANs and a Python-based Software Development Kit (SDK) implementing the proposed abstractions are presented. The authors further clarify that the platform can be effectively leveraged in order to implement typical control tasks such as mobility management and traffic engineering as well as applications and services such as multicast video delivery and dynamic content caching.

By building upon Odin [27–30] and AeroFlux [23], the authors in [35–38] introduces OpenSDWN, an SDN and Network Functions Virtualisation (NFV) based novel Wi-Fi framework. OpenSDWN exploits virtualisaton across the wired and wireless network and introduces datapath programmability to enable service differentiation and fine-grained transmission control, facilitating the prioritisation of critical applications. OpenSDWN implements per-client virtual access points and per-client virtual middleboxes, to render network functions more flexible and support mobility and seamless migration. Moreover, OpenSDWN increases the security of upcoming Wi-Fi HotSpot architectures by following a functional split approach. Finally, the authors clarify that OpenSDWN can also be used to out-source the control over the home network to a participatory interface or an Internet Service Provider (ISP).

The authors in [39] propose ISD-WiFi (Intelligent Software-Defined WiFi), a programmable networking architecture for enterprise WLANs. In ISD-WiFi, an intelligent centre is introduced to enable information sharing between SDN controllers and is responsible for learning about the station's traffic pattern and making predictions. Because user traffic often shows periodicity for regular movement, user traffic pattern can be learned by using historical traffic statistics. By adding new reason type for Packet-in message and new actions for Packet-out message in the extensions of OpenFlow, SDN controllers in ISD-WiFi could periodically request APs for interference and traffic statistics information, which is sent to intelligent centre for storage and further analysis. Based on intelligent SDN, ISD-WiFi could offer programmability and flexible network applications such as access control, mobility management and interference management for enterprise WLANs.

In [40], the authors propose a new SDN-based architecture named Light-weighted Edge Network Virtualization (LENV) to provide a scalable and agile management for enterprise WLANs. Based on the concept of SDN, LENV virtualises edge network functions towards a centralised logical module called Virtual Wireless Access Device (VWAD), which hosts in the wireless AP. Each VWAD corresponds to an associated client and can be seen like a virtual AP to guide the physical entity to provide network functions based on client's requirement. With the help of VWADs, the authors develop two network management services which are access control and mobility management. The proposed access control scheme achieves a feasible joint design between the authentication process and access authority management according to clients'identifies. While for mobility management, the authors consider not only the specific mobility handover scheme but also the handover policies based on reversed signal strength measurement.

The authors in [41] develop a software-defined networking architecture for 802.11 dense WLANs called Ethanol. The Ethanol architecture is composed of two types of devices which are the POX-based SDN controller [42] and Ethanol-enabled APs. The Ethanol-enabled AP consists two parts that are wired and wireless components. The wired component is a configurable switching element that supports the OpenFlow protocol. Because OpenFlow does not provide a control interface for QoS and wireless components, custom APIs are added to the AP by Ethanol agent which is running on top of the AP for controlling wireless links and for defining the QoS of wired flows. Ethanol agent receives custom messages from the Ethanol Controller via a secure channel. With Ethanol-enabled AP which is programmable, the Ethanol architecture refactors the control plane functionalities between the APs and the controller. The centralised controller actuates on APs, controlling features such as QoS support, user mobility and association control, link-level parameters and virtual APs by using custom APIs. The authors demonstrate the feasibility by implementing the proposed architecture on OpenWrt-based commodity APs. Three use cases involving quality of service, the control of client association, and Address Resolution Protocol (ARP) packet suppression are studied, and experiment results show that Ethanol-enabled APs can significantly enhance network performance.

In [43], the authors propose SWAN (Software defined Wireless Access Network), an open campus WLAN framework based on SDN. SWAN uses a logically centralised controller to control APs in the campus WLAN. The controller has a global view of the network and defines programmable interfaces to configure and manage the whole network by programming network applications. Each AP in the system runs an agent, and the agent uses a special protocol called SWAN protocol to communicate with controller. In addition, SWAN uses software access point (SAP) to abstract the connection between the user equipment (UE) and the AP. Each physical AP can hold multiple SAPs, which virtualise association and authentication state and separate them from physical APs. From the point of network operators'view, multiple UEs connected to a single physical AP are treated as a set of logically isolated UEs connected to different ports of a switch. The authors clarify that with the user association abstraction, network applications such as access control, seamless handover and load balancing could be realised in SWAN. However, no physical testbed and experiment results are presented to evaluate the performance of the proposed framework, which raises a question regarding its feasibility.

2.3 Traffic Control in Enterprise WLANs

According to research work [44], the applications used on the WLAN changed dramatically. Initial WLAN usage was dominated by Web traffic. However, with the growing number of smartphones and fast increasing network bandwidth, there are significant increases in peer-to-peer, streaming multimedia, and voice over IP (VoIP) traffic. On-campus traffic now exceeds off-campus traffic, a reversal of the situation at the WLAN's initial deployment. In order to guarantee satisfied QoS and user QoE, modern enterprise WLANs always use specific network services such as mobility management, load balancing, automatic channel switching and bandwidth control to achieve traffic control. In the past decades, many research works have been conducted to improve the performance of these network services. However, the existing solutions continue to exhibit inefficiencies. With the growing popularity of SDN, software-defined enterprise WLANs provide a promising solution for achieving intelligent traffic control.

2.3.1 Mobility Management

In current enterprise WLANs, mobility management service has a great influence on user-perceived QoS due to the service disruption during a handover process. Each of the handover phases incurs different extent of delays. The longest delay is believed to occur during scanning phase based on research work in [45]. Thus, most previous research work has been focused on reducing the scanning delay. In SyncScan [46], authors proposed using implicit time synchronisation to reduce the key cost of scanning new target APs. By synchronising the announcement of beacon frames, clients can arrange to listen to other wireless channels with very low overhead. As a result, the system is able to perform a fast handover in 5ms. However, driver modification to the client network adapter is required to achieve the proposed system. The authors in [47] reduced the probe delay by using a selective scanning algorithm combined with a caching mechanism. With the proposed selective scanning algorithm, the station only scans the orthogonal channels of station's current channel. However, it may still need to perform a full scan depends on the wireless channel setting of neighbour APs.

In FastScan [48], a client-based database which stores information about the neighbour best APs and their corresponding channel settings is used to reduce the scanning delay. However, the database needs to be updated to maintain the correct information which led to additional scanning time. In [49], each AP is equipped with dual wireless network interface cards (NICs) to reduce handoff delay. One interface is used to perform typical AP functions, and the other one is dedicated to handover assistance. With the additional interface and help of the central controller, the mobile station can maintain the association with the current connected AP during the scanning process which guarantees the QoS for WLAN services. However, it requires the central controller to maintain and update a neighbour information table and the states of the second interface of neighbour APs need to be checked by the controller before a handover can be processed. This process could introduce additional delay when the number of neighbour APs increased and as a result, the handover delay could increase.

The necessity for infrastructure-controlled handovers is not only motivated by the requirement for seamless user mobility but also has the potential for being used to balance the network traffic load. Many of these existing research works achieve these handover mechanisms by installing some specialised software on the client side which the infrastructure can use to control client associations [50–52]. The 802.11r amendment [53] specifies “fast basic service set (BSS) transition”, wherein a client that is to handover between two APs can perform a re-association faster. It is done by pre-caching intermediary results of the key negotiation process between the original AP where the client first authenticated, and the authentication server at different parts of the wireless network. The IEEE 802.11k standard [54] enables better management of WLANs by introducing a signalling technique into the protocol for APs to assist wireless clients in selecting the target AP for re-association, as opposed to having clients blindly choose the new AP to handover based on the signal strength, which can potentially lead to traffic overload of APs. The authors in [55] investigate a simple, proactive roaming method using the IEEE 802.11k standard to collect information about the wireless environment prior to roaming. Actual handover is then based on an assessment of the signal strength.

Handover based on signal strength is a common strategy and is easy to be implemented. In [56], the authors study how interference among both APs and STAs affects throughput and propose a decentralised AP selection scheme that avoids interference. In addition, the authors implemented the proposed architecture on actual devices and added a function for avoiding interference. The authors in [57] propose a handover management scheme adaptable to multi-rate WLANs to enhance the practicality of handover scheme in a realistic environment where WLAN supports multiple data rates and automatically changes the rate in response to wireless link condition. The proposed scheme exploits two sorts of information to recognise the wireless condition appropriately that are data rate used most frequently (DRMF) by each interface and frame re-transmission ratio (FRR) on each interface for some duration. The former of two criteria first enables the estimation of the area where the handover should start, and if DRMFs of two interfaces are same in the area, the latter then achieves a precise comparison of wireless condition on two interfaces, thereby giving an optimal handover point. However, both schemes need client-side modification which is not suitable for real-world implementation.

The authors in [58] proposes an adaptive network selection scheme, called as DMMT (Dynamic Match of Mobility Threshold) scheme. By considering the distribution of the whole users'movement character and arrival rate, the Mobility Threshold (MT) can be achieved from the solution of optimisation problem, which minimise the global network blocking probability. By using the MT, mobile users can be differentiated to access different network. To obtain the parameters in the optimisation problem, the authors use the prediction of arrival and mobility characteristics. The heterogeneous networks adjust the MT to adapt to the variation of environment. Based on the simulation results, it shows that the proposed DMMT scheme reduced call blocking and handoff probabilities compared to the referenced schemes.

Rousseau *et al.* in [59] introduce the concept of the virtual access point (VAP) to manage client mobility in infrastructure networks. With the proposed VAP scheme, every client is connected to its own VAP, and when they need to handover between physical APs, the VAPs are migrated along with them to enable seamless client mobility on the same wireless channel. The authors use a packet manipulation framework called *PacMap* to implement the VAP, which is compatible with existing clients without any hardware nor software modification. Based on the concept of VAP, Rousseau *et al.* further propose a mobility management solution called Multichannel Virtual APs (mVAP) in [60]. mVAP uses custom Inter-AP messages that contain CSA elements to achieve seamless client handovers in networks with APs operating on multiple channels. However, mVAP require each AP to have a list of its neighbour APs to send relevant Inter-AP messages for user handover, which adds extra effort to deploy the solution. Moreover, it does not consider AP traffic load when making handover decisions.

In recent years, SDN has gained increasing popularity as a novel network architecture. Some research works have been focusing on utilising SDN to improve the performance of mobility management in enterprise WLANs. The authors in [61] present an integrated energy and mobility management solution, which combines Energino [62] with Odin [27] for enterprise WLANs. With the proposed solution, clients joining the network are handed over by the Mobility Manager to the AP that provides the best performance in terms of

Signal–to–noise ratio (SNR). In [63], the authors merge the work of [60] into [37] to develop a WLAN centralised mobility management solution which is able to provide seamless handovers between APs in different channels and meeting the QoS requirements of real-time services. The proposed solution uses a low-frequency rate when the client remains in the same AP and a high-frequency beacon rate to obtain a fast and seamless handoff. However, it does not consider AP traffic load when making handover decisions which can cause unbalanced traffic load on APs.

The authors in [64] propose an RSSI-based fast handoff scheme for SDN based WLAN system. With the proposed solution, clients start to perform a fast pre-scan process when RSSI falls below the Scan Threshold (ST), and when RSSI falls below the Handoff Threshold (HT), clients execute handoff process. By utilising SDN controller, clients can obtain the best alternative AP after the fast pre-scan process. When handoff process is triggered, clients can perform the authentication and re-association process toward the alternative AP directly that reduces the handoff delay. In addition, the authors adopt Kalman filter to minimise the fluctuation of RSSI, which may lead to unnecessary pre-scan process and serious ping-pong effect. However, this solution only uses RSSI as the handoff metric which is inefficient in enterprise WLANs. In [65], the authors propose a seamless, client-unaware handover mechanism on a software-defined enterprise WLAN framework which is based on the Split-MAC design of CAPWAP [66]. The proposed solution requires APs to report the RSSI value of the user, and if the value at the current AP is less than a threshold value and is also less than the RSSI value at any other AP, the controller will initiate the handover process. However, the solution does not take AP traffic load into consideration for handover decision and is only evaluated by simulations which makes its practicability questionable.

In [67], the authors achieve lossless handover between WLAN and WiMAX nodes on the OpenRoads platform [18] [19] by redirecting the flows dynamically through the network. It shows that OpenRoads has the capability to provide very efficient mobility control and the OpenRoads deployment can be used to support mobility management in OpenFlow Networks. However, OpenRoads does not involve multi-hop wireless communications and integration with existing routing protocols. The authors in [68] propose an intelligent mobility enhancement control and then develop an algorithm to decide which neighbour switches need to be selected for the installation of new flow entries and to allocate the appropriate idle-timeout for the selected switches. The proposed approach provides a simple solution to solve the user mobility problem in wireless OpenFlow environments which can handle the fast migration of user addresses (e.g. IP addresses) between several wireless access points and base stations which leads to improvement in the end users'experience. However, the algorithm is not implemented, and no experiment is run to test the performance of the purposed algorithm.

Benefiting from the extended OpenFlow protocol and the unified control platform, authors [26] proposed the abstraction of "One Big AP" illusion with a client-unaware fast AP handover mechanism in SDWLAN. Simulation results demonstrated that AP handover operation in SDWLAN leads to negligible throughput fluctuation of on-going connection compared to traditional architecture with 802.11 standard handover mechanism. However, the results are based on simulations which make the practicability of this architecture and the handover application on actual physical devices questionable. In [69], the authors introduced a load-aware handoff algorithm for SDN-based WLAN, which considers traffic load of APs in addition to received signal strength at wireless clients to address load imbalance issues among APs and achieve seamless mobility for wireless users. By using the signal strength and traffic load threshold, the proposed algorithm improves the mobile users'throughput as well as static clients compared to legacy handoff algorithms used in WLANs. However, the proposed algorithm uses fixed hysteresis margin of Traffic Intensity, which is not efficient under the certain channel load conditions. Our work uses dynamic hysteresis margin of

AP load and the RSSI value to achieve efficient mobility management in software-defined enterprise WLANs.

Dely *et al.* in [70] proposed an SDN-based WLANs framework for handoff management with real-time video streaming. The system allows clients to associate with multiple APs simultaneously and to perform fast handovers between them, which improves the quality of real-time video streaming over WLANs significantly. Moreover, Dely *et al.* introduce a new mechanism for selecting the best AP based on a novel available bandwidth estimation scheme. The available bandwidth provided by an AP depends mainly on the signal quality and the load on the wireless channel. Based on measurements, the authors first analyse how those factors vary stochastically over time and motivate why a frequent estimation of available bandwidth is necessary. The authors then develop BEST-AP [71], a system for bandwidth estimation of APs, which uses regular data traffic to estimate the available bandwidth from all APs in reach in a non-intrusive way, even if they are on a different channel. Based on OpenFlow, BEST-AP allows the station to be associated with multiple APs simultaneously and to switch between APs with low overhead. By using the available bandwidth estimates, the system exploits the best AP for a longer duration while probing the less good APs for a shorter duration to update the bandwidth estimations.

The authors in [27] [28] propose a simple mobility manager for software-defined enterprise WLANs. The proposed mobility manager utilise LVAP migration to achieve seamless user mobility. However, it can only perform seamless handovers when APs are operated on the same wireless channel. In addition, it only uses fixed hysteresis margin of signal strength to make handover decisions, which is inefficient as APs with the best signal strength may not guarantee the satisfied QoS and user QoE. Our solution supports seamless handover over multiple wireless channels and uses dynamic hysteresis margin of AP load and signal strength to achieve efficient mobility management in software-defined enterprise WLANs.

2.3.2 Load Balancing

Load balancing is an important service in modern enterprise WLANs as, without load balancing, wireless users scan all available APs and independently selects the AP with the best signal strength to associate with by default, without considering the traffic load condition of the AP. This mechanism often causes unbalanced traffic load on APs, leading to issues such as high packet loss rate and low network throughput which can significantly degrade QoS and user QoE. Over the past decade, several solutions have been developed by researchers to solve the unbalanced traffic load problem. To force a client to re-associate to another AP, the authors in [72] present a new load balancing technique by controlling the size of WLAN cells (i.e., AP's coverage range), which is conceptually similar to cell breathing in cellular networks. The proposed scheme does not require any modification to the users neither the IEEE 802.11 standard. It only requires the ability of dynamically changing the transmission power of the AP beacon messages. The authors develop a set of polynomial time algorithms that find the optimal beacon power settings which minimise the load of the most congested AP. However, cell breathing would affect the signal-to-noise (SNR) ratio of all the stations that remain associated with the AP that has just decreased its transmission power and does not solve the handover delay issue for the client. There are also standardisation efforts that are relevant in the direction of re-association based load balancing. The 802.11k amendment [54] deals specifically with load balancing in WLANs. It requires changes in the client devices as well. The amendment defines a mechanism by which APs can provide the client with information such that the latter can make better association decisions.

In [73], authors proposed a new AP selection strategy for high-density IEEE 802.11 WLANs. Instead of using the received signal strength, the new strategy calculates the effective throughput for each neighbouring AP and use the result as a basis for the AP selection. The selection strategy solves the relevant problem of which AP can be selected and associated with a station so that efficient resource utilisation can be obtained. The authors in [74] focus on the presentation of a QoS management solution for wireless communication systems. It mainly defends that a balanced distribution of mobile stations among the available access points leads to better performances of the Wireless LAN. With the proposed approach, better resources allocation and efficiency on QoS metrics are achieved. Moreover, a protocol structure between mobiles and APs is also specified for the implementation of this approach. In [75] [76], the authors present an efficient solution to determine the user-AP associations for max-min fair bandwidth allocation. With proposed algorithms, the authors first compute a fractional association solution, in which users can be associated with multiple APs simultaneously. This solution guarantees the fairest bandwidth allocation in terms of max-min fairness. Then, by utilising a rounding method, the authors obtain the integral solution from the fractional solution. In addition, the authors also consider time fairness and present a polynomial-time algorithm for an optimal integral solution.

The authors in [77] propose a load-balancing scheme for overlapping wireless LAN cells. With the proposed solution, agents on each AP periodically broadcast the local load level via the Ethernet backbone and determine whether the access point is overloaded, balanced or under-loaded by comparing it with the received reports. The load metric is the network throughput. Overloaded access points force the handover of some stations to balance the load. Only the under-loaded access points accept the roaming stations in minimising the number of handovers. In [78], authors propose a load balancing procedure, which acts on two separate levels. First, the AP is either distributed across the channels (if the number of AP is small) or located in the same channel that is specified by taking into account the channel-location and the received RSSI of the neighbour AP. Second, the wireless station (WS) are distributed across all the available AP based not only on RSSI measurements but also on the number of WS already associated with the AP and other link quality measurements. With proposed algorithm, the network performance is improved.

Unlike the most load balancing solutions in WLANs that focuses on the optimisation of AP operations, Xue *et al.* in [79] propose a user sociality-aware AP selection scheme to tackle the load balancing problem in enterprise WLANs. Through the analysis of AP usage based on a real WLAN trace, the authors found that users'social activities would disruptively affect the load balance among APs in enterprise environments. By taking into account the social relationships of users, the proposed scheme assigns users with intense social relationships to different APs so that jointly departure of those users would have a minor impact on the load balance of APs. In addition, the authors clarify that the problem of allocating an AP for each user so that the average of the sums of social relation intensity between any pair of users in each AP is NP-complete and propose an online greedy algorithm to tackle the problem. Extensive trace-driven simulations show that the proposed method can achieve about 64.7 percent balancing performance gain on average compared to the traditional AP selection algorithms. In [80], Xue *et al.* further study the social behaviour of users through analysis of intensive WLANs data traces and find out that users with similar application usage have the potential to leave together. Inspired by those observations, the authors propose a user-friendly AP load balancing scheme for enterprise WLANs. The proposed solution characterises the sociality of users by grouping users with similar application usage profiles and elegantly assign users to different APs based on their history sociality information and application profiles. Moreover, the authors clarify that the proposed scheme does not migrate users from one AP to another and is very resilient to network churns as it can resist sudden traffic demand changes caused by users'co-leaving behaviour. The authors in [81] further extend the work and present an online heuristic algorithm, which can actively learn users'sociality information and assign users with certain social relations to different APs so that users in one AP are more likely to leave at the different time.

With the growing popularity of SDN, researchers have been studied the emergence of SDN in enterprise WLANs which provides a promising solution for improving the performance of load balancing service. In [28], the authors implemented a load balancing application on the Odin [27] framework. The proposed application queries the system once per minute to obtain the list of clients that can be seen by different APs and their corresponding RSSI values. It uses this information to build a map of clients to lists of agents that are candidates for hosting their respective virtual APs. The application then evenly re-distributes virtual APs (clients) across physical APs with round robin algorithm. However, it does not consider the traffic load condition of each AP which can cause unnecessary migration of clients. Moreover, it does not support seamless handover over multiple wireless channels which can disrupt user association during the process of reassigning users to new APs.

The authors in [82] propose a novel load balancing mechanism for WLAN, which is implemented on a client-network collaborative architecture based on SDN and 802.11u [83]. In the presented collaborative architecture, SDN applications monitor load and traffic of each infrastructure. Network Resource Query Protocol (NRQP), based on Generic Advertisement Service (GAS) of 802.11u, enables clients and networks to exchange information of AP selection. The network will calculate the load of APs by the channel utilisation and clients'link speed, and the AP with the lowest load is selected by the load-aware load balancing algorithm as the handover target. However, this solution does not support seamless handover when making the association adjustment. As a result, user experience can be affected significantly because of the service disruption and re-association delay.

In [84], the authors propose an SDN-based adaptive load balancing scheme for Wi-Fi APs through association control. With the help of SDN, the proposed method uses the SDN controller to collect the load of APs and user connection situations. In the case that the load of APs is detected as imbalanced, the SDN controller adaptively configures the beacon power of relevant APs by sending the Beacon-Config messages. Based on the power adjustment, some users are forced to change association from low-power APs to high-power APs. However, the proposed solution is evaluated through simulations which raise questions of its feasibility and practicability.

The authors in [85] introduce Wi-Balance, an SDN-based load balancing mechanism for enterprise WLANs. Wi-Balance is implemented as a network application on the open source 5G–EmPOWER platform [86]. The proposed application gathers the information related to the ongoing transmissions and the channel status of the clients in the network by using the Minstrel rate control algorithm [87] which is presently available in the Linux kernel. By analysing the collected information, the proposed application first estimates the number of concurrent transmissions in the same AP. In this case, it evaluates if the handover of a client connected to one of these affected APs to another available one may isolate the flows and improve the performance. To prevent the flows sharing the same collision domain, the proposed application performs a channel switch of the involved APs after the reassignment of users. However, no experiment is conducted to evaluate the performance of the proposed solution.

In [88], the authors propose an SDN-based AP selection scheme called TPR (Throughput, Packet_drop and RSSI based AP selection) for IEEE 802.11 network. TPR uses the throughput, packet loss rate and RSSI as the metrics for AP selection. All measured metrics are sent to the SDN controller to make centralised load balancing, and the AP with highest packet loss rate will be adjusted first to migrate loads. Moreover, the proposed method endeavours to first migrate the high-throughput client to other AP as it makes more contributions to the network load. However, the proposed method does not support seamless handover over multiple channels which can cause service disruption during the migration of clients.

2.3.3 Automatic Channel Switching

With the increasing deployment of APs, channel interference in the 2.4GHz frequency band remains a key issue in modern enterprise WLANs. In order to minimise the interference problem and guarantee satisfied QoS and user QoE, optimised channel allocation mechanism is required in enterprise networks. Previous research works have proposed several channel switching schemes. For instance, the authors in [89] mentioned one approach called least congested channel switch. With this approach, each AP scans beacon frames that are sent by neighbouring APs to get the number of APs and the total number of clients associated with each AP. It then switches to the channel with least number of clients. However, the channel with least number of clients may not always be the least congested channel as the channel utilisation must be considered as well. In [90], the authors propose an approach of optimising AP placement and channel assignment in WLAN by formulating an optimal integer linear programming (ILP) problem. The optimisation objective is to minimise the maximum of channel utilisation, which qualitatively represents congestion at the hot spot in WLAN service areas.

The authors in [91] propose two fully distributed algorithms that allow multiple interfering 802.11 APs to select their operating frequency to minimize interference, and user devices to choose the AP they attach to, in order to get their fair share of the whole network bandwidth. The proposed algorithms rely on Gibbs sampler and do not require explicit coordination among the wireless devices. They only require the participating wireless nodes to measure local quantities such as interference and transmission delay. In [92], the authors argue that network throughput alone, as obtained by simulations or direct measurement, is not an adequate measure for evaluating a frequency plan, so that any direct optimisation of network throughput would indeed not be advisable as it is especially inadequate for fairness. In addition, the authors propose an efficient frequency planning algorithm to mitigate unfairness service to users based on a novel framework which models the load of WLAN cells considering inter-cell interference.

Yu *et al.* in [93] propose a new framework for radio channel allocation (RCA) strategy for WLAN APs with power control capabilities. The authors formulate the RCA as a min-max optimisation problem regarding channel utilisation with constraints of data rate, channel quality, and transmission power. The authors also derive an expression to evaluate the channel utilisation, which incorporates the condition of wireless channels, such as SNR ratio and transmission power, in addition to the transmitting probability of a station. Moreover, a new RCA algorithm that considers both link adaptation and power control mechanisms, in addition to the impact of co-channel interference (CCI) and multiple data rates is proposed. Based on the same framework, Yu *et al.* in [94] further propose a new algorithm to dynamically estimate the number of active stations by using the least square estimator (LSE), which is unbiased and has minimum variance. The authors also derive an expression to evaluate the impact of the co-channel stations of an AP on the channel utilisation of the AP based on the number of equivalent co-channel stations, which is the difference between the numbers of stations that are sensed by and associated with the AP. Finally, the authors develop a new distributed RCA (DRCA) that considers the changing number of active stations, the impact of co-channel interference (CCI), and different traffic demands for different APs.

Mishra *et al.* [95] propose an efficient client-based approach for channel management (channel assignment and load balancing) in 802.11-based WLANs that lead to better usage of the wireless spectrum. The proposed approach is based on a "conflict set colouring" formulation that jointly performs load balancing along with channel assignment. The proposed formulation explicitly captures interference effects on clients and intrinsically exposes opportunities for better channel re-use. Moreover, the authors clarify that algorithms based on this formulation do not depend on specific physical RF models and hence can be applied efficiently to a wide-range of inbuilding as well as outdoor scenarios. Packet-level simulations and physical experiments show that in addition to single network scenarios, the conflict set colouring formulation is well suited for channel assignment where multiple wireless networks share and contend for spectrum in the same physical space. In [96] Mishra *et al.* also propose a distributed channel management mechanism. The proposed solution is distributed in nature, minimal to zero coordination among APs belonging to different hotspots, simple to implement, and interoperable with existing standards. In particular, a specific algorithm called MAXchop is introduced, which works efficiently when using only non-overlapping wireless channels, but is particularly effective in exploiting partially-overlapped channels. With MAXchop, each AP is assigned to a sequence of channels, and it hops through this sequence to average the throughput of all APs

The authors in [97] present a study of the impact of interference in chaotic 802.11 deployments on end-client performance. By using large-scale measurement data from several cities, the authors show that it is not uncommon to have tens of APs deployed in close proximity of each other and most APs are not configured to minimise interference with their neighbours. To making chaotic wireless networks self-managing, the authors design and evaluate automated power control and rate adaptation algorithms to minimise interference among neighbouring APs, while ensuring robust end-client performance. In [98], the authors use graph colouring algorithm as a theoretical basis for a protocol to effectively assign channels to WLAN APs. The authors in [99] introduced a channel allocation algorithm, which utilises all the channels in the crowded Wi-Fi spectrum in a more efficient way that each frequency has a channel separation of 5.

In [100], the authors propose a traffic-aware channel assignment which used traffic-awareness, including measuring an interference graph, handling non-binary interference, collecting traffic demands, and predicting future demands based on historical information to address key practical issues. The authors in [101] propose a channel assignment algorithm to maximise Signal-to-Interference Ratio (SIR) at the user level. The proposed algorithm starts with the channel assignment at the APs, which is based on minimising the total interference between APs. Based on this initial assignment, the SIR for each user is calculated. The authors clarify that the algorithm can be applied to any WLAN, irrespective of the user distribution and user load.

By utilising the flexibility and programmability of SDN, software-defined enterprise WLANs provides a promising solution for researchers to solve the automatic channel switching problem. In [39], the authors propose an SDN-based interference graph and traffic-oriented channel assignment (GTOA) algorithm. The proposed algorithm sorts the node in the interference graph by its suffering degree which is computed by its out-degree and the predicted traffic to be transmitted by it. The main idea of GTOA algorithm is to find key nodes in the graph and assign channels to them with high priority. However, the proposed solution does not support seamless channel switching over multiple channels which can inevitably degrade user QoE. In addition, the proposed solution is evaluated based on simulations that makes its feasibility and practicability questionable.

The authors in [102] presented high-level abstractions for channel quality and interference with an interference-aware channel assignment approach. In the proposed system, the RSSI value measured at each Wireless Termination Points (WTPs) is used as an approximation of the channel quality to form the *Channel Quality Map*. An *Interference Map* is also constructed to monitor the interference level between two communication links. The authors implement a simple DSATUR [103] algorithm to proof that the *Channel Quality Map* and *Interference Map* abstractions can be used to improve the performance of channel assignment. However, the proposed system inherits the LVAP function from Odin system [27] [28] where seamless channel switching is only supported over the same wireless channel.

In [28], the authors further implement varieties of network services on top of Odin framework [27], and automatic channel selection is one of them. The implemented automatic channel selection application scans across all available channels and computes the average and the max RSSI for each channel centre frequency. Based on multiple subsequent spectral scans, it picks the channel with the smallest maximum and average RSSI. However, AP traffic load is not considered when making channel selection decisions. Furthermore, seamless channel switching over multiple channels is not supported with the proposed channel selection application.

In [104], the authors propose an SDN-based channel optimisation scheme for WLAN environments. In the proposed scheme, Open Network Operating System (ONOS) [105] is used as the SDN controller. Clients gather the channel information and transmission power of APs and send to the SDN controller to make channel allocation decisions. In the case that channel re-allocation is needed, the SDN controller changes the channel of the AP which is causing most interference in the network and reduce the transmission power of the APs using the same channel. However, the proposed scheme does not support seamless channel switching which can significantly affect user QoE. Moreover, no channel switching algorithm is provided with the proposed scheme.

The authors in [106] propose a channel assignment algorithm that addresses spectrum congestion in Wi-Fi networks using SDN. The APs'network arrangement, the current assignment of the channels and the characteristics of the RF channels in IEEE 802.11 have all been taken into account in the proposed model for calculating the optimised channel assignment configuration. However, the evaluation of the proposed algorithm is based on simulations that raises questions of its feasibility and practicability. In [107], the authors present a VAP-based channel switch mechanism. In the proposed system, the operating channel of a Dedicated and Dynamic VAP (DD-VAP) can be assigned by the server when the DD-VAP is created initially, or the DD-VAP is operating. When a DD-VAP is operating, channel switch can be done through the reconfiguration command sent by the server. When the DD-VAP is created initially, the server executes a channel assignment algorithm that considers both the operating channel and operating timing of neighbour DD-VAPs to determine the DD-VAP's operating channel. CSA in the beacon frame is used to notify the user of the channel switching. However, no experiments are conducted to evaluate the performance of the proposed scheme which makes its feasibility questionable.

