Wireless Netw
DOI 10.1007/s11276-017-1504-3

Two-tier dynamic load balancing in SDN-enabled Wi-Fi networks

Ying-Dar Lin[1] · Chih Chiang Wang[2] · Yi-Jen Lu[1] · Yuan-Cheng Lai[3] · Hsi-Chang Yang[1]

Abstract This work proposes a Software Defined Networking (SDN) solution to address Wi-Fi congestion due to an unevenly distributed load among access points (APs). The conventional methods generally let client stations learn of APs' load status and select APs distributively. However, such a client-driven approach lacks a global view to make precise load balancing decisions and may result in repeated changes in client-AP association. Although several studies proposed more efficient network-controlled methods to carry out Wi-Fi load balancing, some of them are distributed methods incurring excessive message exchange among *customized* APs, while the rest centralized methods are found to burden the central controller with unnecessary AP association decisions. In contrast, our solution adopts standardized OpenFlow protocol and SDN controller technology to Wi-Fi networks, organizing the SDN controller and the APs into a two-tier architecture so that the controller can evaluate the degree of load balancing among the APs and decide up to which load level the APs can accept association requests without consulting the controller. From our experiment results, our solution improves Wi-Fi's load balancing degree by 34–41%, and yields an improvement of 28–36% in Wi-Fi's re-association time over generic centralized load balancing methods with positive control.

Keywords IEEE 802.11 WLAN · Wi-Fi load balancing · Software defined networking · OpenFlow

1 Introduction

IEEE 802.11 WLANs, usually known as Wi-Fi networks, are widely deployed in infrastructure mode to provide Internet access in public areas. The smallest building block of a Wi-Fi network is a *basic service set* (BSS), in which an associated *access point* (AP) acts as a wireless bridge for all *client stations* therein to connect to the network. A client station covered by multiple APs may send a re-association request to its new favored AP and roam from one BSS to another. To facilitate seamless roaming between nearby BSSes, the IEEE 802.11 standards allow interconnected BSSes to be abstracted into a single *extended service set* (ESS). Client stations can roam within an ESS without changing network configuration. Nevertheless, association and roaming decisions are mostly made by client stations based on the signal strengths received from APs, and such a client-driven nature tends to create an unevenly distributed load among the APs [1–5]. In the worst scenario, even though an ESS area is covered by many overlapped APs, over-loaded APs therein offer a very low quality of service while the neighboring APs remain under-utilized. Since there is no standardized

✉ Chih Chiang Wang
ccwang@kuas.edu.tw

Ying-Dar Lin
ydlin@cs.nctu.edu.tw

Yuan-Cheng Lai
laiyc@cs.ntust.edu.tw

[1] Department of Computer Science, National Chiao Tung University, No.1001, Ta Hsueh Road, Hsinchu 300, Taiwan

[2] Department of Computer Science, National Kaohsiung University of Applied Sciences, No.415, Jiangong Rd., Kaohsiung 807, Taiwan

[3] Department of Information Management, National Taiwan University of Science and Technology, No. 43, Sec. 4, Keelung Road, Taipei 100, Taiwan

Published online: 09 April 2017

method for solving this problem as yet, how to balance the load among APs within an ESS remains an open issue.

Existing methods for load balancing in Wi-Fi networks, depending on which part of the network is in charge of AP association decisions, can be classified into either the *client-driven* or the *network-controlled*. Most of the existing methods adopt the client-driven approach, whereby each client station learns of APs' load status and makes AP association decisions that maximize its own benefits. Since the client-driven approach lacks a global view to make precise network-wide load balancing decisions, it may take quite a while and require repeated changes in client-AP association to reach an ideal equilibrium state [6–8]. On the other hand, the network-controlled methods aim to achieve a good load balancing performance by either adjusting APs' coverage [9–11] or regulating APs' associated client connections [12–16]. However, some of them are distributed methods built upon collaboration of AP devices, requiring excessive message exchange among APs with custom hardware support. The rest methods manage association between client stations and APs by a central controller, and are found to burden the controller with too many unnecessary AP association decisions. Moreover, their dependence on non-standardized hardware/protocol designs makes Wi-Fi networks difficult to inter-operate with devices from different vendors.

In recent years, research efforts started to adopt Software Defined Networking (SDN) architecture to wireless networks to lift the control plane up and out of wireless devices via OpenFlow interface between them [17, 18]. As SDN holds great promise for simplifying network management, enabling programmatic control of wireless devices and even offering new network functions, there are already quite many service providers and vendors supporting OpenFlow standards, including Google, Microsoft, Cisco, HP, and more. After reviewing the existing work on Wi-Fi load balancing and relevant SDN architecture, we believe that a better Wi-Fi load balancing method, instead of relying on custom hardware/protocol designs, could be developed based on standardized OpenFlow protocol and SDN controller technology, which motivates this work.

In this paper, we propose an SDN-based Wi-Fi load balancing solution which organizes an SDN controller and the APs to be managed into a two-tier architecture. The SDN controller can collect information from the APs and decide up to which load level the APs can accept association requests without consulting the SDN controller. While there already exists some designs that also leverage a central controller to carry out Wi-Fi load balancing, our solution is distinguished from the existing centralized methods in two ways. First of all, the existing centralized methods relinquish all the AP association decisions to a central controller, and our experiment results reveal that such an approach tends to burden the controller with too many unnecessary load balancing tasks and hence prolong the turn-around time of the AP association process. In contrast, our solution can strike a good balance by dynamically adjusting up to which load level the APs can act on their own. Furthermore, our solution is hardware independent and applicable to any wireless devices that support OpenFlow standards.

This work made the following contributions: (1) We have developed a two-tier dynamic load balancing method for SDN-enabled Wi-Fi networks. Its design includes an AP-side software module which reports the AP's status to the SDN controller and manages its associated client connections as instructed by the controller, and a controller-side module which receives status information from the APs, performs load balancing computations, and instructs the APs to take proper actions accordingly. (2) We have implemented our solution on a real Wi-Fi network testbed, and have performed a pressure test on it to obtain the optimal load balancing decisions and control parameters under monotonically increasing load diversity.

The rest of this paper is organized as follows. Section 2 surveys the existing work on Wi-Fi load balancing and relevant SDN architecture. The network model and the problem statement are presented in Sect. 3. Section 4 elaborates the proposed two-tier solution. Section 5 presents various experiments for evaluating the performance of our two-tier load balancing solution. We finally conclude in Sect. 6.

