Load Aware Automatic Channel Switching for Software-Defined Enterprise WLANs

Yunong Han[1], Kun Yang[1,2]
[1]School of Computer Science and Electronic Engineering
University of Essex
Colchester CO4 3SQ, UK
[email: {hyunon, kunyang}@essex.ac.uk]
[2]School of Communication Engineering
University of Electronic Science and Technology of China
Chengdu, China

Received April 25, 2017; revised June 1, 2017; accepted June 21, 2017; published November 30, 2017

Abstract

In the last decade, the 2.4 GHz band of IEEE 802.11 WLANs has become heavily congested due to the explosive increase in demand of Wi-Fi connectivity. With the current deployment of enterprise WLANs, channel switching mechanism continues to exhibit inefficiencies because it cannot adapt to real-time channel condition and the inability to support seamless channel switching. Software Defined Networking (SDN) as an emerging architecture is promising to introduce flexibility and programmability for wireless network management. Leveraging SDN to existing enterprise WLANs, channel switching method can be improved significantly. This paper presents a software-defined enterprise WLAN framework with a load aware automatic channel switching solution, which utilizes AP load and channel interference factor (CIF) to provide seamless channel switching. Two automatic channel switching algorithms named Single Switch (SS) and Double Switch (DS) are proposed to improve the overall user experience and the experience of users with highest traffic load respectively. Experiment results demonstrate that our solution can efficiently improve user experience in terms of jitter, transmission delay and network throughout when compared to the conventional channel switching mechanism.

Keywords: SDN, enterprise WLANs, load aware, automatic channel switching

https://doi.org/10.3837/tiis.2017.11.003 **ISSN : 1976-7277**

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 Quality of Experience (QoE) for users becomes extremely difficult; moreover, the increasing trends such as bring your own device (BYOD), rapidly growing user data traffic and high network bandwidth demand only exacerbate this problem. The IEEE 802.11 WLANs [1] can operate on two unlicensed frequency spectrum bands which are 2.4GHz and 5GHz. **Fig. 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 center 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 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. 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 user to successfully send a data packet. The outcome of channel interference is the degradation on network performance, such as low network throughput, high packet loss rate and transmission delay, which can significantly affect the user experience.

In order to minimize the interference issue and guarantee the quality of service (QoS), optimized channel allocation mechanism is critical to modern wireless networks [2]. With traditional enterprise WLAN management, network administrators manually configure the wireless channel for APs based on detailed site surveys. Past research work [3], [4] 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 centralized automatic channel switching approach is needed, which can adapt to the rapidly changing wireless environment and perform real-time traffic load aware channel switching.

Software Defined Networking (SDN) [5] is an emerging architecture and intensively discussed as one of the most promising technologies to simplify network management and service development. OpenFlow [6] 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 centers and the cloud [7] in order to provide a more flexible network architecture based on

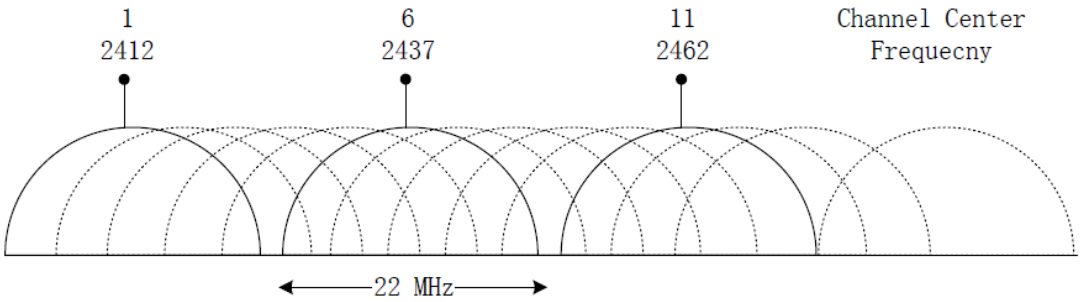

Fig. 1. 2.4GHz frequency band Wi-Fi Channels Overlapping

its simple per-flow management. In the SDN architecture, network management is directly programmable because it is decoupled from forwarding layer; essential network services such as load balancing, mobility management and bandwidth control, can be easily implemented by network administrators.

In this work, we present a software-defined enterprise WLAN system with a load aware automatic channel switching application for 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. We demonstrate 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.

The remainder of this paper is organized as following. 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 automatic channel switching algorithms in detail. The layout of experiment testbed and related performance evaluations are demonstrated in Section V, and finally, the conclusion of this paper is presented in Section VI.

2. Related Work

2.1 Channel Switching

Previous research works have proposed several channel switching schemes. In [8], the authors mentioned one approach called least congested channel switch. With this approach, each AP scans beacon frames that are sent by neighboring 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 utilization must be considered as well. The authors in [9] proposed a fully distributed channel selection algorithm that allows APs to select their

operating channel with the minimum received power in order to minimize interference. In [10], the authors introduced a distributed channel hopping mechanism called MaxChop. 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 [11] used graph coloring algorithm approach to effectively assign channels to WLAN APs. In [12], the authors introduced a channel allocation algorithm, which utilizes all the channels in the crowded Wi-Fi spectrum in a more efficient way that each frequency has a channel separation of 5. The authors in [13] proposed 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. A channel assignment algorithm that aims to maximize Signal-to-Interference Ratio (SIR) at the user level is presented in [14]. In [15], the authors proposed 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. The authors in [16] introduced a new framework to formulate the channel allocation as a min-max optimization problem regarding channel utilization. Based on the same framework, authors further proposed a new channel allocation algorithm that considers both link adaptation and power control mechanisms, in addition to the impact of co-channel interference (CCI) and multiple data rates in [17]. In [18] and [19], author introduced a delay estimation strategy and proposed a QoE-driven delay announcement scheme for media cloud, which improve the user's QoE dramatically.