2.3.4 Bandwidth Control

With the concept of BYOD plays a more important role in modern enterprise WLANs, bandwidth control service becomes a key enabler for achieving efficient network traffic control. Leveraging the high-level programmability introduced by SDN, software-defined enterprise WLAN becomes a promising architecture to improve the performance of bandwidth control. In [108], the authors propose an SDN-based TCP throughput management algorithm that provides a better allocation of the available bandwidth to heterogeneous adaptive video streaming applications. The proposed scheme uses an SDN traffic monitoring module to estimate the throughput required by the clients. After obtaining the estimated throughput, the allocated bandwidth for each user is determined by sending a message to the content provider to allocate the bandwidth.

The authors in [109] propose an SDN-based solution to control traffic flows in WLANs. Based on the value of the Differentiated Services Code Point (DSCP) field, in the IP headers of the packets, the proposed solution classify user traffic and assign traffic flow to the queue with priority at the OpenFlow switch port to ensure QoS. However, packet classification is not suitable for enterprise WLANs where user priority should be considered. In [110], the authors extend current OpenFlow infrastructure to support flexible traffic prioritisation of Wi-Fi MAC layer frames and several queue management strategies such as HTB, SFQ, or FQ_CoDel on Open vSwitch enabled routers and switches to support NFV.

In [111], the authors introduce an SDN-based bandwidth allocation approach called FlowQoS. FlowQoS has two main modules which are the flow classifier and the SDN-based rate limiter. The flow classifier module identifies application types for each flow in real time. Based on the flow classifier module's determination of the associations between flows and applications, the SDN-based rate limiter module assigns each flow to the appropriate rate. However, flow classification is performed within an SDN controller that runs on small devices severely limiting performance. Moreover, application-based bandwidth allocation method is not suitable for enterprise WLAN environments where user priority is the most important factor.

The authors in [112] propose a service-based bandwidth allocation mechanism based on the concept of SDN. With the proposed solution, each service packet is marked with a unique type of service (ToS) value when it is pushed out from the application server. The rate limiter module limits the rate according to the assigned ToS value to the packet. However, service-based bandwidth allocation scheme is inefficient in enterprise WLANs as user priority should be considered firstly.

In [113], the authors present an SDN-based QoS management system with the feature of providing flexible QoS bandwidth allocation in manual with hosts and protocols control. The proposed system works on Ryu SDN controller [114] and utilises a machine learning-based flow classification platform to achieve flow-based QoS management for applications in different slicing networks.

The authors in [115] present an SDN-based bandwidth slicing technique which enables the network administrator to allocate the different amount of bandwidth to users in different groups. The proposed technique utilises the queuing function in the OpenWrt kernel and uses the OpenFlow protocol to forward packets of different user groups to corresponding queues. Because the queuing rules can be adjusted by network administrators, dynamical bandwidth allocation is achieved. In addition, users can be re-assigned to different groups as network administrators can modify the forwarding rules in real time by simply changing the OpenFlow flow entries. However, the proposed solution still requires manual configurations to direct user traffic flows that is inefficient. Moreover, it is not adapted to real-time traffic as AP load is not considered.

Chapter 3

Adaptive Mobility Management and Load Balancing

3.1 Introduction

Over the past decades, IEEE 802.11-based WLANs have become the global trend in wireless communication technologies because of their characteristics, such as wide coverage and high data throughput. Companies have deployed WLANs as wired network extensions to provide mobility support, which enables clients to access network resources and roam freely anywhere within the coverage. With the fast increasing amount of user devices, modern enterprise WLANs require network services such as mobility management and load-balancing to provide efficient network traffic control, which guarantees QoS and user QoE. In order to manage this growing complexity, a novel architecture for enterprise WLANs is needed.

SDN is a novel network paradigm where network control plane is separated from the forwarding plane and is directly programmable. This separation enables network intelligence and state to be logically centralised and the underlying infrastructure to be highly abstracted for network services. With the emergence of OpenFlow enabled SDN technology, the existing inflexible enterprise WLANs infrastructure can be greatly simplified, and network operators are able to achieve fine-grained network traffic control by implementing network services such as mobility management and load balancing.

3.1.1 Problem Statement

Mobility management is a key network service in modern enterprise WLANs. With the trend of fast adaption to implement real-time multimedia services, for instance, video streaming over enterprise WLANs, seamless handover has become one of the crucial criteria to guarantee QoS and user QoE.The conventional handover algorithm only use the fixed RSSI value as the hysteresis margin when making handover decisions. This mechanism is not able to achieve seamless handover over multiple wireless channels and often leads to unbalanced AP traffic load, which significantly degrades network performance.

Load balancing is another necessary network service for modern enterprise WLANs. Although the AP load balancing problem has been studied for years, the existing solutions continue to exhibit inefficiencies. Because without knowing future traffic demands on individual APs, it is very hard to make an optimal assignment or adjustment of WLAN users among a set of APs. Moreover, it can cause link disruptions when dynamically migrating users from heavy-load APs to light-load ones. In the architecture of enterprise WLANs, wireless users scan all available APs and independently select the AP with the best signal strength to associate with by default, without considering the traffic load condition of the AP. This mechanism often causes unbalanced traffic load on APs, leading to high packet loss rate and low network throughput which can significantly degrade QoS and user QoE. The conventional load balancing algorithm based on RSSI continue to exhibit inefficiencies as they are not able to provide seamless user mobility and often cause unbalanced user association on APs, which heavily affects QoS and user QoE. To solve those WLAN-specific mobility management and load balancing problems, commercial vendors offer their solutions

[11–13] through vendor proprietary software and hardware. Unfortunately, these solutions do not have open programmable interface and are unable to be achieved with the low-cost commodity AP hardware that is used by provider networks.

3.1.2 Contributions

With conventional mobility management and load balancing mechanisms, handover decisions based on the RSSI value ensure that the client is always re-associated to the AP with best signal strength. However, these mechanisms do not guarantee the best QoS and user QoE as the AP with best signal strength can be heavily loaded with too many associated users. Therefore, it is vital to consider the AP traffic load when making client handover decisions. In this chapter, the author presents two adaptive algorithms for mobility management and load balancing in software-defined enterprise WLANs that mitigate the unbalanced AP traffic load and support seamless handover over multiple wireless channels. By introducing AP load in addition to the RSSI value, the proposed algorithms running on top of the SDN controller to achieve centralised mobility management and load balancing, which adapt to real-time AP traffic load. The author demonstrates the efficiency of the proposed adaptive algorithms by evaluating their performance through iperf-based TCP and UDP tests.

3.1.3 Structure of Chapter

The remainder of this chapter is organised as follows. Section II further describes the architecture of the selected software-defined enterprise WLANs and the major extension work. Section III presents the AP load metric and the proposed adaptive algorithms for mobility management and load balancing respectively. Section IV demonstrates the layout of experiment testbed and related performance evaluations, and finally, the summary of this chapter is presented in Section V.

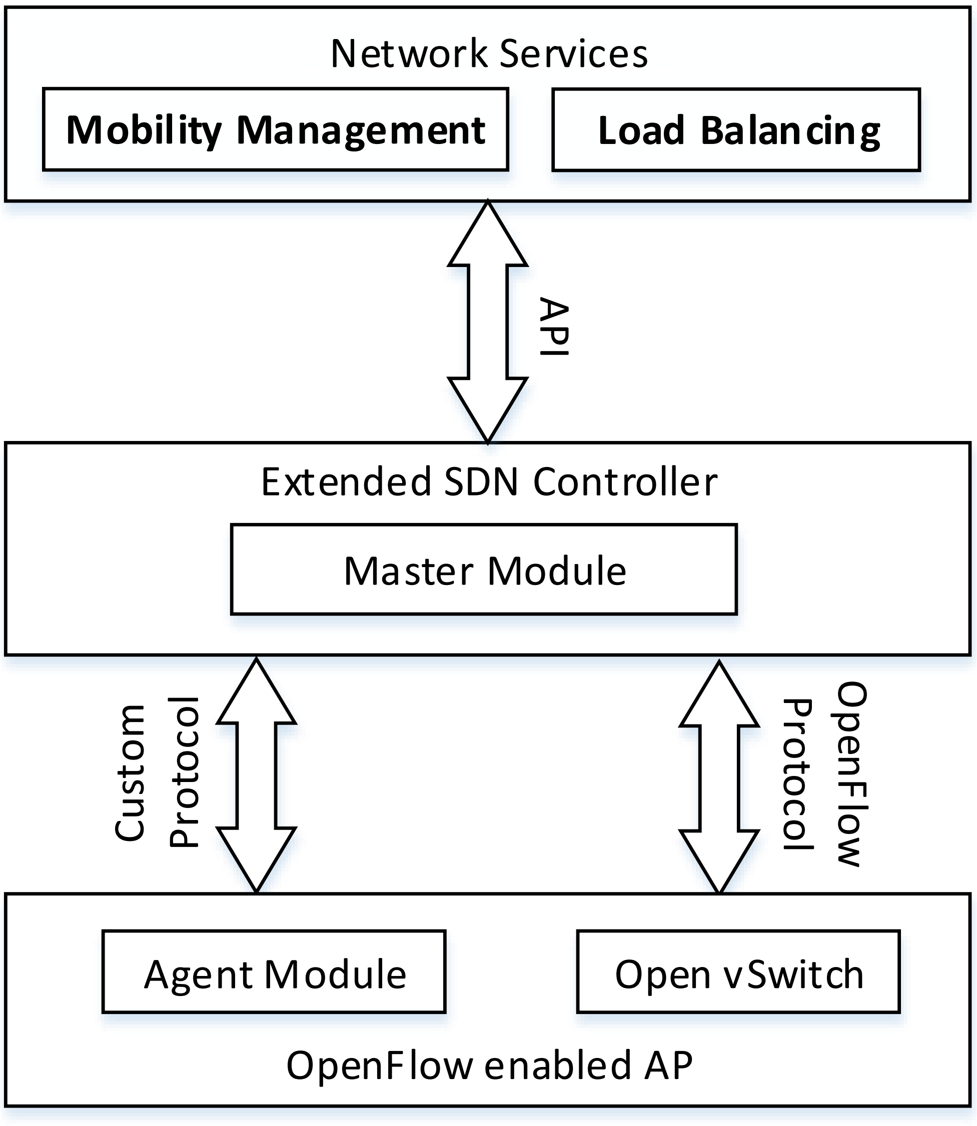


Fig. 3.1 Architecture of the selected software-defined enterprise WLAN system

3.2 The Selected System Architecture

The system that is presented in [27] [28] is selected as the base of the proposed adaptive mobility management and load balancing algorithms because of its light-weight and open source nature.

Fig.3.1 shows the architecture of the selected software-defined enterprise WLAN. The three-layer architecture separates the control plane from the data plane by having a logically centralised Floodlight [116] SDN controller, which uses OpenFlow protocol and a set of custom protocol messages to manage the wired and wireless network respectively. The proposed adaptive mobility management and load balancing algorithms are implemented as network services that run on top of the central controller. The central controller is extended with an add-on master module to have a global view of the network information, which includes data flows, associated APs and clients' association states. All the APs in the architecture are integrated with software OpenFlow switch Open vSwitch [117] [118] and the Click Modular Router [119]. The former one turns the commodity AP into the OpenFlow enabled AP that support the OpenFlow protocol. The latter one works on the AP as the agent module, which significantly simplifies client association management by creating a virtual AP (VAP) for each associated client. Each VAP is identified by a unique BSSID, which is calculated based on the client's MAC address.

With the selected software-defined enterprise WLAN system, mobility management and load balancing services are only based on the RSSI value which is not enough to guarantee satisfied QoS and user QoE. To achieve the proposed adaptive mobility management and load balancing services, the author further extend the selected software-defined enterprise WLAN system. Major extension works are as following:

- To achieve the proposed AP load metric, the ath9k based Wi-Fi driver is modified to export the content of related registers by using debugfs based file system.

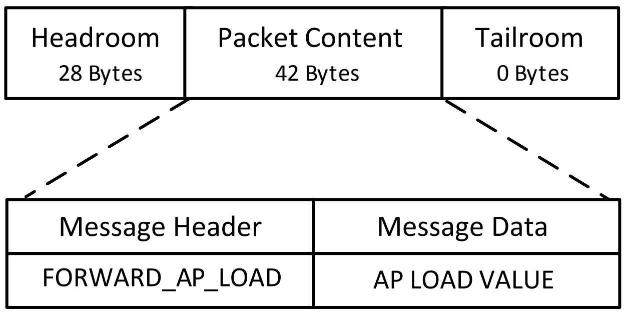


Fig. 3.2 Format of the custom protocol message FORWARD_AP_LOAD

- To make the central controller aware of updated traffic load of each AP, a new custom protocol message called FORWARD_AP_LOAD is introduced.
- To achieve seamless client handover over multiple wireless channels, the CSA element is added into the modified beacon frame.
- To improve network traffic control over software-defined enterprise WLANs, two adaptive algorithms for mobility management and load balancing respectively are proposed and implemented on top of the central controller.

In order to achieve the proposed adaptive algorithms, the central controller requires having the knowledge of the traffic load of each connected AP. Therefore, a new custom protocol message called FORWARD_AP_LOAD is introduced to the system. With the help of this new protocol message, the traffic load of each connected AP that is calculated in the agent module is encapsulated into a data packet with a proper message header before it is finally sent to the central controller via a separate control channel. The format of this custom protocol message is illustrated in Fig.3.2. As it can be seen, the headroom size is 28 bytes by default which contains extra information about the packet like the destination address to be used for routing. The message header and the AP load value are stored as a message string in the packet content with a total size of 30 bytes. The tailroom is configured to be 0 bytes as it is not used in this case.

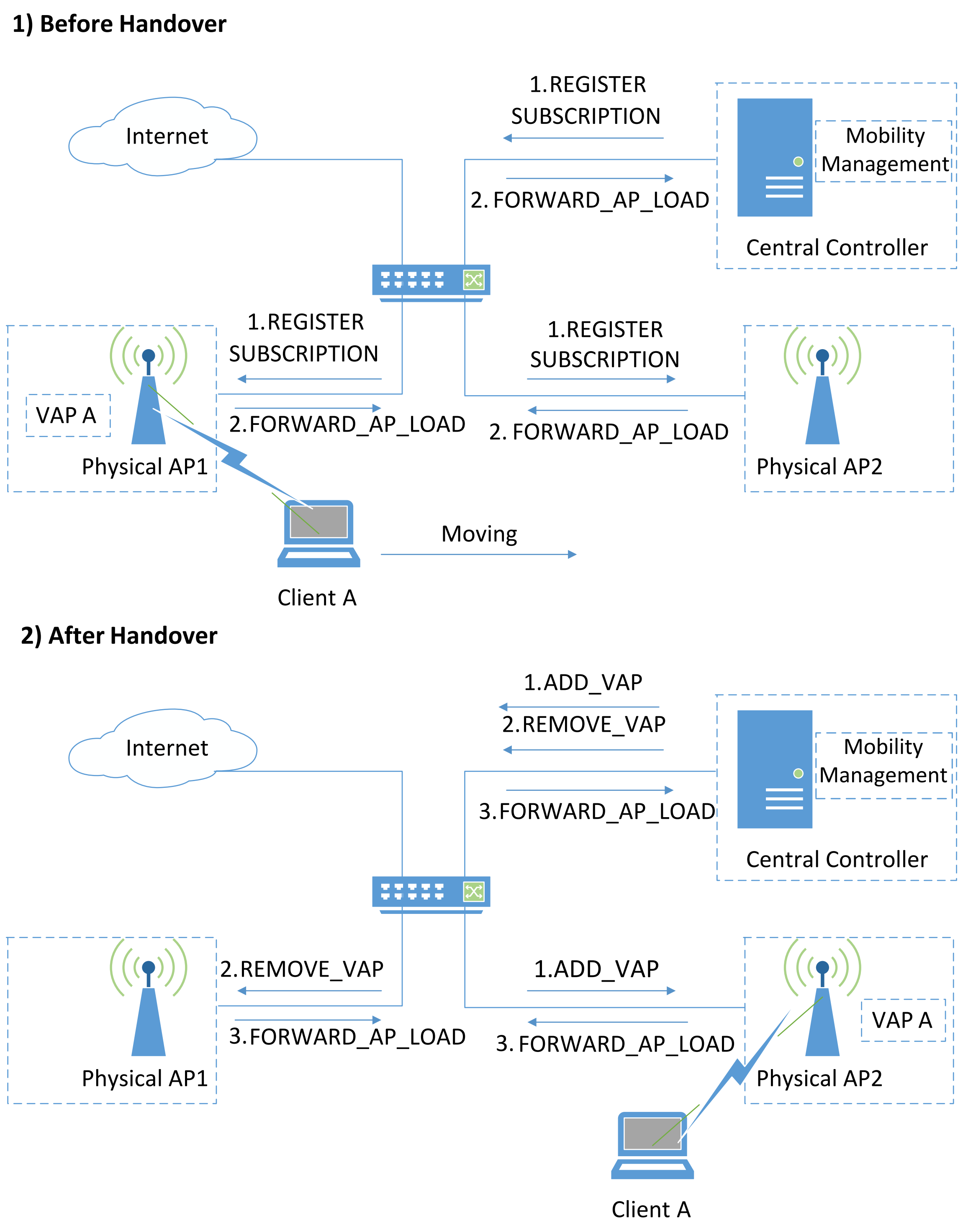


Fig. 3.3 An example of custom protocol messages that are exchanged during the process of seamless handover in the selected software-defined enterprise WLAN system

The central controller stores and updates the AP load value within a hash map matched by the IP address of each connected physical AP. The proposed adaptive mobility management algorithm is implemented reactively based on an event-focused publish-subscribe solution. To enable the mobility management, the proposed algorithm first registers a subscription of the event that the RSSI value of the client from all physical APs is greater than a pre-defined threshold value. If the subscribed event is triggered at one of the physical APs, the PUBLISH message is sent to the central controller with relevant context information including the client's MAC address, the IP address of the physical AP where the event is triggered, and the corresponding up-to-date RSSI value that matches the subscription. All the information is then passed to the adaptive mobility management service as a handover metric together with the AP load value to make handover decisions. Fig.3.3 shows the detailed handover process and relevant custom protocol messages that are transmitted in the selected software-defined enterprise WLAN system. Before the handover, all the AP keep the central controller aware of their AP load value by sending the FORWARD_AP_LOAD custom protocol message periodically. Once the adaptive algorithm performs handover, the central controller sends the ADD_VAP message to the handover target AP and the REMOVE_VAP message to the client's current associated AP respectively. By migrating client's VAP, seamless handover on the same wireless channel is achieved. In order to support seamless handover over multiple wireless channels, the CSA element is added into the modified beacon frame. The CSA element contains three fields which are *channel switch mode*, *new channel number*, and *channel switch count*. The first field is either 0 or 1 which indicates any restrictions on transmission until a channel switch happens. The second field stores the new channel number while the last field contains the value of remaining beacons that will be sent before the channel switch takes place. In our testbed, whenever the client is triggered to handover to an AP on a different wireless channel, the modified beacon frame with CSA element is sent from the client's current associated AP before the client's VAP is removed.

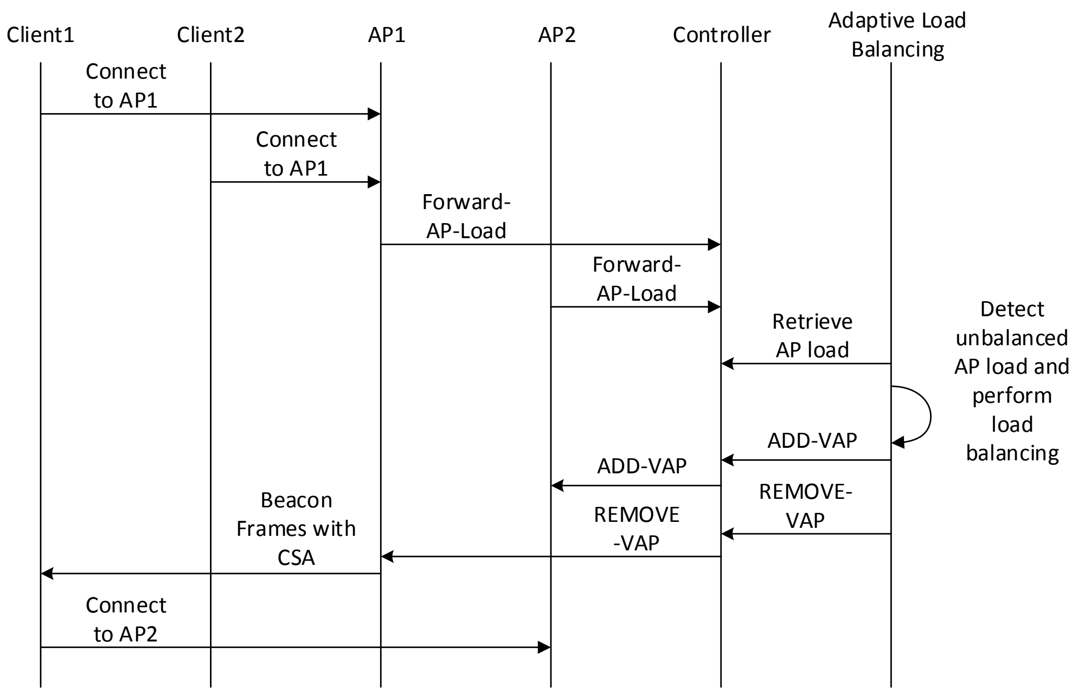


Fig. 3.4 An example of custom protocol messages that are exchanged during the process of adaptive load balancing in the selected software-defined enterprise WLAN system

The proposed adaptive load balancing algorithm is implemented on top of the central controller in a proactive way. It queries the network periodically for the traffic load condition of all physical APs by retrieving their AP load value from the central controller. In the case that the difference between the maximum and minimum value of the AP load is greater than a pre-defined threshold value, the proposed adaptive algorithm detects it as a sign of unbalanced AP traffic load and begins to perform load balancing by migrating wireless clients to less loaded APs with relatively good RSSI value. Fig.3.4 illustrates an example of the adaptive load balancing process and relevant protocol messages that are exchanged in the selected software-defined enterprise WLAN system. As it can be seen, all physical APs in the system keep sending the FORWARD_AP_LOAD custom protocol message to the central controller to update their traffic load information. Once the adaptive algorithm detects the unbalanced traffic load, it performs load balancing by using the ADD_VAP and REMOVE_VAP messages to evenly re-assign clients to physical APs in the system.

Custom beacon frames with CSA element are used to enable seamless handover over different wireless channels.

3.3 Proposed Adaptive Algorithms

3.3.1 AP Load

AP load is introduced as the novel metric to achieve the proposed adaptive algorithms for mobility management and load balancing. Normally, having multiple associated users imply a high possibility of generating heavy traffic load on the AP. Hence, the number of associated users must be considered when defining the AP load. However, the number of associated users cannot represent the AP load as the channel occupancy time of each user varies significantly. Therefore, the author defines the AP load by using both the number of associated users and another parameter called the channel load. Based on the definition in the IEEE 802.11k standard, the channel load is the fraction of time in which the wireless channel is sensed busy or idle. The calculation of the channel load is expressed in equation (3.1), and the AP load is calculated by equation (3.2). Based on test results, the author gives 80% weight to the channel load as user experience is affected significantly when the value of channel load is high. The number of associated users is given 20% weight as user experience is not affected even with multiple associated users that do not generate traffic. An exponential weighted moving average (EWMA) is used to dealing with AP load value in that they are affected by sudden variation of channel load. Based on test results, the frequency of channel load variation is relatively small. Thus, the author gives 90% weight to the current $APLoad_f$, and the final value of AP load is achieved with equation (3.3).

$$ChannelLoad = \frac{ChannelBusyTime}{MeasurementDuration} \tag{3.1}$$

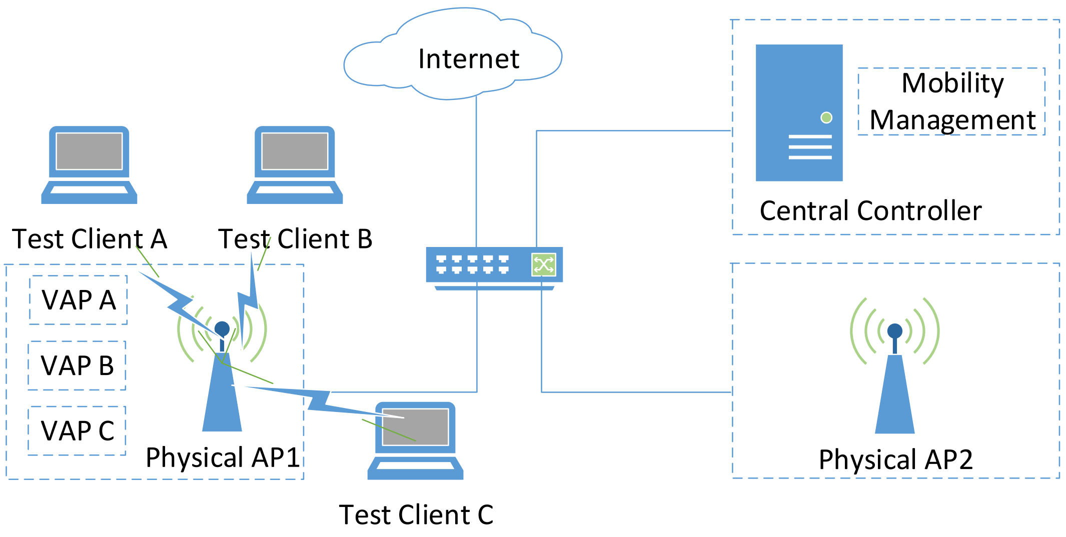


Fig. 3.5 Architecture of the testbed for mobility management experiments

$$APLoad = \begin{cases} ChannelLoad & (N_{au} = 0) \\ ChannelLoad \times 0.8 + N_{au} \times 0.2 & (N_{au} > 0) \end{cases} \tag{3.2}$$

$$APLoad_f = 0.9 \times APLoad_f + 0.1 \times APLoad_{f-1} \tag{3.3}$$

In the practical implementation, the channel load can be achieved from the channel load report of any Wi-Fi cards that support the IEEE 802.11k standard. However, the standard requires specific changes on both the AP and client device to support new measurement frames and procedures, while it is not always supported by commercial Wi-Fi cards. In the current deployment of our testbed, the TP-Link WR1043ND AP equipped with Wi-Fi card based on Qualcomm Atheros QCA9558 802.11 b/g/n chipset is used, and it does not support the IEEE 802.11k standard. However, this chipset applies the ath9k Wi-Fi driver which contains two registers called AR_RCCNT (0x80f4) and AR_CCCNT (0x80f8). The former one stores the number of time slots that are sensed busy by the clear channel assessment (CCA) mechanism and the latter one contains the total number of time slots that have elapsed.

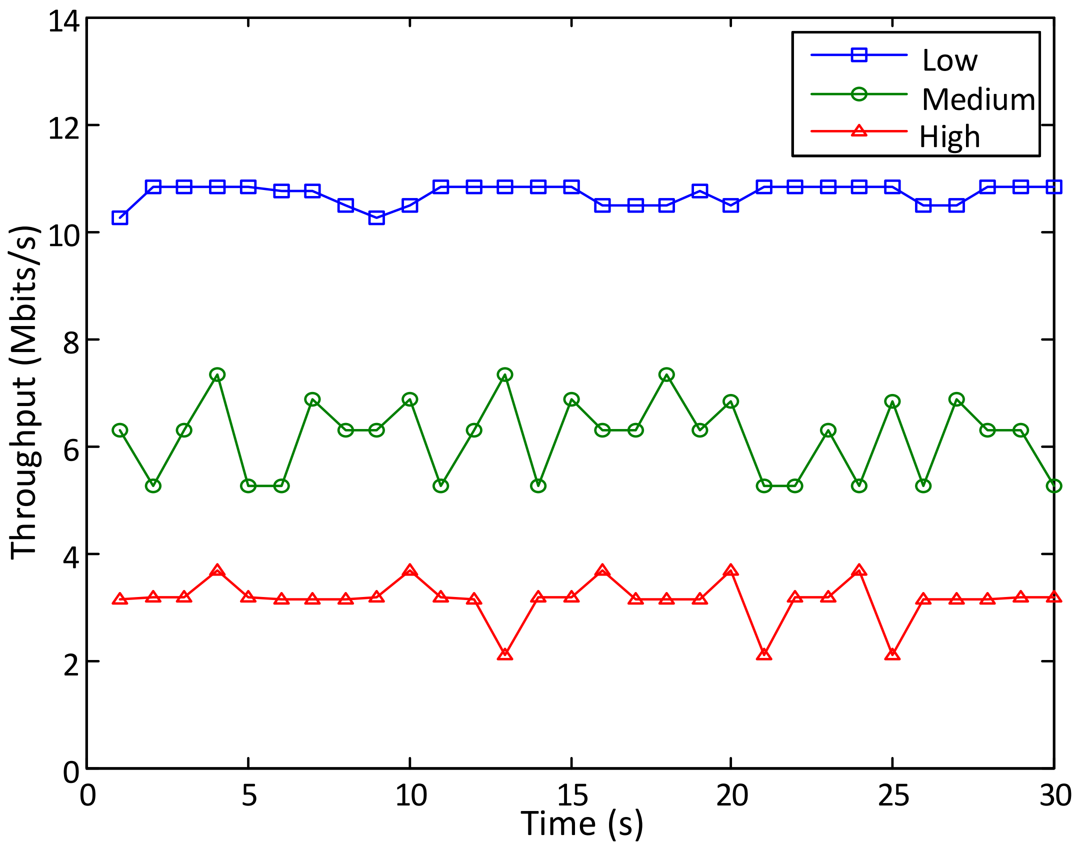


Fig. 3.6 Network throughput achieved with different levels of AP load

To calculate the AP load, the author modifies the ath9k based Wi-Fi driver to expose two debugfs based userspace interfaces which enable the agent module to read the value of both registers.

