Leveraging SDN and OpenFlow to Mitigate Interference in Enterprise WLAN

Dong Zhao, Ming Zhu, and Ming Xu
College of Computer, National University of Defense Technology, Changsha, China
Email: {dongzhao, zhuming, xuming}@nudt.edu.cn

***Abstract*—Today's enterprise WLAN is facing challenges as the rapid growth of user scale and traffic load. Users often experience slow or even intermittent connection in crowded area. This is mainly due to the interference among densely-deployed access points (APs). In this paper, we took advantages of the emerging idea of SDN and OpenFlow technology to mitigate interference in enterprise WLAN. Specifically, we proposed an OpenFlow-based framework for enterprise WLAN. In this framework, a central controller takes control over all the APs via OpenFlow interface. By installing appropriate rules on OpenFlow-enabled APs, the controller can realize fine-grained scheduling of APs' downlink packets. Based on such framework, we proposed a scheduling algorithm to obtain high packet reception rate so that the efficiency of DCF can be improved. Simulation results demonstrate that our solution can significantly increase network throughput and reduce retransmission rate. Moreover, since our solution preserves conventional DCF in 802.11 standard, no modification is required to existing 802.11 clients, which makes our solution practical.**

***Index Terms*—SDN; OpenFlow; Enterprise WLAN; DCF; Interference Mitigation**

I. INTRODUCTION

As Wi-Fi devices become universal, people find increasing number of access points (APs) surrounding them. Enterprise WLANs, with a number of APs that are interconnected by wired backbone, are widely deployed in public places to provide Internet-access for users. To provide continuous coverage, APs are usually placed with considerable overlap among one another. As the expansion of user scale and the increase of traffic load, APs become increasingly denser. Network throughput, however, does not increase proportionally. This is mainly due to the interference among APs and clients.

Basic channel access mechanism in 802.11, Distributed Coordination Function (DCF), has been proved to be unsuitable for high-density wireless network. In crowded area with densely-deployed APs, high collision probability leads to low efficiency of channel access, and thus too much time is spent in dealing with collision for retransmission. This significantly affects networks performance and user's experience.

To improve DCF's efficiency and mitigate the impact of interference in crowded area, many extensions to DCF have proposed. RTS/CTS virtual carrier sense was proposed by Karn, P. [1] to solve notorious "*hidden terminal*" problem and reduce packet collision rate. However, RTS/CTS mechanism brings extra overhead and sometimes leads to congestion [2]. Given that, some adaptive variants of RTS/CTS mechanism [3], [4] were used to reduce the overhead of RTS/CTS and further improve the efficiency of DCF. Many studies also focus on optimizing DCF's efficiency through several other aspects, including adaptive carrier sense threshold [5], multi-channel MAC protocol [6] and reduced slot size [7].

In addition to extensions of DCF, people proposed some other methods to reduce the overhead of retransmission (e.g., ZigZag decoding [8] and CSMA/CN mechanism [9]). These solutions are based on packet recovery techniques in physical communication.

In spite of the improvement above, DCF still cannot avoid collision completely. In crowed area with heavy traffic load, the competition among APs is so fierce that transmission failure and retransmission increase sharply. Therefore, the adverse effect of DCF's limitation on network performance is exacerbated and the optimization described above become less efficient.

Given the ingrained limitation of DCF, some people assert that DCF is obsolete and suggest adopting centralized channel access mechanism in enterprise WLAN. In practice, there exists a centralized channel access mechanism, i.e., Point Coordination Function (PCF), in 802.11 standard. Although PCF provides contention-free access within network, it doesn't apply to scenario where there exists other nearby networks. In addition, PCF was designed to be used in single-AP network, and cannot be directly used in enterprise WLAN that has many APs. MiFi [10] augments PCF to multi-AP scenarios, but it mainly focuses on fairness problem, rather than mitigating interference.

We conclude that any solution that requires modification to client is hard to be widely deployed. This is because of the ubiquitous of 802.11 terminal devices. Therefore, DCF should be preserved to keep backward compatibility.

The emerging Software-Defined Networking (SDN) [11], which is a revolutionary networking paradigm, can be leveraged to mitigate interference in enterprise WLAN. Benefiting from the decoupling of data plane and control plane, SDN greatly simplifies network management, and provides operators a programmable platform for rapid deployment of new features.

doi:10.4304/jnw.9.6.1526-1533

SDN offers a promising alternative architecture for enterprise WLAN to address the issues of interference mitigation. In this paper, we proposed an OpenFlow [12]-based framework for enterprise WLAN. In this framework, central controller can realize fine-grained downlink packets scheduling to promote the efficiency of DCF by installing appropriate rules. Based on OpenFlow-based framework, we presented a downlink packets scheduling algorithm. The core idea of the algorithm is to maintain moderate competition among APs to achieve high packet reception rate. As a result, retransmission is reduced and wasted time is minimized.