2 Related work

2.1 Wi-Fi load balancing methods

Wi-Fi load balancing methods, regardless of which approach is used, always involve associating or re-associating some client stations with lightly loaded APs. In order to do so, client stations must discover lightly loaded APs through either passive scanning or active scanning in the first place. By passive scanning, client stations obtain information of nearby APs by simply listening for their periodically transmitted beacon frames. However, since passive scanning requires client stations to scan all the Wi-Fi channels and dwell in each channel long enough to pick up the beacon signals, the scanning time is typically too high for client handovers. Alternatively, to reduce the discovery latency, client stations can actively probe each channel and collect information from probe responses from nearby APs.

Depending on which part of the network is in charge of AP association decisions, Wi-Fi load balancing methods can be classified into either the *client-driven* or the

network-controlled [19]. Most of the existing methods adopt the client-driven approach, whereby each client station learns of APs' status and makes AP selection decisions that maximize its own benefits. The conventional AP selection techniques are for each client station to selfishly pick the AP with the strongest received signal, but the literature [1–5] demonstrates that the use of the received signal strength in AP selection may result in congested hotspots in Wi-Fi networks, and that new metrics based on packet level information such as the number of calls/registrations admitted to an AP, the average number of packet transmissions associated with an AP, the AP's packet error rate, etc., are required for making better AP selection decisions.

There are several AP selection techniques based on metrics other the signal strengths received from APs. The work of Vasudevan et al. [20] presents a technique for client stations to estimate APs' available bandwidth and use this metric in AP selection. In [21], Gong et al. propose an AP selection method based on the turn-around time that an AP takes to serve each client station one unit of traffic. In [7], Mittal et al. model the distributed AP selection as a selection game and devise a greedy algorithm that leads the game to a Nash equilibrium. In [8], the authors model the distributed AP selection as a variant of the weighted singleton congestion game, and propose an online AP association strategy which maximizes the minimal throughput among all clients. Since the client-driven approach lacks a global view to make precise network-wide load balancing decisions, it generally takes quite a while and requires repeated changes in client-AP association to reach an ideal equilibrium state.

The network-controlled method aims to achieve a good load balancing performance by either adjusting APs' coverage or managing APs' associated client connections. In the coverage adjustment approach, over-loaded and under-loaded APs reduce and raise the transmission power of beacon frames, respectively. In [9], the authors propose a coverage adjustment solution which implements the agents within APs based on Jade Multi-Agent System Platform, but their solution depends on a custom RF hardware and the perfect knowledge of APs' coverage and clients' positions. The work of Bejerano and Han [10] presents an in-depth analysis of a coverage adjustment technique named *cell breathing* plus a centralized solution with two different cell breathing algorithms: one for reducing the load of the most congested AP and the other for solving a min-max load balancing problem. The work of Stanley et al. [11] presents Cisco's centralized management protocol named "CAPWAP" for load balancing in wireless networks based on cell breathing techniques.

On the other hand, the association management approach intends to keep over-loaded APs from accepting new association requests while inducing client stations to (re-)associate to under-utilized APs. The work of [12] employs a centralized *Admission Control Server* (ACS) to perform an admission test on APs' association tables and tell client stations which AP each of them should associate to; however, to implement this solution requires modifying the client devices. An alternative is proposed in [13] such that the APs accept or deny new association requests depending on their load status—under-loaded APs will accept any request, balanced APs will not accept extra load, and over-loaded APs will expel the client stations. In [14], the authors propose load balancing techniques for obtaining the optimal max–min fair bandwidth allocation, along with association management algorithms that achieve constant-factor approximation. The work of Duo and Chen [15] presents a centralized association management mechanism in which a network access device, in response to a new client's association request, selects one of the APs based on their load status, instructs the selected AP to associate with the client, and instructs the remaining APs to reject the client's request. The work of Iyer et al. [16] presents a centralized mechanism which deploys a specialist hardware named *Station Management Logic* (STM) within a wireless network switch to perform load balancing functions. As described in [16], when a client station broadcasts probe requests to nearby APs, the probe requests are routed to STM, and then STM instructs the selected AP to associate with the client station.

In general, the conventional distributed network-controlled methods are built upon collaboration of AP devices, requiring excessive message exchange among APs with custom hardware support. The other methods manage association between client stations and APs by a central controller, and are found to burden the controller with too many unnecessary AP association decisions. Furthermore, their dependence on non-standardized hardware/protocol designs makes Wi-Fi networks difficult to inter-operate with devices from different vendors.

2.2 Software defined wireless networking

In order to provide seamless mobile handovers within a wireless network, coverages of multiple APs often overlap the same Wi-Fi area. Since conventional AP devices handle client stations independently from the rest of the network, they can hardly be dynamically utilized as a whole; some are even over-loaded while the nearby one remains under-utilized. Software Defined Networking (SDN) is an emerging networking paradigm in which network devices operate as pure data planes to be instructed by a software-based central controller via an open interface [22]. Obviously, standardized SDN technology can serve as a

convenient foundation upon which an efficient load balancing method can be built for Wi-Fi networks.

OpenFlow [23] is a prominent example of open interface specifications to realize the idea of SDN. It focuses on the interface between the controller and the switch so that the former can control the latter by using a generated flow table [24]. Each OpenFlow switch, as a pure data plane, runs OpenFlow protocol to communicate with the OpenFlow controller, maintains a flow table to record the relevant data forwarding/processing algorithm, and forwards/processes data according to the flow table. On the other hand, an OpenFlow controller is a software implemented control plane deployed on a powerful server, responsible for generating, updating, and configuring flow tables of OpenFlow switches under its control. Through the OpenFlow controller, users can easily operate and control a particular network with application programming interfaces. Since OpenFlow promises to simplify network management, commoditize network hardware, and enable deployment of third-party network functions on OpenFlow-enabled networks, there are already quite many service providers and vendors supporting OpenFlow standards, including Google, Microsoft, Cisco, HP, and more.

The field of SDN research is quite new. Consequently, adopting the SDN concept to wireless networks has just been recently introduced in a few research work. One of the earliest milestones was set by [25], which applies SDN to low-rate wireless personal area networks (LR-WPANs) for reducing the energy consumption by periodically turning on and off the interface of generic devices. In the work of [26], the authors apply SDN to extremely dense wireless networks (WLANs or cellular networks) and propose an architecture that can reduce the interference and the cost of mobile handovers. This is done by employing the SDN controller to dynamically and intelligently tune IEEE 802.11 parameters for each base station, selecting and muting sub-frames that induce too much interference, and taking certain steps of handovers in parallel. The work of [17] presents a three-layer architecture for software defined wireless networking in which Layer-1 enhances radio access networks with programmability and Layer-2 switches and Layer-3 routers allow setup of unicasting and multicasting at the flow level. In [18], the authors propose a framework for combining wireless network virtualization with SDN and discuss some future challenges.