2.2 SDN Wi-Fi

With the growing popularity of SDN, researchers have studied the emergence of SDN in enterprise WLANs. The authors in [20] introduced a novel SDN Wi-Fi architecture called CloudMAC. In CloudMAC, the Split-MAC design is used where the MAC layer processing is partially offloaded to virtual machines provided by cloud services. CloudMAC achieves similar performance as normal Wi-Fi but a higher level of abstraction and programmability. In [21], the authors presented EmPOWER, a set of high-level programming abstractions for managing wireless networks. The proposed abstractions modeling the fundamental aspects of a wireless network and hide away the implementation details of the underlying wireless technology which enable a flexible management of the network. Odin is introduced as a novel software-defined framework for enterprise WLANs in [22]. It uses Light Virtual Access Points (LVAP) as the user association abstraction which achieves seamless user mobility by migrating LVAPs. The authors in [23] presented a software-defined wireless network named AeroFlux. The system provides low-latency programmatic control of fine-grained wireless transmission settings and supports large enterprise WLAN deployment. By building upon Odin and AeroFlux, the authors in [24] introduced OpenSDWN, an SDN and Network Functions Virtualization (NFV) based novel Wi-Fi framework which exploits data path programmability to enable service differentiation and fine-grained transmission control. By inheriting the LVAP abstraction from Odin, the system achieves seamless mobility and mitigation. The authors in [25] presented a delay model for software-defined wireless virtual networks by using network calculus.

2.3 Channel Switching with SDN Wi-Fi

By utilizing the flexibility and programmability of SDN, software-defined enterprise WLANs provides a promising solution for researchers to solve the automatic channel switching problem. In [26], the authors presented an intelligent SDN-based solution for enterprise

WLANs named ISD-WiFi. They also proposed an interference graph and traffic oriented channel assignment algorithm priorly assign channels to APs with high interference. The authors in [27] introduced an SDN-based channel assignment algorithm which takes the network topology, the current channels assignment across all APs and the orthogonality and overlapping of the channels into account to make channel assignment. The authors in [28] presented high-level abstractions for channel quality and interference with an interference-aware channel assignment approach. In [29], the authors further implement varieties of network services on top of Odin 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 center frequency. Based on multiple subsequent spectral scans, it picks the channel with the smallest maximum and average RSSI.

One common feature of these works is that none of them is adaptive to the real-time AP load to provide seamless automatic channel switching. The conventional automatic channel selection method is AP-driven and it requires the reboot of WLAN interface during the process of channel switching which leads to a short period of service disruption. Users 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. Therefore, it is vital for the channel switching application to be adaptive to the real-time network state and perform seamless channel switching.

3. System Design

In order to effectively implement high-level applications in enterprise WLANs, a central view of the entire network is essential. By utilizing the characteristics of SDN, a three-layer software-defined enterprise WLAN system is designed and implemented. The architecture of the designed system is depicted in **Fig. 2**.

3.1 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 [30] and Click Modular Router [31]. 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. To do so, two protocol messages called FORWARD-AP-LOAD and FORWARD-AP-CHAN are defined. The format of these two protocol messages is shown in **Fig. 3** and **Fig. 4**. By introducing these two messages, the updated load level of each physical AP and its best operating channel 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 utilization value.

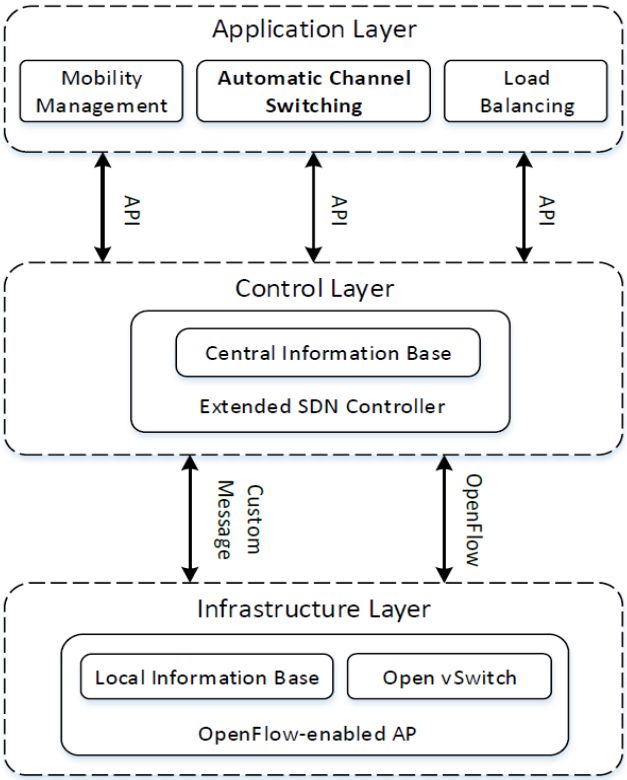

Fig. 2. Architecture of designed Software-Defined Enterprise WLAN

3.2 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 running on physical APs to perform automatic channel switching.

