IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 13, 2019, accepted September 9, 2019, date of publication September 20, 2019, date of current version October 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2942717

Fairness and Load Balancing in SDWN Using Handoff-Delay-Based Association Control and Load Monitoring

SHIRONG LIN[1], NAN CHE[2], FEI YU[1], AND SHOUXU JIANG[1]
[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2]School of Software, Harbin University of Science and Technology, Harbin 150080, China

Corresponding author: Shouxu Jiang (jsx@hit.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61370214 and Grant 61300210.

ABSTRACT Traffic loads in any 802.11 WLAN are distributed unevenly. This imbalance implies that some access points (APs) suffer from traffic congestions, while others are underutilized. The unbalanced load distributions cause annoying packet delay and throughput degradation which is unacceptable in current and future networks. A load-balancing algorithm should solve two challenges. The first is to accurately identify the APs' loads to timely find traffic imbalances. And the second is to associate clients with APs to achieve optimal proportional fairness intelligently. Network metrics such as throughput, delay, jitter, and client amount cannot be used individually to accurately identify APs' loads, because of the complexities of wireless communications. Which metrics to use and how to combine those network metrics to represent AP load are controversial. For intelligent association control, handoff delay (time to move a station from an AP to another) may last for 6 seconds. If the algorithm designers do not consider this delay in their optimization processes, unnecessary re-associations generated in their algorithm will offset the optimization profits. In this paper, we propose novel learning-based methods to monitor the network load to discover real-time load unbalances. We also model the load balancing problem as a utility maximization problem in which costs caused by handoff delay are considered. Then we utilize discretized linear programming theory and general assignment problem theory to solve it. We also compute the approximation ratio of our algorithm. We implement the whole load balancing system and evaluate the performances which show that our method outperforms a state-of-art algorithm in terms of throughput by up to12.7%, and it outperforms the received signal strength indicator (RSSI) based method by up to 28.13%.

INDEX TERMS Load balancing, association control, load monitoring, general assignment problems.

I. INTRODUCTION

Traffic load in an 802.11 wireless LAN (WLAN) is unevenly distributed because clients are free to associate with any desired known AP. This imbalance implies that some APs suffer from traffic congestion, while the others are underutilized. The clients associated with overloaded APs will suffer from lower throughput, longer delays, or longer jitters. Global load balancing algorithms are needed for reasonably distributing network traffic to avoid these unnecessary performance degradations. However, global scheduling in a distributed 802.11 WLAN is difficult [1], [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Omer Farooq.

Software-defined networking (SDN), which enables the controller to manage the whole network in a centralized way, can be extended to implement global load balancing algorithms [2]–[4]. In a software-defined wireless network (SDWN), accurate and prompt load monitoring and logical handover (re-associations) can avoid the performance degradations. The essential problems for load balancing are load monitoring and handover.

Accurately monitoring the load of an SDWN is difficult. First of all, there is no standard definition of when an AP is considered overloaded. Furthermore, there is also no easy way to obtain information about loads from APs [1]. Even if we can obtain all the desired metric measurements, each of them can only reveal the interesting behavioral

aspects of the network [5]. Previous research [5] also points out that the right metrics should be intelligently combined to represent the load of an AP. The word ''right'' means there is a subset of metrics that can dominate the AP load. However, the contents of this subset are currently controversial. Also, our work in this paper shows that queue-related metrics have strong relationships with AP load. At last, the load of an AP may be affected by interfering signals from other devices that work on ISM bands, such as Bluetooth, Zigbee, and micro oven [6], [7]. When the content of the subset is confirmed, we can model the AP load.

Even if the metrics are specified, it is still challenging to build a single unified load model. The relationships between the AP load and these metrics are complex. Besides, various metrics are naturally interdependent of each other [8], for example, signal strength ratio (SNR) and RSSI are highly correlated.

Machine learning method is a feasible way to combine these metrics. Fuzzy logic methods are used to determine AP load: combining signal strength and packet error in [9]; combining signal strength and traffic rate in [10]. However, the subset they chose is not comprehensive, which limits the performances of the machine learning methods.

In this paper, we implemented a load balancing system. This system has accurate load monitoring and logical correlation control algorithms. At first, we implement sample modules in the kernel of OpenWRT [11] to obtain comprehensive real-time network metric measurements, especially the queue-related ones. Then, we utilize a clustering algorithm to cluster and label these real-time data. The labeled data are classified by decision tree algorithms. The generated decision trees are transmitted periodically to the APs. We also implement custom SDN actions to parse and run these decision trees on APs. A decision tree can tell its AP whether it is overloaded by observing current network states, and the APs then message the controller their states. Next, when the amount of overloaded APs exceeds the threshold, our algorithm product a new handover assignment. We implement the handover processes by using the CAPWAP-based [12] light virtual access point (LVAP) of 5G-empower [13].

The association control is designed to associate the clients with the APs to maximize the sum of the logarithms of the actual rates (rates allocated minus handoff cost) of clients. Achieving maximum utility in a WLAN is NP-hard [14]. We transform the initial problem (nonlinear programming) to discretized linear programming (DLP). The DLP is solved to obtain a fractional association solution (for example, $\{0.1, 0.9, \cdots, 0.8\}$), and then, each item of the solution is rounded to an integral value.

Finally, we conduct practical experiments on our extended 5G-empower platform [13] to verify the performance of our methods.