In this work, we also use the SDN/OpenFlow specifications to build the proposed two-tier load balancing solution on a Wi-Fi network testbed. Our solution especially adopts a generic Type-Length-Value message extension offered by OpenFlow version 1.3 and beyond [24] because this extension allows developers to add new network functions without changing the original SDN hardware/software architecture. To date we are not aware of any other research effort that has proposed an SDN-based two-tier Wi-Fi load balancing solution similar to ours which can dynamically adjust up to which load level the APs can act on their own.

3 Network model and problem description

This section presents the network model and the problem statement of our work. Table 1 lists the key mathematical notations used through this paper. Our model considers a public Wi-Fi network in which APs are equipped with bandwidth control whereby the administrator can assign a limited maximum allowable bandwidth, say 256 kbps, for each associated user device. We consider that under such bandwidth limitation, estimating load with the number of associated user devices is appropriate. The public Wi-Fi network is composed of a single ESS which consists of a set of inter-connected BSSes $\{bss_i | i = 1, 2, \ldots, na, na \in \mathbb{N}\}$. For ease of explanation, we use the notation ap_i to denote a specific SDN-enabled AP that is associated with bss_i. As a result, the ESS area is covered by many overlapped SDN-enabled APs which are controlled by a centralized SDN controller.

The SDN controller requires the information about the APs' capacity and load status to determine their load balancing degree. In our model, the capacity information about a particular AP, say ap_i, is maintained as a three-tuple, i.e. $cap_i = (M_i, ns_i, snr_i)$, where M_i denotes the maximum number of client associations that ap_i can maintain in its association table, ns_i denotes the number of client stations currently associated with ap_i, and snr_i denotes the average signal-to-noise ratio of ap_i's associated wireless connections. The load information about ap_i is also a three-tuple, i.e. $load_i = (cpu_i, mem_i, per_i)$, which denotes the CPU utilization, the memory utilization, and the average packet error rate of ap_i's associated wireless connections, respectively. Based on these information, we adopt an evaluation formula from [27] to compute *Jain's fairness index* of the ESS, denoted by B, as:

$$B = \frac{\left(\sum_{i=1}^{na} \frac{ns_i}{M_i} \max(load_i)\right)^2}{na \sum_{i=1}^{na} \left(\frac{ns_i}{M_i} \max(load_i)\right)^2} \quad (1)$$

where the numerator indicates the square of the total loading of the ESS and the denominator is the number of APs times the sum of the square of each AP's loading. Thus, if all the APs are equally loaded, B should approximate 1; on the other hand, $(B \equiv \frac{1}{na})$ means the worst degree of load balancing. As the APs are likely to have different load levels, the SDN controller also computes the *average load level* of the ESS, denoted by L, as

Table 1 Key notations of network model

Symbol	Meaning
na	Number of BSSes in the ESS of the Wi-Fi network
bss_i	A BSS in the ESS
ap_i	The AP associated with bss_i
M_i	Maximum number of client associations that ap_i can support
ns_i	Client stations that are currently associated with ap_i
nss_i	Suggested number of client stations to be maintained by ap_i
snr_i	Average signal-to-noise ratio of ap_i's associated wireless connections
cap_i	Capacity information about ap_i, $cap_i = (M_i, ns_i, snr_i)$
cpu_i, mem_i	CPU and memory utilization of ap_i, respectively
per_i	Average packet error rate of ap_i's associated wireless connections
$load_i$	Load information about ap_i, $load_i = (cpu_i, mem_i, per_i)$
B	Fairness index of the ESS (see Eq. 1)
L	Average load level of the ESS (see Eq. 2)
C	Control level of the ESS (see Eq. 3)

$$L = \frac{\left(\sum_{i=1}^{na} \frac{ns_i}{M_i} \max(load_i)\right)}{na}. \quad (2)$$

The value of L ranges from 0 to 100, and $(L \equiv 0)$ and $(L \equiv 100)$ represent the lightest and the heaviest load, respectively. This average load level is used by the SDN controller to divide the APs associated with the ESS into over-loaded and under-loaded groups so that over-loaded APs can transfer some of their associated client stations to under-loaded APs.

Instead of relinquishing all the AP association decisions to the SDN controller, our solution defines a parameter named *control level* for the SDN controller to decide beyond which load level itself shall make load balancing decisions for the over-loaded APs. The suggested control level for the ESS, denoted by C, is computed as

$$C = 100 \cdot \left(1 - \frac{nss_i}{M_i}\right). \quad (3)$$

In specifics, the SDN controller will not perform the load balancing control on ap_i unless ns_i exceeds nss_i, the suggested number of client stations to be maintained by ap_i. Generally speaking, the higher the value of a control level, the more client stations an over-loaded AP would hand over to other under-loaded APs.

The problem to be addressed thus can be stated as follows.

Problem Statement Consider an arbitrary Wi-Fi network composed of a single ESS which includes a set of interconnected BSSes, $\{bss_i | i = 1, 2, \ldots, na, na \in \mathbb{N}\}$. Assume that bss_i is associated with a specific AP ap_i, whose capacity and load attributes are defined by two three-tuple vectors: $cap_i = (M_i, ns_i, snr_i)$ and $load_i = (cpu_i, mem_i, per_i)$, respectively. The objective is for the SDN controller to calculate the *minimum control level* for the ESS under specific network loading, which is the value of the control level at which the ESS still produces a fairness index close to 1 while the Wi-Fi re-association time is minimized.

4 Two-tier dynamic Wi-Fi load balancing solution

This section presents our two-tier dynamic load balancing solution for SDN-enabled Wi-Fi networks, which is compliant with current OpenFlow specifications and protocol. We first explain its design, including (1) how its AP-side module monitors the AP's status for the SDN controller and manages the AP's associated client connections as instructed by the SDN controller; and (2) how the controller-side module detects and resolves load unbalance among the APs and instructs the AP-side modules to take proper actions accordingly. We then elaborate the OpenFlow extension message formats and other implementation details of our solutions.

4.1 Overview

Our two-tier dynamic Wi-Fi load balancing solution is distinguished from the existing centralized methods in two aspects. First of all, its two-tier architecture, compared with the existing centralized architecture, can effectively alleviate overloading at the central SDN controller by adjusting a parameter named *control level* for offloading part of the AP association control to the APs. Secondly, after collecting the status reports from the APs, our solution automatically calculates the minimum control level at which the Wi-Fi network remains load balanced.

The above design requires collaboration between the APs and the SDN controller. The APs must report their capacity, load, and association-table information to the

SDN controller when they boot up or when their status has changed. In short, the status changes when the AP detects a change in the number of its associated user devices or when the difference between its current and old load values exceeds a fraction η of the old one.[1] After receiving the APs' status reports, the SDN controller computes the fairness index by Eq. 1. If the resultant fairness index is not close to 1, the SDN controller will try to resolve load unbalance among the APs by dispatching some client stations from over-loaded APs to under-loaded APs. The over-loaded APs, once receiving the load balancing instruction from the SDN controller, de-associate some client stations accordingly. As a result, a client station that has been de-associated by an over-loaded AP will try to re-associate with another nearby AP. Eventually, only one specific under-loaded AP assigned by the SDN controller will answer the re-association request issued from that de-associated client station.