To set the hysteresis margin of AP load level, three simple tests with three levels of AP load (i.e. low, medium, and high) respectively are conducted on the testbed shown in Fig.3.5. As it can be seen, two physical APs are deployed and connected to the central controller via a normal layer two switch. The low level of channel load is defined as 0 to 0.49, the medium level is defined as 0.5 to 0.79, and the high level is defined as 0.8 to 1.0. For each test, client C is associated with physical AP1 to perform a TCP test with the length of 30 seconds by using iperf [120], and the result of network throughput is recorded. Each test is conducted five times, and results are averaged. In the first test, only client C is associated to AP1, and by

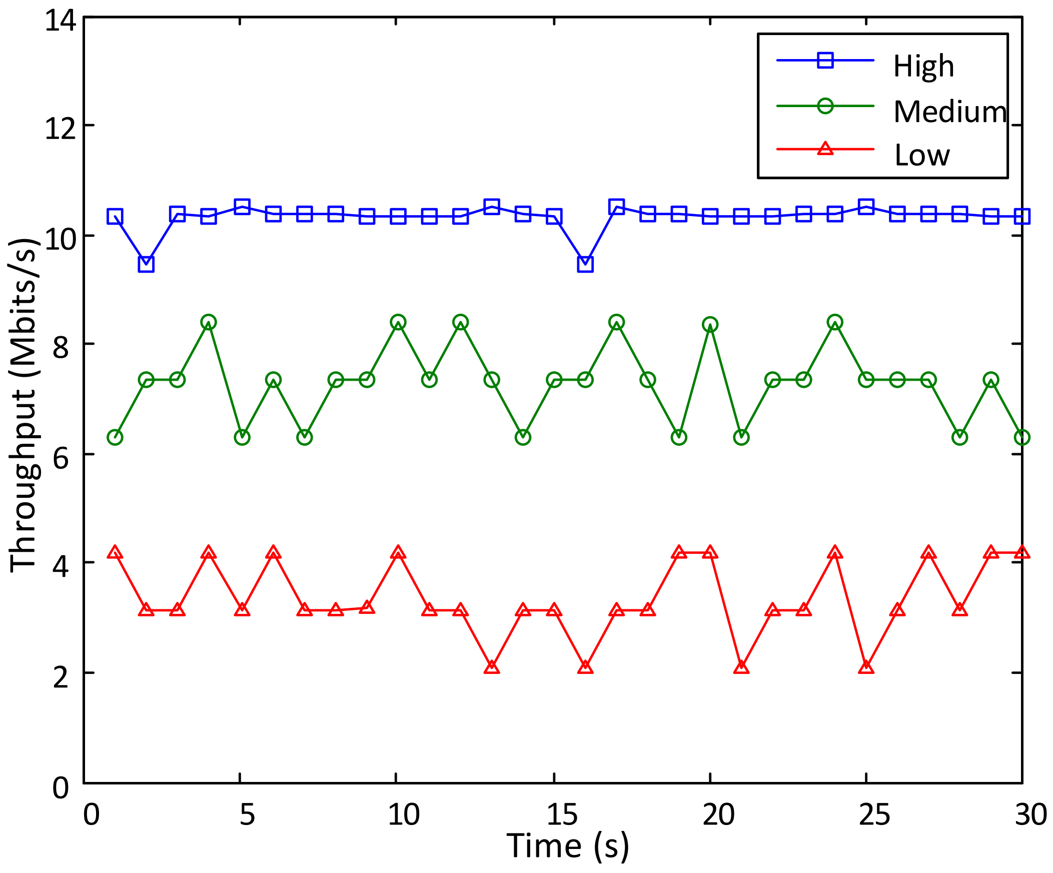


Fig. 3.7 Network throughput achieved with different levels of RSSI

increasing the number of associated clients and network traffic, the effect of different levels of AP load on network throughput is investigated. Based on the result shown in Fig.3.6, the hysteresis margins of AP load level are defined as 0.6 (LH_h), 0.3 (LH_m), and 0.15 (LH_l). As any margin of AP load level is greater than 0.6 can make a significant effect on network throughput and this effect decrease gradually when the margin turns to be 0.3 and 0.15.

To properly define the hysteresis margins of RSSI value, three similar tests are conducted. For each test, only client C is associated with AP1 to perform iperf-based TCP test, and the result of network throughput is recorded. Each test is conducted five times, and results are averaged. By increasing the distance between the test client and the physical AP, the effect of different levels of RSSI on network throughput is investigated. The high signal strength level is defined with any RSSI value which is greater than 205, the medium signal strength level is defined with RSSI value 190 to 204, and the low signal strength level is defined with any RSSI value lower than 189. Based on the result shown in Fig.3.7, the hysteresis margins of RSSI value are defined as 15 (RH_l), 30 (RH_m) and 40 (RH_h) because of its different extent of effect on network throughput. In addition, a dynamic time hysteresis with value of 3 (TH_s) and 6 (TH_l) seconds are used to avoid the influence of sudden variation of two handover metrics and the ping-pong handover effect. The initial threshold of RSSI value (RT_i) for the subscribed handover event to be triggered is defined as 180 where the network throughput starts to get affected significantly.

3.3.2 Adaptive Mobility Management Algorithm

Based on the AP load and RSSI value, an adaptive mobility management algorithm is proposed, which pseudocode is shown in Algorithm 1. To enable adaptive mobility management, the proposed algorithm first register the subscription of handover event with each AP through the central controller, and if the RSSI value of the client on any AP is greater than the pre-defined signal strength threshold RT_i which is 180, then the registered handover event is triggered. The main loop of the algorithm starts from Line 2, the mobility stats which contain the RSSI value of the client, the AP load value, the current time and the initial time when the mobility stats is assigned. The loop first checks the IP address of the AP where the event is triggered so as to see if it is the same of the AP that the client is currently associated, the mobility stats of the client are updated with current system time and the latest value of RSSI and AP load. However, the initial assigned time is not updated at this stage. If the event is triggered by a competing AP, the dynamic back off time hysteresis margin is then decided based on the difference of the RSSI value and the AP load level. If the handover event is triggered by a competing AP, the back off time is then checked to prevent ping-pong handover effect with Line 11. If the notification is received outside the time hysteresis period, the handover decision is then made based on the handover metrics. Line 14 ensures the client

Algorithm 1 Adaptive Mobility Management Algorithm

```
1: Subscribe handover event with APs
2: while RSSI $> RT_i$ do
3:     Assign initial mobility stats of client
4:     if $IP_c = IP_a$ then
5:         Update mobility stats of client
6:     else
7:         if ($RSSI_c$ - $RSSI_a > RH_h$) or
           ($Load_a$ - $Load_c > LH_h$) then
8:             $TH = TH_s$
9:         else
10:            $TH = TH_l$
11:            if ($Time_{cur}$ - $Time_{ass} < TH$) then
12:                back off
13:            else
14:                if ($RSSI_c > RSSI_a + RH_l$) and
                  ($Load_c + LH_l < Load_a$) then
15:                    Handover client and update mobility stats
16:                else if ($RSSI_c > RSSI_a + RH_m$) and
                  ($Load_c < Load_a + LH_l$) then
17:                    Handover client and update mobility stats
18:                else if ($RSSI_c + RH_l > RSSI_a$) and
                  ($Load_c + LH_m < Load_a$) then
19:                    Handover client and update mobility stats
20:                else if ($RSSI_c + RH_m > RSSI_a$) and
                  ($Load_c + LH_h < Load_a$) then
21:                    Handover client and update mobility stats
22:                else if ($RSSI_c > RSSI_a + RH_h$) then
23:                    Handover client and update mobility stats
24:                end if
25:            end if
26:        end if
27:    end if
28: end while
```

Algorithm 2 Adaptive Load Balancing Algorithm

```
    while true do
2:      Sleep $Timer_i$
        for $IP_{ap} \in List_{ap}$ do
4:          Get $MAC_{au}$
            Build $List_{au}$
6:      end for
        Get $APload_{max}$ and $APload_{min}$
8:      if ($APload_{max} > 1$) and
         ($APload_{max}$ - $APload_{min} > 0.6$) then
            for $MAC_{au} \in List_{au}$ do
10:             Get $APload_{min}$
                if ($RSSI_{minap} > RSSI_T$) then
12:                 HandoverClientToAP($APload_{min}$)
                else
14:                 Get the next min value $APload_{min2}$
                    if ($RSSI_{minap2} > RSSI_T$) then
16:                     HandoverClientToAP($APload_{min2}$)
                    end if
18:             end if
            end for
20:     end if
    end while
```

is handed off to less loaded AP with better signal strength. Line 16 hands off the client to the competing AP with much better signal strength even it is slightly heavier loaded than the current associated AP. Line 18 guarantees that the client is handed off to much less loaded AP with relative lower signal strength but well enough to provide good QoS. A similar decision is made in Line 20 but with larger hysteresis margins of RSSI value and AP load value. Line 22 ensures that client is handed off if it is going to disassociate from current associated AP no matter how crowd the competing AP is. Once the handover completes, the mobility stats including the initial assigned time are updated.

3.3.3 Adaptive Load Balancing Algorithm

By utilising AP load and RSSI value as the load balancing metrics, an adaptive load balancing algorithm is proposed, and its pseudocode is shown in Algorithm 2. The proposed adaptive load balancing algorithm is implemented as an application on top of the Floodlight controller and works in a proactive way by querying the network every 30 seconds which is the pre-defined value of time interval $Timer_i$ to get current associated users of each physical AP. It then builds a list of all associated users $List_{au}$ by using their Media Access Control (MAC) address MAC_{au}. Once the list of the associated user is obtained, the application acquires the maximum $APload_{max}$ and minimum $APload_{min}$ value of AP load from the central information management module. The algorithm believes traffic load of APs are balanced unless $APload_{max}$ is greater than 1 and the difference between $APload_{max}$ and $APload_{min}$ is greater than 0.6. The first condition indicates the AP with maximum traffic load have at least two associated users. The second condition is defined based on the test result shown in Fig.3.5 regarding the effect of AP load value on network throughput. The result shows any margin of AP load value that is greater than 0.6 can make a significant effect on network throughput. Both conditions imply unbalanced traffic load of AP and load balancing is required. In the case that load balancing is triggered, the algorithm starts to re-assign users in the list of associated users $List_{au}$. For each associated user, the algorithm acquires the AP with a minimum value of AP load and compares its RSSI value $RSSI_{minap}$ to the RSSI threshold $RSSI_T$ which is configured as 200. The configuration of the threshold value is based on the test result shown in Fig.3.6 regarding the effect of RSSI value on network throughput where it is found out that any RSSI value higher than 200 is able to provide relatively good network throughput. Therefore, users are guaranteed to have a good QoE after been re-assigned to the AP with minimum load value. If the condition in line 11 is not met, the algorithm acquires the AP with second minimum load value and repeat the comparison of its RSSI value $RSSI_{minap2}$ and the RSSI threshold $RSSI_T$ and handover user to the AP

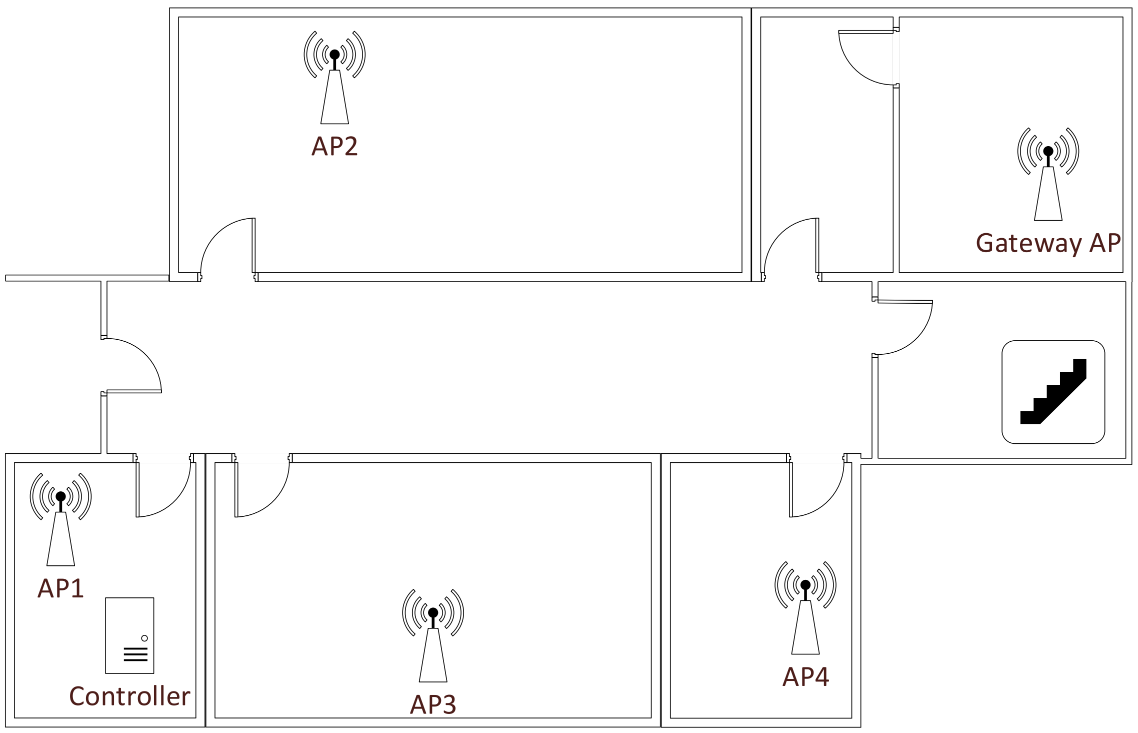


Fig. 3.8 Layout of the extended testbed

if its signal is strong enough to provide a good network throughput. The algorithm keeps repeating above procedures until the last user in the list of associated users is re-assigned to new AP, and after that, it goes to sleep for another time interval $Timer_i$ before its next move.

3.4 Performance Evaluation

3.4.1 Experiment Setup

The testbed for mobility management experiments is shown in Fig.3.5. Two physical APs are deployed in the same room. The distance between both APs is less than 10 meters which make the test environment heavily overlapped. For load balancing experiments, an extended testbed is deployed with its layout depicted in Fig.3.8. Currently, the extended testbed is deployed on the 5th floor of the networking building at the main campus. More specifically, the author has deployed five APs with one AP works as the default gateway, and

Table 3.1 Device information and Experiment configuration

AP Model	TP-Link WR1043ND Ver 2.1
AP Firmware	Openwrt Chaos Calmer 15.05
Number of APs	5
User Operating System	Android 5.0 and Linux Ubuntu 14.04
Number of User Devices	6
Wi-Fi Channels	1, 6 and 11
User Traffic	wget, iperf-based TCP and UDP test
SDN Controller	Floodlight 1.0
Integrated Software	Open vSwitch 2.3, Click Modular Router 2.0.1

the rest four APs perform as OpenFlow-enabled APs. The distance between each APs is less than 15 meters which make the test environment heavily overlapped. In addition, one PC is deployed as the central controller which monitors the traffic load of connected APs and runs proposed adaptive load balancing application. The connection between the default gateway AP and four OpenFlow-enabled APs are achieved via a normal layer two switch. The default gateway AP runs the Dynamic Host Configuration Protocol (DHCP) server which assigns IP address to OpenFlow-enabled APs and the central controller. In order to eliminate DHCP and authentication related delays, all test client devices are assigned with static IP addresses, and Wi-Fi authentication is disabled. Table 1. shows the information and related configurations of hardware and software used for the experiments.

3.4.2 Adaptive Mobility Management Experiments

To investigate the performance of the proposed adaptive mobility management algorithm, four iperf-based Transmission Control Protocol (TCP) tests are conducted with the iperf server running on the central controller. Collected results of network throughput are compared to

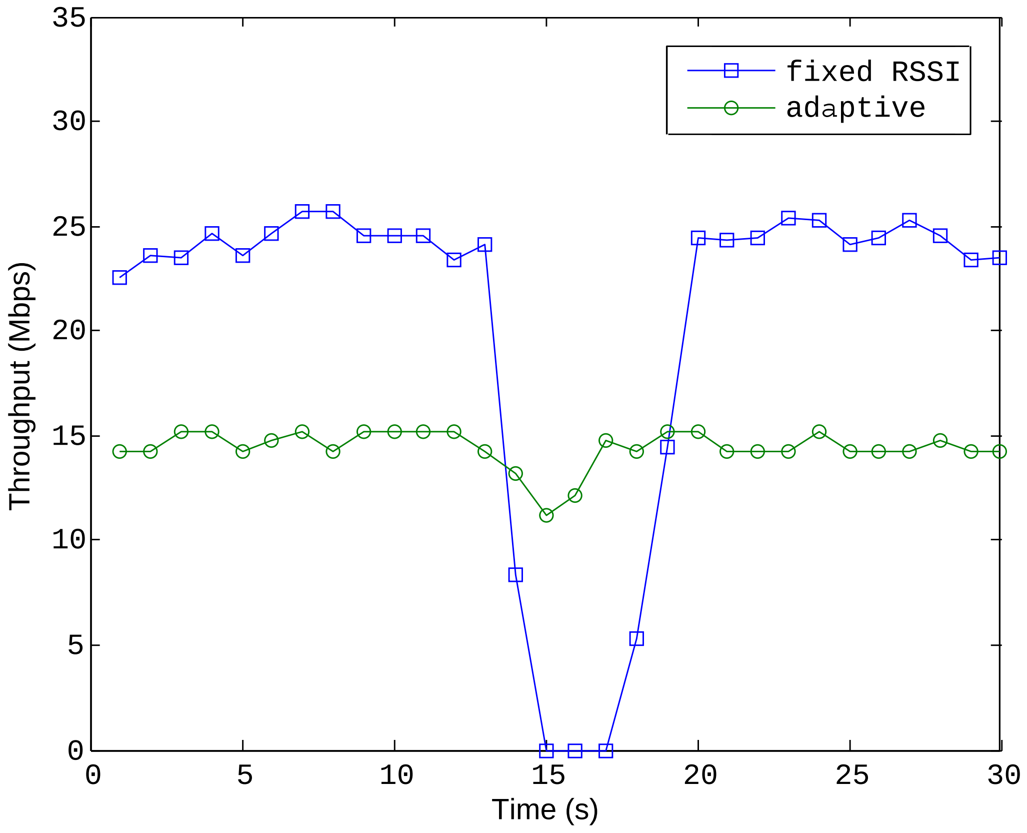


Fig. 3.9 Network throughput comparison for mobility management experiment scenario 1

the one achieved with the conventional mobility management algorithm, which uses the fixed RSSI value as the handover metric.

The aim of the first experiment is to show how the proposed mobility management solution supports seamless client handover over multiple wireless channels. In this experiment, one smartphone with Android 5.0 OS is used as the test client *A*, which is initially associated with AP1. At the beginning of the experiment, test client *A* starts an iperf-based TCP test with 30 seconds duration. During the TCP test, test client *A* is moved from AP1 to AP2. Experiment results of network throughput are collected in one-second interval. The same experiment is conducted five times to get the averaged results, which are shown in Fig.3.9. There are two key points to be noticed with the result. First, the network throughput achieved with the proposed adaptive mobility management algorithm is approximate 10 Mbps lower

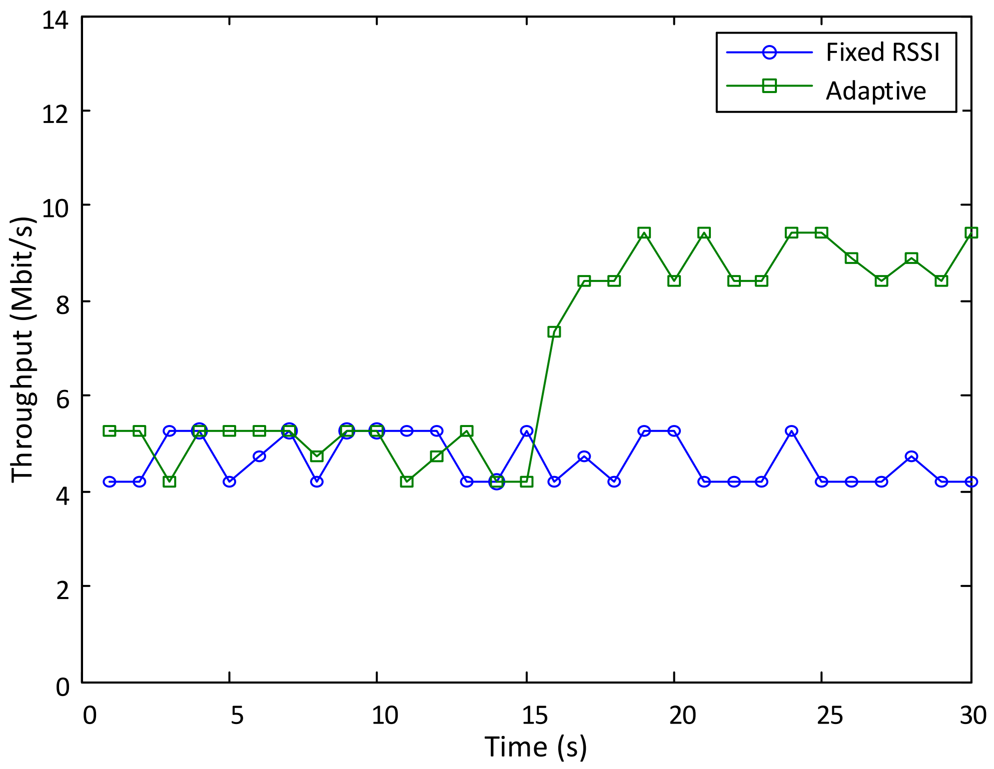


Fig. 3.10 Network throughput comparison for mobility management experiment scenario 2

than the result of the conventional mobility management algorithm. This reduced data rate is due to the fact that the userspace click modular router is used in the system that causes slower and jittery forwarding performance on the AP hardware. Consequently, TCP is forced to throttle down. Second, it can be observed that the network throughput achieved with the conventional mobility management algorithm is dropped to 0 after 15 seconds of the TCP test. This is because test client *A* is disconnected from AP1 and re-connect to AP2 to finish the handover process. This process only takes few seconds but can cause service disruption that significantly degrades QoS and user QoE. By contrast, there is no throughput disruption at the time of handover with the proposed adaptive mobility management algorithm as the proposed solution uses CSA-enabled beacon frame to change client's working channel seamlessly during handover.

The purpose of the second experiment is to show that the proposed adaptive mobility management algorithm can perform efficiently when the difference between both APs'signal

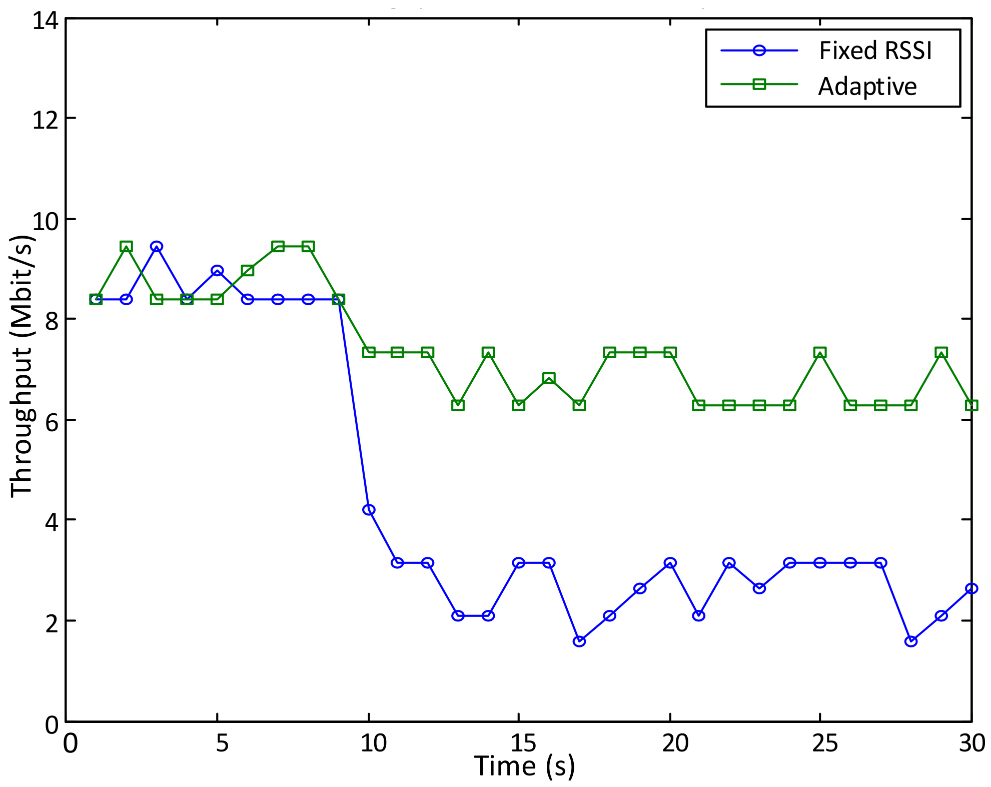


Fig. 3.11 Network throughput comparison for mobility management experiment scenario 3

strength is very small. Two smartphones with Android 5.0 OS are used as the test client devices. Initially, both test client devices *A* and *B* are associated with AP1 to start the iperf-based TCP test with 30 seconds duration. Once the TCP test session begins, client device *A* starts to move slowly towards AP2 until it reaches the overlapping area of two APs where the difference of the signal strength between those two APs is quite small. After that, test client *A* stops moving until the end of the iperf TCP session. The network throughput result of test client *A* is collected over one-second interval. The same experiment is conducted five times and results are averaged and presented in Fig.3.10. It can be seen that the network throughput achieved with the conventional mobility management algorithm remains around 5Mbps during the whole TCP session. Because the difference of RSSI value between AP1 and AP2 is not big enough to trigger the handover of test client *A*, test client *A* is associated to AP1 during the whole session and have to share bandwidth with test client *B* until the end of the experiment that leads to low network throughput. However, with the proposed adaptive mobility management algorithm, test client *A* is associated with AP2 which is less congested after 15 seconds of the TCP session as the AP load of AP2 is much lower than AP1 that trigger the handover of test client *A*. It can be observed that the network throughput achieved with the proposed mobility management algorithm is improved by more than 5 Mbps compared to the results achieved with the conventional mobility management algorithm, which uses fixed RSSI hysteresis as the handover metric.

The aim of the third experiment is to show that how efficiently the proposed adaptive mobility management algorithm can perform when the AP with better signal strength is heavily loaded. One smartphone with Android 5.0 OS and one laptop with Ubuntu 14.04 OS are used as the test client *A* and *B* respectively. At the beginning of the experiment, test client *A* is associated with AP1 to run the iperf-based TCP test, while test client *B* is associated with AP2 to download a large file by using wget [121]. After the start of the TCP session, test client *A* begins to move towards AP2 until it gets to a very close range of AP2. The whole TCP session lasts 30 seconds and is repeated five times to average the results of network throughput, which is shown in Fig.3.11. It can be observed that the network throughput achieved with the conventional mobility management algorithm is dropped by more than 6 Mbps after 10 seconds compared to the result achieved with the proposed mobility management algorithm. Because test client *A* is moved closely to AP2, which makes the signal strength difference between AP1 and AP2 big enough to hand over the test client *A* to AP2 with the conventional mobility management algorithm. As a result, test client *A* has to share the bandwidth with test client *B* that makes both of them suffer low network throughput. In contrast to the conventional mobility management algorithm, the proposed adaptive mobility management algorithm works more efficiently as it detects the heavy traffic load of AP2, and instead of performing a handover, the proposed adaptive algorithm remains

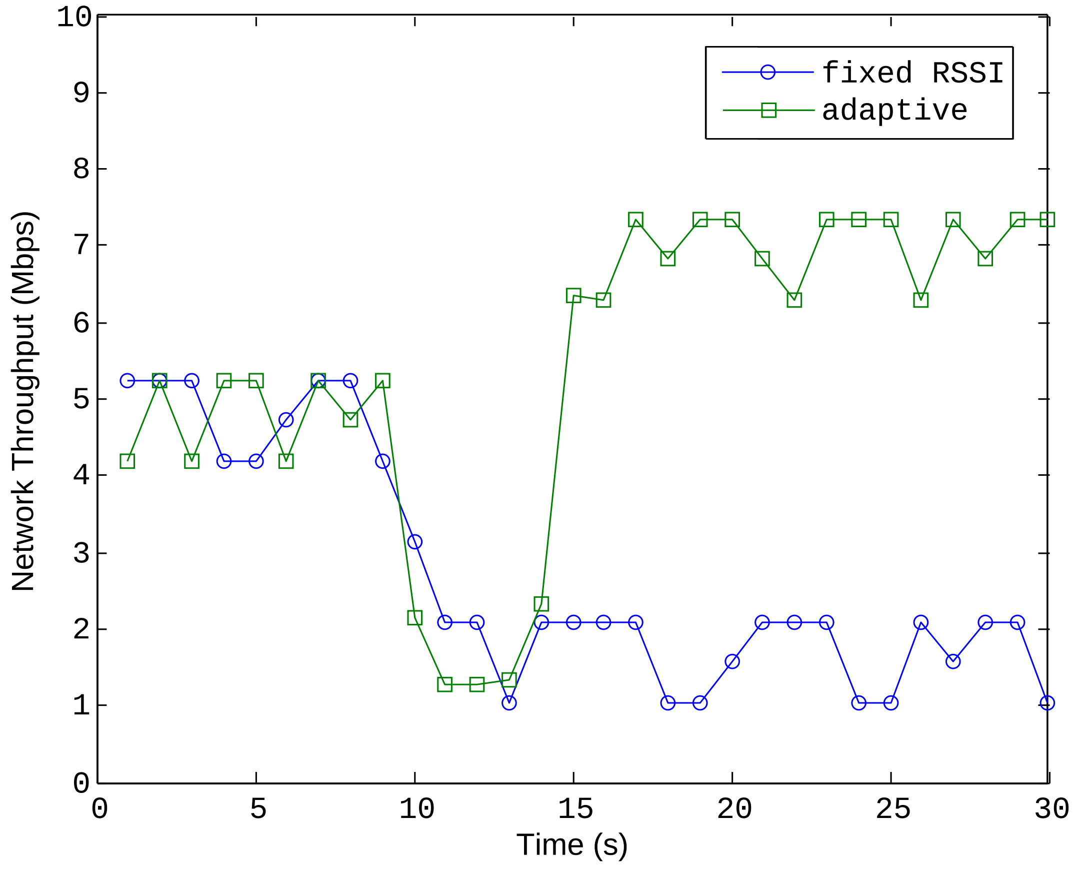


Fig. 3.12 Network throughput comparison for mobility management experiment scenario 4

the current association of test client *A* with less loaded AP1, which is able to provide better QoS and user QoE.

The purpose of the fourth experiment is to investigate the performance of the proposed adaptive mobility management algorithm with dynamic changing traffic load. Four smartphones with Android 5.0 OS are used as test client devices. Both test clients *A* and *B* are initially associated with AP1 and remain static until the end of the experiment. Both test clients *C* and *D* are initially connected to AP2, and test client *D* remains static until the end of the experiment. In the beginning, all test clients start iperf-based TCP test which lasts 30 seconds. After around 6 seconds of the TCP session, test client *C* is manually associated to AP1 to perform a fresh iperf-based TCP test with 30 seconds duration. The results of network throughput of test client *A* is collected over one-second interval. The same experiment is conducted five times, and results are averaged and presented in Fig.3.12. It can be observed that the decline of network throughput achieved with the conventional mobility management algorithm is approximately 4Mbps after 10 seconds of the experiment because of the extra traffic load generated by test client *C*. However, with the proposed adaptive mobility management algorithm, network throughput of test client *A* is increased around 6 Mbps after 15 seconds as the adaptive algorithm detects the sudden increasing traffic load on AP1 and performs handover of test client *A* by changing its connection from AP1 to AP2, which is less congested and able to achieve better network throughput.

3.4.3 Adaptive Load Balancing Experiments

Four experiments are conducted to investigate the performance of the proposed adaptive load balancing algorithm. For each experiment, two laptops with Linux Ubuntu 14.04 OS and four smartphones with Android 5.0 OS are used as the test user devices. TCP and User Datagram Protocol (UDP) tests based on iperf are performed with three different scenarios which are without load balancing service, with load balancing algorithm based on round robin method and with proposed adaptive load balancing algorithm respectively. Experiment results in terms of average packet loss rate and network throughput are collected and compared to evaluate the efficiency of the proposed adaptive load balancing algorithm.

The aim of the first experiment is to evaluate the performance of the proposed adaptive load balancing algorithm on decreasing user packet loss rate. To set up the experiment, only AP1 and AP4 are used, and all six test users are gradually added to the network with static IP addresses. Test users are evenly placed within a meter of both APs but are all associated with AP1 initially and remained stationary during the whole experiment session. To start the experiment, all test users run an iperf-based UDP test session to generate fixed-size UDP segments at a constant rate to the iperf server, which is running on the default gateway AP.