3.3 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 centralized 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. 5** illustrates an example of the automatic channel switching process and relevant protocol messages that are exchanged in the designed software-defined enterprise WLANs architecture. In order to support seamless channel switching, beacon frames contain channel switch announcement (CSA) are broadcast to

associated users before the AP switches its channel. The format of the CSA element includes three fields which are channel switch mode, new channel number, and channel switch count. The channel switch mode is set to either 0 or 1 which indicates any restrictions on transmission until a channel switch happens. The new channel number stores the new operating channel of the AP, and the channel switch count is the value of remaining beacon frames that will be broadcast on the current operating channel.

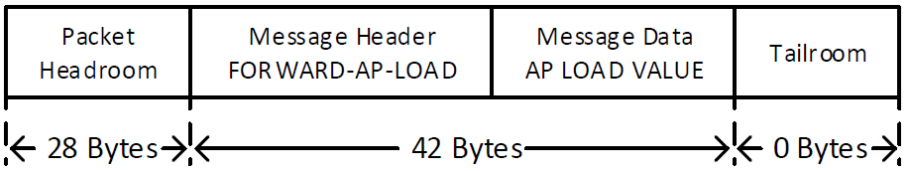

Fig. 3. Format of protocol message FORWARD-AP-LOAD

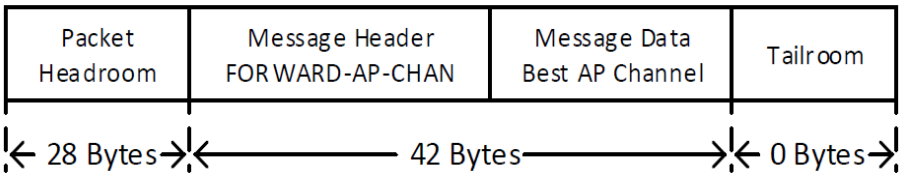

Fig. 4. Format of protocol message FORWARD-AP-CHAN

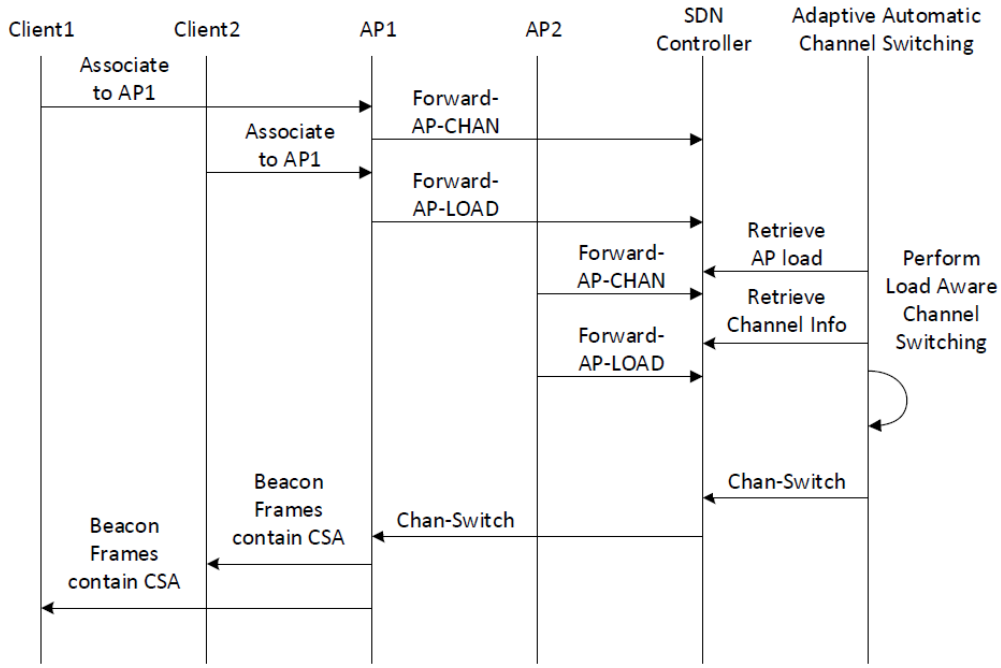

Fig. 5. Sequence diagram of protocol messages for proposed automatic channel switching

4. Proposed Automatic Channel Switching Algorithms

4.1 AP Load

In order to support load aware automatic channel switching, the AP load is introduced as a channel switching metric in this work. 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, we define 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 (1) and the AP load is calculated by equation (2). We give 80% weight to the channel load as it makes the bigger impact on user experience when compared with the number of associated users that is given 20% weight. 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. To calculate the AP load, we modified the ath9k based Wi-Fi driver to expose two debugfs based user space interfaces which enable the LIB to read the value of both registers.