It is worth noting that our solution requires no modification to 802.11 clients. We preserve as much features of 802.11 standard as possible. Thus, little hardware modification to APs hardware is needed.

The main contribution of this paper includes:

- We propose an OpenFlow-based framework of enterprise WLAN. This framework leverages SDN and OpenFlow technology to conduct downlink packets scheduling for interference mitigation. Our solution requires no modifications to client, which make it quite practical and easily deployed.
- Benefiting from the OpenFlow-based framework, we propose a packet scheduling algorithm. By keeping competition not very fierce, this algorithm can significantly improve the efficiency of DCF, and network performance.
- We evaluate the impact of scheduling algorithm on interference mitigation through simulation. The results demonstrate that the feasibility and efficiency of our solution.

II. Background and Motivation

A. Distributed Channel Access Mechanism in 802.11

In enterprise WLAN, all the stations, including APs and clients, should contend for channel access opportunity. Due to the limited number of available non-overlapping channel in 802.11, it is impossible to ensure that APs in close proximity work on different channel when the density of APs is very high. Interference coordination mechanism used in 802.11 standard, i.e., DCF (Distributed Coordination Function), has poor performance in high-density area, where APs are closely deployed and users are crowded.

DCF is primary channel access mechanism in 802.11 standard. However, DCF is not suitable for crowed area. In DCF, station contends for the opportunity to grab the channel in a distributed fashion. However, DCF itself cannot avoid the occurrence of collision. As shown in Fig. 1, if collision occurs and packet fails to be decoded by the receiver, the time used to contend for channel access opportunity and packet transmission, along with waiting for timeout, is a waste. In crowded area, completion is very fierce, and the probability of collision occurrence is very high, leading to low channel utilization. In this case, the throughput of users will be decreased and the retransmission rate will be very high.

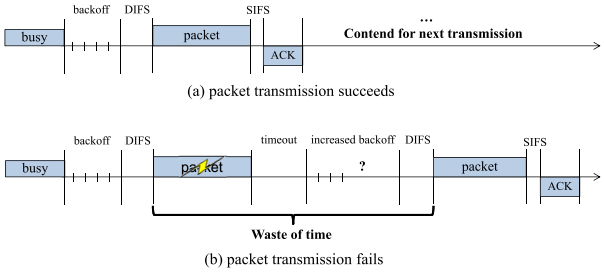

Figure 1. Timeline of DCF

B. SDN and OpenFlow

Software Defined Networking (SDN) [11] has attracted considerable attention from both academia and industry. SDN extracts intelligence from network devices and places control logic onto a centralized controller. In SDN, original various network devices with complex functions are reduced to some uniform SDN switches. Benefiting from the decoupling of data plane and control plane, SDN greatly simplifies network management and provides operators a programmable platform for rapid deployment of new features.

SDN envisions a three-layer architecture, including infrastructure layer, control layer, and application layer (as shown in Fig. 2). Infrastructure layer consists of switches. Control layer is a centralized controller that conducts control over switches via standardized interface (e.g., OpenFlow [12] and ForCES [13]). Controller implements some services to facilitate the development of new features. Application layer includes applications that leverage API provided by controller to implement a variety of functions.

According to basic principle of SDN, OpenFlow [12] embodies SDN switch and SDN controller, and standardizes the communication protocol between them. In OpenFlow, SDN switch, or OpenFlow switch, contains some *flow tables* that consist of a set of rules. Each packet received by OpenFlow switch will be processed according to these rules. OpenFlow defines interfaces to add, delete, and modify rules in flow table. By installing appropriate rules, controller implements various network functionalities. In recent year, SDN and OpenFlow technology have been widely adopted to solve various problems [14], [15].

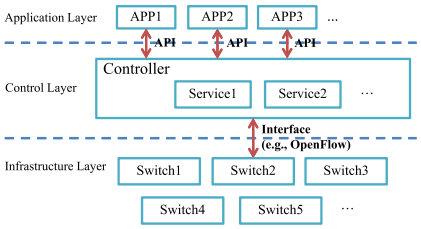

Figure 2. SDN: three-layer architecture

C. Scheduling Downlink Packets via OpenFlow

As we discussed in prior section, DCF performs quite well when contention rarely happens. We seek to

maintain a moderate competition among APs. To achieve this, we need a downlink packets scheduling mechanism. Since downlink traffic accounts for nearly 80% of all the traffic in enterprise WLAN [16], optimization of downlink traffic will significantly improve of the whole network's performance.

In OpenFlow, whether or not a switch shall forward a packet is determined by controller. We take advantage of this property to construct an OpenFlow-based framework for enterprise WLAN in which we can perform fine-grained downlink packet scheduling.