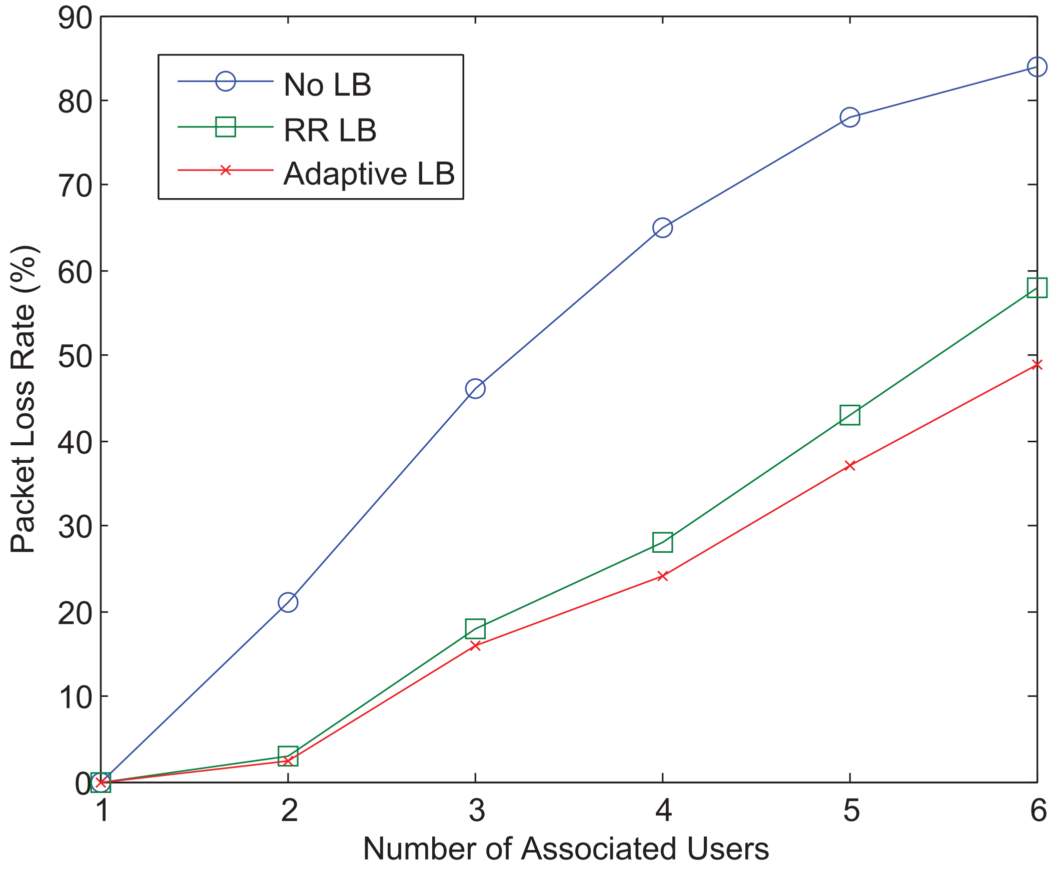


Fig. 3.13 Average packet loss rate comparison with 2 APs

The UDP buffer size is set to the default value of 160 Kbytes, and the source data rate is configured to be 54 Mbps. Each UDP test session lasts 60 seconds and is conducted ten times to get the averaged statistical results of packet loss rate. Fig.3.13 shows the average packet loss rate achieved with three different load balancing mechanisms. As it can be observed, with the increasing number of associated users, the average packet loss rate achieved without load balancing mechanism is significantly high (i.e. around 80% with all test clients connected to AP1) compared to the results achieved with the other two load balancing algorithms. This result is due to network congestion as all test users are associated with AP1 to contend for the channel resource and the effective bandwidth that each user received is reduced, which leads to increased packet loss rate of all test users. In contrast, with the round robin and the proposed adaptive load balancing algorithms, test users are evenly migrated to AP2 when

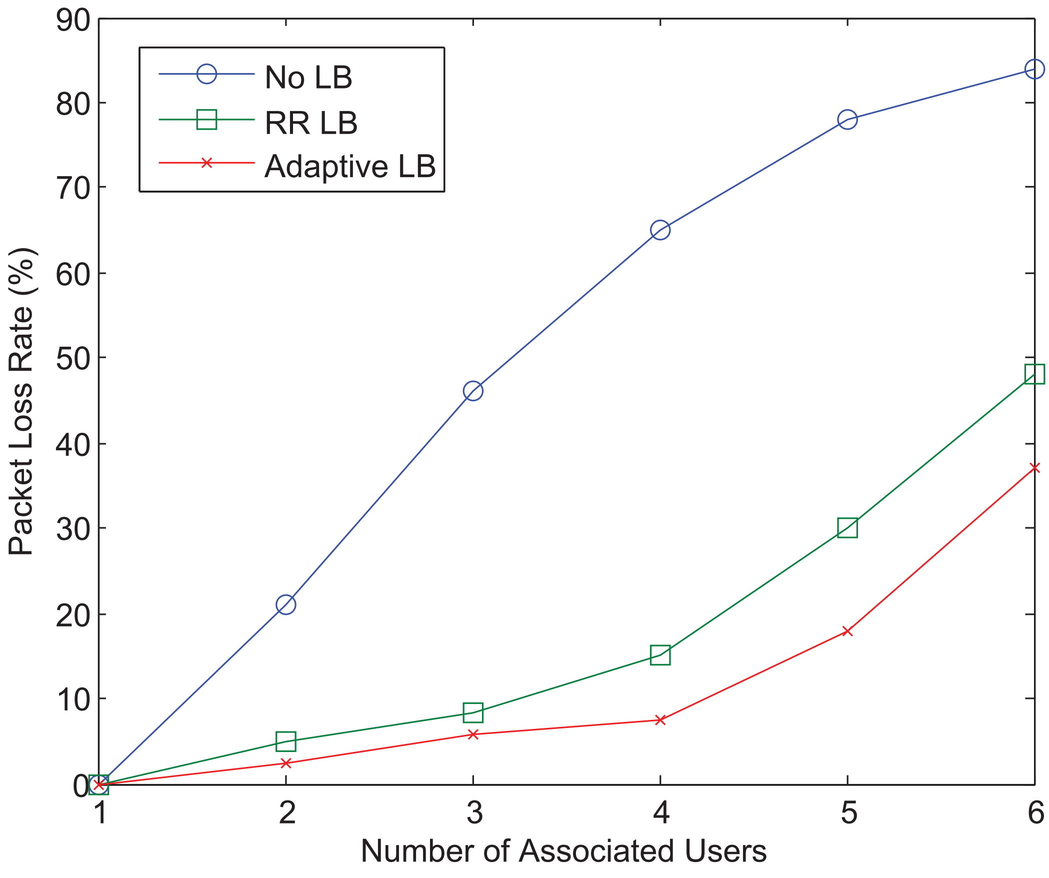


Fig. 3.14 Average packet loss rate comparison with 4 APs

the load balancing service detects the unbalanced traffic load on AP1. It can be seen that the average packet loss rate decreases dramatically (i.e. approximately 30% with all test users added to the network) compared to the results achieved without load balancing mechanism. In addition, the average packet loss rate achieved with the proposed adaptive load balancing algorithm is about 10% lower than the result achieved with the round robin algorithm when all test users are added to the network. This result is because load balancing based on round robin algorithm randomly re-distributes test users without considering the signal strength of each AP in the system.

In the second experiment, the same iperf-based UDP tests of experiment one are conducted but on a larger scale of the testbed by adding AP2 and AP3 to the network in addition to the initial setup of the first experiment. Result in terms of average packet loss rate is

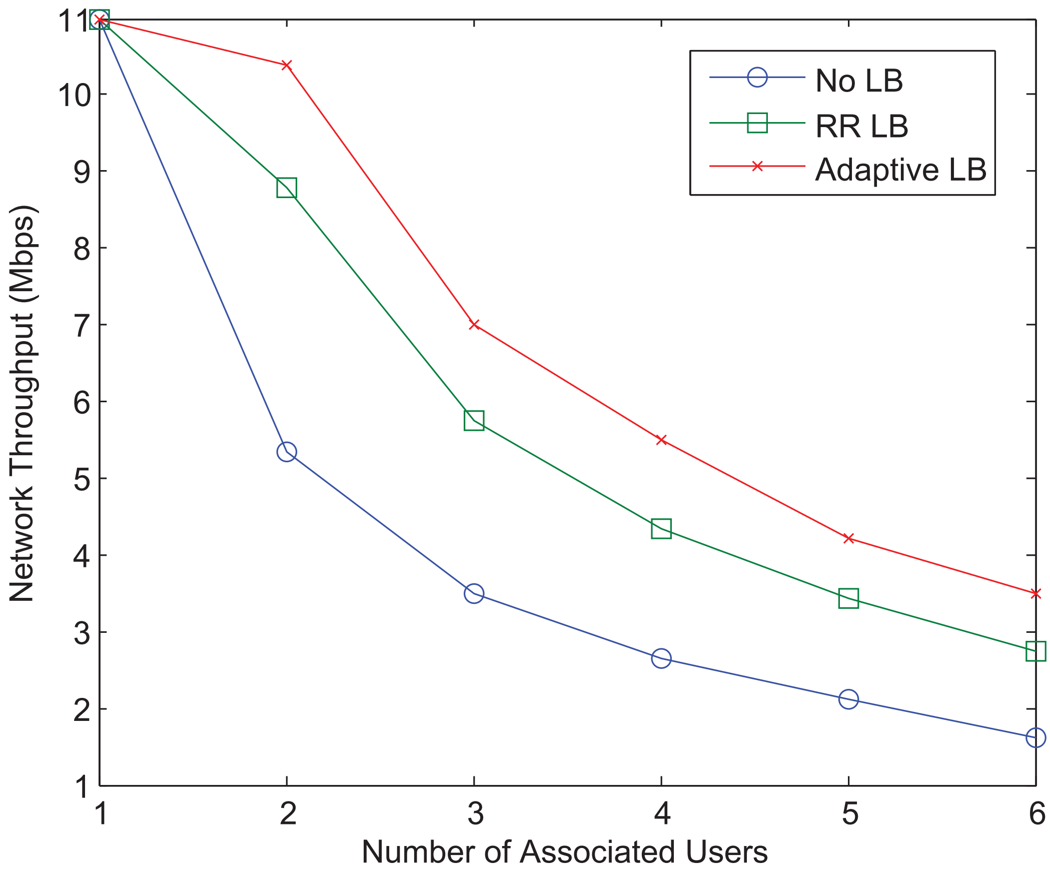


Fig. 3.15 Average network throughput comparison with 2 APs

illustrated in Fig.3.14. It can be observed that the average packet loss rate achieved with the round robin load balancing algorithm and the proposed adaptive load balancing algorithm is decreased significantly compared to the result of the first experiment. This result is because the re-distribution of test users are more balanced and the contending of the channel resource is further decreased. Also, the proposed adaptive load balancing algorithm is able to achieve better average packet loss rate compared to the round robin load balancing algorithm as with a larger scale of the testbed, test users have a much higher possibility of being re-assigned to the AP with better signal strength with the proposed adaptive load balancing algorithm.

The purpose of the third experiment is to investigate the performance of the proposed adaptive load balancing algorithm on increasing average user throughput. The initial setup of test users is the same to the first experiment. To begin the experiment, all test users run

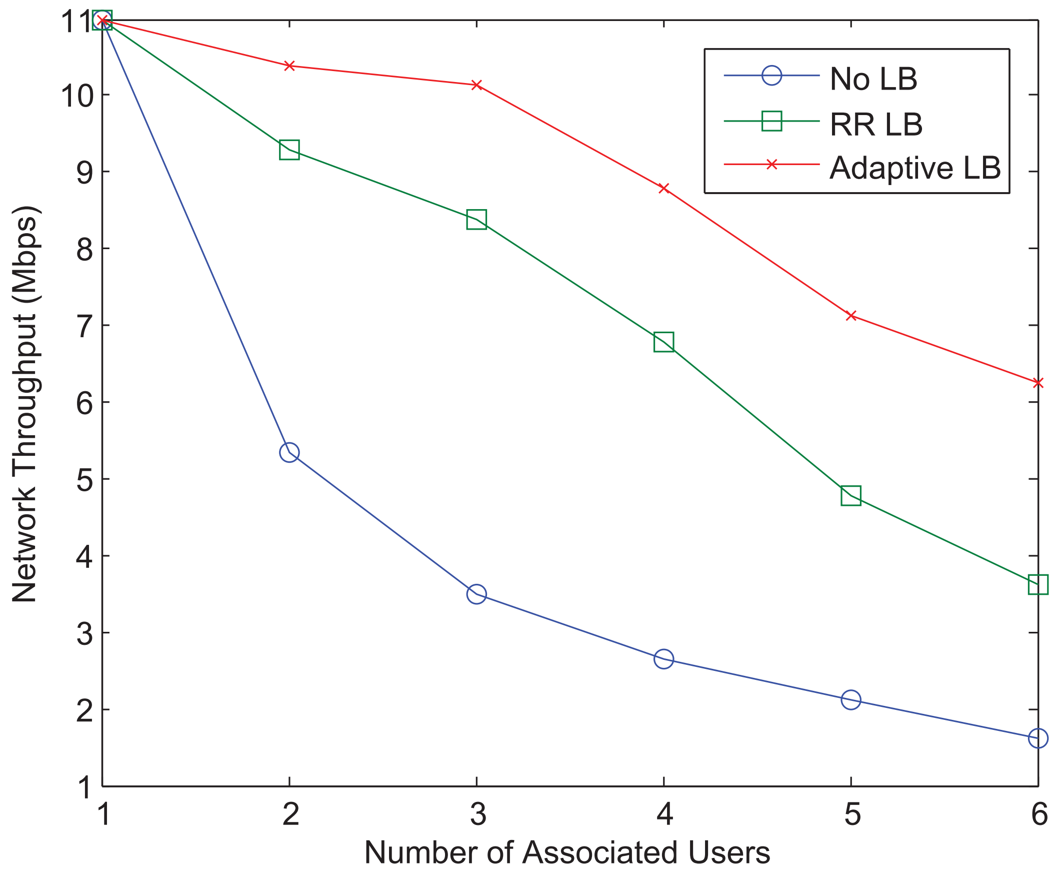


Fig. 3.16 Average network throughput comparison with 4 APs

an iperf-based TCP test session which lasts 60 seconds. Test results of network throughput are collected over one-second interval. The same experiments run ten times with averaged result shown in Fig.3.15. It can be seen that with the increasing number of associated users, the average network throughput achieved without load balancing mechanism is declined significantly (i.e. less than 2 Mbps with all test clients connected to AP1) compared to the results achieved with the other two load balancing algorithms. The reason for low network throughput is because no load balancing mechanism is applied and all test users are associated AP1 to share the bandwidth that causes heavy network congestion. By contrast, test users are evenly re-assigned to AP2 when the round robin and the proposed adaptive load balancing algorithms detect the unbalanced traffic load on AP1. This re-distribution guarantees test users to receive better QoS and user QoE which is reflected in the result. It also can be noticed that the average network throughput achieved with the proposed adaptive load balancing algorithm is around 1 Mbps higher than the result achieved with the round robin algorithm when all test users are added to the network. This result is due to the random re-distribution of test users with the round robin load balancing algorithm. Without considering the signal strength of each AP in the system, test users are not guaranteed to be re-assigned to the AP that provides the best QoS and user QoE.

In the fourth experiment, the same iperf-based TCP tests of experiment three are conducted but on a larger scale of the testbed by adding AP2 and AP3 to the network. The initial setup of test users is the same to the third experiment. Instead of sending UDP traffic, a TCP client with default window size 64 Kbytes is run on each test user to send TCP traffic to the iperf server running on the default gateway AP. The whole TCP session lasts 60 seconds and network throughput is recorded within one-second interval. The same experiments run ten times, and results are averaged. Fig.3.16 depicted the average network throughput achieved with three different load balancing mechanisms. It can be observed that the performance of the proposed adaptive load balancing algorithm increased noticeably by improving the average network throughput with more than 2.5 Mbps compared to the result achieved with the round robin load balancing algorithm.

3.5 Chapter Summary

This chapter investigates how to achieve intelligent network traffic control in enterprise WLANs from a client-side perspective. An adaptive mobility management algorithm and an adaptive load balancing algorithm are proposed to improve the client-based traffic control in software-defined enterprise WLANs. The proposed adaptive algorithms take AP load into account in addition to RSSI to achieve efficient client mobility management and AP load balancing that address the issue of unbalanced AP traffic load. A custom protocol message called Forward_AP_Load is introduced to enable the central controller to have the knowledge of each AP's traffic load information. The CSA element is added into beacon frame to achieve seamless client handover over multiple wireless channels. TCP and UDP test based on iperf are conducted to evaluate the efficiency of the proposed adaptive algorithms. Experiment results show that the proposed adaptive mobility management and load balancing algorithms can improve QoS and user QoE significantly in terms of network throughput and packet loss rate when compared to the conventional mobility management and load balancing algorithms, which are based on the fixed-RSSI and round robin method.

In the next chapter, the author studies how to improve network traffic control from an AP-side perspective by presenting a software-defined enterprise WLAN system and two load-aware automatic channel switching algorithms which improve the performance of channel switching control on the 2.4GHz frequency band.

Chapter 4

Load-Aware Automatic Channel Switching

4.1 Introduction

In recent years, the deployment of enterprises WLANs has increased at a remarkable rate which makes the effective management of such networks significantly important. Due to the broadcast nature of wireless communication, the task of providing satisfied QoS and user QoE becomes extremely difficult; moreover, the increasing trends such as BYOD, rapidly growing user data traffic and high network bandwidth demand only exacerbate this problem. The IEEE 802.11 WLANs can operate on two unlicensed frequency spectrum bands which are 2.4GHz and 5GHz. Fig.4.1 shows the 2.4GHz frequency band channels overlapping information. As it can be seen, the 2.4 GHz spectrum has 14 channels for normal operation, and there are only three non-overlapping channels, which are 1, 6 and 11 with centre frequencies 2412 MHz, 2437 MHz and 2465 MHz respectively. The 5GHz spectrum provides 23 20MHz wide channels which are well suited for high-density Wi-Fi deployment due to its higher number of non-overlapping channels when compared with the 2.4GHz spectrum. The unlicensed frequency bands suffer interference issues during operation, which are the major problems and cannot be solved completely. Two types of interference mainly seen in WLAN are co-channel interference and adjacent channel interference. Co-channel interference (CCI) is caused by APs and users that are operating on the same channel frequency. In case that two users transmit at the same time, their radio signals collide which results in data corruption and packet loss. In order to avoid collisions, 802.11 users perform the CCA to detect if other users are transmitting on the channel before starting its own transmission. If another transmission is detected, it will wait for a random short period after which it would perform another check before attempting to transmit again. With the increasing number of contenders on the same channel, users have to wait longer to send data. Adjacent channel interference (ACI) result from APs and users which are working on different channel frequencies but are overlapped. Users on overlapped channels transmit at the same time can cause data corruption and packet loss as the CCA check may not detect the collision due to channel overlapping. As the number of interfering users increases, the potential of re-transmission is raised, and it takes longer for the user to successfully send a data packet. The outcome of channel interference is the degradation of network performance, such as low network throughput, high packet loss rate and transmission delay, which can significantly affect the user experience.

SDN is an emerging architecture and intensively discussed as one of the most promising technologies to simplify network management and service development. OpenFlow is the first standard communication interface defined between the central controller and the network forwarding devices of an SDN framework. It is a key enabler for SDN that allows direct manipulation of the forwarding plane of network devices. In recent years, SDN has rapidly transformed campus networks, data centres and the cloud to provide a more flexible network architecture based on its simple per-flow management. In the SDN architecture, network management is directly programmable because it is decoupled from forwarding

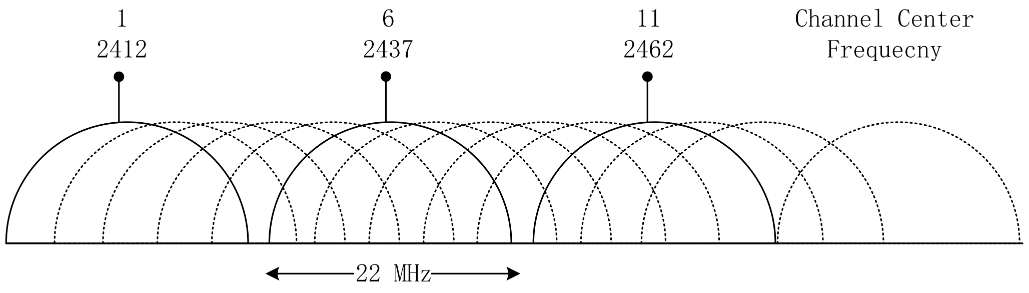


Fig. 4.1 The 2.4GHz frequency band Wi-Fi channels overlapping

layer; essential network services such as load balancing, mobility management and bandwidth control, can be easily implemented by network administrators.

4.1.1 Problem Statement

To minimize the interference issue and guarantee QoS and user QoE, optimised channel allocation mechanism is critical to modern wireless networks [122]. With traditional enterprise WLAN management, network administrators manually configure the wireless channel for APs based on detailed site surveys. Past research work [94] [103] also proposed similar static solutions. However, with the increasing traffic variability and user mobility, the performance of existing static channel assignment approach is bound to suffer; furthermore, it is insufficient for modern enterprise WLANs as the airtime demand in enterprise WLANs can vary significantly both across APs and across time. To be effective for usage patterns of modern enterprise WLANs, a centralised automatic channel switching approach is needed, which can adapt to the rapidly changing wireless environment and perform real-time traffic load aware channel switching.

The conventional channel switching mechanism is based on RSSI which is inefficient. Most commodity APs use automatic channel selection mechanism, but it only works at boot time. In addition, it requires a reboot of WLAN interface during the process of channel switching which leads to a short period of service disruption. User devices have to perform re-scan, re-authentication and re-association to connect to AP after channel switching. These processes can cause problems such as decreased network throughout, increased transmission delay and jitter which affects user experience significantly. Commercial vendors offer their solutions through vendor proprietary software and hardware. Unfortunately, these solutions do not have open programmable interface and are unable to be achieved with the low-cost commodity AP hardware that is used by provider networks. Therefore, it is vital for the channel switching algorithm to be aware of real-time network traffic load and provide seamless channel switching for users.

4.1.2 Contributions

In this chapter, the author presents a software-defined enterprise WLAN system and two load-aware automatic channel switching algorithms to improve the performance of channel switching control on the 2.4GHz band as it is much more crowd than the 5GHz band. Leveraging the characteristics of SDN, the proposed automatic channel switching application runs on top of the SDN controller to maintain a central view of the network state, and to provide real-time seamless channel switching based on the received AP load and channel quality information from each connected AP in the system. Two channel switching algorithms named the Single Switch (SS), and the Double Switch (DS) are proposed. The SS algorithm aims to improve overall user experience while the DS algorithm targets to guarantee the experience of users with highest traffic demand. The author demonstrates the efficiency of proposed automatic channel switching algorithms by evaluating their performance on improving user experience in terms of network throughput, jitter, and transmission delay.

4.1.3 Structure of Chapter

The remainder of this chapter is organised as follows. In Section II, the related work is presented. Section III introduces the system architecture of the proposed software-defined enterprise WLANs. Section IV further presents the channel switching metrics and the proposed load-aware automatic channel switching algorithms in detail. Performance evaluations of the proposed SS and DS algorithms are demonstrated in Section V, and finally, the summary of this chapter is presented in Section VI.

4.2 System Design

In order to achieve efficient AP channel control in enterprise WLANs, a centralised view of the entire network status is essential. By utilising the characteristics of SDN, a three-layer software-defined enterprise WLAN system is designed. The architecture of the designed system is depicted in Fig.4.2.

4.2.1 Local Information Base

In the infrastructure layer, physical APs are installed with Linux based OpenWRT Chaos Calmer firmware which is then integrated with Open vSwitch and Click Modular Router. The former one turns a normal AP into an OpenFlow-enabled AP to support the communication between AP and SDN controller via OpenFlow protocol. The latter one works as the Local Information Base (LIB) that stores network state information, such as channel quality map, AP traffic load, etc. As one of the key prerequisites for achieving the proposed automatic channel switching application, the central controller needs to be aware of the traffic load of each physical AP and its best operating channel in the system. The custom protocol message FORWARD_AP_LOAD that is defined in chapter 3 is used to update the central controller with each AP's traffic load and a new custom protocol message called FORWARD_AP_CHAN is introduced to send the best operating channel of each AP to the central controller. The format of the proposed custom protocol message is shown in Fig.4.3. The headroom size is 28 bytes by default which contains extra information about the packet

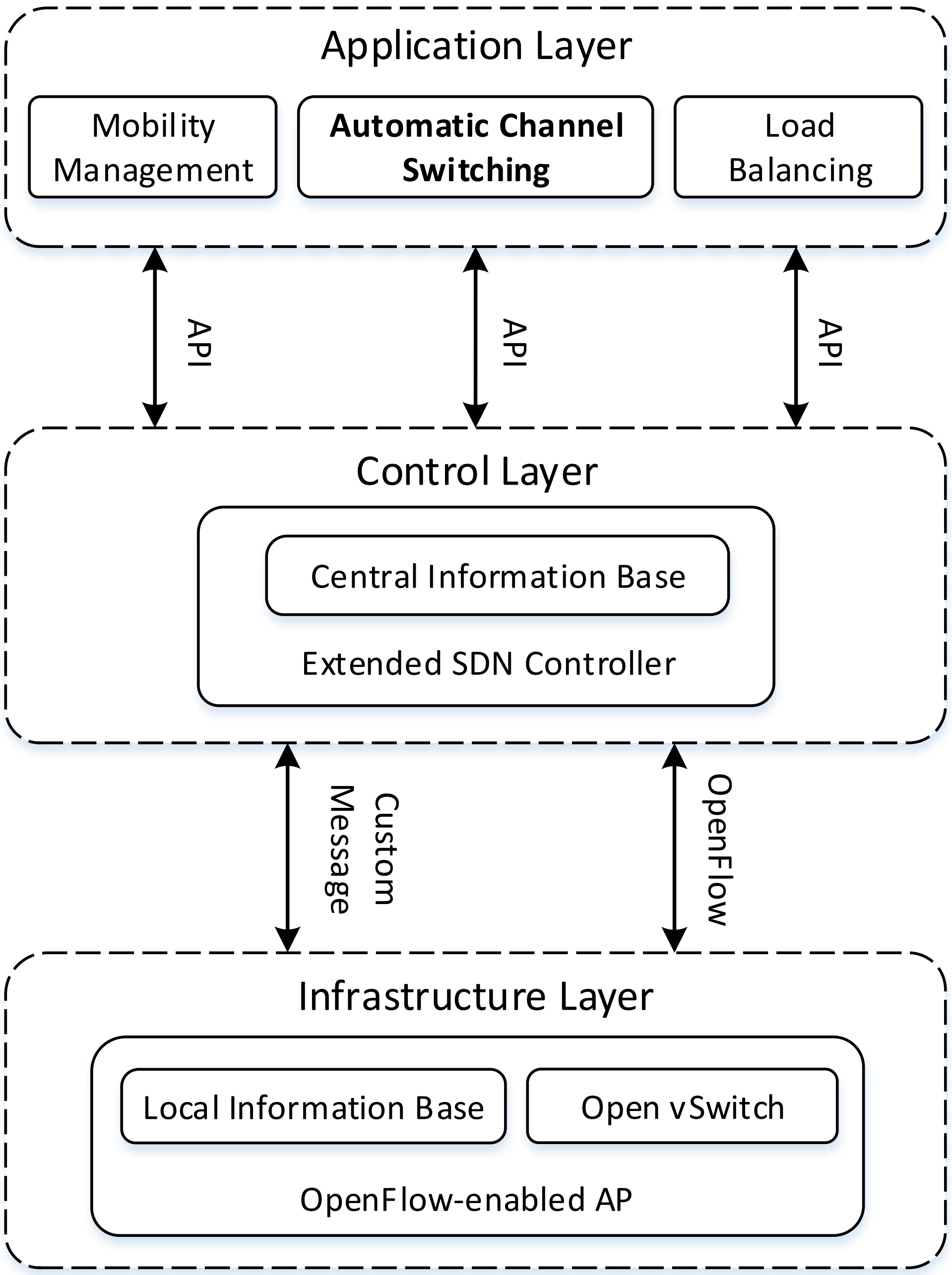


Fig. 4.2 Architecture of the designed software-defined enterprise WLAN system

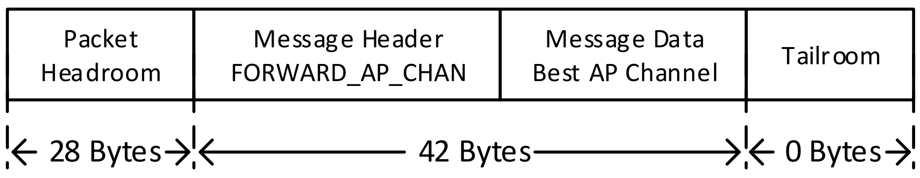


Fig. 4.3 Format of the custom protocol message FORWARD_AP_CHAN

like the destination address to be used for routing. The message header and the AP load value are stored as a message string in the packet content with a total size of 42 bytes. The tailroom is configured to be 0 bytes as it is not used in this case. By introducing these two messages, the updated traffic load of each physical AP and their best operating channel information can be forwarded to the central controller periodically. LIB running on physical AP stores updated AP load and channel quality information. It periodically scans all available Wi-Fi channels and constructs a channel information map, which contains the signal strength value of each AP and the channel utilisation value.

4.2.2 Central Information Base

Floodlight is used as the SDN controller which is operated in the control layer. It is extended with an add-on module called the Central Information Management (CIB) to maintain a global view of the state information of the entire network. CIB stores AP load and the best operating channel of each AP in hash maps matched by their IP addresses. Both values are updated periodically by receiving custom protocol messages sent from LIB. To make load aware automatic channel switching, the proposed application first acquires AP load from CIB by custom API. In case that the channel switching is needed, the application then retrieves the best operating channel from CIB and sends the channel switching message to CIB through custom APIs. Finally, CIB forwards a custom protocol message called Chan_Switch which contains the best operating channel to LIB to perform automatic channel switching.