4.2 Controller-side and AP-side load balancing operations

In our design, the SDN controller is responsible for collecting the APs' status reports, calculating the fairness index, and deciding a re-association/de-association list (a list of client stations to be re-associated/de-associated) for each under-/over-loaded AP. Figure 1a shows the operation of our two-tier dynamic Wi-Fi load balancing solution on the SDN-controller side. Initially, the SDN controller listens for OpenFlow extension messages from the Wi-Fi network under its control. When receiving the APs' status reports from the ESS, the SDN controller computes the fairness index, the average load level, and the suggested control level accordingly, and sends these results to all the APs in the ESS. The APs with load levels greater than the received average load level must report their association tables to the SDN controller so that the SDN controller can decide de-association lists for over-loaded APs and re-association lists for other under-loaded APs.

The operation on the AP side is illustrated in Fig. 1b. As shown in the figure, there is a detection mechanism running in background to monitor the AP's status for the SDN controller. This detection mechanism reports the AP's capacity, load, and association-table information to the SDN controller when the AP has just successfully established an OpenFlow session to the SDN controller or when the AP's status has just changed. The management plane running in foreground also triggers the detection mechanism when an association request from some client station has arrived. After an AP reports its status to the SDN controller, the AP will receive from the SDN controller the computation results of the fairness index, the average load level, and the suggested control level. If the number of its currently associated client stations exceeds the threshold set by the suggested control level, the AP will send its association table to the SDN controller. Such an over-loaded AP will eventually receive a de-association list from the SDN controller, and will de-associate those specified in the list accordingly.

4.3 OpenFlow extension message formats

We have specified four payload formats for the OpenFlow extension messages, as shown in Fig. 2, to support the load-balancing communication between the SDN controller and the APs. These payload formats are for carrying the information about an AP's load and capacity status, the computation results from the SDN controller, the association table from an over-loaded AP, and the de-association or re-association decisions from the SDN controller, respectively.

In the figure, the payload format starting with "apInfo_prefix@" is used by an AP to report its status to the SDN controller; the fields in the payload store the following information: the daemon process ID (Dpid), the service set ID (ssid), the source AP's MAC address (apMAC-1), its capacity information (Capacity(...)) and load information (Load(...)). The payload formats with the "result_prefix@" and the "assoc_prefix@" prefix are similar to the previous one except that the former is used to carry the computation results of the fairness index, the average load level, and the control level from the SDN controller to all the APs, whereas the latter is used to carry a list of an over-loaded AP's associated client stations along with their MAC and RSSI information. The last payload format starting with "decision_prefix@" is used to carry the de-association or re-association decisions from the SDN controller to the designated AP. Its fields store the destination AP's MAC address, the action to be taken by the AP, and the MAC addresses of a list of the client stations as the target of the action.

4.4 Implementation details

To realize the proposed two-tier Wi-Fi load balancing design, we need to implement: (1) the communication protocol between the SDN controller and the APs, (2) the load balancing calculations and decisions to be performed by the SDN controller, and (3) the detection mechanism and the actions to be invoked at the APs. Figure 3 illustrates our implementation framework in a state-transition diagram and shows how the APs interact with the SDN controller. From the state-transition diagram, we can see that when an AP just boots up and successfully establishes

[1] In our experiment, η is set to 0.2 by default.

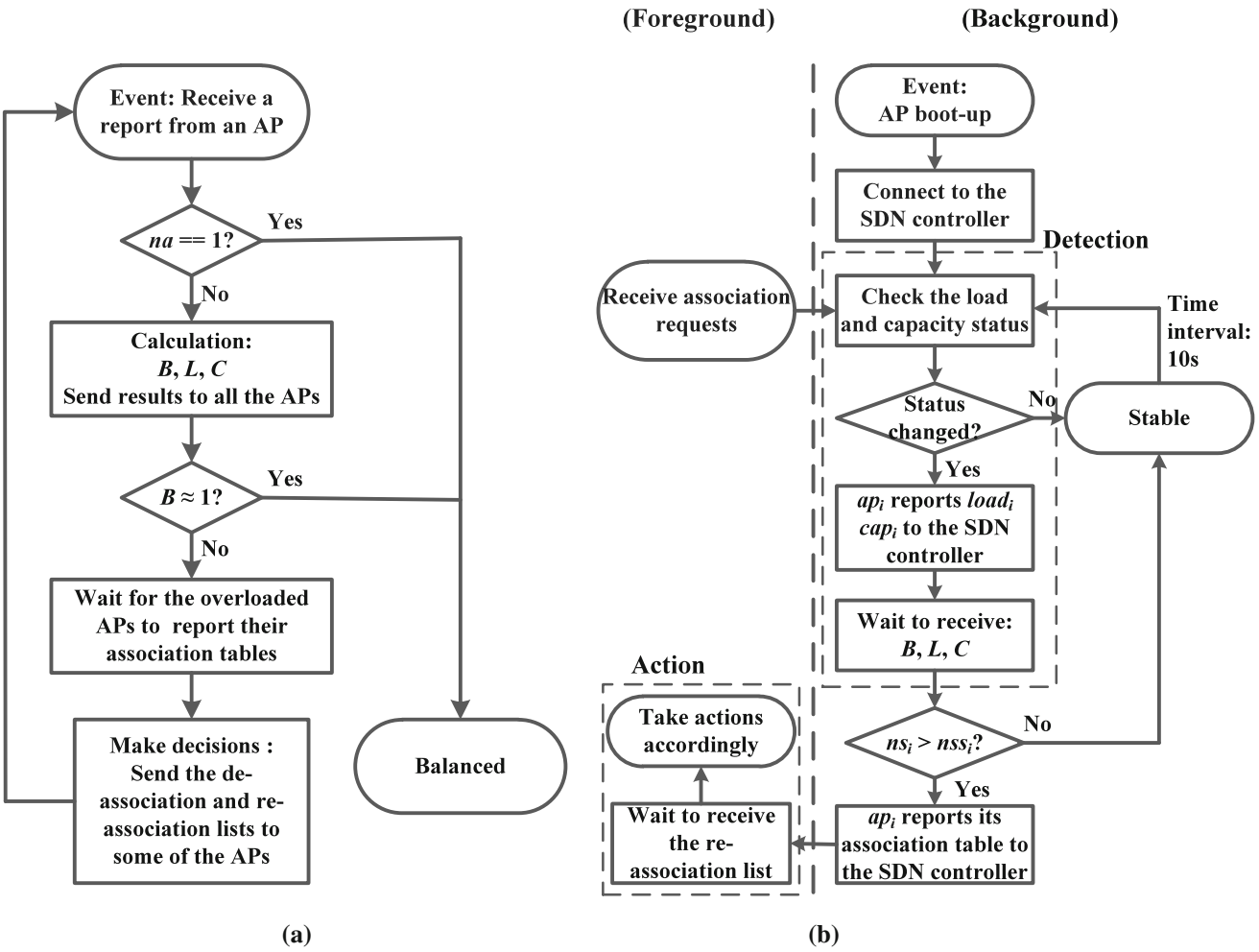

Fig. 1 The operations of our two-tier Wi-Fi load balancing solution: **a** on the SDN-controller side and **b** on the AP side