Overall, the key contributions of this paper are as follows:

1) Based on the comprehensive measurement study of network load in WLAN, we proposed a learning-based load monitoring method. Our method can adapt to the dynamics of WLANs. Moreover, our method can be used in real-time applications because it is implemented on the AP to obtain the AP load instantly. We believe these works have implications for all players in the mobile Internet ecosystem.
2) We formulate the association control problem, which aims at high throughput, optimal proportional fairness among users and minimum resource guarantees, under a complex WLAN traffic. These factors are necessary and significant for load balancing applications. To the best of our knowledge, the association control scheme taking account of all these factors has not been reported yet in the literature.
3) We model the association control problem as a utility maximization problem. Our algorithm can be divided into two sub-algorithms. The first sub-algorithm is a Linear Programming relaxation based-algorithm that yields a fractional solution. The second sub-algorithm is an extended general assignment problem (GAP) algorithm which rounds the fractional solution into the binary solution. And, we derive the approximation ratio of our algorithm.
4) Our work shows that queue-related network metrics have strong relationships with AP load, this is helpful for the other load balancing researches in WLAN.

The paper is organized as follows: we generalize the related works in §II. In §III we introduce the implementation of our load balancing framework. In §IV, we introduce our works on network metrics sampling and AP load modeling. In §V, we analyze the association control problem and propose an association algorithm with a good approximation ratio. In §VI, we evaluate our algorithms and system; the results show that they work well.

II. RELATED WORK

We focus on the definition of network load, the formulation of delay-related network utility, the analysis of algorithm and the implementation of load balancing application in this paper. So, we introduce the related work from these four aspects mentioned above.

A. AP LOAD DEFINITION

Load monitoring is vital for load monitoring applications. Load monitoring modules are implemented to quantify the AP loads by monitoring, processing, and computing network metrics measurements. However, there are no standard definitions of the AP load. Which metrics to monitor and how to combine these metrics are urgent problems. Based on the used metrics and how the metrics are combined, we category related works about AP load definition into three classes.

Some researches, especially the early ones, utilize a single metric as the AP load. Table 1 shows the details of these researches. Research work [5] points out that each metric can only reveal the interesting behavioral aspects of the link. And research work [1], [5] points out that multiple metrics should be intelligently combined to yield an accurate representation of the AP load.

TABLE 1. Metrics used to model the AP load in previous researches.

Metrics	Researches
Data arriving rates	[30]–[33]
The amount of associated stations	[34], [35]
RSSI, the amount of associated stations	[36]–[38]
Signal strength, aggregated traffic rates	[39]
Channel utilization	[18], [40]–[42]
Packet errors	[20], [43], [44]
Aggregated traffic rates, protocols	[19]

AP load is also modeled as mathematical formulations of multiple network metrics. In [17] load is defined as $\mathcal{T} = \sum_{k=1}^{M} T_k$, where M is the number of users, $T_k = S_i/r$ where S_i is the size of data arriving for user i and r is the transmission rate of user i and AP j, this means the total amount of time needed for current arriving data. In [18], the load is defined as $on\_state\_time/(off\_state\_time + on\_state\_time) * packet\_size * number\_of\_stations/inter\_arrival\_time$ which means the data arriving rate. Research work [19] defined mac efficiency to evaluate an AP association decision. The user mac efficiency which is defined as

$$a_i = \frac{s_i^{up} + s_i^{down}}{\min\{1, u_i + d_i\} r_{i,a_i}},$$

where S_i^{up} is the estimated up-link throughput of client i, S_i^{down} is the estimated down-link throughput of client i, u_i is the up-link traffic probability while d_i is the one for down-link, $r_{i,a}$ is the optimal data rate between client i and AP a. Mac efficiency means the time needed for user i to transmit its down-link and up-link data. Also, in this paper, the author proposes and solves an algorithm that maximizes the mac-efficiency-based utility function.

Machine learning is also used for load definition. In [9], the fuzzy logic method is used to combine signal quality and packet errors to get the AP load. In [20], the fuzzy logic method is also used to combine signal strength, packet error, and traffic rate to get the AP load.

B. HANDOFF DELAY HANDLING

As another essential problem of load balancing, association control aims to intelligently associate clients with APs to achieve optimal proportional fairness in a network of APs [14], [21]–[23]. For associations between practical APs and stations, handoff delays may last for 6 seconds [24]. If the corresponding costs of handoff delays are not considered when computing new associations, the outputs may contain some unnecessary associations with profits less than their handoff losses. Similar problems are considered in [24]–[26]; the author of [25] decreases the losses caused by channel switching delay using two methods: increasing the duration between two schedules to diminish losses and increasing wireless interfaces to guarantee connectivity. The author of [26] utilizes a Markov decision process (MDP) to model the throughputs (contain handoff cost) of all possible association schedules and get the optimal schedule. The one-AP-multiple-interfaces architecture is used in [24] to alleviate the handoff costs. For a one-AP-multiple-interfaces architecture, each client can associate with multiple APs, and handoff becomes a process that idles current associations and activates new associations without large time cost. These three methods have drawbacks when used to solve the handoff cost problem in this article; it is not practical to add a wireless interface to a deployed WLAN, and if the controller's computational load is supportable, then the MDP is a better choice.

C. THE PROBLEM FORMULATIONS AND ALGORITHM ANALYSES

Most of the load balancing optimizations in WLAN are NP-hard [14], [24]. A greedy algorithm is given in [24], but they do not prove the time complexity of their algorithm. The author in [27] analyzes the complexity of their inter-packet delay minimization problem, and utilizes the approximate algorithm in [28] to solve it. To model the load balancing problem as utilization maximization problems is another scheme. The load balancing utilization maximization problem itself is a general assignment problem. So, the author in [14] first transforms their initial non-linear problem to be a linear problem and then utilizes the GAP algorithm proposed in [29] to get the final solution. We consult the non-linear to linear transformation in [14], but our problem is different from theirs, and we use our customized assignment (rounding) algorithm.