$$ChannelLoad = \frac{ChannelBusyTime}{MeasurementDuration} \tag{1}$$

$$APLoad = \begin{cases} ChannelLoad & (N_{au} = 0) \\ 0.8 \times ChannelLoad + 0.2 \times N_{au} & (N_{au} > 0) \end{cases} \tag{2}$$

$$APLoad_f = 0.9 \times APLoad_f + 0.1 \times APLoad_{f-1} \tag{3}$$

An exponential weighted moving average (EWMA) is used to dealing with AP load value in that they are effected by sudden variation of channel load. We give 90% weight to the current $APLoad_f$ and therefore the final value of AP load is achieved with equation (3)

4.2 Channel Interference Factor

In this work, we define the Channel Interference Factor (CIF) as an approximation of the channel quality. The channel with the minimum value of CIF is selected as the best operating channel for automatic channel switching. In order to calculate the CIF, the signal to noise ratio (SNR) have to be considered. SNR is defined as the ratio of the power of received signal and the background noise level (noise floor) with equation (4). 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. As 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. Therefore, 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, we define a weight factor *w* where its value is decided by the RSSI of the AP that operated on overlapped channel. In order to find out the effect of RSSI on network throughput, we conducted a simple experiment by associating one user to AP1 to test the throughput under the different level of RSSI. Experiment result is shown in **Fig. 6**. 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 experiment results, we defined three level of *w* which is high with RSSI higher than -69dBm, medium

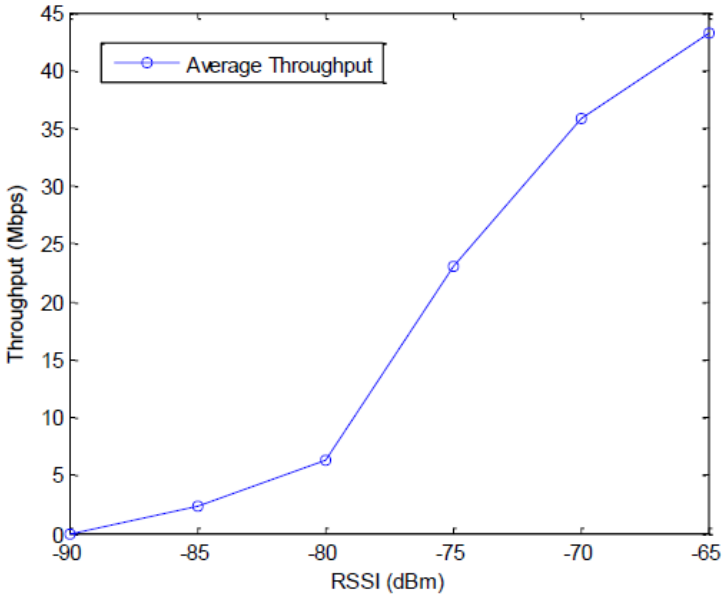

Fig. 6. Average throughput over RSSI

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.

$$SNR_{dB} = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \tag{4}$$

$$CIF_i = \begin{cases} \dfrac{CO_i + AD_{j,k}}{N_{i,k}} & (i=1,\ j=2,k=5) \\ \dfrac{CO_i + AD_{j,k}}{N_{j,k}} & (i=6,\ j=2,k=10) \\ \dfrac{CO_i + AD_{j,k}}{N_{j,i}} & (i=11,\ j=7,k=10) \end{cases} \tag{5}$$

$$CO_i = CU_i \times \sum_{x=1}^{n} \left| RSSI_{i,x} \right| \tag{6}$$

$$AD_{j,k} = \sum_{c=j}^{k} \left(CU_c \times \sum_{x=1}^{n} \left(\left| RSSI_{c,x} \right| \times w_x \right) \right) \tag{7}$$

$$CIF_f = 0.9 \times CIF_f + 0.1 \times CIF_{f-1} \tag{8}$$

Because only the three non-overlapping channels are considered for proposed channel switching mechanism, the CIF is then calculated by equation (5) where *i* is the channel number and $N_{i,k}$ is the total number of APs that operate on channels from *i* to *k*. CO_i is the absolute value of the co-channel interference of channel *i* which is achieved by equation (6) where CU_i is the channel utilization of channel *i*. $AD_{j,k}$ which is denoted by equation (7) is the total adjacent channel inference of APs that operated on channel range from *j* to *k*. 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. 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 (8). 90% weight is given to the current CIF_f.

4.3 Proposed SS and DS Algorithms

Based on above metrics, two 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 1 and 2 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}. We configure $load_t$ with 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 highest load value have the 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.

Algorithm 1: Single Switch Algorithm

```
1: while true do
2:     Sleep Timer_i
3:     Sort LoadMAP_ap by load_max
4:     for load_ap ∈ 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 ≠ 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
```

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 automatic channel switching application iterates the AP load map to check the AP 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 which further improves 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.