III. OpenFlow-based Framework for Enterprise WLAN

A. Overview

In this section, we will describe how to apply SDN and OpenFlow technology to enterprise WLAN and provide a flexible AP control platform to mitigate interference among APs by scheduling downlink packets in a fine-grained fashion.

As shown in Fig. 3, the OpenFlow-based framework includes a centralized OpenFlow controller, a set of OpenFlow-enabled access points (APs). All the APs are at the behest of the controller. Network functionalities are implemented as applications running on top of controller. This architecture not only simplifies the management and control of APs, but provides a flexible platform to realize the coordination among APs for interference mitigation.

B. OpenFlow-enabled Access Point

In the proposed framework, AP has been greatly simplified relative to original 802.11 AP. Specifically, each AP has at least two network interfaces: an 802.11 interface that communicates with 802.11 clients and an Ethernet interface that is connected to wired backbone (as shown in Fig. 4). Packets that received from client and delivered by wired backbone will be forward from one side to another side. Whether or not to forward any packet is determined by looking up the flow tables.

Flow table, which consists of a set of rules, is maintained by controller through OpenFlow interface. If OpenFlow-enabled AP doesn't find any matching rules in flow table for a packet, it will ask controller how to deal with the packet.

As you can see, our design requires no modifications to 802.11 clients. Furthermore we don't modify 802.11 MAC protocol. Therefore, the advantages of DCF are preserved, and the compatibility with off-the-shelf devices is maintained.

The scheduling of OpenFlow controller coordinates the transmission between APs. The Responsibility of scheduling algorithm operating in controller is to reduce the occurrence of conflicts and retransmission and resulting overhead.

C. Downlink Packets Scheduling

As described above, flow table in every AP determines whether or not to send a specified packet and we argue that we can realized downlink packets scheduling in the proposed framework to mitigate interference among APs. When a downlink packet from Internet arrives at an AP, the AP won't immediately forward the packet to its wireless interface. Instead, AP will check the flow table to find whether or not to forward the packet. In the first place, it cannot find any matching rules for the packet, so it will send an OpenFlow message to the controller. As a result, the controller knows the arrival of any packet. The controller can easily collect relevant information from all the APs, constructing a global view of the whole network, and decides which packet to be sent and when to send. APs won't send the packet until it receives the message from the controller to modify corresponding rules in flow table.

DCF is still used by the packet transmission between APs and clients: once find matching rule to “output” the packet, AP forward the packet to 802.11 wireless interface, which sends the packet to corresponding client using DCF mechanism. By collecting statistical information in flow table, the controller is also informed that whether or not a downlink packet is successfully sent by an AP's wireless interface.

Generally speaking, by taking advantage of the proposed OpenFlow-based framework, and installing appropriate rules in appropriate APs, we can realize a downlink packets scheduling in enterprise WLAN to improve the efficiency of DCF mechanism.

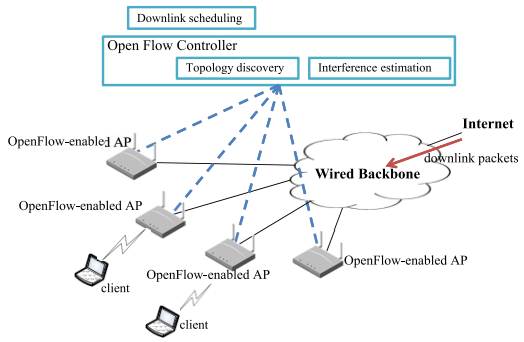

Figure 3. OpenFlow-based framework of enterprise WLAN

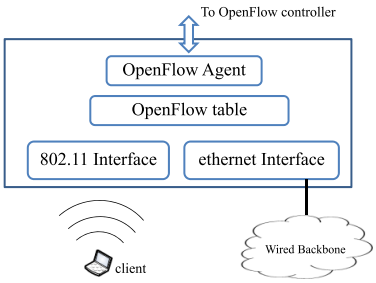

Figure 4. OpenFlow-enabled access point

IV. Downlink Packets Scheduling Algorithm

In this section, we will describe downlink packets scheduling algorithm in OpenFlow-based framework. The basic objective of our scheduling algorithm is to ensure any transmission of downlink packet have as high success rate as possible. As a result, we can reduce waste of time due to retransmission, and improve the efficiency of DCF mechanism.

The basic idea of scheduling algorithm is to maintain moderate competition between APs to achieve high transmission success rate. As a result, retransmission is reduced and the waste due to retransmission is minimized. Since interference estimation is essential in evaluating competition for scheduling algorithm, we first introduce some details of interference estimation and then explain the proposed downlink packets scheduling algorithm.