D. THE IMPLEMENTATIONS

Before the appearance of SDWN, it is not easy for a controller to manage the handover behaviors of the wireless termination points. Control and Provisioning of Wireless Access Points (CAPWAP) protocol can help the controller to do these works. CAPWAP is a protocol that enables an access controller (AC) to manage a collection of wireless termination points [12]. Authors of project ODIN [3] implement the CAPWAP protocol in their light virtual access points (LVAPs) to enable support seamless handovers. Another open-source SDN platform, 5G-empower proposed in [13] also implements the LVAP based on the click [45], and 5G-empower supports seamless handovers too. The following two load balancing researches are implemented on 5G-empower. The authors of the research work [40] propose an SDN-based solution for joint user association and channel assignment. Their channel assignment aims to get the maximal number of channels that can be used concurrently. Their user association process monitors the average channel occupancy across all the APs. If a significant difference between the average channel occupancy and any occupancy ratios of the APs is found, a user re-association process is triggered for the affected AP. The re-association algorithm selects a neighbor AP and migrates clients to it if the average occupancy can be decreased. The authors of the research work [33] consider the mobility statistics and throughput statistics in two of their load balancing algorithms. Instead of monitoring and finding the overloaded AP and then triggering handover processes, their algorithm runs periodically to evenly distribute clients

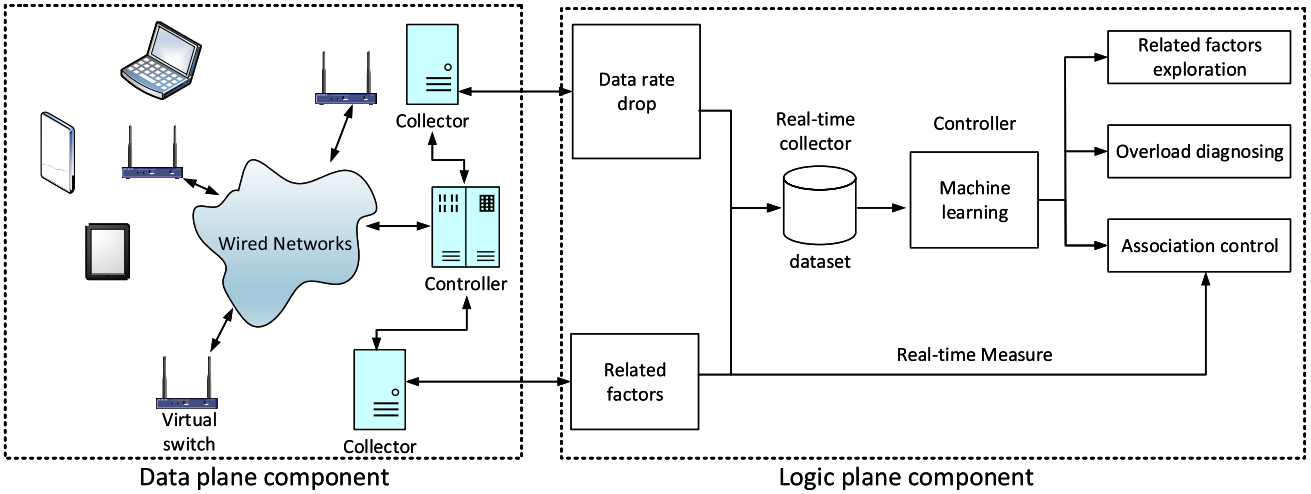


FIGURE 1. Overview of our load balancing platform.

among the APs. We extend the 5G-empower platform to support our AP-based load monitoring application, and then we implement our algorithms on the extended 5G-empower platform.

III. OVERVIEW OF TRIGGER-BASED DYNAMIC LOAD-BALANCING PLATFORM

In this section, we introduce the design of our load balancing platform. The introductions describe how our platform monitors network loads and how it acts when network traffic should be redistributed.

We implement a load balancing platform in this paper, which is shown in Fig. 1. This platform consists of two parts, and it has six functions. The two parts are the data plane part and the logic plane part. The six functions are rate control, data sampling, load monitoring, association control, overload diagnosing, and related factors exploration. Rate control, data sampling, and load monitoring work on the data plane part, and the other three functions work on the logic plane part. Custom messages are used for data and messages transmission between the data plane and logic plane, and there are two kinds of custom messages: monitoring messages and control messages. We introduce these items in the following paragraphs.

A. DATA PLANE PART

We utilize the OpenFlow [46] protocol to convert commercial APs into virtual switches (data planes). The OpenFlow protocol needs two software tools: OpenWRT and Open vSwitch [47]. OpenWRT is an open-source project for the embedded operating system based on Linux. OpenWRT converts commercial APs into Linux devices. Open vSwitch running on OpenWRT can convert the APs into virtual switches and enable them to support the OpenFlow protocol. With the OpenFlow protocol, the SDWN can be created.

Rate control, data sampling and load monitoring work on the data plane part and we introduce them in the following paragraphs.

TABLE 2. Categories, sources and sample rates of sampled information.

Module	Location	Rate
Link	Mac80211, kernel	250 Hz
Channel	Mac80211, kernel	250 Hz
Queue	sch_ generic.c, kernel	250 Hz
Beacon	Mac80211, kernel	10 Hz
Drops	Mac80211, codel.h, kernel	Each drop

1) RATE CONTROL

We focus on TCP traffic in this paper. For TCP, windows play an important role in rate control. There are three kinds of windows for TCP (except BBR [48]): the congestion window (CWND), the send window, and receive window (advertised window in packets). The send window controls the data rate; it must be smaller than the congestion window and the receive window. Receive window can be modified on APs. So, if we can get the congestion window, we can modify the receive window to be a lower value which smaller than the congestion window to limit the data rate (send window must smaller than this modified value). Fortunately, authors of research work in [49] propose a machine learning method that can infer the congestion window on APs. We use this method in this paper to control data rates.