The Payload of the AP's Status Information (apInfo_prefix@)

apInfo_prefix@	Dpid	ssid	apMac-1	Capacity(max_conn, cur_conn, snr)	Load(cpu, mem, per)

The Payload of the SDN Controller's Calculation Information (result_prefix@)

result_prefix@	Dpid	ssid	apMac-1	Fairness index	Average load level	Control level

The Payload of the AP's Association Table (assoc_prefix@)

assoc_prefix@	Dpid	ssid	apMac-1	Station1(mac, rssi)	Station2(mac, rssi)	...

The Payload of the SDN Controller's Re-association Decision List (decision_prefix@)

decision_prefix@	ssid	apMac-1	Action(del)	StationMac-1	...	apMac-2	Action(add)	...

Fig. 2 The payload formats of the OpenFlow extension messages that are used by our two-tier Wi-Fi load balancing solution

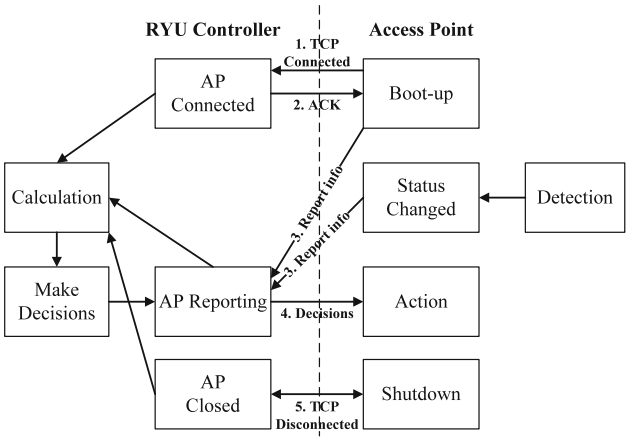

Fig. 3 The implementation framework illustrated by a state-transition diagram

Exponential Moving Average (EMA)

$EMA_t = \alpha * V_t + (1-\alpha) * EMA_{t-1}$

- α: a smoothing constant (weight) between 0 and 1
- V_t: the value of the load at time t
- $EMA_0 = V_0$

For example:

- Initial phase: $\alpha = 0.6$, $EMA_0 = V_0 = 10$
- Next time interval: $V_1 = 20$, $EMA_1 = 0.6*20+0.4*10 = 16$
- Compare the current and the old EMA to see if changed significantly

Fig. 4 The exponential moving average method used by the detection mechanism on the AP side

a session to the SDN controller, or reports its status to the controller, or disconnects from the controller, it causes the state of the SDN controller to transit to "AP Connected", "AP Reporting", or "AP Closed", respectively. Transition to any of the aforementioned state triggers the SDN controller to re-calculate the load balance, the average load level, and the suggested control level and to make load balancing decisions accordingly. Moreover, the SDN controller always maintains the device instances of APs and switches involved in the load balancing operation.

For information exchange between the controller and the APs, we use the OpenFlow extension messages defined previously to carry the information. The computations of the fairness index, the average load level, and the control level are straightforward based on Eqs. 1, 2, and 3. In regard to deciding on a de-association list, the SDN controller picks from the received association table a list of client stations to de-associate by Last-In-First-Out (LIFO) policy. The implementation of the detection mechanism on the AP side, however, must take into account the variations in the AP's capacity and load status. To address this issue, we use an exponential moving average method exhibited in Fig. 4 to detect changes in an AP's status. Finally, in our implementation, APs invoke an utility named *hostapd_cli* to perform client association or de-association.

5 Performance evaluation

In this section, we firstly elaborate the experiment setup and the experiment plan for evaluating how the proposed two-tier load balancing solution performs as compared with generic centralized load balancing methods with positive control. We then present some representative results and discuss their implications. The load balancing performance is evaluated in terms of the resultant fairness index and the average re-association time perceived by the client stations.

5.1 Experiment setup and experiment plan

Our experiment aims to evaluate how the proposed load balancing solution performs on a real Wi-Fi network composed of heterogeneous AP devices. For this purpose, we set up a Wi-Fi network testbed which consists of three real APs of AP222 model from EstiNet [28] and one RYU SDN controller from [29]. The flow of our experiment is illustrated in Fig. 5. At first the APs report their status (dev_info) to the SDN controller when detecting changes in the status. Based on the newly received status information, the SDN controller computes the fairness index, the average load level, and the control level, and sends the most updated computation results to the APs. The APs check their status against the received computation results, reporting their association table to the SDN controller when needed. If the SDN controller receives association tables from the APs, it will make load balancing decisions and send the re-association/de-association lists to the under-/over-loaded APs.

The three APs have different transmission powers and are set with different IP addresses under the same ESS: AP_1 with $(IP = 10.0.0.1/24, TxPower = 19dBm)$, AP_2 with $(IP = 20.0.0.1/24, TxPower = 15dBm)$, and AP_3 with $(IP = 30.0.0.1/24, TxPower = 11dBm)$. Since this experiment needs to create a pressure test on the network testbed for testing our load balancing solution under monotonically increasing load diversity, we run a station generator in each AP to simulate the dynamic behaviors of a large set of client stations. In the experiment we tune the APs' loading to observe the corresponding changes in the number of client connections and the fairness index at each AP as the control level varies.