A. Interference Estimation

All the scheduling-related problems have to address a crucial problem: determining whether a certain set of links can be activated at the same time. In essence, we require an interference model to predict whether a given link will be interfered with by other activated links and estimate interference degree. Interference model has been proved to have significant impact on the complexity of many optimization problems in wireless networks, such as scheduling and channel assignment.

Interference model: At the very beginning, k-hop model [17] and protocol model [18] are two simplest interference models. We call these models pairwise model in that they have a common feature that they consider every interferer independently, ignoring the fact that interference is a accumulated effect the accumulated effects when multiple interferers exist, people proposed physical interference [18], or SINR-based model. In SINR-based model, whether a packet can be correctly received by the receiver depends on the Signal-Interference-plus-Noise Ratio (SINR) at the receiver. SINR-based model assumes that packet will be correctly received only when SINR at receiver is greater than a specific threshold value.

However, all the models above are under the assumption that interference is a binary phenomenon.

Graded model further captures the phenomenon that the reception of packet under interference is stochastic, and argues that the success rate of packet reception rises with the decrease of SINR. In this model, interference degree is assessed by *Packet Reception Rate* (PRR), which is a function of SINR. Several works studied how to construct the PRR-SINR curve [19] in off-line fashion. Considering the practicality, the study of Ahmed *et al.* [20] proposed to send some test packets to measure PRR in single-hop AP-based networks. In really-deployed testbed, several works proposed some measurement-based model to predict PRR through measuring received signal strength (RSS) of radios, e.g., Ref. [21].

Computation of packet reception rate: For the sake of accuracy and practicability, we choose graded SINR-based model to construct interference relation of APs. To calculate SINR, we need to get the power of useful signal, interfering signals, and background noise. Furthermore, in graded SINR-based model, PRR is a function of SINR value, so we need to construct the PRR-SINR curve. We benefit from many existing works [21] which developed measurement-based techniques to estimate interference. Moreover, we incorporate off-line RSS measurement with online probing technique to construct interference relation of the whole network.

Given a target link and a set of simultaneously activated links, we need to estimate the PRR of the target link can achieve under the interference of these interfering links. Let $l_{i,j}$ denote a link, i is the source node and j is the destination node, $P_r(s)$ is the received power of signal from node s, at node r. Using the approach from Ref. [21], we can get $P_r(s)$ value of any node s and r. So, we obtain the SINR of the link $l_{i,j}$:

$$SINR_{l_{i,k}} = \frac{P_j(i)}{N_j + \sum_{l_{i,k} \in \mathrm{I}} P_j(k)} \quad (1)$$

where N_j is the background noise at node j, and I is the set of all simultaneously-activated links.

With the SINR, we next attempt to estimate PRR of link using PRR-SINR curve. We choose the method in Ref. [19].

B. Scheduling Algorithm

In OpenFlow controller, downlink packets can be categorized into two sets: sending packets, and waiting packets. Sending packets includes the packets that are undergoing transmission between APs and corresponding client and waiting packets are in effect the packets that don't have any matching rule in APs and sent to controller to ask for the subsequent processing. In addition, we define and record the set of activated links which means the links that have packets sending on it. The controller updates these three sets continuously during network's operating, according to the message from APs and message it have sent to APs.

Waiting packets are also classified into several groups according to their corresponding links. Each link has an ordered queue of downlink packets called waiting queue, is identified by its source and destination. Source is AP and destination is client. The waiting queues are kept to update to maintain the order of every packet's arrival. This is to ensure that packet arrives first will be sent first. This is very important for TCP performance, because out-of-order packets in TCP protocol will be regarded to be congestion, leading to shrinking of sending window. When receiving a message from AP that informs the table miss of downlink packet, the controller inserts that packet at the back of corresponding waiting queue.

For easy of exposition, we at first define some symbols to be used in section. $WQ_{i,j}$ denotes the waiting queue of link $l_{i,j}$, AL denotes the set of all the activated links. Moreover, $source_p$ and $destination_p$ denote the source and destination of packet p respectively.

Our scheduling intelligence is presented in terms of pseudo-code of algorithms (as shown in TABLE I. , 0, TABLE III. , and TABLE IV.). As for the downlink packets scheduling problem, we cannot ensure that the proposed algorithm can get optimal solution. Instead, our aim is to provide as significant improvement as possible relative to state-of-the-art mechanism, i.e., DCF without

any scheduling, by taking advantage of some intuitive heuristic view.

When a downlink packet arrives at AP and table miss occurs, the controller will invoke the algorithm *HandNewArrivedPacket* (as shown in TABLE I.). This algorithm inserts the new arrived packet into corresponding wait queue and trigger the invoking to the algorithm *Schedule* (shown in TABLE IV.). When a packet is successfully transmitted, the algorithm *HandSendSuccessPacket* (shown in 0) will be invoked. This algorithm removes corresponding link from *AL* and also trigger the invoking to the algorithm *Schedule*. Once a transmission of packet fails, the controller will invoke the algorithm *HandSendFailPacket* (shown in Table. III) to insert the failing packet in the front of waiting queue. As a result, out-of-order of packets can be avoided.