2) DATA SAMPLING

We sample all the network metrics that may affect wireless data communications. These measurements are comprised of channel information, link information, beacon information, queue information, packet information, and dropped packet information. The locations where we implement sampling modules and the corresponding sampling rates are listed in Table 2. The precision of these measurements is high. The link, channel, and queue information is sampled every 4 ms. Beacon information varies slowly, so we sample it every 0.1 seconds. High precision samples cause a large amount of data that needs to be transmitted to the controller in real-time, so some redundancy reduction methods are used to remove the redundancies.

TABLE 3. Metrics chosen for learning.

Symbol	Semantics
congestion drop	Drop rate in queue process.
noncongestion drop	Drop rate in Mac80211.
time_busy	The amount of time when channel is busy.
time_rx	Time for receiving.
time_tx	Time for transmitting.
time_scan	Time for scan channel.
noise	Noise.
packets_3 / packet_5	Packets received in the third / fifth queue.
drops_3 / drops_5	Packet drops in the third / fifth queue.
requeues_3 / requeues_5	Requeues in the third / fifth queue.
backlog_3 / backlog_5	Backlog in the third / fifth queue.
bytes_3 / bytes_5	Bytes received in the third / fifth queue.
inactive_time	Inactive time of AP.
expected_throughput	Expected throughput.
sta_count	Amount of associated clients.

Table 3 lists the sampled and computed network statistics. These metrics are statistics computed from real-time sampled data from APs.

Some details about these metrics are explained as follows:

1) Packets are mainly dropped by queue algorithms and the Mac80211 driver. Drops by the queue algorithms are called congestion drops, and those dropped by the Mac80211 are called noncongestion drops.
2) There are five queues for the OpenWRT-based APs, with almost no data in queues 1, 2, and 4. Data are mainly transmitted in queue 3, and some data are transmitted by queue 5 when a high arriving data rate happens.
3) Expected_throughput are computed by Mac80211.
4) There are four kinds of channel utilities in this paper: time_busy, time_rx, time_tx, and time_scan. time_busy is a measure of how much time channel is in use during the last 1000 milliseconds. time_rx is a measure of how much time the current AP is receiving data during the last 1000 milliseconds. time_tx is a measure of how much time the current AP is transmitting data during the last 1000 milliseconds. time_scan is a measure of how much time current AP is scanning channels during the last 1000 milliseconds.

3) LOAD MONITORING

In a typical SDWN, a controller controls several wired connected APs, the wireless termination points are connected with these APs. The data transmission performance is important for SDWN, and re-associations are needed when the traffic is severely uneven. We implement the load monitoring module to characterize the AP's data transmission performance and trigger re-associations. The details of the load monitoring module will be introduced in §IV.

4) CUSTOM MESSAGES AND ACTIONS

We implement the monitoring messages carry real-time sampled data to collectors and the controller; we extend the monitoring message in 5G-empower to do these works. We also implement a control message to transmit the generated decision trees to the APs and feedback the transmission results. The overheads of these messages are described in §IV. We also implement custom actions to process the control messages and maintain trees (create, insert, delete, update, search.).

B. LOGIC PLANE PART

The logic plane is the "brain" of our platform. As shown in Fig. 1: association control, overload diagnosing, related factors exploration, real-time data receiving, and data preprocessing are working on it. We will introduce these functional modules in the following paragraphs.

1) DATA RECEIVING AND PROCESSING

Typical SDWN will produce large amounts of data samples that should be processed and transmitted in real-time to other functions. It is challenging to guarantee time performance when the scale of the WLAN is big enough to cause abundant computational loads. There are several kinds of computations. For example, maintaining data to compute link statistics, packet statistics, channel statistics, packet statistics; merging these statistics to be useful data items, and preprocessing these data items. To guarantee time performance, we code all the data receiving modules with the C language, and apply numerous optimization methods in these modules.

2) ASSOCIATION CONTROL

Based on Algorithm 2, association control computes new associations for all the clients when the amount of overloaded APs exceeds a threshold. The details of association control will be described in V.

3) OVERLOAD DIAGNOSING

This function is designed to diagnose the overload reasons, i.e., to find the patterns of network metrics when overloaded events happen. Additionally, decision trees that recognize network state on APs are the outputs of this function; the details will be introduced in §IV.

4) RELATED FACTORS EXPLORATION

This function is designed to explore the relationships between AP load and metrics, and Table 3 lists the results. The metrics without effects (on load) will be removed from information sampling. The details will be introduced in §IV.

IV. LOAD MONITORING

Load monitoring tells the administrator whether the AP is in the overloaded state in which throughput degradation, delay increasing, and jitter increasing will happen. Load monitoring is to forecast or deduce the data transmission states by observing current network metrics. The value of load belongs to a load state set, which is a set of labeled states. And our load monitoring method is based on a set of rules. These rules output the AP load value after the current network measurements are inputted. This section aims to get these rules.

TABLE 4. The details of the collected data.

Category	Data Size	Explanations
Link information	225.7 GB	802.11n, 5 GHz
Channel information	137.4 GB	802.11n, 5 GHz
Queue information	147.3 GB	-
Drop information	4.3 GB	-
Beacon information	92.6 GB	-

A. LARGE-SCALE MEASUREMENT STUDY

We deployed 8 APs in the fourth-floor of Zonghe building of the Harbin Institute of Technology and 12 APs in the library of the Harbin Institute of Technology. We did not set passwords for these APs, and all the potential users nearby can utilize our APs to access the Internet for free. The metrics in Table 3 are sampled when there exist users in our networks. This data sampling program lasted for about two months. We collected about 607.3 GB data. The details of these data are listed in table 8.

Except for these historical training data, our system generates real-time training data when running too. We utilize all the data for the analysis in §IV-B and §IV-C.