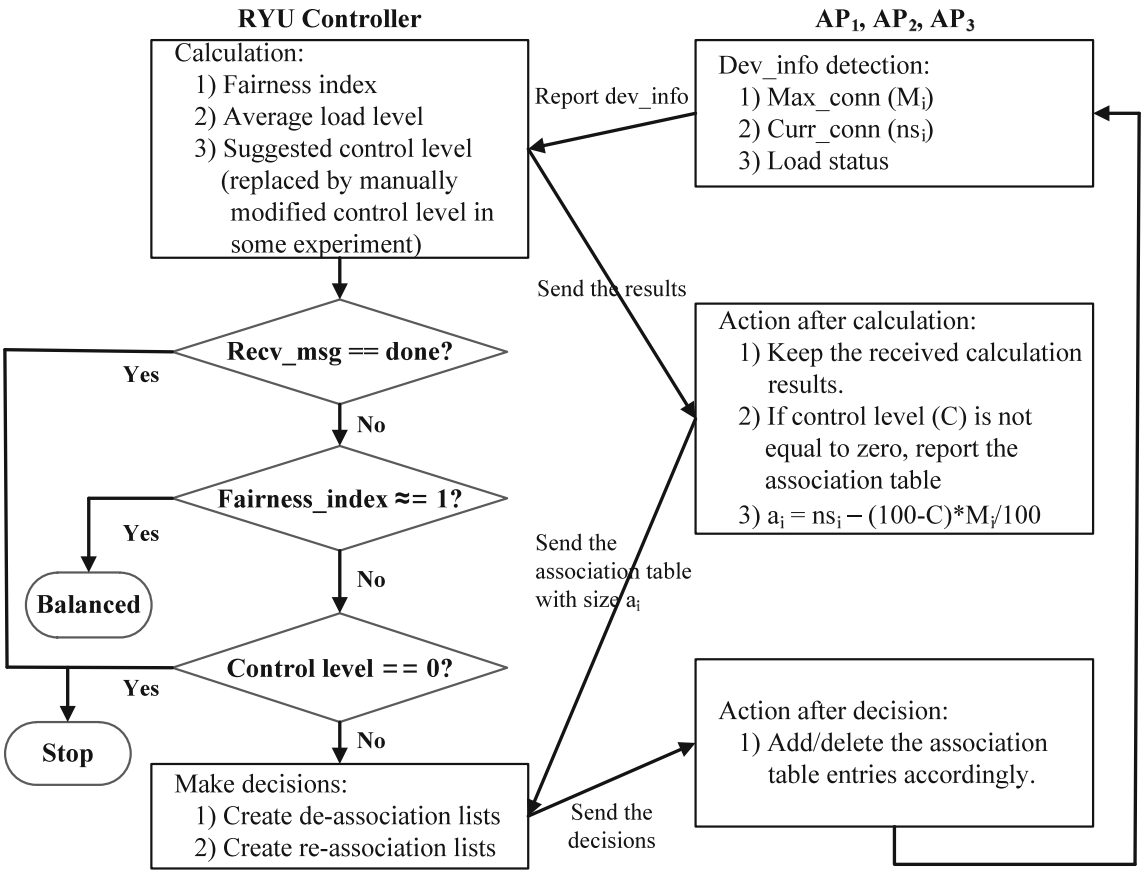

Fig. 5 The flow of the experiment

The experiment is organized into two scenarios whose parameters are summarized in Table 2. In both the scenarios, each AP can maintain at most 50 client connections in its association table, and the three APs, AP_1, AP_2, and AP_3, are initialized with association tables of different sizes of 49, 34, and 1 client connections, respectively. The first scenario, in which all the APs are initialized with the same load level of value 60, is intended for studying how the numbers of client connections maintained by the APs affect the load balancing result. The second scenario represents the case when the APs are initialized with different load levels, which shows how the re-association time is affected by tuning of the suggested control level as the load diversity increases. For ease of explanation, we define a parameter named *loading difference* to represent the sum of the differences between the highest load level and the load levels of the other APs. For example, given that the load levels of AP_1, AP_2, and AP_3 are 100, 90, and 50, respectively, their loading difference is computed as: $(100 - 90) + (100 - 50) = 60$. This allows us to observe how the suggested control level and the re-association time change as the loading difference is increased from 0 to 100.

5.2 Experiment results

Here we present the experiment results of the proposed two-tier load balancing solution. Figures 6 and 7 exhibit the resultant fairness index and re-association time of the first and the second experiment scenario under different control levels. These results reveal that: (1) the resultant fairness index increases with the control level and converges to 1 before the control level reaches 100; (2) the re-association time increases with the control level as well due to an increasing number of client stations that need to be

Table 2 Experiment parameters

Devices	Tx power	Max. Conn.	Curr. Conn.	Load (same)	Load (diff.)	Range of Loading
AP_1	19 dBm	50	49	60	60	100
AP_2	15 dBm	50	34	60	40	100,90,80,70,60,50
AP_3	11 dBm	50	1	60	20	100,90,80,70,60,50

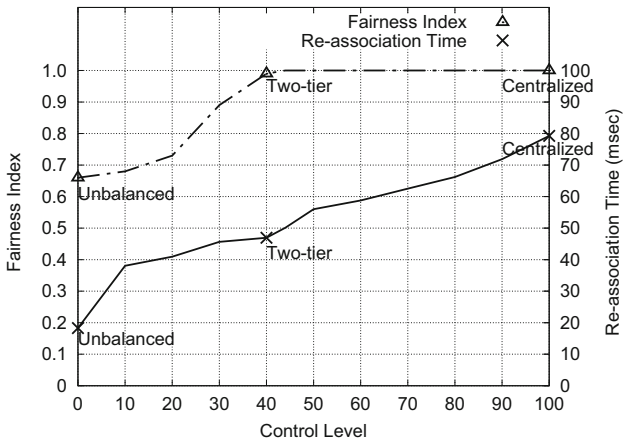

Fig. 6 The experiment results of the first scenario where the APs are initialized with the same load levels

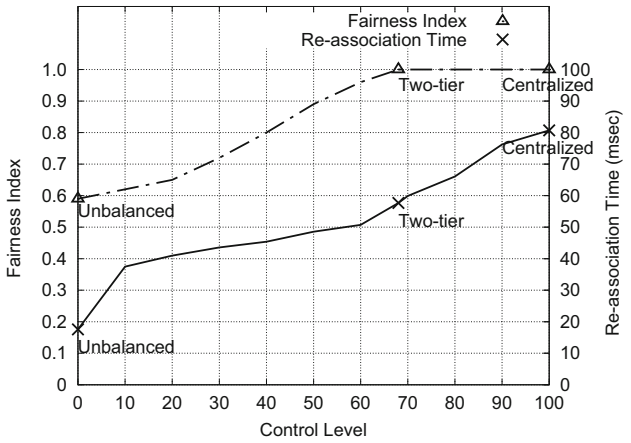

Fig. 7 The experiment results of the second scenario where the APs are initialized with different load levels

handed over from one AP to another; (3) a large re-association table causes a long data transfer time and a long computation process at the SDN controller; (4) the SDN controller should intelligently suggest the minimum control level at which the Wi-Fi network produces a fairness index close to 1 while minimizing the re-association time. Note that the case of (control level $\equiv 0$) represents an unbalanced Wi-Fi network whereas the case of (control level $\equiv 100$) resembles the conventional centralized scheme in which all the AP association decisions are 100% made by a centralized controller. In summary, Figs. 6 and 7 show that by automatically calculating the minimum control level, the proposed two-tier solution improves Wi-Fi's load balancing degree by 34–41%, and yields an improvement of 28–36% in Wi-Fi's re-association time over the conventional centralized load balancing scheme.