Algorithm 2: 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 ≠ CH_MAXcurrent) then
9:             for load_ap ∈ LoadMAP_ap do
10:                Get CH_APcurrent of AP with load_ap
11:                if (CH_APcurrent = CH_MAXbest) then
12:                    Break loop
```

end if
end for
ChanSwitch($CH_{MAXcurrent}$) for AP $load_{ap}$
ChanUpdate($CH_{MAXcurrent}$) for AP $load_{ap}$
ChanSwitch($CH_{MAXbest}$) for AP $load_{max}$
ChanUpdate($CH_{MAXbest}$) for AP $load_{max}$
else
AP remain at $CH_{MAXcurrent}$
end if
end if
end while

5. Performance Evaluation

5.1 Experiment Setup

To set up the experiment, we deployed the testbed on the 5th floor of the networking building. More specifically, five APs are deployed in different rooms with one normal AP functions as the default gateway and the rest four OpenFlow-enabled APs perform as experiment APs for channel switching test. In addition, one desktop is deployed to work as the Floodlight SDN controller with the CIB and the proposed automatic channel switching application running on top of it. The layout and the initial channel configuration of the experiment APs are depicted in **Fig. 7**. A normal layer 2 switch is used to connect the default gateway AP and the four OpenFlow-enabled APs. The default gateway AP runs the DHCP server which assigns IP address for the APs and user devices in the system. **Fig. 8** briefly shows the 2.4GHz Wi-Fi channel usage information of the experiment environment which is achieved by the android app 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 the test. **Table. 1** shows the information of experiment devices and initial configurations.

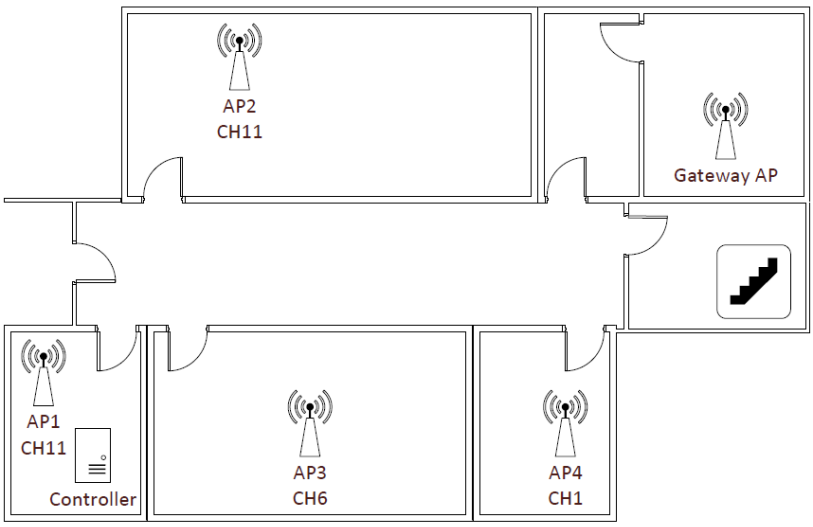

Fig. 7. Layout of the current experiment testbed

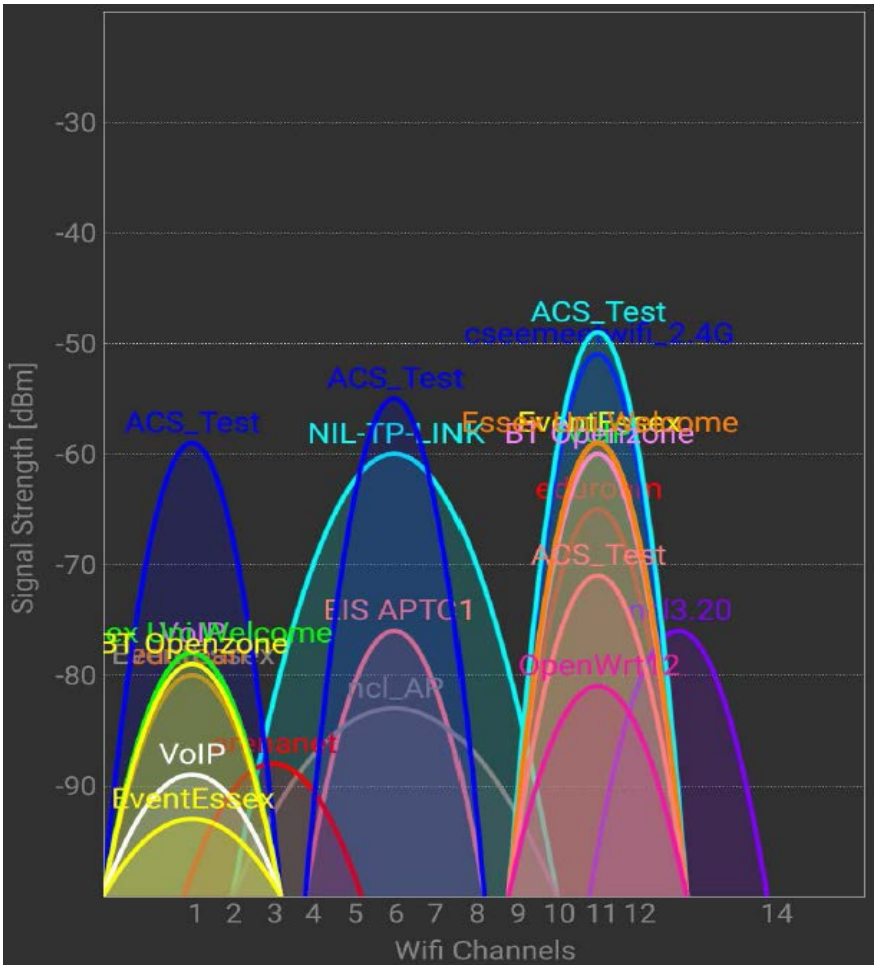

Fig. 8. 2.4GHz band channel information of experiment environment

Table 1. Experiment devices and initial configurations

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
OS of User Devices	Android 5.0 and Linux Ubuntu 14.04
Number of User Devices	4
User Traffic	Iperf based UDP and TCP Test
SDN Controller	Floodlight 1.0
Integrated Software	Open vSwitch and Click Modular Router 2.0.1

5.2 Performance Evaluation

Five experiments are conducted to investigate the performance of the proposed SS and DS automatic channel switching algorithms. Transmission Control Protocol (TCP) tests based on iperf are conducted to evaluate the average and total network throughput, while User Datagram Protocol (UDP) tests based on Iperf are conducted to measure the jitter and delay of data transmission.

1) Seamless Channel Switching: The first experiment is conducted to evaluate the performance of providing seamless channel switching with proposed automatic channel switching algorithm. To set up this experiment, the iperf TCP server is running on AP1. Only one user device is associated to 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. 9** shows the experiment 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 is because the WLAN interface is required to reboot in order to switch to the new channel 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 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.

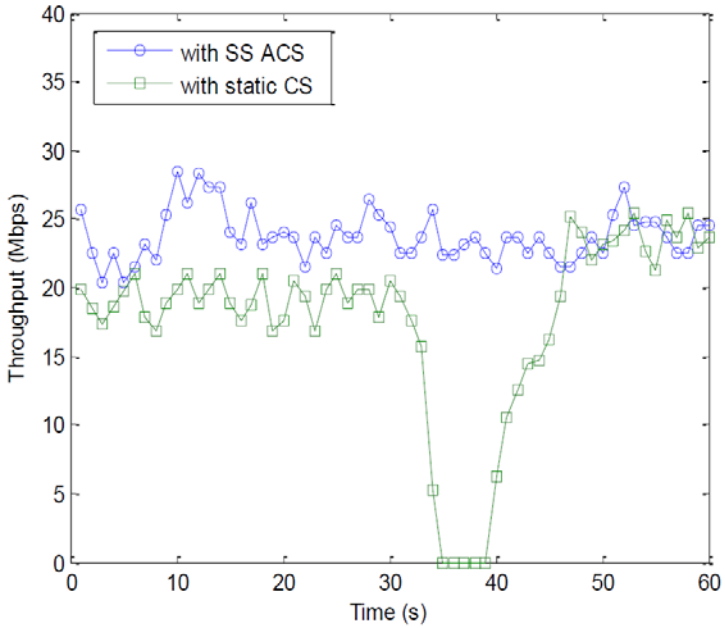