1) THE PREPROCESSING PROCESSES

The main challenge for our load monitoring is data preprocessing. This challenge comes from the following three aspects: a)

1) Different sample rates of the data. Different kinds of have their varying rates. Mac80211'registers update different wireless information at different rates. Queue statistics fluctuate in the OpenWRT system when data bursts happen. The packet dropping information can be triggered by congestion or non-congestion events. Beacon information updates every 0.1 seconds. Therefore, we sample these data with different rates as described in Table 2.
2) The out-sequence measurement problem, i.e., the transmission delays of data samples between different APs to the controller are not the same.
3) The class imbalance problem. The number of samples belongs to different classes are vastly different.

To solve the sample rate and out-sequence problem, we add the μs level time label to each data sample. We also implement data cache for each data source and re-sample the data as we need to obtain real-time training data items. We solve the imbalance problem with re-sampling methods. For example, the data amounts for class A, B, C and D are 3000, 4000, 5000 and 7000; we will sample only 3000 items randomly from set B, C, and D.

B. THE RELATIONSHIPS BETWEEN THE AP LOAD AND THE DATA TRANSMISSION PERFORMANCES

Two typical data transmission performances of the APs, the throughput statistics, and the queue statistics (which determines the delay and jitter performances), have qualitative relationships with the other network metrics measurements. As the prediction of the data transmission performance, the AP load should also have qualitative relationships with the network metrics measurements. And when the AP is in the overloaded state, some of its network metric measurements should also be in the overloaded state. So, we label the raw training data by the K-means method and then learn the qualitative relationships by the decision tree method. The analysis above is the logic of this section.

First of all, the AP load should be modeled with multiple network metrics. As described in §II, a single network metric is not accurate to be used as the AP load. Moreover, which network metrics should be chosen to model the AP load are controversial; this is also described in §II. At last, it is difficult to model the AP load as a single unified function. The natural interdependences among the network metrics are difficult to handle when modeling the AP load [8].

Since the AP load should be modeled with multiple network metrics; we show that there exist qualitative relationships between the throughput and related network metrics. Firstly, the throughput has strong monotonic relationships with the other network metrics, and it can be predicted by them. Secondly, previous research [8] points out that the Kendall index between delay and channel utilization is as high as 0.886, and the corresponding information gain (IG) is 0.1708. Kendall and IG are complementary. The Kendall score quantifies the monotone relationships, and the IG quantifies how well we can predict the object item [8]. In this section, we also implement experiments to show the qualitative relationships between the queue backlog and the other network metric measurements, and the relationships between the throughput and the other network metric measurements.

The feature selection in this paper is based on the Kendall correlation and IG values. We remove the expected_throughput, time_scan, and inactive_time features from the training set because their Kendall correlation and IG is too small (approach to zero) when computing with throughput. The rest of the features in Table 3 are used in following K-means and decision tree algorithms.

For throughput prediction, we classify the throughput into four classes and predict them by using the decision tree algorithm with other metric measurements. The resulting precision, recall, and F-score of each class are shown in Table 5; the mean and standard deviation (SD) of the cross-validation score (CS) are also shown in this table. The F-score for this 4-class classifier is acceptable [8], [51]; additionally, the results of the cross-validation scores indicate that there exist strong qualitative relationships between the throughput and the other network metric measurements.

For queue backlog prediction, we classify the backlogs into four classes and predict it by decision tree with the other metrics. The results are as in Table 5. The results are similar to those of throughput prediction, i.e., there exist strong qualitative relationships between the queue backlog and the other network metric measurements.

Now, we learn that delay, jitter (determined by the queue backlog) and throughput have strong qualitative relationships

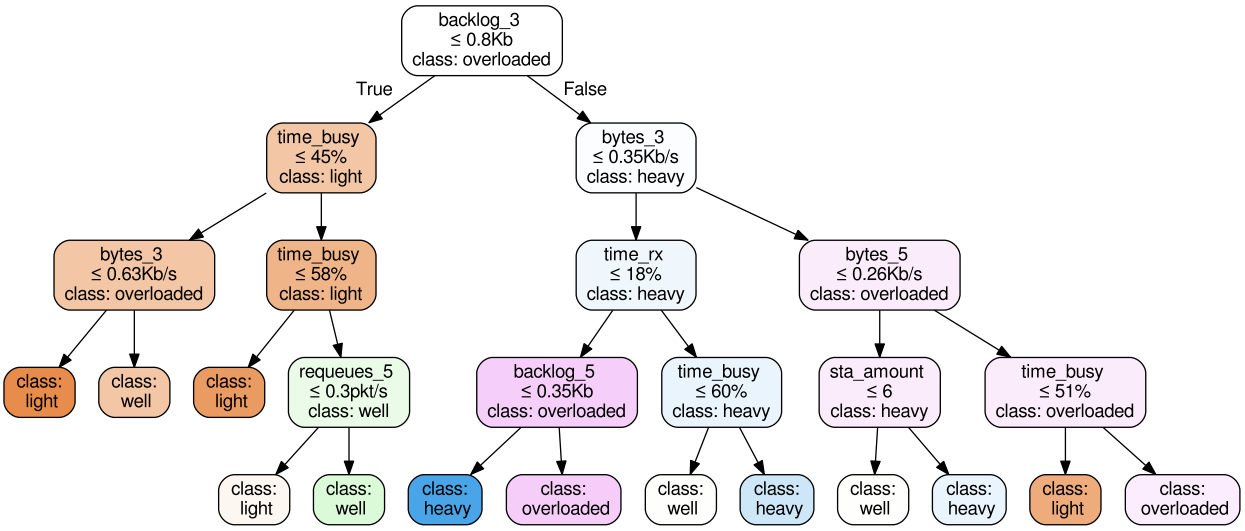


FIGURE 2. Decision tree generated on the controller and running on the APs.

TABLE 5. Performances of the 4-class throughput and backlog classifiers.