The algorithm *Schedule* (shown in TABLE IV.) iterates through each non-empty waiting queue, and examines whether the packet in the front can be sent. At the end of the algorithm, *PS* records all the packets chosen to be scheduled in next round. Then, the controller installs rules in relevant APs.

The primary idea behind the algorithm is that packet is supposed to achieve high PPR under the prerequisite of only not affecting PRR of links in *AL* . Specifically, we have two metrics: first, if the packet was sent in the air, the sum of PRR of links in *AL* and PRR of the candidate packet's corresponding link should not be less than the sum of original PRR of links in *AL* when it was not sent; second, if the packet was sent in the air, the PRR of all the activated links including the candidate link and links in *AL* should not be less than a threshold value (we set it 90%). The second metric is to ensure activated links achieve high PRR, so that transmitting failures and retransmissions would occur not very frequently.

For the function *CalcPRR* has two parameters: the target link and the set of activated links, and returns the PRR of the target link under the interference of the activated links. We use the method mentioned in section IV-A to derive the pseudo code of *CalcPRR*.

TABLE I. THE ALGORITHM HANDLENEWARRIVEDPACKET

Algorithm HandleNewArrivedPacket
Input: the new arrived packet p
1: Insert p to the end of $WQ_{source_p, destionation_p}$; 2: Invoke the algorithm Schedule (shown in Table IV); 3: return

It is noted that the scheduling may bring about fairness problem. If no measurement was taken, certain link would suffer from "starvation". To solve the problem, we link all the waiting queues of all the links as a circular linked list. In algorithm *Schedule*, the iteration procedure goes along the circular linked list. As a result, link being visited earlier, the probability that it is chosen is higher. To achieve fairness among APs and clients, every time the algorithm *Schedule* is invoked, the starting point of iteration is forwarded by one position. In this way, all the candidate links have comparatively fair opportunities to be chosen, and the proposed scheduling algorithm can prevent starvation of certain links and achieve relative fairness.

TABLE II. THE ALGORITHM HANDLESENDSUCCESSPACKET

Algorithm HandleSendSuccessPacket
Input: the packet p that has been sent successfully
1: $AL \leftarrow AL - l_{source_p, destionation_p}$; 2: Invoke the algorithm Schedule (shown in Table IV); 3: return

TABLE III. THE ALGORITHM HANDLEFAILPACKET

Algorithm HandleSendFailPacket
Input: the packet p that fails to be sent
1: Insert p to the front of $WQ_{source_p, destionation_p}$ 2: Invoke the algorithm Schedule (shown in Table IV); 3: return

TABLE IV. THE ALGORITHM SCHEDULE

Algorithm *Schedule:* Find links to be activated nextt
Input: waiting queue $WQ_{i,j}$ for any AP i and any client j set of all the activated links AL
Output: set of all the packets scheduled to be sent next: PS
1: $PS \leftarrow \Phi$; 2: $sum \leftarrow 0$; 3: for each non-activated link whose waiting queue $WQ_{i,j}$ is not empty do 4: $sum\_prr \leftarrow 0$; 5: for each link $l \in AL \cup \{l_{i,j}\}$ do 6: $prr \leftarrow CalcPRR(l, AL \cup \{l_{i,j}\})$; 7: if $prr \le threshold$ then 8: goto 3; *//Once a link's PRR is lower than threshold, skip directly and turn to the next candidate* $WQ_{i,j}$ 9: end if 10: $sum\_prr \leftarrow sum\_prr + prr$; 11: end for 12: if $sum\_prr \ge sum$ then 13: $ca\_p \leftarrow$ dequeue the packet in front of $WQ_{i,j}$; 14: $PS \leftarrow PS \cup \{ca\_p\}$; 15: $AL \leftarrow AL \cup \{l_{i,j}\}$; 16: $sum \leftarrow sum\_prr$; 17: end if 18: end for 19: return PS

V. EVALUATION

We evaluated the performance of OpenFlow-based enterprise WLAN and the proposed downlink packets scheduling algorithm through simulations. Our experiment topology consists of 10 APs and 50 clients, each client associated with one AP. The relative position of links covers every possible scenario, including "*hidden terminal*" and "*exposed terminal*".

We let each AP generate downlink traffic to its associated clients, and client also generates uplink traffic to its associated AP. For UDP, the traffic was CBR; for TCP, the traffic was generated in a fashion that emulates real behavior of Internet-access. The wireless interfaces of APs and clients operated under 802.11g 54Mbps

constant rate mode and RTS/CTS handshake was disabled. We varied traffic data rate to examine the effect of increasing load on network throughput. We kept the ratio of downlink and uplink traffic at 4:1.