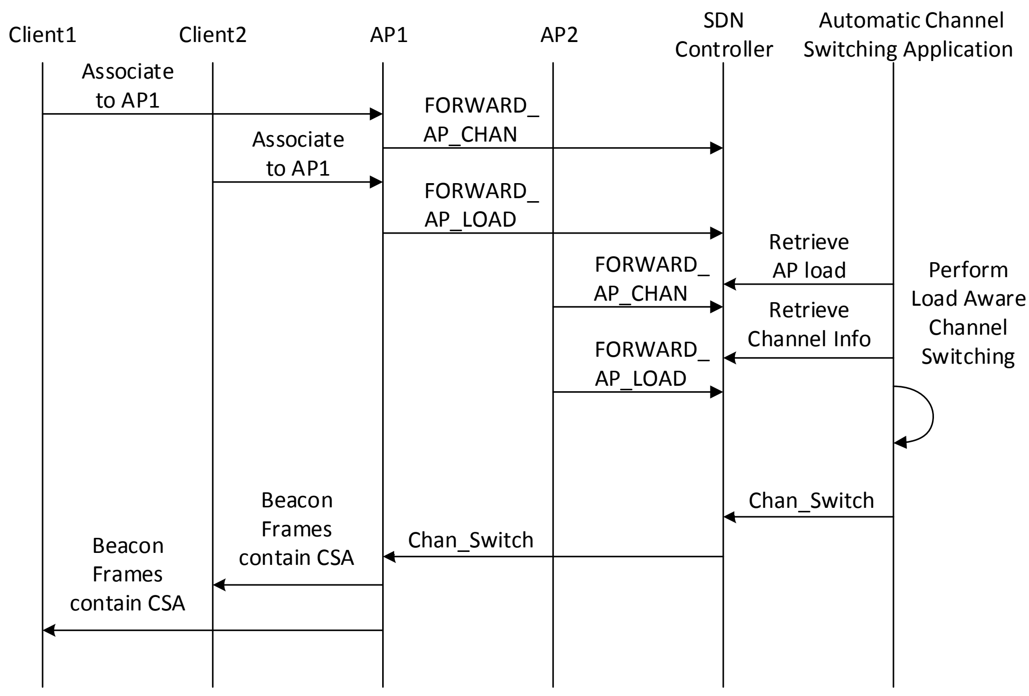


Fig. 4.4 An example of custom protocol messages that are exchanged during the process of automatic channel switching in the designed software-defined enterprise WLAN system

4.2.3 Automatic Channel Switching Application

The proposed automatic channel switching application is running in the application layer. It is implemented on top of the Floodlight controller to achieve centralised channel switching management. The application operates in a proactive way by querying the network periodically for the AP load condition and the channel information. It then makes channel switching decision based on this information. Fig.4.4 illustrates an example of the automatic channel switching process and relevant protocol messages that are exchanged in the designed software-defined enterprise WLANs architecture. As it can be seen, all physical APs in the system keep forwarding the FORWARD_AP_LOAD and FORWARD_AP_CHAN protocol messages to the central controller to update their traffic load and best operating channel information. In case that the proposed automatic channel switching application decides to make channel switching on the AP, it sends the Chan_Switch message through the CIB to the Click-based LIB to switch the operating channel of the AP. Beacon frames with CSA element are broadcast to all connected users before the AP switches its operating channel to achieve seamless channel switching, which is critical to guarantee satisfied QoS and user QoE.

4.3 Proposed Channel Switching Algorithms

To enable load-aware channel switching, AP load that is defined in chapter 3 is used as one of the channel switching metrics. The ath9k based Wi-Fi driver is modified to expose two debugfs based userspace interfaces which enable the LIB to read the value of both registers AR_RCCNT (0x80f4) and AR_CCCNT (0x80f8) that stores the number of time slots that are sensed busy by the CCA mechanism and the total number of time slots that have elapsed respectively. After reading the value of the registers, the LIB calculate the value of AP load and forward it to the central controller by using the protocol message FORWARD_AP_LOAD.

4.3.1 Channel Interference Factor

In this work, the author defines the Channel Interference Factor (CIF) as an approximation of the channel quality. To calculate the CIF, the signal to noise ratio (SNR) have to be considered. SNR is defined as the ratio of the power of the received signal and the background noise level (noise floor) with equation (4.1). Based on the test results, the measured noise floor for APs in our system is the same. Therefore, the received signal strength value (RSSI) is used as one of the factors for the calculation of the CIF. Because channel overlapping is a major contributor to the degraded performance of a Wi-Fi network where significant channel overlap can cause networks even with high received signal strengths to perform poorly, the co and adjacent channel interference have to be considered as another factor when calculating the CIF. To estimate the interference level of channel overlapping, the author defines a weight factor w

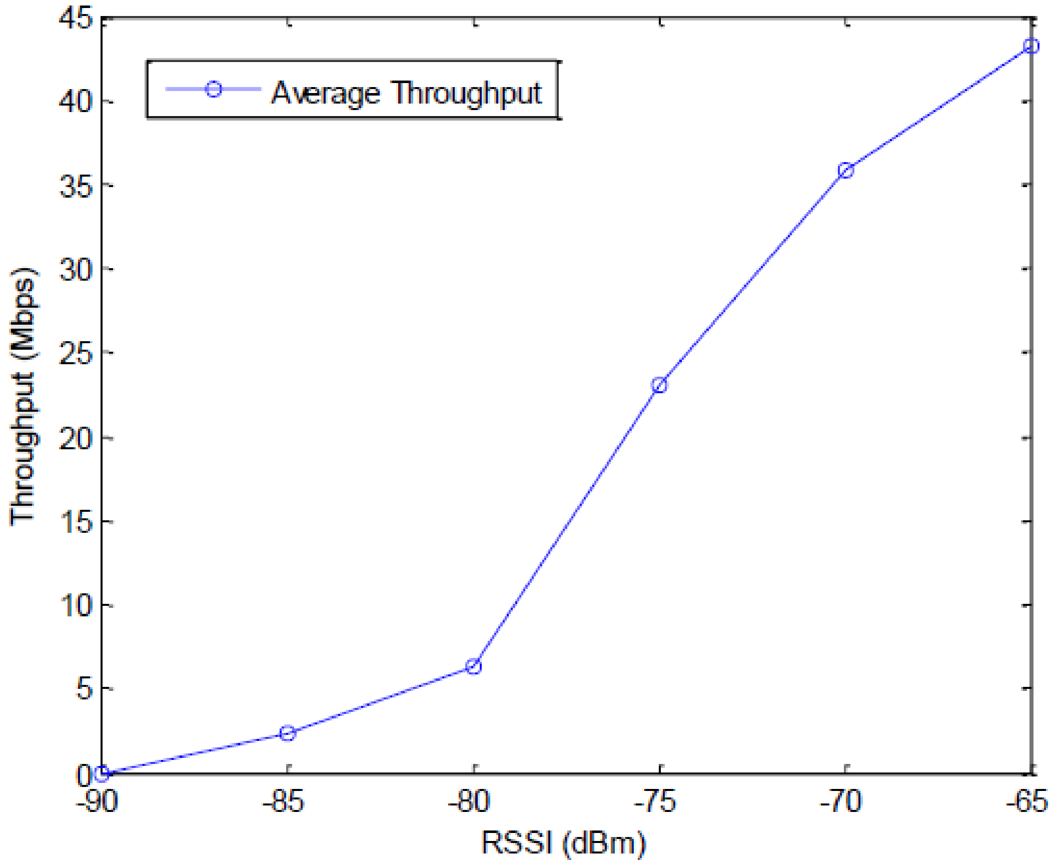


Fig. 4.5 Average network throughput achieved with different levels of RSSI

where its value is decided by the RSSI of the AP that operated on the overlapped channel. In order to find out the effect of RSSI on network throughput, the author conducts a simple experiment by associating one user to AP1 to test the throughput under the different level of RSSI. Experimental result is shown in Fig.4.5. As it can be observed, network throughput is decreased slightly if the RSSI is lower than -70 dBm and it is further degraded significantly if the RSSI is lower than -80 dBm. Based on the experimental results, the author defines three level of w which is high with RSSI higher than -69dBm, medium with RSSI between -70dBm to -79dBm and low with RSSI lower than -80dBm. The value of w is defined as 0.9, 0.6 and 0.3 corresponding to the RSSI level from high to low as the higher RSSI of the overlapped AP, the heavier interference that it can cause.

$$SNR_{dB} = 10\log_{10}(\tfrac{P_{signal}}{P_{noise}}) \tag{4.1}$$

$$CIF_i = \tfrac{CO_i + AD_j}{N_{ap}} \tag{4.2}$$

$$CO_i = CU_i \times \sum_{x=1}^{n} |RSSI_x| \tag{4.3}$$

$$AD_j = \sum_{c=1}^{n} (CU_c \times \sum_{x=1}^{n} (|RSSI_x| \times w_x)) \tag{4.4}$$

$$CIF_j = 0.9 \times CIF_j + 0.1 \times CIF_{j-1} \tag{4.5}$$

The value of the CIF is calculated by equation (4.2) where i is the channel number and N_{ap} is the total number of APs that operate on channels that can cause inference to channel i. CO_i is the absolute value of the co-channel interference of channel i which is achieved by equation (4.3) where CU_i is the channel utilisation of channel i. AD_j which is denoted by equation (4.4) is the total adjacent channel inference of APs that operated on channels that can cause adjacent channel interference to channel i. w_x is the weight factor of the AP that is overlapped with channel i, and its value is based on the RSSI of the AP. Because only the three non-overlapping channels are considered for proposed channel switching mechanism, the value of i can only be 1, 6 and 11. In order to prevent the CIF from the sudden variation of RSSI, an EWMA is used to calculate the final value of the CIF, which is denoted by equation (4.5). 90% weight is given to the current CIF_f. After calculating the value of CIF, the LIB select the channel with the maximum value of CIF as the best operating channel for automatic channel switching and send it to the central controller by using the protocol message FORWARD_AP_CHAN.

4.3.2 Proposed SS and DS Algorithms

By utilising AP and CIF as the channel switching metric, two load-aware automatic channel switching algorithms are proposed which are the Single Switch (SS) and the Double Switch (DS). The pseudo code of both algorithms is shown in Algorithm 3 and 4 respectively. Both algorithm work in a proactive way every pre-configured time interval $Timer_i$. However, the SS algorithm aims to improve overall user experience while the DS algorithm focuses on guaranteeing the performance of users with highest traffic load.

With the SS algorithm, the automatic channel switching application first retrieves the AP load map from the CIB and then sorts the load value from max to min to get a new AP load map $LoadMAP_{ap}$. Then it iterates $LoadMAP_{ap}$ to get $load_{ap}$ which is the load value of each AP and check if any AP need to switch its operating channel. If the $load_{ap}$ is higher than a pre-configured threshold value $load_t$, it then retrieves the best operating channel CH_{best} of this AP from the CIB and performs automatic channel switching if CH_{best} is not equal to the APs current operating channel $CH_{current}$. After automatic channel switching, AP, current channel information is updated with new channel CH_{best}. The author configures $load_t$ with

Algorithm 3 Single Switch Algorithm

```
 1: while true do
 2:     Sleep $Timer_i$
 3:     Sort $LoadMAP_{ap}$ by $load_{max}$
 4:     for $load_{ap} \in LoadMAP_{ap}$ do
 5:         Get $load_{ap}$
 6:         if $(load_{ap} > load_t)$ then
 7:             Get $CH_{best}$ of AP with $load_{ap}$
 8:             Get $CH_{current}$ of AP with $load_{ap}$
 9:             if $(CH_{best} \neq CH_{current})$ then
10:                 ChanSwitch($CH_{best}$)
11:                 ChanUpdate($CH_{best}$)
12:             else
13:                 AP remain at $CH_{current}$
14:             end if
15:         end if
16:     end for
17: end while
```

the value of 0.8 as with at least one associated user, which is generating traffic, the AP load value is higher than 0.8 based on our experiment test. No channel switching is needed for AP without associated users, and AP with the maximum load value has the highest priority for channel switching. Because finding the optimal channel assignment is known to be NP-hard, SS algorithm should achieve the close-to-optimal performance in the case of the channel load condition varies significantly. However, if the channel load condition is almost the same, SS algorithm may not be able to improve user QoE. Also, it may affect the experience of some users, but the overall network performance should be improved.

In contrast to the SS algorithm, the DS algorithm does not check each AP but only the AP with highest load value. With the DS algorithm, the automatic channel switching application first iterates the AP load map which is stored on CIB to get the AP with highest load value $load_{max}$. The application is triggered in case that $load_{max}$ is higher than the pre-defined threshold value $load_t$. It then retrieves the best operating channel $CH_{MAXbest}$ and the current operating channel $CH_{MAXcurrent}$ of the AP from the CIB. If the AP is not operating on its best channel, the ACS application iterates the AP load map to check the AP

Algorithm 4 Double Switch Algorithm

```
1: while true do
2:     Sleep $Timer_i$
3:     Sort $LoadMAP_{ap}$ by $load_{max}$
4:     Get $load_{max}$ from $LoadMAP_{ap}$
5:     if ($load_{max} > load_t$) then
6:         Get $CH_{MAXbest}$ of AP with $load_{max}$
7:         Get $CH_{MAXcurrent}$ of AP with $load_{max}$
8:         if ($CH_{MAXbest} \neq CH_{MAXcurrent}$) then
9:             for $load_{ap} \in LoadMAP_{ap}$ do
10:                Get $CH_{APcurrent}$ of AP with $load_{ap}$
11:                if ($CH_{APcurrent} = CH_{MAXbest}$) then
12:                    Break loop
13:                end if
14:            end for
15:            ChanSwitch($CH_{MAXcurrent}$) for AP $load_{ap}$
16:            ChanUpdate($CH_{MAXcurrent}$) for AP $load_{ap}$
17:            ChanSwitch($CH_{MAXbest}$) for AP $load_{max}$
18:            ChanUpdate($CH_{MAXbest}$) for AP $load_{max}$
19:        else
20:            AP remain at $CH_{MAXcurrent}$
21:        end if
22:    end if
23: end while
```

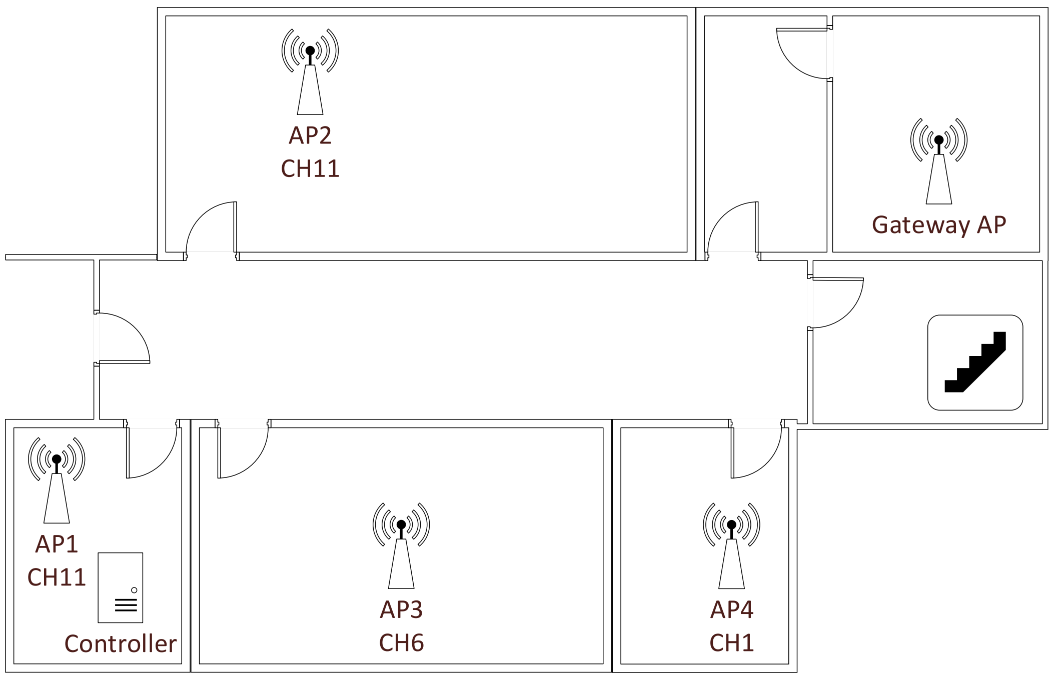


Fig. 4.6 Initial channel configurations of experiment APs

with next highest load value with its operating channel $CH_{APcurrent}$ equal to $CH_{MAXbest}$. It then switches the AP with $load_{max}$ to its best channel $CH_{MAXbest}$ to improve the experience of users with highest data traffic demanding. At the same time, the AP with highest load value $load_{ap}$ on channel $CH_{MAXbest}$ is switched to channel $CH_{MAXcurrent}$ to reduce the possible co-channel interference that further improve the experience of users with highest traffic load. Theoretically, DS algorithm should give a close-to-optimal channel switching performance, which guarantees QoE of the user group with highest traffic load. However, as a trade-off, it will sacrifice the experience of users associated to the AP that is going to switch to a busy channel.

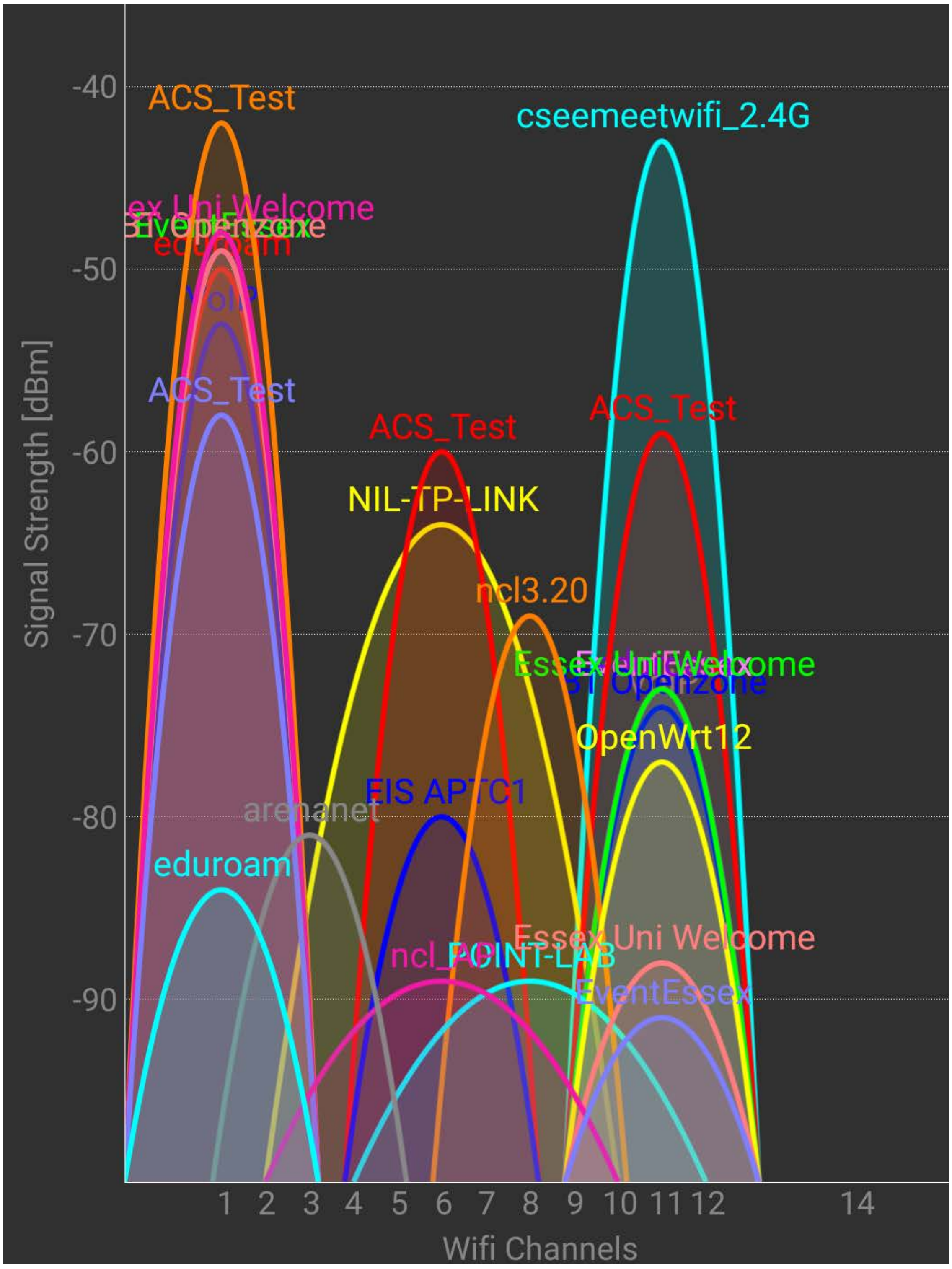


Fig. 4.7 The 2.4GHz band channel information of experiment environment

4.4 Performance Evaluation

4.4.1 Experiment Setup

To implement the designed software-defined enterprise WLAN system, the testbed that is deployed in chapter 3 is used. The distance between each APs is less than 15 meters which make the test environment heavily overlapped. The initial channel configuration of the experiment APs are depicted in Fig.4.6. The proposed load-aware channel switching algorithms are implemented on top of the Floodlight controller, which is running on a physical PC. A normal layer 2 switch is used to connect the default gateway AP and four experiment APs. The Wi-Fi function of the default gateway AP is disabled to reduce channel interference, while the DHCP server is enabled to assign IP address for experiment APs and user devices. Fig.4.7 shows an example of the 2.4GHz Wi-Fi channel usage information of the experiment environment, which is scanned by the Android application called WiFi Analyzer. As it can be seen, the 2.4GHz band is quite crowded with many APs operating on the same channel or channels which are overlapped. APs with SSID ACS Test are the experiment APs. In order to eliminate authentication related delays, Wi-Fi authentication is disabled during all experiments. Table. 1 shows the information of experiment devices and initial configurations.

4.4.2 Performance Evaluation

Five experiments are conducted to investigate the performance of the proposed SS and DS automatic channel switching algorithms. TCP tests based on iperf are conducted to evaluate the average and total network throughput, while iperf-based UDP tests are conducted to measure the jitter and ping-based tests are used to investigate the performance of the proposed algorithms on transmission delay.

Table 4.1 List of software and hardware information for experiments

AP Model	TP-Link WR1043ND Ver 2.1
AP Firmware	Openwrt Chaos Calmer 15.01
Number of APs	5
IEEE 802.11 Standard	802.11n (20MHz)
Configured AP Channels	1, 6 and 11
User Operating System	Android 5.0 and Linux Ubuntu 14.04
Number of User Devices	4
User Traffic	iperf-based TCP and UDP session
SDN Controller	Floodlight 1.0
Integrated Software	Open vSwitch 2.3 and Click Modular Router 2.0.1

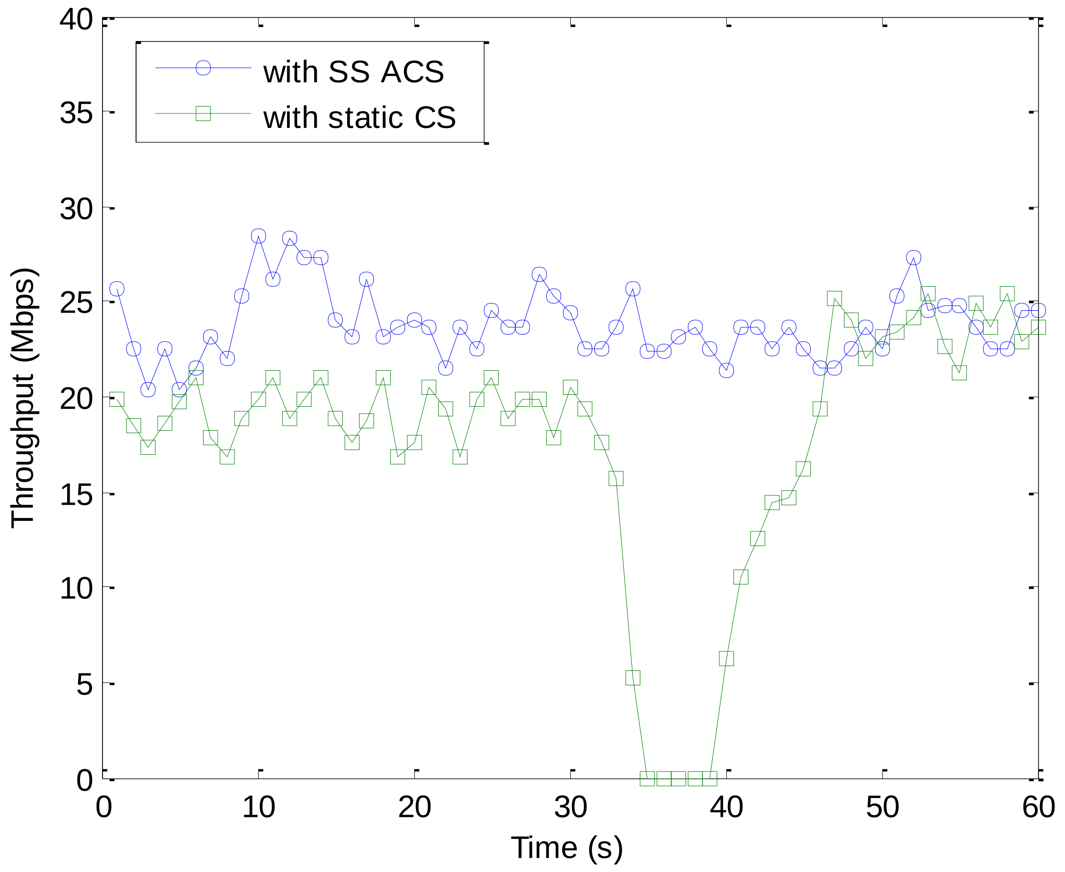


Fig. 4.8 Single user throughput SS vs static method

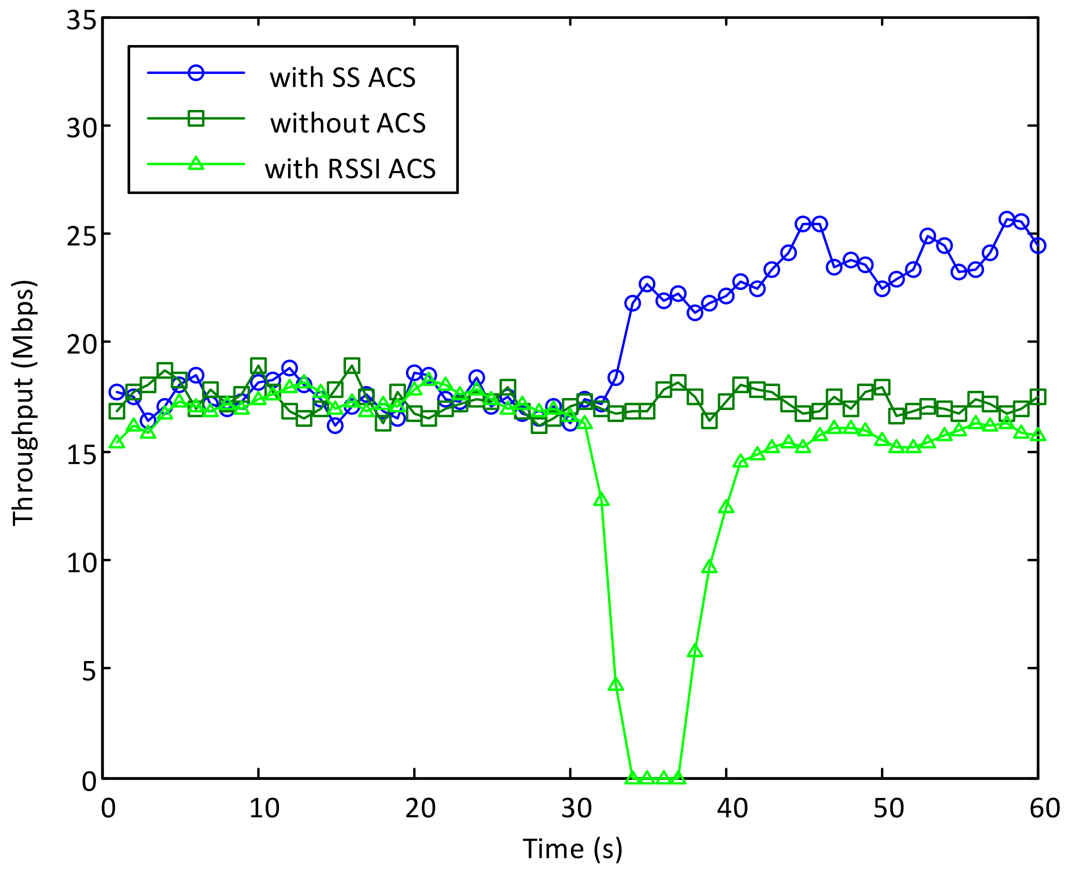


Fig. 4.9 Average user throughput comparison SS vs RSSI

Seamless Channel Switching

The first experiment is conducted to evaluate the performance of providing seamless channel switching with the proposed automatic channel switching algorithm. To set up this experiment, the iperf TCP server is running on AP1. Only one user device is associated with AP1 to perform iperf TCP test which lasts 60 seconds. During the experiment, AP1 perform channel switching for once, and user throughput is recorded in one-second interval. The same test runs ten times and results are averaged. Fig.4.8 shows the experimental results achieved with the SS algorithm and the static manual mechanism respectively. As it can be observed, user throughput is dropped to 0 for few seconds with static manual channel switching mechanism. This result is because the WLAN interface is required to reboot to switch to the new channel

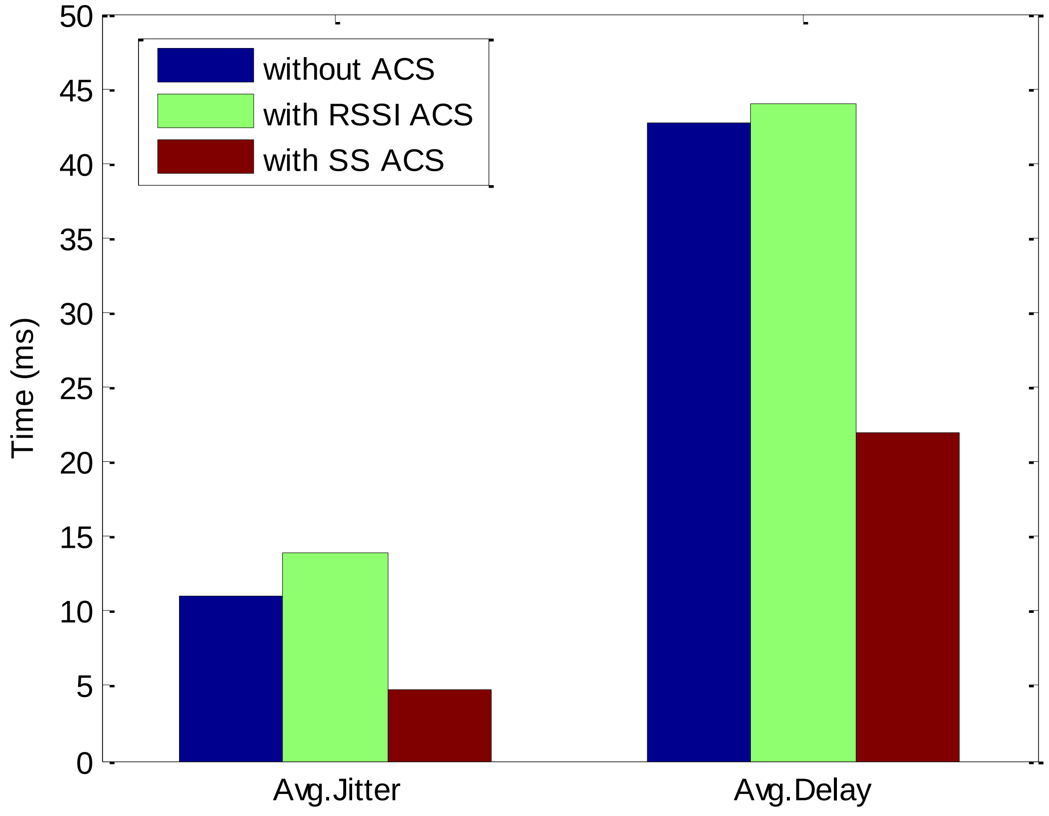


Fig. 4.10 Average jitter and delay comparison SS vs RSSI

frequency. Therefore, the associated user has to perform re-scan and re-association with AP1, which adds extra delays before the recovering of TCP connection. In contrast to the static manual channel switching mechanism, the proposed automatic channel switching application supports seamless channel switching, which maintains user association and makes no effects on user throughput during the process of channel switching. This mechanism is achieved by broadcasting beacon frames contain CSA elements to the associated user before channel switching. Thus, user devices that support CSA are able to switch channel without losing TCP connection.