Items	Throughput	Backlog
Precision	[0.86 0.69 0.59 0.81]	[0.92 0.59 0.56 0.56]
Recall	[0.81 0.68 0.68 0.56]	[0.76 0.42 0.64 0.66]
F-score	[0.84 0.68 0.64 0.67]	[0.83 0.50 0.60 0.61]
CS mean	0.533	0.488
CS SD	0.112	0.079

with the other network metric measurements. As the metric to represent the delay performance, jitter performance, and the throughput performance of data transmission, the AP load must also have qualitative relationships with the network metric measurements.

C. TRAINING SET LABELING AND LEARNING

Because of the qualitative relationships, when an AP is in the ''overloaded'' state, most of the metrics should have either significantly high values or significantly low values. Therefore, we utilize clustering algorithms on historical data to characterize the classes of network states. Among the output classes, the one with extreme ''bad'' values must be the ''overloaded'' state. As in the experiments of throughput and queue backlog, a four-class solution is suitable for the AP load. In [8], network delays are categorized into four classes: ''EX fast'', ''fast'', ''slow'', and ''EX slow''. Analogously, we categorize the AP load into four classes: ''light'', ''well'', ''heavy'' and ''overloaded''. Among these four load states, the only one we need is ''overloaded''.

We utilize the K-means algorithm to cluster the pre-processed datasets. We remove the expected_throughput, time_scan, and inactive_time features from the training set because their Kendall correlation and IG is too small (approach to zero) when computing with throughput and queue backlog. So, the rest of the features in Table 3 are used in the following K-means, decision tree, and random forest algorithms. The outputs of clustering are high-dimensional vectors and challenging to be recognized. The decision tree algorithm can aid in distinguishing them.

When computing the decision tree of the AP load, we only utilize the latest 10000 items of the training data, i.e., the window size is 10000. Historical data will offset the importance of the latest training data if all the data are used. We employ this window in the learning processes in §IV-D and §VI. Based on the learning processes, we noticed that the generated decision trees were relatively stable. The window size decides this stability; large windows emphasize the historical experiences while small windows care more about the network dynamics. The rationality of this window size will be explored in our future research works.

The decision tree is generated and update 1time/s. The newly generated training items are added to the data set to learn the new decision tree. The new tree is transmitted to the APs by the control messages and updated by our custom APIs.

D. PERFORMANCE EVALUATION OF THE DECISION-TREE BASED LOAD MONITORING ALGORITHM

As an example, we draw a 4-depth decision tree as in Fig. 2. There are only several metrics shown in this figure; this is because the height of the tree is limited by space. The scale of the practical decision trees that run on AP has higher depth. The decision tree puts important metrics near the root; metrics close to the root have more effect on AP load. From Fig. 2, we find that the backlog of queue 3 and channel busy time are the two most important metrics. The time_rx, bytes_5, and sta_amount impact low-level branches. Other radio factors that do not show up play a relatively less important role for AP load.

To explore the qualitative relationships between AP load and the metric measurements, we drop the data of ''well''

TABLE 6. Kendall index and IG index between AP load and network metrics.

Metrics	Kendall	IG
backlog_3	0.2570	0.1889
time_busy	0.2020	0.0947
bytes_3	0.1862	0.0952
time_rx	0.1268	0.0540
time_tx	0.1691	0.0287
bytes_5	0.1403	0.0340
sta_amount	0.1331	0.0006
requeues_5	0.1142	0.0654

and ''heavy''; and label the ''light'' and ''overloaded'' ones. We compute the Kendall value and IG value over the AP load with the remained data. The results of the features with significant values are shown in Table 6. These results indicate that the queue backlog, channel utilization, arrive rate of the queue, and the transmit rate of the queue have strong qualitative relationships with the AP load.

It is easy to understand that backlog_3 acts as the most critical metric for characterizing the AP load. The third queue is the primary data queue in OpenWRT, and the backlog only appears when data burst happens and data can not be transmitted timely. Also, the backlog means the packets are undergoing extra queuing delays. Except for the backlog of queue 3, the channel utilization (time_busy in this article) is more important than the other metrics. This result is similar to the conclusion in the research work [8], which shows that channel utilization is the most critical metric for characterizing the packet delay. Of course, the queue-related metrics measurements are not used in [8]. The throughput performance (bytes_3 in this article) is one of data transmission performances, so it is reasonable that the throughput acts the third important metric is also reasonable.

As a contrast method to the decision tree algorithm, we also utilize the random forest algorithm to learn the qualitative relationships between the AP load and the network metric measurements. Table 7 lists the performance comparisons of the decision tree and random forest algorithms. Although the amount of classes is four, the precisions are acceptable [51]. The performances of the random forest algorithm are slightly better than the ones of the decision tree algorithm.

The SSE (Error Sum of Squares) is used to evaluate the performance of the clustering algorithms. The smaller the SSE value, the closer the data points are to their centroids, and the better the clustering effect. One way to reduce SSE is to increase the number of K, but this violates the original intention of our clustering. In order to improve the performance of the K-means algorithm. We conduct a K-means ($K = 2$) process on the cluster with the largest SSE to divide it into two clusters. Then we merge the two clusters with the smallest SSEs into one cluster.

E. DECISION TREE ALGORITHM ON THE APS

We choose the decision tree algorithm as the classification method for two reasons. The first is it can explore and draw

TABLE 7. The AP load prediction performances of the 4-class classifiers.

Items	Decision tree	Random forest
Precision	[0.881 0.897 0.988 0.811]	[0.896 0.896 0.989 0.825]
Recall	[0.843 0.928 0.993 0.813]	[0.853 0.941 0.993 0.819]
F-score	[0.862 0.912 0.990 0.812]	[0.874 0.918 0.991 0.822]
Accuracy	0.894	0.902
CS mean	0.890	0.897
CS SD	0.002	0.002

the relationships between network metrics and these classes. Besides the Kendall and IG indexes, the output of the decision tree can also be seen as quantitative relationships between AP load and network metrics. The second is the decision tree rules are a set of if-else items that can be implemented on APs. Load monitoring on APs significant for promptly deducing or predicting the overloaded states.This method avoids the transmission delays of both the data samples and the feedback results.