From the first experiment, we can see that the loading difference among the APs directly affects the resultant fairness index and re-association time because more client stations are handed over from heavily loaded APs to lightly loaded APs as the load diversity increases. Furthermore, for any Wi-Fi network with a specific loading difference value, the SDN controller shall be able to find out and suggest the corresponding minimum control level for the APs therein to use. To examine how such a suggested control level and the resultant re-association time evolve as the load diversity increases, we re-run the previous experiment with loading difference $= \{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, and exhibit the results in Table 3 and plot them in Figs. 8 and 9. As shown in the table and the figures, the suggested control level grows linearly with the loading difference, but the resultant re-association time grows from 48.9 to 62.12 ms and then drops to 55.74 as the loading difference increases from 0 to 60 then to 100. This is because the setup of (loading difference $\equiv 60$) incurs the most number of client handovers in the Wi-Fi network.

Some may wonder what would happen, in the proposed scheme, if a client cannot connect to another AP? Our results show that this scenario rarely occurs because the SDN controller will not instruct an over-loaded AP to de-associate its clients unless some under-loaded AP in the ESS can take over those clients. However, in case that a client cannot connect to any other AP, the client will simply re-associate with the old AP, and the whole process will incur in some delays.

Table 3 The suggested minimum control level and the resultant re-association time as the loading difference among the APs increases

Loading difference	0	10	20	30	40	50
APs' loading	(100, 100, 100)	(100, 100, 90)	(100, 100, 80)	(100, 100, 70)	(100, 100, 60)	(100, 100, 50)
Re-association time (ms)	48.9	51.02	53.96	56.73	58.11	61.95
Suggested control level	44	46	48	50	54	58
Loading diff.	60	70	80	90	100	
APs' loading	(100, 90, 50)	(100, 80, 50)	(100, 70, 50)	(100, 60, 50)	(100, 50, 50)	
Re-association time (ms)	62.12	59.08	58.62	57.34	55.74	
Suggested control level	58	60	62	64	66	

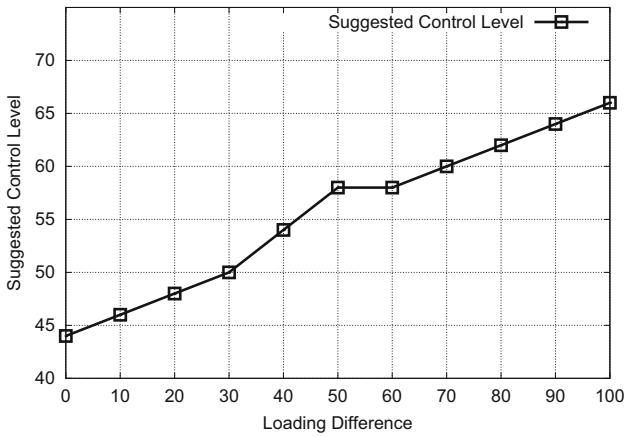

Fig. 8 The resultant minimum control level as a function of the loading difference among the APs

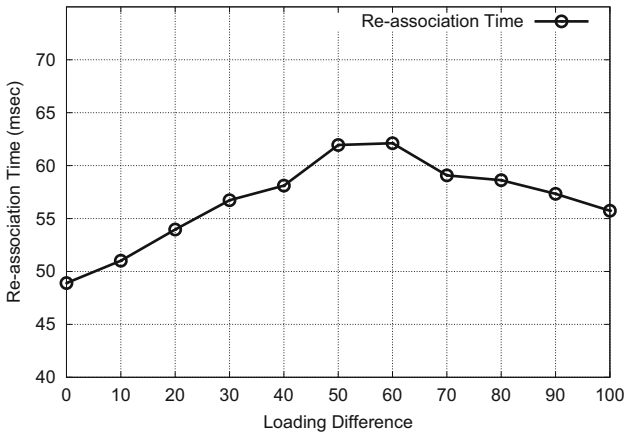

Fig. 9 The resultant re-association time as a function of the loading difference among the APs

6 Conclusions

Our two-tier Wi-Fi load balancing solution is hardware independent and applicable to any devices that support OpenFlow standards. The proposed solution is good for the management of network congestion because the network administrator can easily divide the APs into over-loaded and under-loaded groups, maneuver the control level parameter to balance the load among the APs while minimizing the re-association time of the Wi-Fi network. Moreover, because the APs and the SDN controller communicate through OpenFlow extension messages, developers can easily add new functions to our solution without changing the original SDN hardware/software architecture. From our experiment results, the proposed two-tier solution improves Wi-Fi's load balancing degree by 34–41%, and yields an improvement of 28–36% in Wi-Fi's re-association time over generic centralized load balancing methods with positive control.

In the future, we like to enhance the proposed two-tier solution by accounting for more factors such as traffic patterns of the associated devices, user priorities and QoS constraints in the load balancing decisions. Another possible future work is to extend the experiments to broader scenarios such as highly crowded Wi-Fi environment, users with different mobility, etc. There are also a couple of open issues that we like to examine in the future. First, we now use the Last-In-First-Out (LIFO) policy to pick client stations for de-association. It is of great relevance to further examine how alternative policies may affect or even improve the network performance. Second, it would be of interest to introduce different service classes and different user priority to our two-tier Wi-Fi load balancing solution.