Throughput Evaluation with SS algorithm

In the second experiment, the efficiency on improving average user throughput with the proposed SS ACS application is studied. Initially, four user devices are associated with four experiment APs respectively. All users run iperf TCP test with TCP server running on their associated APs. The experiment runs ten times, and each lasts 60 seconds. Network throughput of all users is collected in one-second interval and are averaged. Fig.4.9 illustrates the results achieved with SS and RSSI-based automatic channel switching algorithm. It can be seen that average user throughput is improved significantly after 30 seconds with SS automatic channel switching algorithm. This result is achieved because the application detects that AP1 is not operating on its best channel and decide to switch its current operating channel from channel 11 to channel 1that reduces the channel interference and improve the user experience. On the contrary, the average user throughput is decreased slightly after channel switching with the RSSI-based automatic channel switching algorithm, which selects the channel with the smallest maximum and average RSSI. However, the channel with the smallest maximum and average RSSI is not always the best channel as the channel load condition is not considered. In addition, this algorithm does not support seamless user connection during channel switching, which affects user QoE significantly.

Delay and Jitter Evaluation with SS Algorithm

The efficiency of improving jitter and transmission delay with the SS automatic channel switching algorithm is evaluated in the third experiment. To set up this experiment, four users are connected to four experiment APs respectively. To measure jitter, all users run iperf UDP test with UDP server running on their associated APs. The experiment runs ten times, and each lasts 60 seconds. The jitter of all users is collected in one-second interval and are averaged. The same test is conducted to measure transmission delay, but instead of using iperf, Ping is used. Fig.4.10 shows results of average jitter and delay achieved with

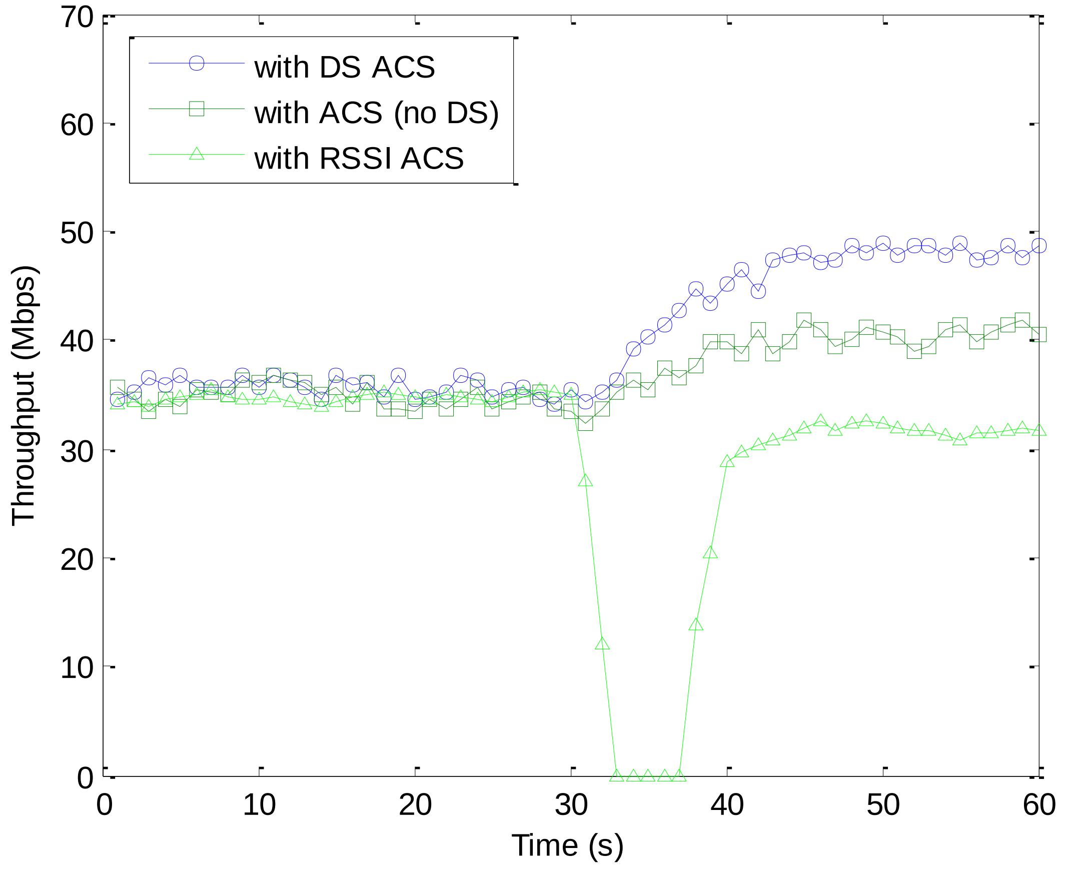


Fig. 4.11 Total user throughput comparison DS vs RSSI

SS and RSSI based automatic channel switching algorithm. It can be observed that both average jitter and delay are much higher with RSSI based channel switching algorithm due to the lack of channel load awareness and adaption. However, with SS automatic channel switching algorithm, the application performs seamless channel switching on AP1 that decreases channel interference on AP1 and AP2. As the result of channel switching, both jitter and delay are improved significantly.

Throughput Evaluation with DS Algorithm

The fourth experiment is conducted to evaluate the efficiency of improving total user throughput with the proposed DS automatic channel switching algorithm. Initially, two users are associated with AP1, and the other two users are associated with AP2 and AP4 respectively.

All users run iperf TCP test with TCP server running on their associated APs. The experiment runs ten times, and each lasts 60 seconds. Network throughput of two users that are connected to AP1 are collected, and averaged values are added to get the final results. Fig.4.11 illustrates the total user throughput achieved with DS algorithm, single switching algorithm, and RSSI based algorithm respectively. It can be seen that the total user throughput is decreased slightly with RSSI based algorithm due to the lack of channel load awareness and adaption. The result is increased slightly with single switching algorithm as AP1 performs load aware seamless channel switching. However, with DS automatic channel switching algorithm, the application performs a double switching by changing AP1 to channel 1 and AP4 to channel 11 to further reduce the possible channel interference from AP4. It can be observed that the total user throughput is increased significantly with DS algorithm when compared with single switch method.

Delay and Jitter Evaluation with DS Algorithm

In the fifth experiment, the efficiency of improving jitter and transmission delay with the proposed DS automatic channel switching algorithm is studied. To set up this experiment, two users are connected to AP1 and the other two users are associated with AP2 and AP4 respectively. All users run iperf UDP test with UDP server running on their associated APs. The experiment runs ten times, and each lasts 60 seconds. The jitter of two users that are connected to AP1 is added to achieve the final results. The same test is conducted to measure transmission delay but instead of running iperf UDP test, Ping test is conducted. The results of total jitter and delay achieved with DS algorithm, the single switching method, and the RSSI-based algorithm are shown in Fig.4.12. As it can be observed, both jitter and delay are much higher with RSSI based algorithm. This result is achieved because the DS automatic channel switching algorithm performs double channel switching on AP1 and AP4 to guarantee the experience of users with highest traffic load. As a result of this double

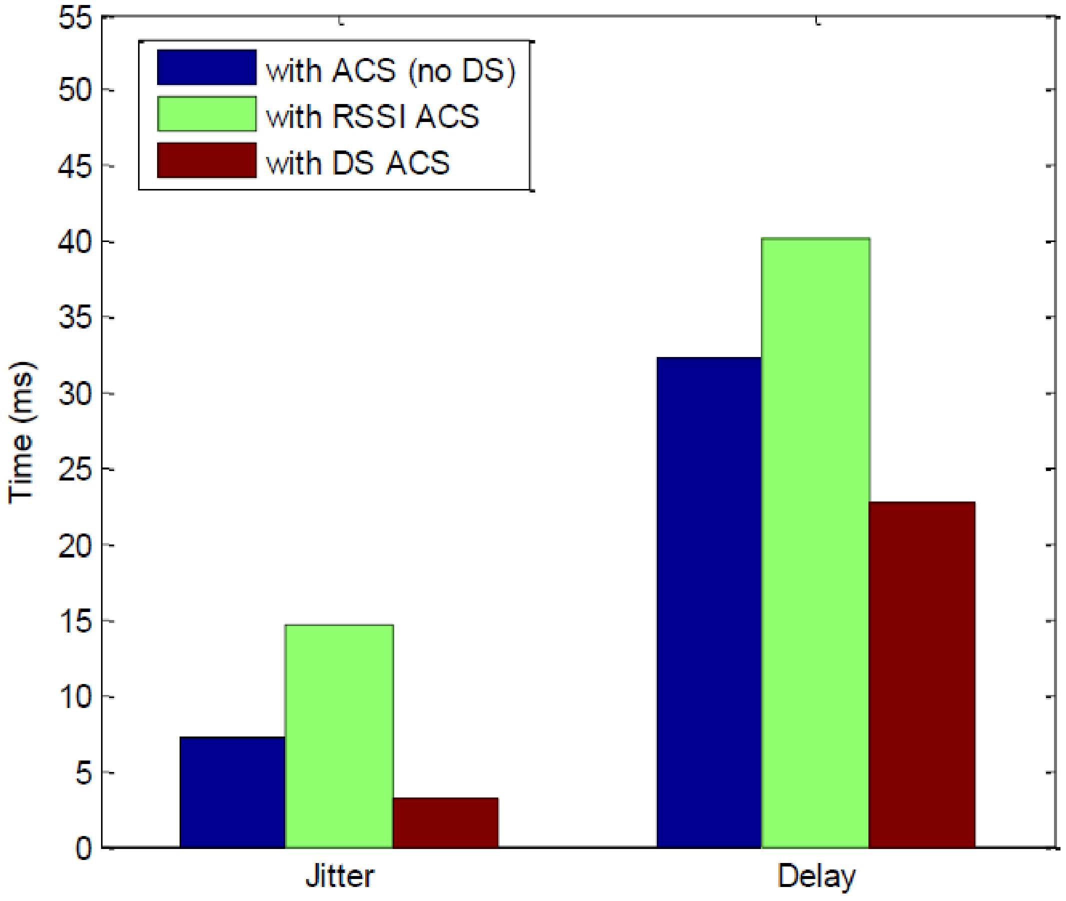


Fig. 4.12 Total jitter and delay comparison DS vs RSSI

channel switching, both jitter and delay are significantly decreased which is reflected in the experimental results.

4.5 Chapter Summary

This chapter investigates how to achieve intelligent network traffic control in enterprise WLANs from an AP-side perspective. Leveraging SDN to the existing enterprise WLAN architecture, a software-defined enterprise WLAN system, which enables centralised network management and highly flexible programmability is presented. Furthermore, two load-aware automatic channel switching algorithms are proposed and implemented on a physical testbed to improve the performance of channel switching control on the 2.4GHz frequency band.

By utilising AP load and CIF as channel switching metrics, the proposed SS and DS algorithms achieve efficient channel switching control, which adapts to real-time traffic load and channel condition and supports seamless handover of users over multiple wireless channels. Experimental results show that the proposed SS and DS channel switching algorithms can significantly improve QoS and user QoE by decreasing jitter and transmission delay and increasing network throughput compared to the conventional RSSI-based channel switching mechanism.

In the next chapter, the author further investigates how to improve network traffic control from an AP-side perspective by proposing a QoS-aware bandwidth control algorithm for adaptive bandwidth control. The proposed algorithm is implemented on the software-defined enterprise WLAN system that is designed in this chapter.

Chapter 5

QoS-Aware Bandwidth Control

5.1 Introduction

Over the past decades, IEEE 802.11 WLAN has become one of the most widely used wireless communication technologies for the enterprise because of its characteristics such as easy deployment, low cost and high data rate. With the growing popularity of BYOD, data traffic in modern enterprise WLANs is significantly increased that can cause network congestion. To guarantee QoS and user QoE, an efficient bandwidth control service is needed in modern enterprise WLANs. Conventional bandwidth control mechanism requires network administrators to manually configure on the AP to the allocate the different amount of bandwidth to users with different priorities. However, users may roam from one AP to another that makes the conventional solution inefficient. Therefore, the author argues that an efficient bandwidth control algorithm needs to be logically centralised and adapt to real-time network traffic. Although enterprise vendors have provided commercial solutions [11] [12] [13] to solve this WLAN-specific bandwidth control issue, these existing solutions require vendor proprietary software and hardware that are not able to be achieved with the low-cost commodity AP hardware.

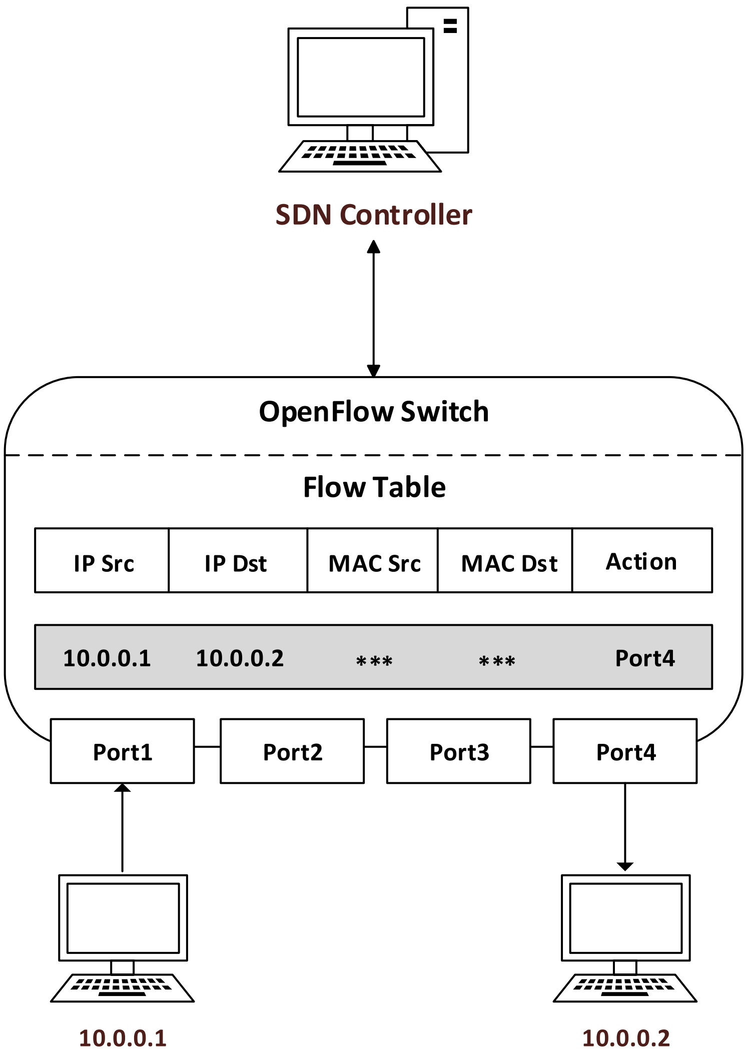


Fig. 5.1 An example of forwarding rules and flow table of an OpenFlow switch

SDN is a novel network architecture that has the great potential to simplify the existing complex and inflexible network infrastructure, making it an ideal solution for the modern enterprise WLANs deployment. SDN architecture decouples the network control and forwarding functions that enables the network management to be logically centralised and directly programmable and the underlying forwarding infrastructure is abstracted for network applications. OpenFlow is the first communication protocol defined in SDN architecture. It is a key element which enables the SDN controller to directly interact with the forwarding devices such as switches and routers. The architecture of an OpenFlow network is centralised

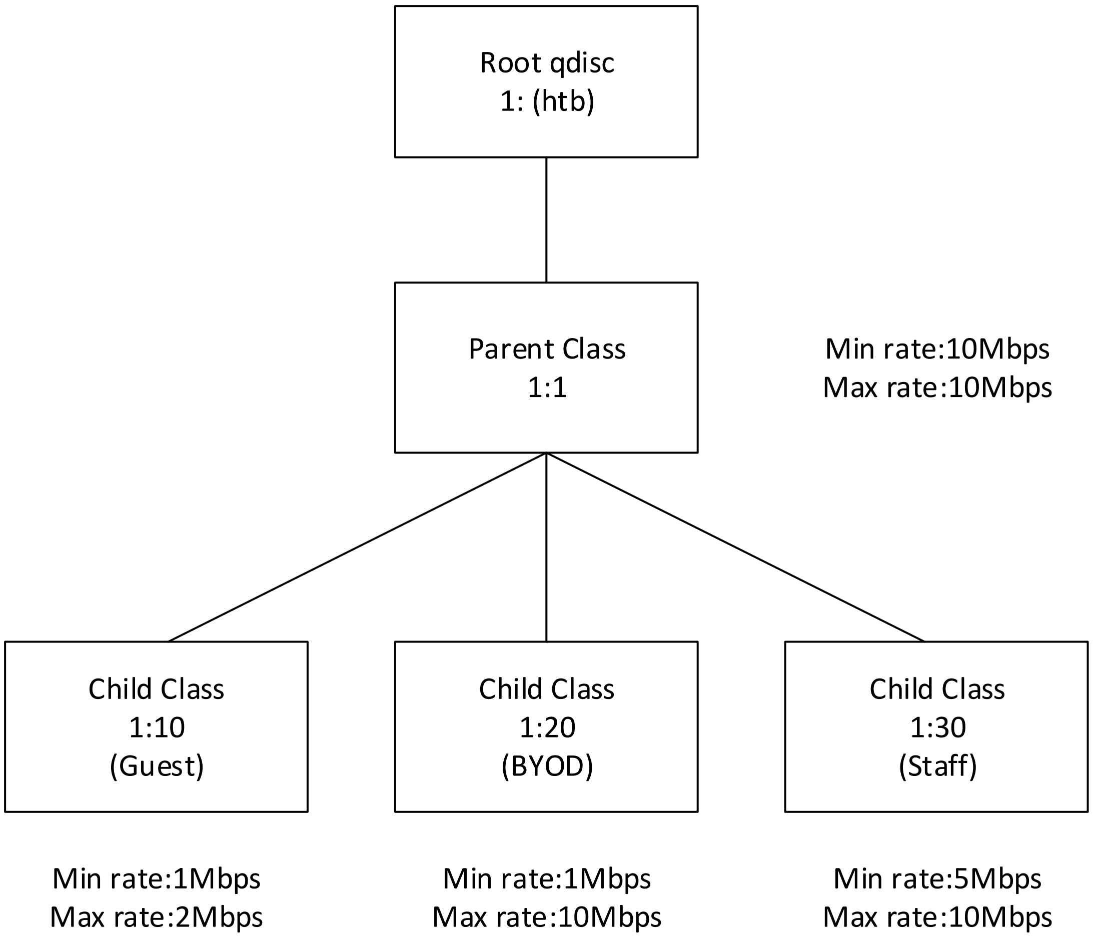


Fig. 5.2 An example of the conventional bandwidth control queuing architecture

where a central controller manages the packet forwarding rules of the data plane. To define the forwarding rules, the central controller configures flow entries in the flow table of the OpenFlow switch. Fig.5.1 shows an example of forwarding rules and flow table of an OpenFlow switch. It can be seen that a flow entry defines a forwarding rule that if the packet's source IP address is 10.0.0.1 and its destination IP address is 10.0.0.2, send the packet to port 4 of the switch. However, the terminal with IP address 10.0.0.2 cannot forward packets to the terminal with IP address 10.0.0.1 as there are no proper forwarding rules defined in the flow table. In this case, the switch has to send requests to the central controller for the corresponding flow entries. By configuring the flow table of the OpenFlow switch, the central controller can change the forwarding behaviours flexibly. By utilising this characteristic, a software-defined enterprise WLAN can achieve dynamic bandwidth control on users with different priorities.

5.1.1 Problem Statement

APs with OpenWrt firmware support conventional bandwidth control mechanism which is based on the Hierarchical Token Bucket (HTB) queuing discipline (qdisc). Fig.5.2 shows an example of the conventional bandwidth control queuing architecture. In this architecture, the parent class specifies the minimum guaranteed bandwidth and the maximum data bandwidth of a network interface. All the child classes are inherited from the parent class with different amount of guaranteed bandwidth. User groups with different priorities are assigned to corresponding classes to achieve dynamic bandwidth control.

Although the HTB qdisc based mechanism support dynamic bandwidth control, it still requires manual configuration by network administrators and it is not adapt to real-time network traffic load. Because network traffic varies with the fast-changing users'association state, the conventional bandwidth control mechanism is inefficient and can not guarantee QoS and QoE for high priority users. Therefore, it's vital for the bandwidth control algorithm to be logically centralised and have the knowledge of AP traffic load in order to perform efficient bandwidth control.

5.1.2 Contributions

In this chapter, the author presents a QoS-aware bandwidth control algorithm and implement the proposed algorithm on the software-defined enterprise WLAN system that is designed in Chapter 4. Leveraging the characteristics of SDN, the proposed QoS-aware bandwidth control algorithm runs on top of the SDN controller to maintain a centralised view of the real-time network traffic. By utilising AP load which is defined in Chapter 3 as the bandwidth control metric, the proposed algorithm achieves QoS-aware uplink bandwidth control on users with different priorities. The author evaluates the efficiency of the proposed algorithm by conducting five experiments based on iperf TCP and UDP tests. Results show that the proposed algorithm can achieve efficient bandwidth control on low priority users and improve the QoS and QoE for users with high priority.

5.1.3 Structure of Chapter

The remainder of this chapter is organised as follows. Section II further describes the working principle of the implemented QoS-aware bandwidth control application. Section III presents the proposed QoS-aware bandwidth control algorithm. The experiment setup and performance evaluation are demonstrated in Section IV before the chapter summaries at Section V.

5.2 System Introduction

To achieve efficient bandwidth control, a centralised view of network traffic state is essential. By introducing SDN to the existing enterprise WLAN architecture, centralised bandwidth control can be easily implemented. The proposed QoS-aware bandwidth control algorithm is implemented on the software-defined enterprise WLAN system that is designed in Chapter 4. The three-layer architecture of the updated system is depicted in Fig.5.3. In the SDN infrastructure layer, physical APs with Linux-based OpenWrt firmware are used as the forwarding device. Open vSwitch is installed to turn the commodity AP into OpenFlow-enabled AP. Click-based LIB runs on the physical APs to update the central controller with each AP's traffic load by using the custom protocol message FORWARD_AP_LOAD, which is defined in Chapter 3. In the SDN control layer, Floodlight is used as the SDN controller. It runs the add-on module CIB to enable the communication with the LIB and

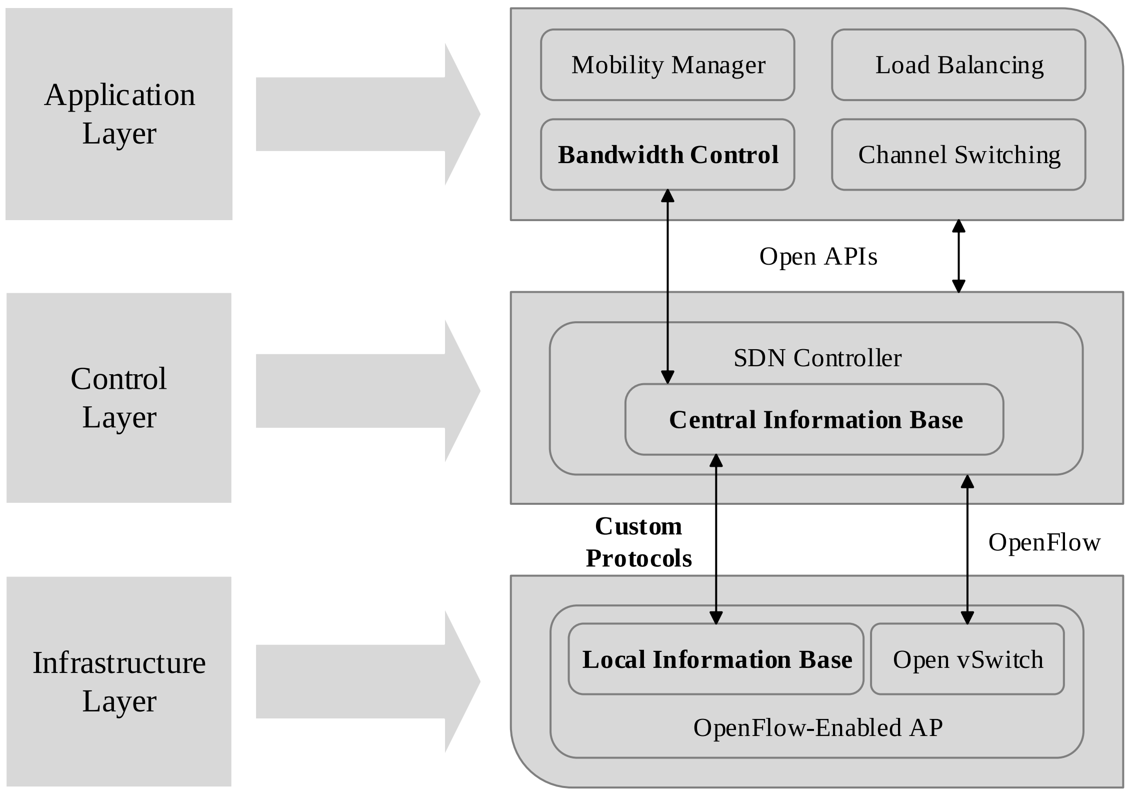


Fig. 5.3 The proposed QoS-aware bandwidth control algorithm is implemented in the SDN application layer

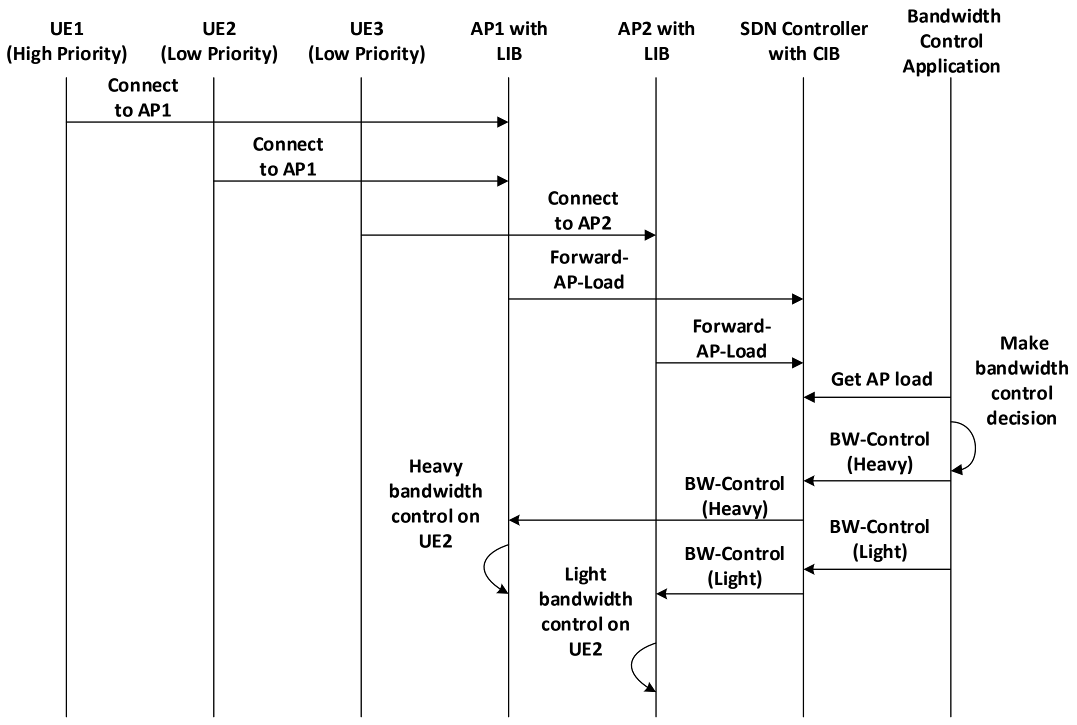
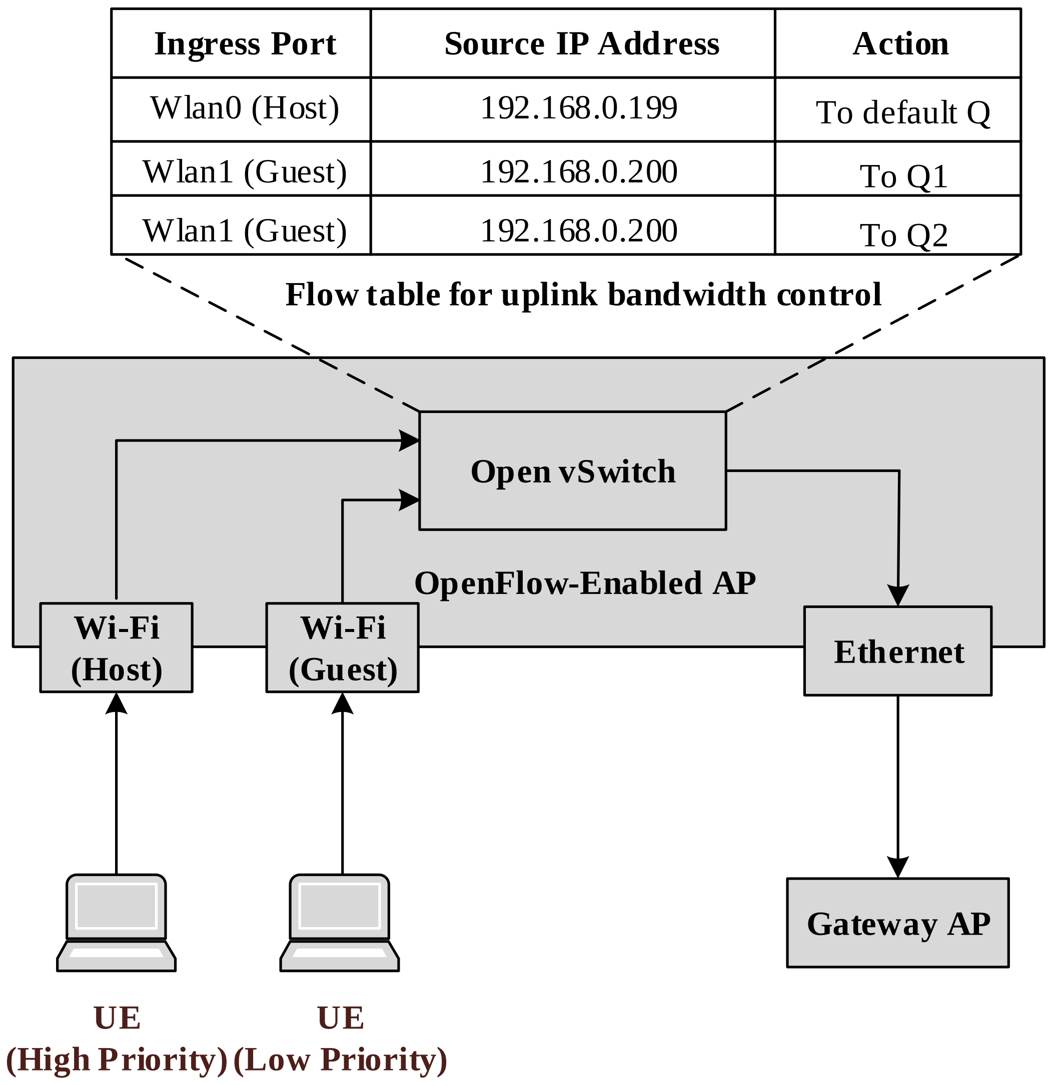


Fig. 5.4 Sequence diagram of protocol messages exchanged with the proposed QoS-aware bandwidth control algorithm

the implemented bandwidth control algorithm. After receiving the custom protocol message FORWARD_AP_LOAD, CIB stores and updates the traffic load of each AP in a hash map matched by their IP addresses. In the SDN application layer, the proposed QoS-aware bandwidth control algorithm is implemented as a WLAN-specific application to achieve centralised uplink bandwidth control. The application operates in a proactive way by waking up every pre-defined time.