Algorithm 1 The Load Monitoring Algorithm

```
Require: None (no inputs).
Ensure: Load states.
  while 1 do
    ret[] = recvfrom(skfd, ...);
    fstate = get_state(ret);
    load = bst_search(fstate);
    if load == ''overloaded'' then
      message_controller();
    else
      usleep(100);
    end if
  end while
```

The decision tree is transmitted from the controller to the corresponding AP. We implement a parsing function to decode received information and create the decision tree on the AP. After the decision tree is implemented on the AP, the primary challenge of load monitoring is on how to obtain the network metric measurements. The decision tree is running in the user space, while the data sampling modules are running in the kernel space. We chose the socket communication approach to transmit the network metric measurements, i.e., the recvfrom(...) function in Algorithm 1. The algorithm periodically formats the network metric measurements and input them into the searching function of the decision tree, and the output is the AP load. The algorithm will message the controller if the result is ''overloaded''. Algorithm 1 describes the load monitoring processes.

F. THE OVERLOADS OF LOAD MONITORING

We analyze the overheads of data sampling in the network. The overheads of data sampling comes from two aspects: the sampling overhead, and transmission overhead. We sample network states by using global variables, the overhead of sampling comes from the functions that maintain these global

variables. We observed that the utilization of NetGear 4300's CPU is under 12% when AP is in full load state. Table 2 lists the sampled measurements, and it shows that link info, channel info, and queue info have the highest sample rate. We now calculate the total data rate for all the real-time data samples. Let z is the number of stations associated with the AP. x is the varying frequency for the queue process. y is the number of neighbors APs. η is the retransmission ratio. The sample rates are listed in Table 2. The data length for each sample element of these data are (link, 64B), (channel, 64B), (beacon, 32B), (drop, 48B), (queue, 48B). The data rate of sampling data r_s is as the following

$$r_s = \frac{(64 * z + 64 + 48 * x) * 250 + 32 * 10 * y + 48 * \eta}{1024} = 15.625z + 15.625 + 11.7186x + 0.315y + 0.0468\eta.$$

Suppose $z = 10$, $x = 1$, $y = 20$ and $\eta = 0.3$, $r_s = 189.9076$ $kByte/s$. The realistic rate for the data samples is much lower than r_s, which is because of three reasons. The first is our redundancy minimization module. This module compares the current data item with the previous one. Present data item will not be transmitted if they are the same, and the receiver side will recover it by using the past data item. The second is that the queue statistics infrequently change when there is no data burst. So, there is not much queue info that needs to be transmitted. The third is that the coherence time is longer than 4 ms (1000 / 250), during the coherence time, the channel information and link information are assumed to be constants. The coherence time for 802.11 WLAN is approximately 25.3885 ms when the velocity of the client is 1 m/s [50]. Even if r_s is 189.9076 kB, it is also sustainable in Ethernet. Moreover, when the number of clients increases, we could also set multiple data collectors to receive and preprocess the data samples.

V. ASSOCIATION CONTROL

When the load-monitoring algorithm alerts a network traffic load imbalance, the loads should be distributed efficiently and fairly among APs of the network to obtain high QoE performance. The efficiencies and fairness of the network are usually defined as utilities. To maximize the overall utilities of IEEE 802.11 WLAN is NP-hard [14], and it is challenging to design an algorithm with an acceptable approximation ratio. In this section, we introduce how we model the load balancing problem as a utility-maximization problem and how we solve that problem using an approximation method.

The notations are summarized in Table 8. We consider an SDWN in which a controller controls multiple wired associated APs. The set of APs is denoted as $A = \{a_1, a_2, \ldots, a_m\}$, where $m = |A|$ is the amount of APs. Each AP accesses the Internet via Ethernet and provides wireless access services to associated clients via the 802.11 protocol. The set of clients is $U = \{u_1, u_2, \ldots, u_n\}$, where $n = |U|$ is the amount of clients. Variable r_j is the data rate of client j. Binary variable $x_{ij} = 1$ if client j is associated with AP i and 0 if they are not associated. Fractional variable $y_{ij} \in [0, 1]$ denotes the fraction of time client j occupies the channels for transmission. Combinations of x_{ij} and y_{ij} are the outputs of the AP selection algorithm.

TABLE 8. Notations used in this paper.

Notation	Description
A	The set of all access points (APs).
a_i	The i-th AP.
m	The number of APs, i.e., $m = \|A\|$.
U	The set of all clients.
n	The number of clients, i.e., $n = \|U\|$
r_j	Bit rate of client j.
$x_{ij}(t)$	Fractional association of client j with AP i at time t.
x_{ij}	Abbreviate of $x_{ij}(t+1)$.
u_{ij}	The utility client j obtains from a_i.
y_{ij}	Fraction of time client j is associated with a_i.
e_{ij}	Link for client j and a_i.
d	Handoff cost.

Before introducing association control, we first introduce some instructions about how the association algorithm runs.

1) Trigger: The AP announces the controller when the output of its decision tree is "overloaded". When the amount of overloaded APs exceeds a threshold, a trigger is generated. If there is a trigger, the controller should consider running the association control algorithm.
2) Interval of algorithms: The minimal interval between two re-associations is set to T_1. If there is a trigger at time t, and the time when the last association control algorithm ran is t_b, and if $(t - t_b) \geq T_1$, this new trigger will be applied. Run the association control algorithm to generate new associations otherwise.
3) New association: We assume that x_{ij} changes when outputs (new associations) of the association control algorithm arrive at the APs.
4) Rate control: We guarantee y_{ij} by controlling the client data rate; the process is introduced in §III.
5) Minimum channel usage: We assume $y_{ij} \geq \beta$, i.e., each client can obtain channel occupancy time at least β.