Fig. 9. Single user throughput SS vs static method

2) Throughput Evaluation with SS algorithm: In the second experiment, the efficiency on improving average user throughput with 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. 10** 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 is 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 the channel switching with 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.

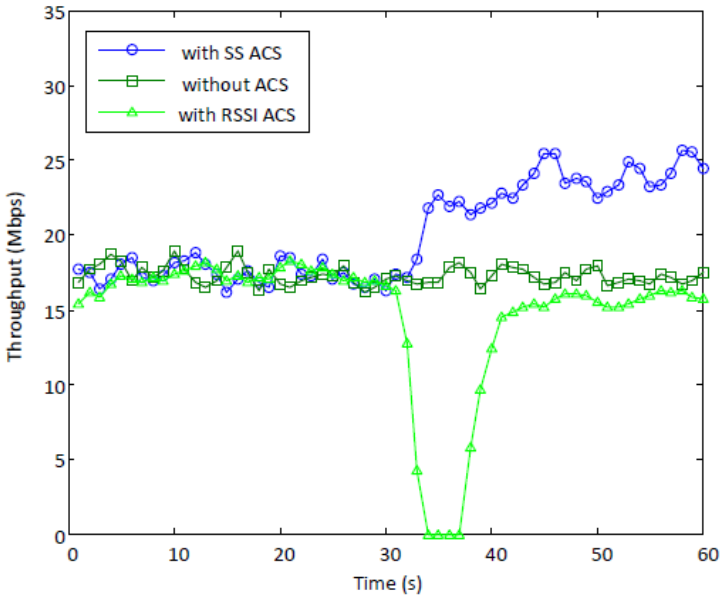

Fig. 10. Average user throughput comparison SS vs RSSI

3) Delay and Jitter Evaluation with SS Algorithm: The efficiency on 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. 11** shows results of average jitter and delay achieved with 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.

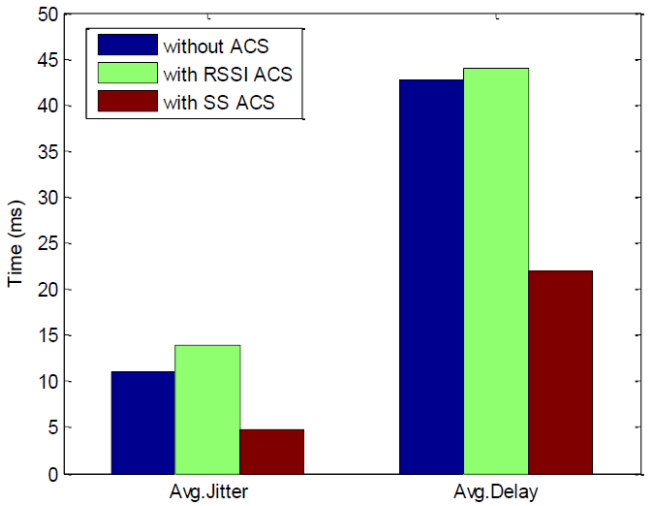

Fig. 11. Average jitter and delay comparison SS vs RSSI

4) Throughput Evaluation with DS Algorithm: The fourth experiment is conducted to evaluate the efficiency on improving total user throughput with 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 value are added to get the final results. **Fig. 12** 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.

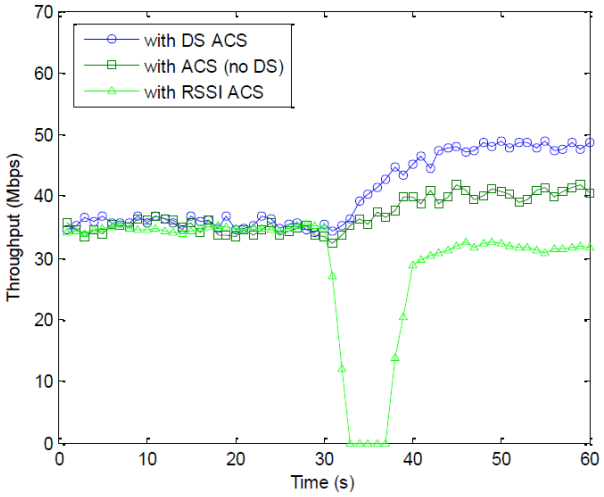

Fig. 12. Total user throughput comparison DS vs RSSI

5) Delay and Jitter Evaluation with DS Algorithm: In the fifth experiment, the efficiency on improving jitter and transmission delay with 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, single switching method, and RSSI based algorithm are shown in **Fig. 13**. As it can be observed, both jitter and delay are much higher with RSSI based algorithm. This is 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 channel switching, both jitter and delay are decreased significantly which is reflected in the experiment results.

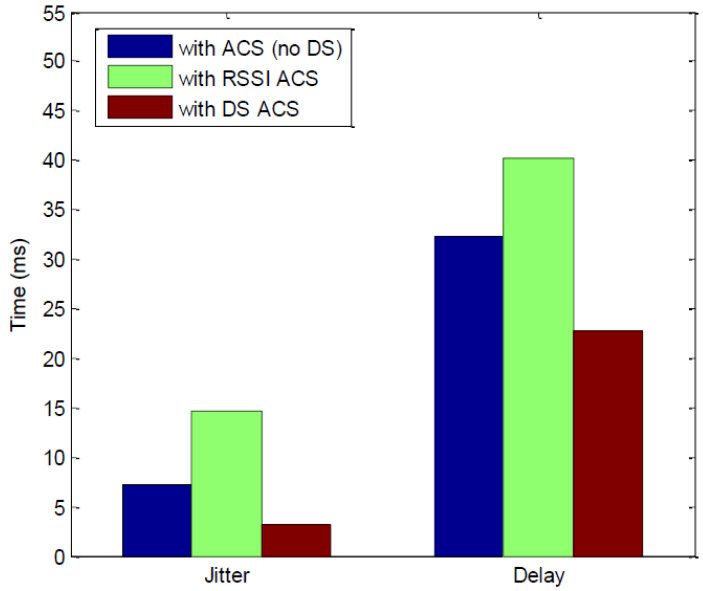

Fig. 13. total jitter and delay comparison DS vs RSSI

6. Conclusion

This work presents a software-defined enterprise WLANs framework with an automatic channel switching application. By utilizing the AP load value and the channel interference factor as channel switching metrics, our proposed SS and DS channel switching algorithms achieve seamless channel switching which adapts to real-time traffic load and channel condition. Experiment results show the effectiveness and efficiency of proposed SS and DS channel switching algorithms in terms of improving jitter, transmission delay, and network throughput compared to traditional channel switching method. Future work includes further extending the system to support Quality of Service and to implement a more diverse set of enterprise WLAN-specific applications.

Acknowledgment

This work was partially funded by UK EPSRC Project NIRVANA (EP/K002643/1), EU FP7 Project CROWN (GA- 2013-610524), Natural Science Foundation of China (Grant No. 61572389, 61620106011) and International Science & Technology Cooperation Program of China under grant No. 2015DFE12860.

References

[1] "IEEE Standard, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1-3534, Dec. 2016. Article (CrossRef Link).

[2] 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, no. , pp. 980-992, 2016. Article (CrossRef Link).