We first compared the average throughput of all the 50 links in OpenFlow-based framework with downlink packets scheduling against that in traditional framework. The results are given in Fig. 5. OpenFlow-based framework with the proposed downlink packets scheduling improves the performance of the whole network in term of TCP and UDP throughput. We vary the offered load from every client, and found that when the offered load is low, the improvement is not quite obvious. With the increase of offered load, the capacity of some links cannot support the traffic load and these links enter saturated state. At that time we find that the increase of average throughput cannot match that of offered load. The throughput in traditional framework displays a decline under heavy load due to the increased transmission failure and retransmission caused by fierce competition. This reflects the poor performance of DCF in heavy load network. On the contrast, our solution performed well. The proposed OpenFlow-based framework with downlink packets scheduling improves average UDP throughput by more than 40% over that of traditional framework without any scheduling. The proposed solution has even greater effect on TCP's performance. For TCP, the overall gain is nearly 70% in saturated state.

To further understand the advantage of the proposed OpenFlow-based solution, we choose 20 representative links. We examined the throughput of these links when offer load is 6.5Mbps and the results are given in Fig.6. We find that our solution provides significant UDP and TCP throughput gains throughout all the links.

We further evaluated UDP delay. We extracted the data when downlink rate was 10Mbps. The results are shown in Fig. 7. We find that the delay has been reduced by at least 30% for all the links.

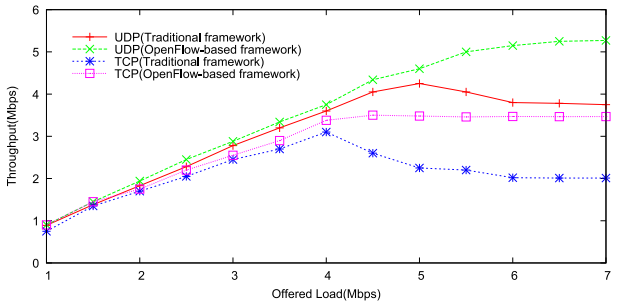

Figure 5. UDP and TCP throughput as the increase of offered load

The ratio of the number of successfully transmitted packets and the number of transmit attempts reflects packet retransmission rate. Fig. 8 shows the results of UDP traffic. In traditional framework without any scheduling, the ratio decreases rapidly as the increase of traffic load. This indicates that fierce competition leads to high collision and thus many retransmissions. On the contrast, the ratio in our solution keeps at about 0.9. That is to say, our scheduling maintains moderate completion, which promotes the efficiency of DCF.

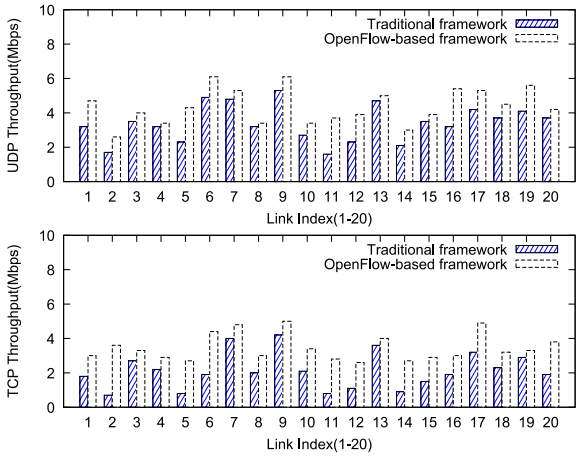

Figure 6. Throughput of representative links (offered load is 6.5Mbps)

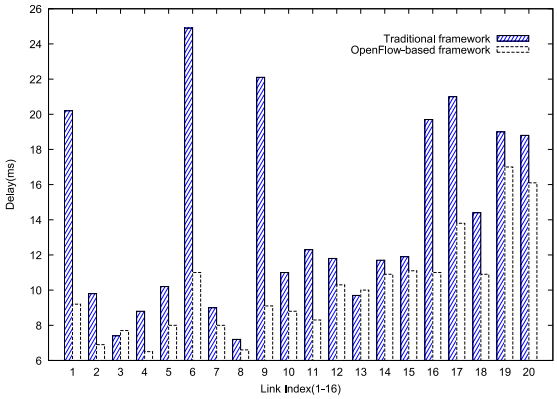

Figure 7. UDP delay of representative links (offered load is 6.5Mbps)