5.2.1 QoS-Aware Bandwidth Control

To perform QoS-aware bandwidth control, the proposed algorithm first check the traffic load condition of each AP by acquiring the AP load from CIB through custom API. After that, the proposed algorithm chooses the corresponding level of bandwidth control based on the AP load information and sends the bandwidth control message to CIB through custom



Ingress Port	Source IP Address	Action
Wlan0 (Host)	192.168.0.199	To default Q
Wlan1 (Guest)	192.168.0.200	To Q1
Wlan1 (Guest)	192.168.0.200	To Q2

Fig. 5.5 Traffic flow for uplink bandwidth control

API. Finally, CIB forwards a custom protocol message called Band_Control to LIB that runs on the AP to perform different levels of bandwidth control on users with low priority. Fig.5.4 illustrates an example of the process of the proposed QoS-aware bandwidth control and related custom protocol messages that are exchanged between the data and control plane. Fig.5.5 shows an example of the traffic flow and flow table for uplink bandwidth control. Packets of uplink traffic are generated by User Equipments (UEs) with different priorities. When receiving data packets from Wi-Fi ports, OpenWrt directs them to the virtual OpenFlow switch Open vSwitch. According to pre-defined flow actions, OpenFlow protocol forwards data packets to proper queues before these packets are sent to the gateway AP. In this work, two queues with different maximum data rate are created to enable various levels of bandwidth control on users with low priority. Data packets generated from high priority users are forwarded to the default queue which has no bandwidth limitation.

Algorithm 5 Load-Aware Adaptive Bandwidth Control Algorithm

```
1: while true do
2:     Sleep $Timer_i$
3:     Sort $LoadMAP_{ap}$ by $load_{max}$
4:     for $load_{ap} \in LoadMAP_{ap}$ do
5:         Get $load_{ap}$
6:         if ($load_{ap} > load_t$) then
7:             Heavy bandwidth control on low priority users
8:             Get IP address $LoadMAP_{ap}$.getKey($load_{ap}$)
9:             BWControlHeavy($LoadMAP_{ap}$.getKey($load_{ap}$))
10:        else
11:            Light bandwidth control on low priority Users
12:            Get IP address $LoadMAP_{ap}$.getKey($load_{ap}$)
13:            BWControlLight($LoadMAP_{ap}$.getKey($load_{ap}$))
14:        end if
15:    end for
16: end while
```

5.3 Proposed QoS-Aware Bandwidth Control Algorithm

By utilising AP load which is defined in Chapter 3 as the bandwidth control metric, a QoS-aware bandwidth control algorithm is proposed. The pseudo code of the proposed algorithm is shown in Algorithm 5. With every pre-defined time interval $Time_i$, the adaptive bandwidth control application starts to working by acquiring the AP load map from the CIB and then sort the AP load value in descending order to get a new AP load map $LoadMAP_{ap}$. Then it iterates $LoadMAP_{ap}$ to get $load_{ap}$ which is the load value of each AP and check if any AP need to perform bandwidth control. If the $load_{ap}$ is higher than a pre-configured threshold value $load_t$, the application performs heavy bandwidth control on low priority users by sending the bandwidth control message to related AP. After receiving the bandwidth control message, the AP uses OpenFlow to direct the data traffic of low priority users to the related queue that is created via Open vSwitch. If the $load_{ap}$ is lower than $load_t$, the application performs light bandwidth control to allocate a higher amount of bandwidth to low priority users compare to the heavy bandwidth control. In this algorithm, the author configures the $load_t$ with value of 1 because the AP load value is always smaller than one if the AP only have one associated user that is generating traffic and is bigger than one if there are at least two associated users that are generating traffic. Based on the value of AP load, the application detects the traffic load on the AP as idle or busy and performs light and heavy bandwidth control accordingly to achieve efficient bandwidth allocation on users with different priorities.

5.4 Performance Evaluation

5.4.1 Experiment Setup

To set up the experiment, the testbed that is deployed in chapter 3 is used (i.e. the testbed shown in Fig3.8). The proposed QoS-aware bandwidth control algorithm is implemented on top of the Floodlight controller, which is running on a physical PC. The gateway AP runs the

Table 5.1 List of software and hardware information for experiments

AP Model	TP-Link WR1043ND Ver 2.1
AP Firmware	OpenWrt Chaos Calmer 15.01
Number of APs	5
IEEE 802.11 Standard	802.11n (20MHz)
AP Channels	1, 6 and 11
AP SSIDs	Host (high priority) and Guest (low priority)
User Operating System	Android 5.0 and Linux Ubuntu 14.04
Number of UEs	2
User Traffic	iperf-based TCP and UDP Tests
SDN Controller	Floodlight 1.0

DHCP server, which assigns IP addresses to OpenFlow-enabled APs and associated UEs. Two Service Set Identifier (SSID) named Host and Guest are created for UEs with high and low priorities respectively. Table. 1 shows the information of experiment devices and initial configurations.

5.4.2 Performance Evaluation

Five experiments are conducted to investigate the performance of the proposed QoS-aware bandwidth control algorithm. Iperf-based TCP and UDP tests are conducted to evaluate the network throughput and packet loss rate of both high and low priority users.

Light Bandwidth Control

The first experiment is conducted to evaluate the performance of the proposed light bandwidth control mechanism on limiting network throughput of the low priority user. To set up this experiment, the iperf TCP server is running on AP1. Only one low priority user is associated to AP1 through the guest Wi-Fi to perform the iperf TCP test which lasts 60 seconds. User

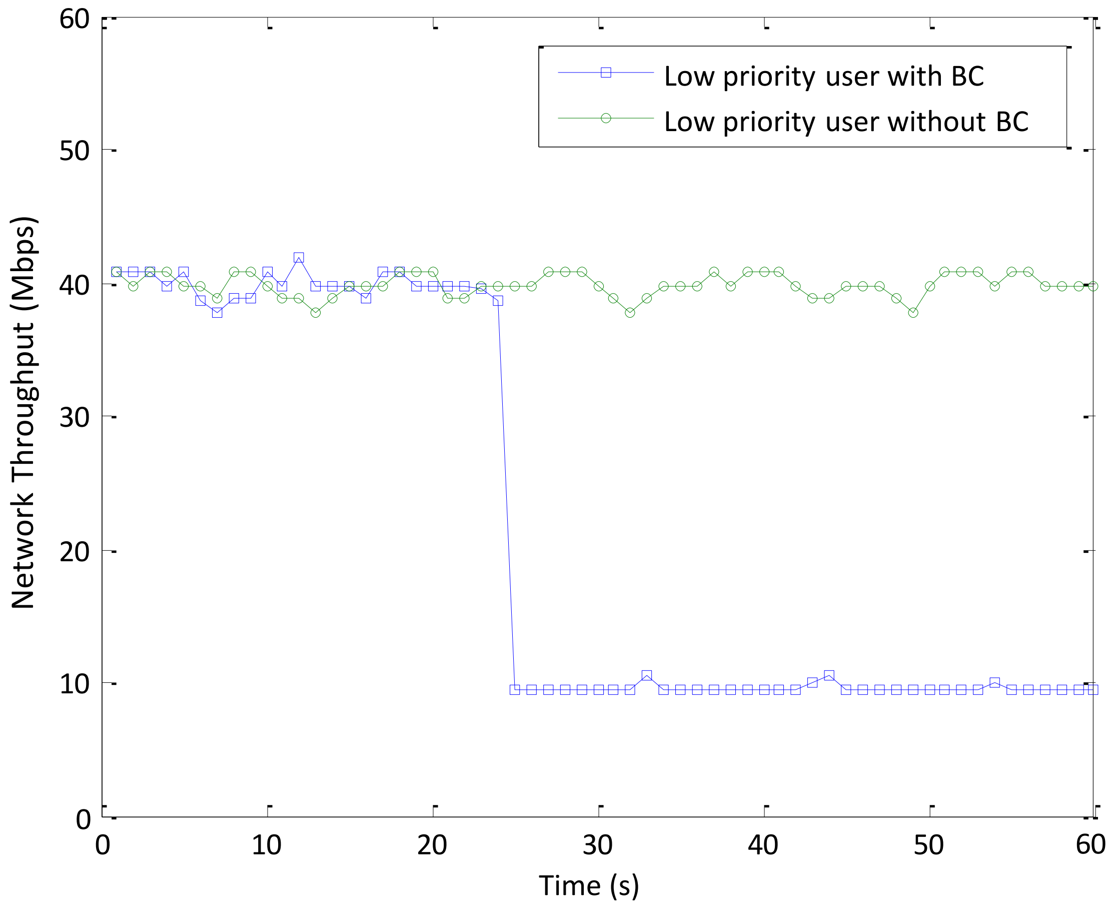


Fig. 5.6 Network throughput achieved with the proposed light bandwidth control

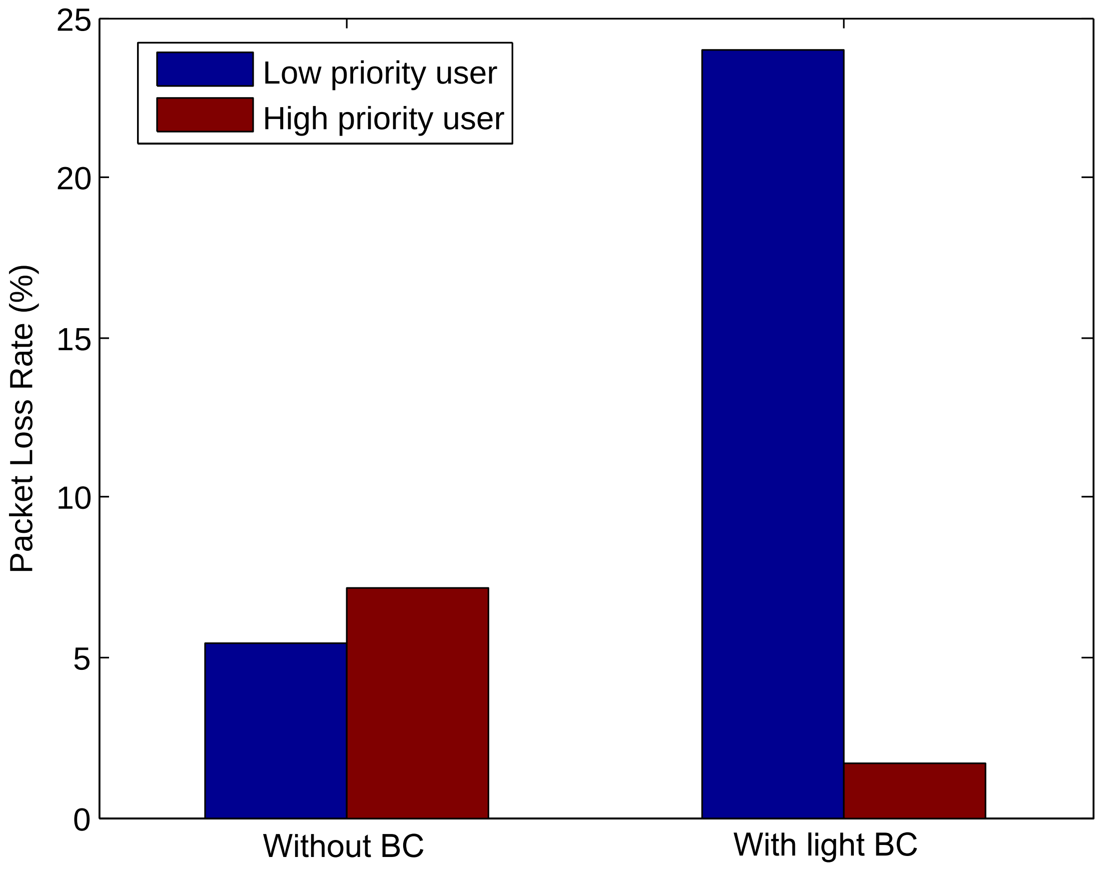


Fig. 5.7 Packet loss rate achieved without bandwidth control and with the proposed light bandwidth control

network throughput is recorded in one-second interval. The same test runs ten times and results are averaged. Fig.5.6 shows the network throughput achieved with and without the proposed light bandwidth control respectively. As it can be observed, at the begin of the test, the network throughput of the low priority user is around 40 Mbps without any bandwidth limitation. After 23 seconds of the TCP test, the proposed algorithm detects the AP load of AP1 is lower than 1 and performs light bandwidth control by directing data traffic of the low priority user to the queue which has a pre-defined maximum data rate of 10 Mbps. It is reflected in the result that the network throughput of the low priority user is limited to 10 Mbps approximately until the end of the TCP test.

The aim of the second experiment is to investigate the performance of the proposed light bandwidth control mechanism on improving packet loss rate of the high priority user. To set up the second experiment, the iperf UDP server is running on AP1. Both the low priority and high priority user are connected to AP1 to run the iperf UDP session which lasts 60 seconds. The UDP buffer size is set to the default value of 160 Kbytes, and the source data rate is configured to be 100 Mbps. Both users'packet loss rate are recorded in one-second interval. The same UDP test runs ten times and results are averaged. Fig.5.7 illustrates the packet loss rate achieved with and without the proposed light bandwidth control mechanism respectively. It can be seen that, without applying bandwidth control mechanism, the packet loss rate of the high priority user is slightly higher than the result of the low priority user as both users can contend the bandwidth and there is no policy to guarantee the QoS and QoE for the high priority user. On the other hand, with the proposed light bandwidth control mechanism, the allowed bandwidth for the low priority user is limited to 10 Mbps according to the pre-defined QoS policy, which significantly increases the packet loss rate of the low priority user. As a result, the user with high priority is guaranteed with higher bandwidth, and its packet loss rate is decreased about 6% compared to the result achieved without bandwidth control mechanism.

Heavy Bandwidth Control

The third experiment is conducted to evaluate the performance of the proposed heavy bandwidth control mechanism on limiting network throughput of the low priority user. To set up this experiment, the iperf TCP server is running on AP1. Two users with high and low priority respectively are associated with AP1 simultaneously to perform iperf TCP test which lasts 60 seconds. Both users'network throughput are recorded in one-second interval. The same test runs ten times and results are averaged. Fig.5.8 shows the network throughput achieved without the proposed algorithm. It can be seen that both users are competing for

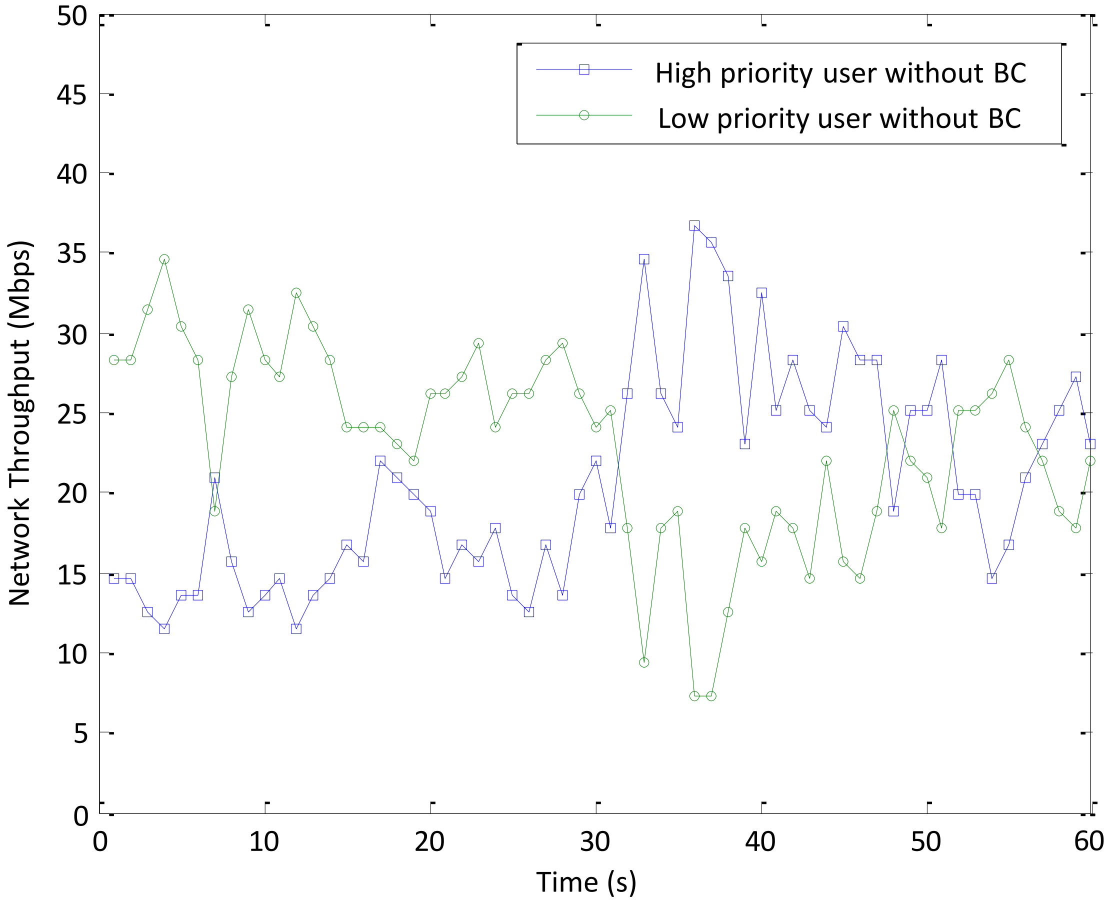


Fig. 5.8 Network throughput achieved without bandwidth control

uplink bandwidth and there is no guarantee for the high priority user to achieve a higher network throughput than the low priority user. Fig.5.9 demonstrates the network throughput achieved with the proposed heavy bandwidth control. As it can be observed, at the begin of the test, the low priority user achieves a higher network throughput than the high priority user as there is no bandwidth control applied on the low priority user. After 23 seconds of the TCP test, the proposed algorithm detects the AP load of AP1 is higher than 1 and performs heavy bandwidth control by directing data traffic of the low priority user to the queue which has a pre-defined maximum data rate of 5 Mbps. It is reflected in the result that the network throughput of the low priority user is limited to 5 Mbps approximately until the end of the

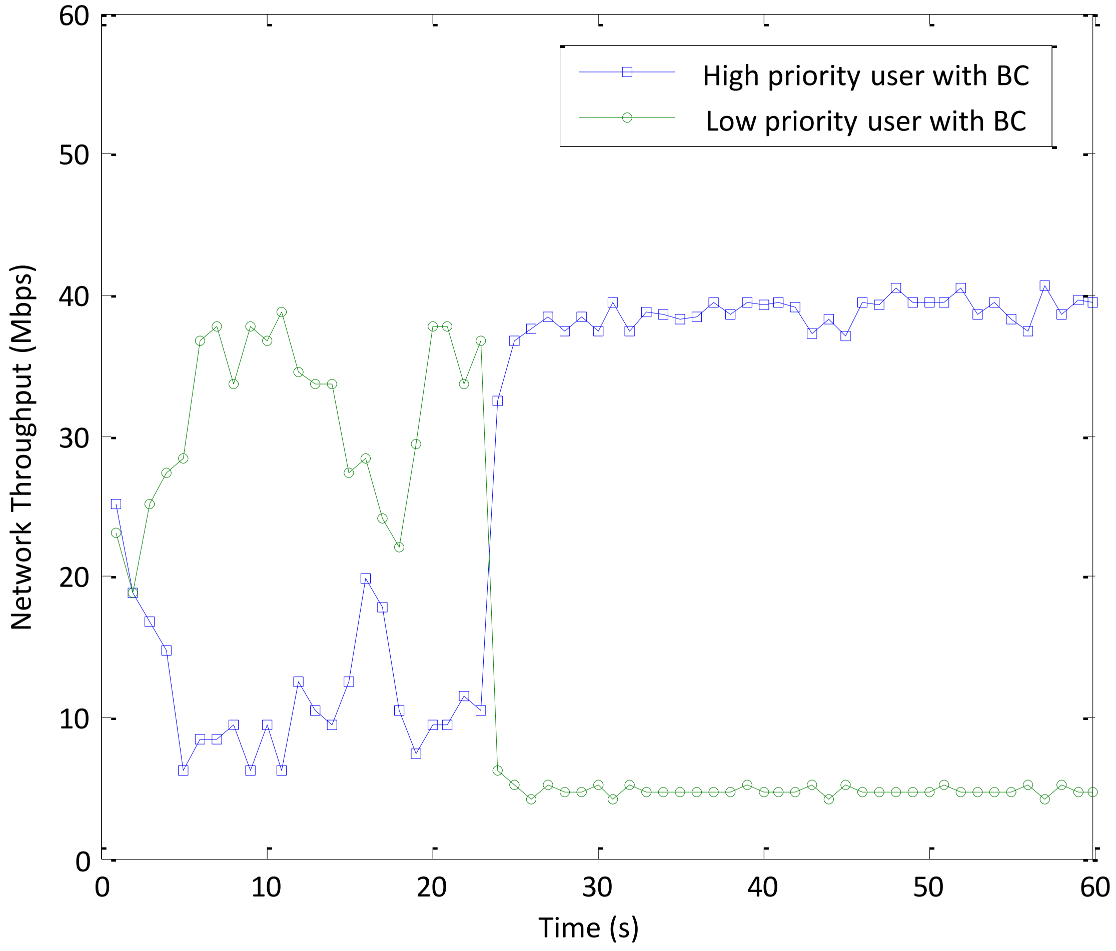


Fig. 5.9 Network throughput achieved with the proposed heavy bandwidth control

TCP test. Meanwhile, the network throughput of the high priority user has improved to 40 Mbps approximately.

The aim of the fourth experiment is to investigate the performance of the proposed heavy bandwidth control mechanism on improving packet loss rate of the high priority user. To set up the experiment, the iperf UDP server is running on AP1. Both the low priority and high priority user are connected to AP1 to run the iperf UDP session which lasts 60 seconds. The UDP buffer size is set to the default value of 160 Kbytes, and the source data rate is configured to be 100 Mbps. Both users'packet loss rate are recorded in one-second interval. The same UDP test runs ten times and results are averaged. Fig.5.10 illustrates the

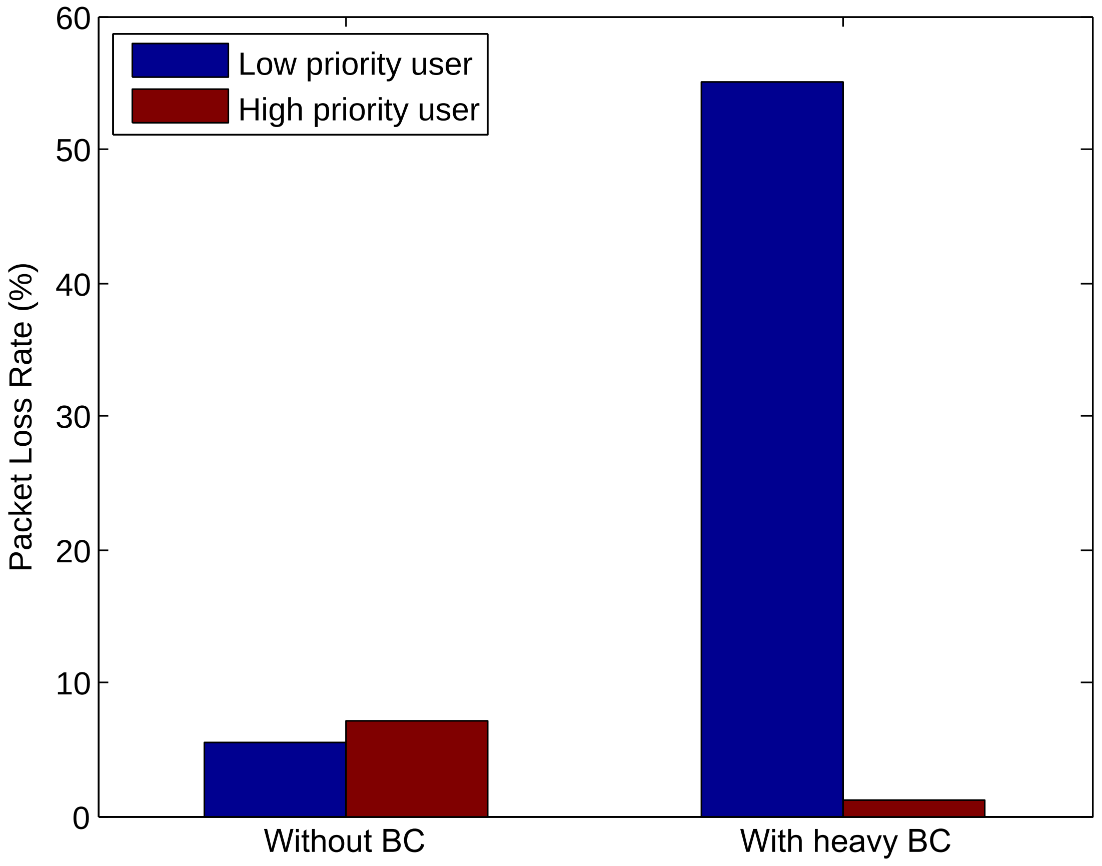


Fig. 5.10 Packet loss rate achieved without bandwidth control and with the proposed heavy bandwidth control

packet loss rate achieved with and without the proposed heavy bandwidth control mechanism respectively. It can be observed that, without applying bandwidth control mechanism, the packet loss rate of the high priority user is slightly higher than the result of the low priority user as both users can contend the bandwidth, and there is no policy to guarantee the QoS and QoE for the high priority user. On the other hand, with the proposed heavy bandwidth control mechanism, the allocated bandwidth for the low priority user is further limited to 5 Mbps according to the pre-defined QoS policy, which further increases the packet loss rate of the low priority user compared to the result achieved in the second experiment. As a result, the guaranteed bandwidth for the high priority user is further increased. The packet loss rate of the high priority users is reduced by 7% compared to the result achieved without bandwidth

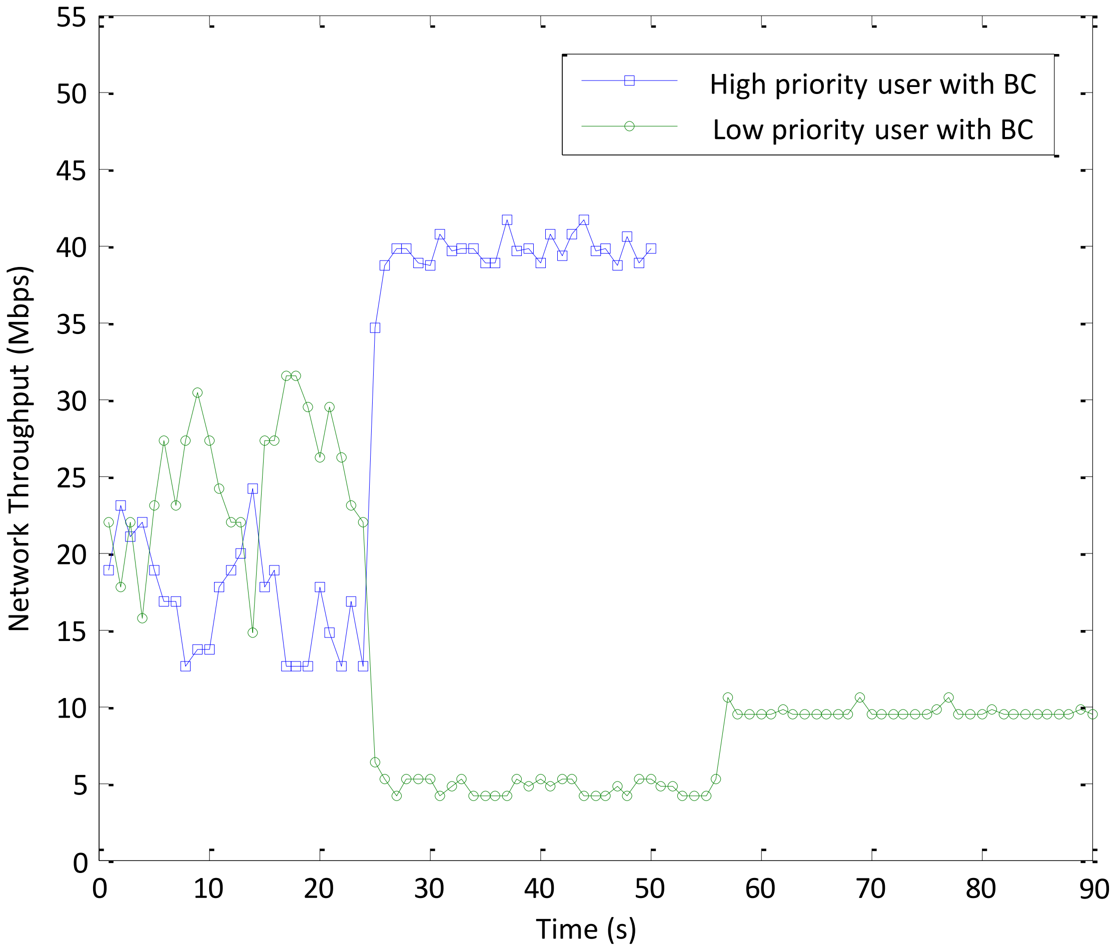


Fig. 5.11 Network throughput achieved with the proposed adaptive bandwidth control

control mechanism. Because the packet loss rate of the high priority user is already very low, the heavy bandwidth control mechanism achieves similar performance on reducing high priority users' packet loss rate when compared to the result achieved the light bandwidth control mechanism.

Adaptive Bandwidth Control

The fifth experiment is conducted to evaluate the efficiency of the proposed adaptive bandwidth control mechanism on dynamic allocating bandwidth for the low priority user. To set up this experiment, the iperf TCP server is running on AP1. Two users with high and low priority respectively are associated with AP1 simultaneously to perform iperf TCP test. However, the high priority user only runs the test for 50 seconds while the low priority user runs the test for 90 seconds. Both users'network throughput are recorded in one-second interval. The same test runs ten times and results are averaged and shown in Fig.5.11. It can be seen that both users are competing for the uplink bandwidth at the beginning of the test with the low priority user achieves a higher network throughput than the high priority user. This result is achieved because no bandwidth control is applied to the low priority user. After 23 seconds of the TCP test, the proposed algorithm detects the AP load of AP1 is higher than 1 and performs heavy bandwidth control. It is reflected in the result that the network throughput of the low priority user is limited to 5 Mbps approximately after 23 seconds. Meanwhile, the network throughput of the high priority user is improved to around 40 Mbps. After 50 seconds of the test, the high priority user finishes its TCP test and is dis-associated with AP1. It can be noticed that the network throughput of the low priority user is increased to around 10 Mbps as the proposed algorithm detects the change of the AP load and performs light bandwidth control to improve the QoE for the low priority user.