[3] Y. Lee, K. Kim, and Y. Choi, "Optimization of AP Placement and Channel Assignment in Wireless LANs," in *Proc. of the 27th Annual IEEE Conference on Local Computer Networks*, ser. LCN '02. Washington, DC, USA: IEEE Computer Society, pp. 0831–, 2002. Article (CrossRef Link).

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

[5] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," Apr 2012, White Paper. Article (CrossRef Link).

[6] 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. Article (CrossRef Link).

[7] X. Tang, P. Zhang, C. Cao. "SDN-Based Broadband Network for Cloud Services", *J. of ZTE Communications*, Volume 12, Issue 2, pp. 18-22, Dec. 2014. Article (CrossRef Link).

[8] 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. Article (CrossRef Link).

[9] 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 *Proc. of IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pp. 1451–1459, May 2007. Article (CrossRef Link).

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

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

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

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

[14] M. Haidar and R. Ghimire and H. Al-Rizzo and R. Akl and Yupo Chan, "Channel assignment in an IEEE 802.11 WLAN based on Signal-To-Interference Ratio," in *Proc. of 2008 Canadian Conference on Electrical and Computer Engineering*, pp. 001 169–001 174, May 2008. Article (CrossRef Link).
[15] 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. Article (CrossRef Link).
[16] 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 *Proc. of IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*, pp. 4893–4898, Nov. 2007. Article (CrossRef Link).
[17] 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. Article (CrossRef Link).
[18] L. Zhou, "On Data-Driven Delay Estimation for Media Cloud," in *IEEE Transactions on Multimedia*, vol. 18, no. 5, pp. 905-915, May 2016. Article (CrossRef Link).
[19] L. Zhou, "QoE-Driven Delay Announcement for Cloud Mobile Media," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 1, pp. 84-94, Jan. 2017. Article (CrossRef Link).
[20] J. Vestin, P. Dely, A. Kassler, N. Bayer, H. Einsiedler and C. Peylo, "CloudMAC: Towards Software Defined WLANs," in *Proc. of the 18th Annual International Conference on Mobile Computing and Networking*, ser. Mobicom '12. New York, NY, USA: ACM, pp. 393–396, 2012. Article (CrossRef Link).
[21] 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. Article (CrossRef Link).
[22] 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. Article (CrossRef Link).
[23] J. Schulz-Zander, N. Sarrar, and S. Schmid, "Towards a Scalable and Near-sighted Control Plane Architecture for WiFi SDNs," in *Proc. of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14. New York, NY, USA: ACM, pp. 217–218, 2014. Article (CrossRef Link).
[24] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, and A. Feldmann, "OpenSDWN: Programmatic Control over Home and Enterprise WiFi," in *Proc. of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, ser. SOSR '15. New York, NY, USA: ACM, pp.16:1–16:12, 2015. Article (CrossRef Link).
[25] L. Zhang, J. Liu, K. Yang. "Virtualized Wireless SDNs: Modelling Through the Use of Stochastic Network Calculus," *J. of ZTE Communications*, Volume 12, Issue 2, pp. 50-56, Dec 2014. Article (CrossRef Link).