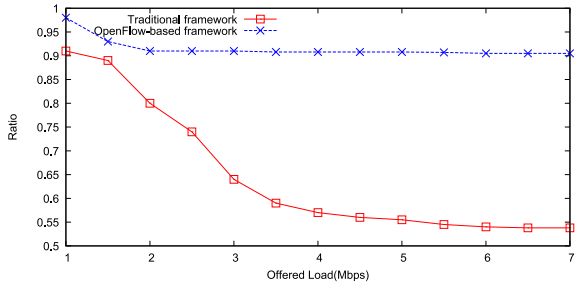

Figure 8. Packet transmission ratio of UDP

VI. Related Work

A. Centralized Scheduling to Improve DCF's Performance

Inspired by the fact that downlink traffic accounts for nearly 80% of the whole traffic in enterprise WLAN [16], Shrivastava *et al.* proposed a framework called Centaur [22] that leverages centralized scheduling to improve DCF's performance in enterprise WLAN. In Centaur, packets are judiciously chosen to be simultaneously sent during every period of time, Centaur can solve some problems such as "*hidden terminal*" and "*exposed terminal*" that DCF suffers from.

Similar with Centaur, we also attempt to preserve DCF and proposed a centralized framework to promote DCF's efficiency. Compared with Centaur, our scheduling algorithm has some advantages. Firstly, our algorithm goes beyond epoch-based scheduling method. In our algorithm, a waiting packet may be forwarded as soon as some activated links retreat from the set of activated links and give the corresponding link of the waiting packet a high PRR, rather than waiting to a batch of sending packets all finish their sending. Secondly, our algorithm is based on a graded SINR-based model that is more accurate. This model complicates the scheduling problem, posing a new challenge.

B. SDN in Enterprise WLAN

Attempts of applying SDN to WLAN can be found in Ref. [23], Ref. [24] and Ref. [25].

Ref. [23] proposed to use OpenFlow to monitor traffic flows and provided a GUI to control traffic flows. Ref. [23] provides to use SDN to support network virtualization, and they propose to slice network according to user's requirements or application characteristics. For instance, operator can create a dedicated network slice for service with special QoS settings (e.g., VoIP).

In Odin [24], users' association states are kept on a central controller and AP is responsible for authentication and beacon generation. Odin introduces LVAP, which records a user's association context. With a user's LVAP, AP can communicate with the user. Odin realizes AP handoff by removing LVAP from old AP and spawning it in new AP. However, spawning a new LVAP inevitably takes quite some time, while AP handoff in SDWLAN has no such overhead. Since LVAP contains user's key, multiple copies of the key scattered in several APs increases security risk.

CloudMAC [25] also lifts MAC-layer management function onto central controller. However, CloudMAC does not mention how to unify the wired controller and wireless controller. In addition, CloudMAC only supports switching all the associated clients from one AP to a new AP at the same time.

All the works above haven't proposed to take advantages of SDN and OpenFlow to conduct fine-grained downlink packets scheduling in enterprise WLAN for interference mitigation.

VII. CONCLUSION

In this paper, we leveraged the emerging idea of SDN and OpenFlow technology to reorganize the architecture of enterprise WLAN to mitigate the impact of interference, which cannot be handled very well in conventional architecture. In the proposed OpenFlow-based framework, we can conduct fine-grained downlink packets scheduling by installing appropriate rules in corresponding APs. We retained conventional DCF mechanism in 802.11 standard and thus keep backward compatibility with existing billions of 802.11 terminal devices. Based on the framework, we proposed a downlink packets scheduling algorithm to mitigate the impact of interference among APs and clients. The basic principle of the algorithm is to ensure that every downlink packet achieve high PPR without affecting the PRR of activated links. By keep moderate competition between APs, we can increase the efficiency of DCF due to high packet transmission rate. We demonstrated by simulation that the proposed algorithm can significantly promote network performance and user's experience in terms of high performance and low retransmission rate.

ACKNOWLEDGMENT

This work was partially supported by the National Science Foundation of China under Grant No. 61379144, 2014-2017. The authors wish to thank Prof. Jiannong Cao, from The Hong Kong Polytechnic University, for his suggestion on SDN.

REFERENCES

[1] P. Karn, "MACA-a new channel access method for packet radio," *in ARRL/CRRL Amateur Radio Computer Networking Conf.,* 1990.

[2] S. Ray, J. Carruthers, and D. Starobinski, "RTS/CTS-induced congestion in ad hoc wireless LANs," *in Proc. IEEE WCNC*, vol. 3, 2003, pp. 1516–1521 vol.3.

[3] E. S. and A. Campbell, "E-CSMA: supporting enhanced CSMA performance in experimental sensor networks using per-neighbor transmission probability thresholds," *in Proc. IEEE INFOCOM*, 2007, pp. 1208–1216.