References

1. Tang, D., & Baker, M. (1999). Analysis of a metropolitan-area wireless network. In *Proceedings of ACM mobicom* (pp 13–23).
2. Tang, D., & Baker, M. (2000). Analysis of a local-area wireless network. In *Proceedings of ACM mobicom* (pp. 1–10).
3. Balachandran, A., Voelker, G. M., Bahl, P., & Rangan, P. V. (2002). Characterizing user behavior and network performance in a public wireless LAN. In *Proceedings of ACM SIGMETRICS* (pp. 195–205).
4. Henderson, T., Kotz, D., & Abyzov, I. (2004). The changing usage of a mature campus-wide wireless network. In *Proceedings of ACM mobicom* (pp. 187–201).
5. Jardosh, A. P., Ramachandran, K. N., Almeroth, K. C., & Belding-Royer, E. M. (2005). Understanding congestion in IEEE 802.11B wireless networks. In *Proceedings of ACM IMC* (pp. 25–25).
6. Suri, S., Tóth, C. D., & Zhou, Y. (2007). Selfish load balancing and atomic congestion games. *Algorithmica*, *47*(1), 79–96.
7. Mittal, K., Belding, E. M., & Suri, S. (2008). A game-theoretic analysis of wireless access point selection by mobile users. *Computer Communications*, *31*(10), 2049–2062.
8. Xu, F., Tan, C. C., Li, Q., Yan, G., & Wu, J. (2010). Designing a practical access point association protocol. In *Proceedings of INFOCOM* (pp. 1361–1369).
9. Bigham, J., Wang, Y., & Cuthbert, L. G. (2004). Intelligent radio resource management for IEEE 802.11 WLAN. In *Proceedings of WCNC* (pp. 1365–1370).
10. Bejerano, Y., & Han, S.-J. (2009). Cell breathing techniques for load balancing in wireless LANs. *IEEE Transactions on Mobile Computing*, *8*(6), 735–749.
11. Stanley, D., Montemurro, M., & Calhoun, P. R. (2015). Control and provisioning of wireless access points (CAPWAP) protocol specification. IETF RFC 5415, October 14.
12. Balachandran, A., Bahl, P., & Voelker, G. M. (2002). Hot-spot congestion relief in public-area wireless network. In *Proceedings of WMCSA* (pp. 70–80).
13. Munos H. V., Aleo, V., & Karlsson, G. (2004). Load balancing in overlapping wireless LAN cells. In *Proceedings of IEEE ICC* pp. 3833–3836.

14. Bejerano, Y., Han, S.-J., & Li, E. L. (2007). Fairness and load balancing in wireless LANs using association control. *IEEE/ACM Transactions on Networking*, *15*(3), 560–573.
15. Duo, Z., & Chen, Z. (2009). Centralized wireless LAN load balancing. US Patent No. 7,480,264, January 20.
16. Iyer, P. J., Narasimhan, P., Andrade, M., & Taylor, J. (2011). System and method for centralized station management. US Patent No. 7,969,937, June 28.
17. Bernardos, C. J., de la Oliva, A., Serrano, P., Banchs, A., Contreras, L. M., Jin, H., et al. (2014). An architecture for software defined wireless networking. *IEEE Wireless Communications*, *21*(3), 52–61.
18. Cao, B., He, F., Li, Y., Wang, C., & Lang, W. (2015). Software defined virtual wireless network: Framework and challenges. *IEEE Network*, *29*(4), 6–12.
19. Yen, L.-H., Yeh, T.-T., & Chi, K.-H. (2009). Load balancing in IEEE 802.11 networks. *IEEE Internet Computing*, *13*(1), 56–64.
20. Vasudevan, S., Papagiannaki, K., Diot, C., Kurose, J., & Towsley, D. (2005). Facilitating access point selection in IEEE 802.11 wireless networks. In *Proceedings of IMC* (pp. 293–298).
21. Gong, H., & Kim, J. W. (2008). Dynamic load balancing through association control of mobile users in Wi-Fi networks. *IIEEE Transactions on Consumer Electronics*, *54*(2), 342–348.
22. Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software defined networking: A comprehensive survey. In *Proceedings of IEEE,* Vol. 103, No. 1 (pp. 14–76).
23. McKeown, N., Anderson, T. Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J. et al. (2008). OpenFlow: Enabling innovation in campus networks. *SIGCOMM Computer Communication Review*, *38*(2), 69–74.
24. Open Networking Foundation. OpenFlow switch specification version 1.3.0. https://www.opennetworking.org/, Jun 2012.
25. Costanzo, S., Galluccio, L., Morabito, G., & Palazzo, S. (2012). Software defined wireless networks: Unbridling SDNs. In *Proceedings of European workshop on software defined networking* (pp. 1–6).
26. Ali-Ahmad, H., Cicconetti, C., de la Oliva, A., Mancuso, V., Sama, M. R., Seite, P., & Sivasothy, S. (2013). An SDN-based network architecture for extremely dense wireless networks. In *Proceedings of IEEE SDN for future networks and services* (pp. 1–7).
27. Chiu, D.-M., & Jain, R. (1989). Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, *17*(1), 1–14.
28. EstiNet Technologies. http://www.estinet.com/.
29. Sriram, N. RYU Controller. http://sdnhub.org/tutorials/ryu/.

Ying-Dar Lin is a Distinguished Professor of computer science at National Chiao Tung University (NCTU), Taiwan. He received his Ph.D. in computer science from the University of California at Los Angeles (UCLA) in 1993. He was a visiting scholar at Cisco Systems in San Jose, California, during 2007–2008, and the CEO at Telecom Technology Center, Taipei, Taiwan, during 2010–2011. Since 2002, he has been the founder and director of Network Benchmarking Lab (NBL, www.nbl.org.tw), which reviews network products with real traffic and has been an approved test lab of the Open Networking Foundation (ONF) since July 2014. He also cofounded L7 Networks Inc. in 2002, which was later acquired by D-Link Corp. His research interests include network security, wireless communications, and network cloudification. His work on multihop cellular was the first along this line, and has been cited over 800 times and standardized into IEEE 802.11s, IEEE 802.15.5, IEEE 802.16j, and 3GPP LTE-Advanced. He is an IEEE Fellow (class of 2013), IEEE Distinguished Lecturer (2014–2017), and ONF Research Associate. He currently serves on the editorial boards of several IEEE journals and magazines, and is the Editor-in-Chief of IEEE Communications Surveys and Tutorials (COMST). He published a textbook, Computer Networks: An Open Source Approach (www.mhhe.com/lin), with Ren-Hung Hwang and Fred Baker (McGraw-Hill, 2011).

Chih-Chiang Wang obtained his Ph.D. degree in Computer Science from North Carolina State University, USA, in 2007. He is currently a faculty member in Department of Computer Science and Information Engineering at National Kaohsiung University of Applied Sciences in Taiwan. His research interests are in the general areas of network protocols, peer-to-peer systems, software-defined networking, and wireless networks.

Yi-Jen Lu graduated from the National Chiao Tung University (NCTU), Taiwan, and received her Master Degree of Computer Science in 2015. She currently works as a senior software engineer for ZYXEL, a network equipment supplier.

Yuan-Cheng Lai received his Ph.D. degree in computer science from National Chiao Tung University in 1997. He joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in 2001 and has been a professor since 2008. His research interests include wireless networks, network performance evaluation, network security, and social networks.

Hsi-Chang Yang was born in Taiwan in 1972. He received the B.S. degree from Chung Hua University and M.S. degree from National Chung Cheng University, Taiwan, in 1995 and 1997, respectively. Since then, he has been worked for Foxonn Inc. and Mediatek Inc., where he is currently the Senior Manager of Connectivity Technology Development of Chip Development Division. His development area includes Bluetooth, Wireless and Data Center system.