[26] D. Tu and Z. Zhao and H. Zhang, "ISD-WiFi: An intelligent SDN based solution for enterprise WLANs," in *Proc. of 2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, pp. 1–6, Oct. 2016. Article (CrossRef Link).
[27] M. Seyedebrahimi and F. Bouhafs and A. Raschell and M. Mackay and Q. Shi, "SDN-based channel assignment algorithm for interference management in dense Wi-Fi networks," in *Proc. of 2016 European Conference on Networks and Communications (EuCNC)*, pp. 128–132, June 2016. Article (CrossRef Link).
[28] R. Riggio and M. K. Marina and T. Rasheed, "Interference management in software-defined mobile networks," in *Proc. of 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 626–632, May 2015. Article (CrossRef Link).
[29] J. Schulz-Zander, L. Suresh, N. Sarrar, A. Feldmann, T. Hu¨ hn, and R. Merz, "Programmatic Orchestration of WiFi Networks," in *Proc. of the 2014 USENIX Conference on USENIX Annual*

Technical Conference, ser. USENIX ATC'14. Berkeley, CA, USA: USENIX Association, pp.347–358, 2014. Article (CrossRef Link).

[30] 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 *Proc. of 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, pp. 117–130, May 2015. Article (CrossRef Link).

[31] 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. Article (CrossRef Link).

Yunong Han received the B.Sc. degree in computing and network computing, and the M.Sc. degree in broadband networking from Oxford Brookes University ,Oxford, U.K., in 2010 and 2011 respectively, and is currently pursuing the Ph.D. degree in computer science and electronic engineering at University of Essex, Colchester, U.K. His research interests include wireless local area networks, software-defined networking, and software defined wireless networks.

Kun Yang received the B.Sc. and M.Sc. degrees in computer science from Jilin University, Changchun, China, and the Ph.D. degree in electronic and electrical engineering from the University College London (UCL), London, U.K., in 1991, 1994, and 2006, respectively. He is currently a Chair Professor with the School of Computer Science and Electronic Engineering, University of Essex, Colchester, U.K., leading the Network Convergence Laboratory (NCL), UK. Before joining the University of Essex in 2003, he worked with UCL on several European Union (EU) research projects for several years. He has authored more than 80 journal papers. His research interests include wireless networks, network convergence, future Internet technology and network virtualization, mobile cloud computing, and networking. Prof. Yang is a Fellow of IET. He serves on the Editorial Boards of both the IEEE and non-IEEE journals.

Dongdai Zhou obtained his BEng and MSc, both from Changchun University of Science and Technology, China, in 1992 and 1997 respectively. He received the Ph.D. degree from the Jilin University, China in 2001. He is currently a professor at School of Information Science and Technology, Northeast Normal University, China. His research interests include software architecture and software code auto-generation, especially the architecture design and codeless development of e-learning software.