[4] M. Cesana, D. Maniezzo, P. Bergamo, and M. Gerla, "Interference Aware (IA) MAC: an enhancement to IEEE 802.11b DCF," *in Proc. IEEE Vehicular Technology Conf.*, 2003, pp. 2799–2803.

[5] X. Yang and N. Vaidya, "On physical carrier sensing in wireless ad hoc networks," *in Proc. IEEE INFOCOM,* vol. 4, 2005, pp. 2525–2535 vol. 4.

[6] Y. Sun, B. Zhou, Z. Wu, Q. Ni, and R. Zhu, "Multi-channel MAC protocol in cognitive radio networks," *Journal of Networks (JNW)*, vol. 8, no. 11, pp. 2478–2490, 2013.

[7] E. Magistretti, K. K. Chintalapudi, B. Radunovic, and R. Ramjee, "WiFi-Nano: reclaiming wifi efficiency through 800 ns slots," *in Proc. ACM MobiCom,* 2011, pp. 37–48.

[8] S. Gollakota and D. Katabi, "Zigzag decoding: combating hidden terminals in wireless networks," *in Proc. ACM SIGCOMM*, 2008, pp. 159–170.

[9] S. Sen, R. Roy Choudhury, and S. Nelakuditi, "CSMA/CN: carrier sense multiple access with collision notification," *in Proc. ACM MobiCom*, 2010, pp. 25–36.

[10] Y. Bejerano and R. Bhatia, "MiFi: A framework for fairness and QoS assurance in current IEEE 802.11 networks with multiple access points," *in Proc. IEEE INFOCOM,* vol. 2, 2004, pp. 1229–1240 vol.2.

[11] "Software-Defined Networking: The new norm for networks," *White Paper, Open Networking Foundation (ONF)*, April 2012.

[12] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[13] A. Doria, J. H. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal, and J. Halpern, "Forwarding and control element separation (ForCES) protocol specification," *RFC 5810*, March 2010.

[14] A. Banjar, P. Pupatwibuli, and R. Braun, "DAIM: a mechanism to distribute control functions within OpenFlow switches," *Journal of Networks (JNW)*, vol. 9, no. 1, pp. 1–9, 2014.
[15] T. Feng, J. Bi, H. Hu, and H. Cao, "Networking as a service: a cloud-based network architecture," *Journal of Networks (JNW)*, vol. 6, no. 7, pp. 1084–1090, 2011.
[16] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage, "Jigsaw: Solving the puzzle of enterprise 802.11 analysis," *in Proc. ACM SIGCOMM*, 2006, pp. 39–50.
[17] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," *in Proc. ACM MobiCom*, 2006, pp. 227–238.
[18] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.
[19] R. Maheshwari, S. Jain, and S. R. Das, "A measurement study of interference modeling and scheduling in low-power wireless networks," *in Proc. ACM SenSys*, 2008, pp. 141–154.
[20] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki, "Online estimation of rf interference," *in Proc. ACM CoNEXT*, 2008, pp. 1–12.
[21] A. Kashyap, S. Ganguly, and S. R. Das, "A measurement-based approach to modeling link capacity in 802.11-based wireless networks," *in Proc. ACM MobiCom*, 2007, pp. 242–253.
[22] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra, "CENTAUR: Realizing the full potential of centralized wlans through a hybrid data path," *in Proc. ACM MobiCom*, 2009, pp. 297–308.
[23] Y. Yiakoumis, K.-K. Yap, S. Katti, G. Parulkar, and N. McKeown, "Slicing home networks," *in Proc. ACM SIGCOMM Workshop on Home networks (HomeNets)*, 2011, pp. 1–6.
[24] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise WLANs with Odin," *in Proc. ACM SIGCOMM Workshop on Hot Topics in Software Defined Networks (HotSDN)*, 2012, pp. 49–54.
[25] P. Dely, A. Kassler, J. Vestin, N. Bayer, H. Einsiedler, and C. Peylo, "CloudMAC: An OpenFlow-based architecture for 802.11 MAC Layer processing in the cloud," *in Proc. IEEE Broadband Wireless Access Workshop*, 2012.

Dong Zhao, born in 1985, received his MSc degree in computer science from National University of Defense Technology (NUDT) in 2009. From 2010 he has been a PhD candidate in NUDT. His main research interests include wireless mesh networks, software-defined networking.

Ming Zhu, born in 1985, received his MSc degree in computer science from National University of Defense Technology (NUDT) in 2010. From 2011 he has been a PhD candidate in NUDT. His main research interests include unmanned aerial vehicle (UAV) networks, wireless vehicle networks and software-defined networking.

Ming Xu, born 1964, received his PhD degree in computer science from National University of Defense Technology (NUDT), professor of Department of Networking Engineering of College of Computer at NUDT. His main research interests include mobile computing, wireless sensor networks, wireless vehicle networks and wireless cognitive networks.