5.5 Chapter Summary

This chapter further investigates how to achieve intelligent network traffic control in enterprise WLANs from an AP-side perspective. By introducing SDN to the existing architecture of enterprise WLANs, a QoS-aware bandwidth control algorithm is presented and implemented as a WLAN-specific application on the software-defined enterprise WLAN system that is introduced in Chapter 4. By utilising AP load as the bandwidth control metric, the proposed algorithm achieves centralised uplink bandwidth control which is adapted to real-time AP traffic load. In addition, Open vSwitch is used to enable QoS-aware bandwidth control on users with different priorities. Experiment results show the proposed QoS-aware bandwidth control algorithm achieves efficient control on the network throughput of users with low priority, which significantly improves the network throughput and decreases the packet loss rate for high priority users.

In the next chapter, the author concludes this thesis with major contributions and future work.

Chapter 6

Conclusions

During the past decades, the telecom world has experienced an explosively increasing in the use of wireless devices (e.g. smartphones and laptops) and the consequent deployments of extensive wireless networks in areas such as enterprise, campuses or even entire cities. Because of the random additions of APs, unmanaged WLANs suffers interference and unbalanced traffic, which lead to significant performance degradation. This situation has made clear the need for the solution including a set of coordinated APs, usually known as enterprise WLANs. Modern enterprise WLANs always consist of multiple APs to meet the fast-increasing demand for Internet access and real-time network traffic. In order to avoid network congestion which leads to issues such as suboptimal QoS and degraded user QoE, intelligent network traffic control is needed in the deployment of enterprise WLANs. Network services such as mobility management, load balancing, automatic channel switching and bandwidth control are implemented to enable proper management on APs and user devices. Although commercial solutions exist, these are vendor proprietary and costly, which in most of the cases make them unfeasible for many organisations.

SDN is an emerging architecture and intensively discussed as one of the most promising technologies to simplify network management and service development. In the architecture of SDN, forwarding devices in the infrastructure layer provide access to the data plane and communicate with the SDN controller, which acts as the control plane for the network. The network intelligence is logically centralised in the software-based SDN controller to maintain a global view of the network. As a result, the network appears to the applications and network services as a single, logical switch, which greatly simplifies the network management service development. Leveraging SDN to the existing enterprise WLANs framework, network services can be easily implemented via open programming interfaces to improve the performance of network traffic control.

With the trend of fast adaption to implement real-time multimedia services, for instance, video streaming over enterprise WLANs, mobility management has become one of the crucial criteria to guarantee QoS and user QoE. Load balancing is another key network service to enable network traffic control as it can avoid unbalanced traffic load on APs. To improve the performance of both network services, an adaptive mobility management algorithm and an adaptive load balancing algorithm are proposed in Chapter 3. The proposed adaptive algorithms take AP load into account in addition to RSSI to achieve efficient client mobility management and AP load balancing that address the issue of unbalanced AP traffic load. A custom protocol message called Forward_AP_Load is introduced to enable the central controller to have the knowledge of each AP's traffic load information. The CSA element is added into beacon frame to achieve seamless client handover over multiple wireless channels. The author conducts iperf-based TCP tests to evaluate the performance of the proposed adaptive mobility management algorithm. Experiment results show that the proposed algorithm achieves seamless user mobility over multiple wireless channels and significantly improve network throughput when compared to the conventional fixed-RSSI mobility management algorithm. Furthermore, the author carries out iperf-based TCP and UDP tests to analyse the efficiency of the proposed adaptive load balancing algorithm. Experiment results show that the proposed algorithm can improve QoS and user

QoE significantly in terms of network throughput and packet loss rate when compared to the conventional load balancing algorithm, which is based on round robin method.

Currently, the 2.4GHz frequency band is extremely crowded because of the extensive deployment of WLANs. To minimise the interference issue caused by channel overlapping, optimised channel allocation mechanism is critical to modern enterprise WLANs. In Chapter 4, the author investigates how to achieve intelligent network traffic control in enterprise WLANs from an AP-side perspective. Leveraging SDN to the existing enterprise WLAN architecture, a software-defined enterprise WLAN system, which enables centralised network management and highly flexible programmability is presented. Furthermore, two load-aware automatic channel switching algorithms are proposed and implemented on a physical testbed to improve the performance of channel switching control on the 2.4GHz frequency band. CIF is defined as an approximation of the channel quality. By utilising AP load that is defined in Chapter 3, and CIF as channel switching metrics, the proposed SS and DS algorithms achieve efficient channel switching control, which adapts to real-time traffic load and channel condition. The author conducts iperf-based TCP and UDP tests to evaluate the performance of the proposed channel switching algorithms. Experiment results demonstrate that the proposed algorithms support seamless channel switching over multiple wireless channels. Moreover, the proposed algorithm can significantly improve QoS and user QoE by increasing network throughput, decreasing jitter and transmission delay when compared to the conventional RSSI-based channel switching mechanism.

With the growing popularity of BYOD, modern enterprise WLANs require efficient bandwidth control mechanism to enable dynamic bandwidth resource allocation on users with different priorities. In Chapter 5, the author further investigates how to achieve intelligent network traffic control in enterprise WLANs from an AP-side perspective. By introducing SDN to the existing architecture of enterprise WLANs, a QoS-aware bandwidth control algorithm is presented and implemented as a WLAN-specific application on the software-defined enterprise WLAN system that is designed in Chapter 4. By utilising AP load that is introduced in Chapter 3 as the bandwidth control metric, the proposed algorithm achieves centralised uplink bandwidth control which is adapted to real-time AP traffic load. In addition, Open vSwitch is integrated to the commodity AP to enable QoS-aware bandwidth control on users with different priorities. To evaluate the efficiency of the proposed algorithm, the author conducts iperf-based TCP and UDP tests. Experiment results show the proposed QoS-aware bandwidth control algorithm achieves efficient control on the network throughput and packet loss rate of users with low priority, which significantly improves the QoS and QoE for high priority users.

6.1 Future Work

The author's aim for the future work is to further extend the designed software-defined enterprise WLAN system to implement a more diverse set of WLAN-specific network services. Although the current SDN provides rich support for wired packet-switched networks, OpenFlow-Based SDN is not extensively adopted in the wireless part of enterprise networks yet as the current state of OpenFlow is ill-suited for WLANs. It does not have a real view of a wireless interface (i.e. OpenFlow sees the IEEE 802.11 interface as an unknown interface connected to the switch) and does not support for the matching of IEEE 802.11 specific fields. The designed system will be refined by adding a set of programming abstractions modelling the fundamental aspects of a wireless network such as user association state management, network resource allocation and network traffic monitoring. An improved bandwidth control algorithm will be implemented on top of the controller, which supports service differentiation for users with the same priority.

To enable efficient energy control on APs, a load-aware power management algorithm will be designed. The custom protocol messages will be refined accordingly and implemented. The designed algorithm will be implemented and evaluated on the testbed that is introduced in Chapter 3. The bandwidth control algorithm will be improved by utilising the SDN controller to dynamically monitor the network traffic load to estimate the network bandwidth requirement of the devices. With the estimation, dynamic bandwidth allocation can be achieved. The improved algorithm will be implemented and evaluated on the testbed that is introduced in Chapter 3. As most of the testbeds, the currently deployed testbed is relatively small. Therefore, scalability properties cannot be studied adequately. The next step is to perform experiments in a larger testbed with increased user devices that would allow understanding how scalable the proposed ideas are and how the proposed algorithms perform with a large amount of traffic workloads.

References

[1] "IEEE Standard, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, March 2012.

[2] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. Gude, N. McKeown and S. Shenker, "Rethinking Enterprise Network Control," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1270–1283, Aug 2009.

[3] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," White Paper, Apr 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[4] F. Silva, J. Pereira, P. Rosa and S. Kofuji, "Enabling Future Internet Architecture Research and Experimentation by Using Software Defined Networking," in *2012 European Workshop on Software Defined Networking*, Oct 2012, pp. 73–78.

[5] S. Hasan, Y. Ben-David, C. Scott, E. Brewer and S. Shenker, "Enhancing Rural Connectivity with Software Defined Networks," in *Proceedings of the 3rd ACM Symposium on Computing for Development*, ser. ACM DEV '13. New York, NY, USA: ACM, 2013, pp. 49:1–49:2.

[6] B. Boughzala, R. Ben Ali, M. Lemay, Y. Lemieux and O. Cherkaoui, "OpenFlow supporting inter-domain virtual machine migration," in *2011 Eighth International Conference on Wireless and Optical Communications Networks*, May 2011, pp. 1–7.

[7] A. Gember, C. Dragga and A. Akella, "ECOS: Leveraging Software-Defined Networks to support mobile application offloading," in *2012 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, Oct 2012, pp. 199–210.

[8] V. Mann, A. Vishnoi, K. Kannan and S. Kalyanaraman, "CrossRoads: Seamless VM mobility across data centers through software defined networking," in *2012 IEEE Network Operations and Management Symposium*, April 2012, pp. 88–96.

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[10] "Bring Your Own Device: The Facts and the Future," https://www.gartner.com/doc/2422315/bring-device-facts-future.

[11] "Meru Networks," http://www.fortinet.com/meru.

[12] "Cisco Meraki," https//meraki.cisco.com.

[13] "Aruba," http://www.arubanetworks.com/.

[14] "OpenWrt," https://openwrt.org/.

[15] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon and M. Casado, "The Design and Implementation of Open vSwitch," in *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 117–130.

[16] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, R. Johari, N. McKeown and G. Parulkar, "Carving Research Slices out of Your Production Networks with OpenFlow," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 129–130, Jan. 2010.

[17] R. Sherwood, G. Gibb, K. Yap, G. Appenzeller, M. Casado, N. McKeown and G. Parulkar, "FlowVisor: A Network Virtualization Layer," Tech. Rep, Oct 2009. [Online]. Available: http://archive.openflow.org/downloads/technicalreports/openflow-tr-2009-1-flowvisor.pdf

[18] K. Yap, M. Kobayashi, R. Sherwood, Rob, T. Huang, M. Chan, N. Handigol and N. McKeown, "OpenRoads: Empowering Research in Mobile Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 125–126, Jan. 2010.

[19] K. Yap, R. Sherwood, M. Kobayashi, T. Huang, M. Chan, N. Handigol, N. McKeown and G. Parulkar, "Blueprint for Introducing Innovation into Wireless Mobile Networks," in *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, ser. VISA '10. New York, NY, USA: ACM, 2010, pp. 25–32.

[20] P. Dely, J. Vestin, A. Kassler, N. Bayer, H. Einsiedler and C. Peylo, "CloudMAC-An OpenFlow based architecture for 802.11 MAC layer processing in the cloud," in *2012 IEEE Globecom Workshops*, Dec 2012, pp. 186–191.

[21] J. Vestin, P. Dely, A. Kassler, N. Bayer, H. Einsiedler and C. Peylo, "CloudMAC: Towards Software Defined WLANs," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 16, no. 4, pp. 42–45, Feb. 2013.

[22] W. S. Kim, S. H. Chung, C. W. Ahn, M. R. Do, "Seamless Handoff and Performance Anomaly Reduction Schemes Based on OpenFlow Access Points," in *2014 28th International Conference on Advanced Information Networking and Applications Workshops*, May 2014, pp. 316–321.

[23] J. Schulz-Zander, N. Sarrar and S. Schmid, "Towards a Scalable and Near-sighted Control Plane Architecture for WiFi SDNs," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14. New York, NY, USA: ACM, 2014, pp. 217–218.

[24] J. Schulz-Zander, N. Sarrar and S. Schmid, “AeroFlux: A Near-Sighted Controller Architecture for Software-Defined Wireless Networks,” in *Presented as part of the Open Networking Summit 2014 (ONS 2014)*. Santa Clara, CA: USENIX, 2014.

[25] D. Zhao, M. Zhu and M. Xu, “SDWLAN: A flexible architecture of enterprise WLAN for client-unaware fast AP handoff,” in *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, July 2014, pp. 1–6.

[26] D. Zhao, M. Zhu and M. Xu, “Supporting "One Big AP" illusion in enterprise WLAN: An SDN-based solution,” in *2014 Sixth International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct 2014, pp. 1–6.

[27] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann and T. Vazao, “Towards Programmable Enterprise WLANS with Odin,” in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN ’12. New York, NY, USA: ACM, 2012, pp. 115–120.

[28] J. Schulz-Zander, L. Suresh, N. Sarrar, A. Feldmann, Anja and T. Hühn and R. Merz, “Programmatic Orchestration of WiFi Networks,” in *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIX ATC’14. Berkeley, CA, USA: USENIX Association, 2014, pp. 347–358.

[29] L. Suresh, J. Schulz-Zander, R. Merz and A. Feldmann, “Demo: Programming Enterprise WLANs with Odin,” *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 279–280, Aug. 2012.

[30] L. Suresh, J. Schulz-Zander, R. Merz and A. Feldmann, “Demo: Programming Enterprise WLANs with Odin,” in *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM ’12. New York, NY, USA: ACM, 2012, pp. 279–280.

[31] R. Riggio, T. Rasheed and F. Granelli, “EmPOWER: A Testbed for Network Function Virtualization Research and Experimentation,” in *2013 IEEE SDN for Future Networks and Services (SDN4FNS)*, Nov 2013, pp. 1–5.

[32] R. Riggio, K. M. Gomez, T. Rasheed, J. Schulz-Zander, S. Kuklinski and M. K. Marina, “Programming Software-Defined wireless networks,” in *10th International Conference on Network and Service Management (CNSM) and Workshop*, Nov 2014, pp. 118–126.

[33] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski and T. Rasheed, “Programming Abstractions for Software-Defined Wireless Networks,” *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, June 2015.

[34] E. Coronado, R. Riggio, J. Villalón and A. Garrido, “Programming abstractions for wireless multicasting in software-defined enterprise WLANs,” in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, May 2017, pp. 263–271.

[35] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid and A. Feldmann, "OpenSDWN: Programmatic Control over Home and Enterprise WiFi," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, ser. SOSR '15. New York, NY, USA: ACM, 2015, pp. 16:1–16:12.

[36] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, A. Feldmann and R. Riggio, "Programming the Home and Enterprise WiFi with OpenSDWN," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 117–118, Aug. 2015.

[37] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, A. Feldmann and R. Riggio, "Programming the Home and Enterprise WiFi with OpenSDWN," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM '15. New York, NY, USA: ACM, 2015, pp. 117–118.

[38] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid and A. Feldmann, "Unified Programmability of Virtualized Network Functions and Software-Defined Wireless Networks," *IEEE Transactions on Network and Service Management*, vol. PP, no. 99, pp. 1–1, 2017.

[39] D. Tu, Z. Zhao and H. Zhang, "ISD-WiFi: An intelligent SDN based solution for enterprise WLANs," in *2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, Oct 2016, pp. 1–6.

[40] T. Hu, K. Xue, W. Wei and W. Jiang, "LENV: A new light-weighted edge network virtualization framework in software-defined wireless networks," in *2015 International Conference on Wireless Communications Signal Processing (WCSP)*, Oct 2015, pp. 1–6.

[41] H. Moura, G. V. C. Bessa, M. A. M. Vieira and D. F. Macedo, "Ethanol: Software defined networking for 802.11 Wireless Networks," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 388–396.

[42] "POX SDN Controller," https://github.com/noxrepo/pox.

[43] T. Lei, Z. Lu, X. Wen, X. Zhao and L. Wang, "SWAN: An SDN based campus WLAN framework," in *2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace Electronic Systems (VITAE)*, May 2014, pp. 1–5.

[44] T. Henderson, D. Kotz, David and I. Abyzov, "The Changing Usage of a Mature Campus-wide Wireless Network," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '04. New York, NY, USA: ACM, 2004, pp. 187–201.

[45] A. Mishra, M. Shin and W. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 93–102, Apr. 2003.

[46] I. Ramani and S. Savage, "SyncScan: practical fast handoff for 802.11 infrastructure networks," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 1, March 2005, pp. 675–684 vol. 1.

[47] S. Shin, A. Forte, A. Rawat and H. Schulzrinne, "Reducing MAC Layer Handoff Latency in IEEE 802.11 Wireless LANs," in *Proceedings of the Second International Workshop on Mobility Management &Amp; Wireless Access Protocols*, ser. MobiWac '04. New York, NY, USA: ACM, 2004, pp. 19–26.

[48] I. Purushothaman and S. Roy, "FastScan: A Handoff Scheme for Voice over IEEE 802.11 WLANs," *Wirel. Netw.*, vol. 16, no. 7, pp. 2049–2063, Oct. 2010.

[49] J. P. Jeong, Y. Deok Park and Y. J. Suh, "Handoff performance enhancement scheme for multi-interface enterprise WLANs," in *2013 Fourth International Conference on the Network of the Future (NoF)*, Oct 2013, pp. 1–5.

[50] I. Papanikos and M. Logothetis, "A study on dynamic load balance for ieee 802.11b wireless lan," in *in COMCON 2001*, 2001.

[51] "IEEE Standard for Local and metropolitan area networks–Port-Based Network Access Control," *IEEE Std 802.1X-2010 (Revision of IEEE Std 802.1X-2004)*, pp. 1–205, Feb 2010.

[52] "IEEE Standard for Local and metropolitan area networks - Media Independent Handover Services," *IEEE Std 802.21-2008*, pp. 1–323, Jan 2009.

[53] "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS) Transition," *IEEE Std 802.11r-2008 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008)*, pp. 1–126, July 2008.

[54] "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs," *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)*, pp. 1–244, June 2008.

[55] S. Feirer and T. Sauter, "Seamless handover in industrial WLAN using IEEE 802.11k," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, June 2017, pp. 1234–1239.

[56] Y. Fukuda, M. Honjo and Y. Oie, "Development of Access Point Selection Architecture with Avoiding Interference For WLANs," in *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, Sept 2006, pp. 1–5.

[57] K. Tsukamoto, Y. Oie, S. Kashihara and Y. Taenaka, "Seamless Handover Management Scheme Under Multi-rate WLANs," in *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia*, ser. MoMM '10. New York, NY, USA: ACM, 2010, pp. 148–158.

[58] Y. Meng, J. D. Li, H. Y. Li and P. Zhang, "An adaptive network selection scheme based on the combination of user mobility and traffic load," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, Sept 2012, pp. 1491–1496.

[59] Y. Grunenberger and F. Rousseau, “Virtual Access Points for Transparent Mobility in Wireless LANs,” in *2010 IEEE Wireless Communication and Networking Conference*, April 2010, pp. 1–6.

[60] M. E. Berezin, F. Rousseau and A. Duda, “Multichannel Virtual Access Points for Seamless Handoffs in IEEE 802.11 Wireless Networks,” in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–5.

[61] R. Riggio, C. Sengul, L. Suresh, J. Schulz–zander and A. Feldmann, “Thor: Energy programmable WiFi networks,” in *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2013, pp. 21–22.

[62] R. Riggio, C. Sengul, K. Mabell Gomez and T. Rasheed, “Energino: Energy Saving Tips for Your Wireless Network,” in *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM ’12. New York, NY, USA: ACM, 2012, pp. 273–274.

[63] L. Sequeira, J. L. de la Cruz, J. Ruiz-Mas, J. Saldana, J. Fernandez-Navajas and J. Almodovar, “Building an SDN Enterprise WLAN Based on Virtual APs,” *IEEE Communications Letters*, vol. 21, no. 2, pp. 374–377, Feb 2017.

[64] B. Zhang, X. Wen, Z. Lu, T. Lei and X. Zhao, “A fast handoff scheme for ieee 802.11 networks using software defined networking,” in *2016 19th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, Nov 2016, pp. 476–481.

[65] A. Sen and K. M. Sivalingam, “An SDN framework for seamless mobility in enterprise WLANs,” in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Aug 2015, pp. 1985–1990.

[66] P. Calhoun, M. Montemurro, and D. Stanley, “Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification,” Mar 2009. [Online]. Available: https://tools.ietf.org/html/rfc5415

[67] K. Yap, T. Huang, M. Kobayashi. M. Chan, R. Sherwood, G. Parulkar and N. Mckeown, “Lossless Handover with n-casting between WiFi-WiMAX on OpenRoads,” in *In ACM Mobicom (Demo*, 2009.

[68] A. AL Sabbagh, P. Pupatwibul, A. Banjar and R. Braun, “Optimization of the OpenFlow controller in wireless environments for enhancing mobility,” in *38th Annual IEEE Conference on Local Computer Networks - Workshops*, Oct 2013, pp. 930–935.

[69] A. K. Rangisetti, H. B. Baldaniya, P. K. B and B. R. Tamma, “Load-aware hand-offs in software defined wireless LANs,” in *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct 2014, pp. 685–690.

[70] P. Dely, A. Kassler, L. Chow, N. Bambos, N. Bayer, H. Einsiedler, C. Peylo, D. Mellado and M. Sanchez, “A software-defined networking approach for handover management with real-time video in WLANs,” *Journal of Modern Transportation*, vol. 21, no. 1, pp. 58–65, Mar 2013.

[71] P. Dely, A. Kassler, L. Chow, N. Bambos, N. Bayer, H. Einsiedler and C. Peylo, "BEST-AP: Non-intrusive Estimation of Available Bandwidth and Its Application for Dynamic Access Point Selection," *Comput. Commun.*, vol. 39, pp. 78–91, Feb. 2014.

[72] Y. Bejerano and S. J. Han, "Cell Breathing Techniques for Load Balancing in Wireless LANs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 735–749, June 2009.

[73] X. Wan, X. Wang, U. Heo and J. Choi, "A New AP-Selection Strategy for High Density IEEE802.11 WLANs," in *2010 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Oct 2010, pp. 52–58.

[74] I. Jabri, N. Krommenacker, T. Divoux and A. Soudani, "IEEE 802.11 Load Balancing: An Approach for QoS Enhancement," *International Journal of Wireless Information Networks*, vol. 15, no. 1, pp. 16–30, Mar 2008.

[75] Y. Bejerano, S. J. Han and L. Li, "Fairness and Load Balancing in Wireless LANs Using Association Control," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 560–573, June 2007.

[76] Y. Bejerano, S. J. Han and L. Li, "Fairness and Load Balancing in Wireless LANs Using Association Control," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '04. New York, NY, USA: ACM, 2004, pp. 315–329.

[77] H. Velayos, V. Aleo and G. Karlsson, "Load balancing in overlapping wireless LAN cells," in *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*, vol. 7, June 2004, pp. 3833–3836 Vol.7.

[78] I. Papanikos and M. Logothetis, "A study on dynamic load balance for IEEE 802.11b wireless LAN," in *Proceedings of the 8th International Conference on Advances in Communication and Control*, ser. COMCON 8, 01 2001, pp. 83–89.

[79] G. Xue, Q. He, H. Zhu, T. He and Y. Liu, "Sociality-Aware Access Point Selection in Enterprise Wireless LANs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 10, pp. 2069–2078, Oct 2013.

[80] G. Xue, Y. Zhu, Z. Hu, H. Zhu, C. Yue and J. Yu, "Characterizing sociality for user-friendly steady load balancing in enterprise WLANs," *IEEE Network*, vol. 29, no. 6, pp. 26–32, Nov 2015.

[81] W. Chen and G. Xue, "Sociality Analysis in Wireless Networks," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2015, pp. 999–1008.

[82] X. Sang, Q. Wu and H. Li, "Client-network collaborative load balancing mechanism for WLAN based on SDN and 802.11u," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2017, pp. 506–511.

[83] “IEEE Standard for Information Technology-Telecommunications and information exchange between systems-Local and Metropolitan networks-specific requirements-Part II: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 9: Interworking with External Networks,” *Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11w-2009, IEEE Std 802.11n-2009, IEEE Std 802.11p-2010, IEEE Std 802.11z-2010, and IEEE Std 802.11v-2011*, pp. 1–208, Feb 2011.

[84] C. Y. Lin, W. P. Tsai, M. H. Tsai and Y. Z. Cai, “Adaptive Load-Balancing Scheme through Wireless SDN-Based Association Control,” in *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, March 2017, pp. 546–553.

[85] E. Coronado, J. Villalon and A. Garrido, “Wi-balance: SDN-based load-balancing in enterprise WLANs,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, July 2017, pp. 1–2.

[86] “5G–EmPOWER,” http://empower.create-net.org/.

[87] D. Xia, J. Hart and Q. Fu, “Evaluation of the Minstrel rate adaptation algorithm in IEEE 802.11g WLANs,” in *2013 IEEE International Conference on Communications (ICC)*, June 2013, pp. 2223–2228.

[88] C. Chen, C. Wang, H. Liu, M. Hu and Z. Ren, “A novel AP selection scheme in software defined networking enabled WLAN,” *Computers and Electrical Engineering*, 2017.

[89] I. Katzela and M. Naghshineh, “Channel Assignment Schemes for Cellular Mobile Telecommunication Systems: A Comprehensive Survey,” *IEEE Communications Surveys and Tutorials.*, vol. 3, no. 2, pp. 10–31, Apr. 2000.

[90] Y. Lee, K. Kim and Y. Choi, “Optimization of AP placement and channel assignment in wireless LANs,” in *27th Annual IEEE Conference on Local Computer Networks, 2002. Proceedings. LCN 2002.*, Nov 2002, pp. 831–836.

[91] B. Kauffmann and F. Baccelli and A. Chaintreau and V. Mhatre and K. Papagiannaki and C. Diot, “Measurement-Based Self Organization of Interfering 802.11 Wireless Access Networks,” in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, May 2007, pp. 1451–1459.

[92] Y. Bejerano, S. J. Han and M. Smith, “A Novel Frequency Planning Algorithm for Mitigating Unfairness in Wireless LANs,” *Comput. Netw.*, vol. 54, no. 15, pp. 2575–2590, Oct. 2010.

[93] M. Yu and W. Su and A. Malvankar and B. K. Kwan, “A New Radio Channel Allocation Strategy For WLAN APs With Power Control Capabilities,” in *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*, Nov 2007, pp. 4893–4898.

[94] M. Yu and A. Malvankar and W. Su, "A distributed radio channel allocation scheme for WLANs with multiple data rates," *IEEE Transactions on Communications*, vol. 56, no. 3, pp. 454–465, March 2008.

[95] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan and W. Arbaugh, "A Client-Driven Approach for Channel Management in Wireless LANs," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, April 2006, pp. 1–12.

[96] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee and S. Ganguly, "Distributed Channel Management in Uncoordinated Wireless Environments," in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '06. New York, NY, USA: ACM, 2006, pp. 170–181.

[97] A. Akella, G. Judd, S. Seshan and P. Steenkiste, "Self-management in Chaotic Wireless Deployments," in *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '05. New York, NY, USA: ACM, 2005, pp. 185–199.

[98] P. Mahonen, J. Riihijarvi and M. Petrova, "Automatic channel allocation for small wireless local area networks using graph colouring algorithm approach," in *2004 IEEE 15th International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE Cat. No.04TH8754)*, vol. 1, Sept 2004, pp. 536–539 Vol.1.

[99] N. Abraham, P. P. E. Winston and M. Vadivel, "Adaptive channel allocation algorithm for WiFi networks," in *2014 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2014]*, March 2014, pp. 1307–1311.

[100] E. Rozner, Y. Mehta, A. Akella and L. Qiu, "Traffic-Aware Channel Assignment in Enterprise Wireless LANs," in *2007 IEEE International Conference on Network Protocols*, Oct 2007, pp. 133–143.

[101] M. Haidar, R. Ghimire, H. Al-Rizzo, R. Akl and Y. Chan, "Channel assignment in an IEEE 802.11 WLAN based on Signal-To-Interference Ratio," in *2008 Canadian Conference on Electrical and Computer Engineering*, May 2008, pp. 001 169–001 174.

[102] R. Riggio and M. K. Marina and T. Rasheed, "Interference management in software-defined mobile networks," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 626–632.

[103] P. San Segundo, "A New DSATUR-based Algorithm for Exact Vertex Coloring," *Comput. Oper. Res.*, vol. 39, no. 7, pp. 1724–1733, Jul. 2012.

[104] H. Lee, W. Lee, M. Shin and M. Chung, "Channel allocation and transmission power management scheme in software defined network-based WLAN environments," in *2017 International Conference on Information Networking (ICOIN)*, Jan 2017, pp. 138–142.

[105] "Open Network Operating System," https://onosproject.org//.

[106] M. Seyedebrahimi, F. Bouhafs, A. Raschellà, M. Mackay and Q. Shi, "SDN-based channel assignment algorithm for interference management in dense Wi-Fi networks," in *2016 European Conference on Networks and Communications (EuCNC)*, June 2016, pp. 128–132.

[107] Y. E. Lin and T. M. Tsai, "Creation, Management and Migration of Virtual Access Points in Software Defined WLAN," in *2015 International Conference on Cloud Computing and Big Data (CCBD)*, Nov 2015, pp. 313–320.

[108] M. Taha, L. Garcia, J. M. Jimenez, and J. Lloret, "SDN-based throughput allocation in wireless networks for heterogeneous adaptive video streaming applications," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2017, pp. 963–968.

[109] A. Amelyanovich, M. Shpakov, A. Muthanna, M. Buinevich and A. Vladyko, "Centralized control of traffic flows in wireless LANs based on the SDN concept," in *2017 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SINKHROINFO)*, July 2017, pp. 1–5.

[110] J. Vestin and A. Kassler, "QoS enabled WiFi MAC layer processing as an example of a NFV service," in *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, April 2015, pp. 1–9.

[111] M. Seddiki, M. Shahbaz, S. Donovan, S. Grover, M. Park, N. Feamster and Y. Song, "FlowQoS: QoS for the Rest of Us," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14. New York, NY, USA: ACM, 2014, pp. 207–208.

[112] J. Chen and S. Kao, "QoS Mechanism for Virtualized Wireless Networks with Software-defined Networking," *Int. J. Commun. Syst.*, vol. 28, no. 11, pp. 1741–1752, Jul. 2015.

[113] N. F. Huang, S. J. Wu, I. J. Liao, and C. W. Lin, "Bandwidth distribution for applications in slicing network toward SDN on vCPE framework," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Oct 2016, pp. 1–4.

[114] "Ryu SDN Framework," https://osrg.github.io/ryu/.

[115] K. L. Huang, C. L. Liu, C. H. Gan, M. L. Wang, and C. T. Huang, "SDN-based wireless bandwidth slicing," in *International Conference on Software Intelligence Technologies and Applications International Conference on Frontiers of Internet of Things 2014*, Dec 2014, pp. 77–81.

[116] "Floodlight OpenFlow Controller," http://www.projectfloodlight.org.

[117] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon and M. Casado, "The Design and Implementation of Open vSwitch," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 117–130.

[118] B. Pfaff, J. Pettit, T. Koponen, K. Amidon, M. Casado and S. Shenkerz, "Extending Networking into the Virtualization Layer," in *8th ACM workshop on hot topics in networks (HotNets)*, 01 2009.

[119] E. Kohler, R. Morris, B. Chen and J. Jannotti and M.F. Kaashoek, "The Click Modular Router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[120] "iPerf-The ultimate speed test tool for TCP, UDP and SCTP," https://iperf.fr/.

[121] "Wget," https://www.gnu.org/software/wget/.

[122] N. Ul Hasan, W. Ejaz, N. Ejaz, H. S. Kim, A. Anpalagan and M. Jo, "Network Selection and Channel Allocation for Spectrum Sharing in 5G Heterogeneous Networks," *IEEE Access*, vol. 4, pp. 980–992, 2016.