For client j, the current schedule is $x_{ij}(t)$, and the schedule in the next period is $x_{i'j}(t+1)$; if $i' \neq i$, then a handoff cost d will be caused. Thus, the utility of client j is:

$$b_j = x_{ij}\left(y_{ij}R_{ij} - d + x_{ij}(t)d\right), \tag{1}$$

where R_{ij} is the theoretical data rate computed with the Shannon-Hartley equation [52]. The independent variables of Shannon-Hartley equation are *SINR* and bandwidth collected by the sampling module.

The utility of the whole WLAN is

$$f_{origin} = \sum_{j \in U} \sum_{i \in A} r_j \log b_j. \tag{2}$$

To maximize f_{origin} is the utility-maximization problem of efficiencies and fairnesses (UMPEF); and UMPEF has been proven to be NP-hard in [14] when the handoff cost d is zero.

Obviously, UMPEF is also not convex when $d \neq 0$. Thus, traditional methods that solve the convex problem and then round the fractional solutions to binary values are not suitable

for UMPEF. However, UMPEF still belongs to the general assignment problems. As explained in [29], the key to the general assignment problem is to find a proper method to solve the origin problem and then utilize rounding methods to round these fractional values back into binary values. A feasible way for UMPEF is trying to solve it by discretized linear programing and then round the solution into an binary one.

The first step of the approximation is to move x_{ij} out of the log function. Because $x_{ij} \in \{0, 1\}$, and $\sum_{i\in A} x_{ij} = 1$, we have the following equation:

$$\log \sum_{i\in A} x_{ij} \left(R_{ij}y_{ij} - d + x_{ij}(t)d\right) = \sum_{i\in A} x_{ij} \log \left(R_{ij}y_{ij} - d + x_{ij}(t)d\right). \tag{3}$$

Here, the utility item $R_{ij}y_{ij} - d$ is larger than 0, this is because the object of optimization is to maximize the total utility, thus a client changes its associated AP only when the loss is less than the gain. Thus, the objective function (2) changes to

$$f_{nlp}(x, y) = \sum_{j\in U} \sum_{i\in A} r_j x_{ij} \log \left(R_{ij}y_{ij} - d + x_{ij}(t)d\right). \tag{4}$$

Add the constraints to (4), and we have the following:

$$\max \ f_{nlp}(x, y) \tag{5}$$

$$\text{s.t.} \ \sum_{i\in A} x_{ij} = 1, \quad \forall j \in U \tag{6}$$

$$\sum_{j\in U} x_{ij} = 1, \quad \forall i \in A \tag{7}$$

$$\sum_{j\in U} y_{ij}x_{ij} \le 1, \quad \forall i \in A \tag{8}$$

$$\sum_{i\in A} x_{ij}R_{ij} \ge \beta, \quad \forall j \in U \tag{9}$$

$$y_{ij} \in [0, 1], \quad \forall i \in A, \forall j \in U \tag{10}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i \in A, \forall j \in U. \tag{11}$$

As the third instruction in the previous paragraph, the association between an AP and each of its corresponding clients lasts for the entire scheduling period; we utilize constraint (7) to represent this constraint. Constraint (8) means the access time allocated to all the associated clients should be less than 1. Constraint (9) arises because the minimum throughput should be guaranteed for each client's applications, and it is logical for data flows of applications to have a minimum data rate; for example, a video stream of 20 KB/s will make you crazy. This nonlinear programming (NLP) optimization is also difficult to solve because it imposes an integral constraint on x_{ij}.

Although this NLP is challenging to be solved, resources should be fairly allocated to different clients as fairness is considered when problem formulation even when there is only one AP. If there is only a single AP, this nonlinear programming problem can be solved by the Lagrange multiplier method. In this solution, the AP divides the channel occupancy time based on proportional fairness and data rates of clients. In addition, even if $|A| \neq 1$, the AP also allocates channel occupancy time based on proportional fairness and data rate.

Lemma 1: *If* $|A| = 1$, *then* $y_{1j} = \frac{r_j + r_j \sum_{j\in U} \frac{K}{R_{1j}}}{\sum_{j\in U} r_j} - \frac{K}{R_{1j}}$, *where* $K = d * x_{ij}(t) - d$.

Proof: Substituting $m = 1$ into constraint (7) and (8), and we have $x_{1j} = 1$ and $\sum_{j\in U} y_{1j} \le 1$; substituting these two into (5) and utilizing the Lagrange multiplier method, we have the following:

$$\sum_{j\in U} r_j \log \left(R_{1j}y_{1j} + K\right) - \lambda \left(\sum_{j\in U} y_{1j} - 1\right). \tag{12}$$

The partial derivative of (12) with respect to y_{1j} is

$$\frac{r_j R_{1j}}{R_{1j}y_{1j} + K} = \lambda. \tag{13}$$

Solving this function, we can obtain y_{ij} as follows:

$$y_{1j} = \left(\frac{r_j R_{1j}}{\lambda} - K\right) \frac{1}{R_{1j}}. \tag{14}$$

Taking the partial derivative of (12) with respect to λ, we have the following:

$$\sum_{j\in U} y_{1j} = 1. \tag{15}$$

Substituting y_{1j} in (14) into (15), we have the following:

$$\sum_{j\in U} \left(\left(\frac{r_j R_{1j}}{\lambda} - K\right) \frac{1}{R_{1j}}\right) = 1. \tag{16}$$

Solving (16), we can obtain λ as the following:

$$\lambda = \frac{\sum_{j\in U} r_j}{1 + \sum_{j\in U} \frac{K}{R_{1j}}}. \tag{17}$$

Substituting (17) into (14), and we have the following:

$$y_{1j} = \frac{r_j + r_j \sum_{j\in U} \frac{K}{R_{1j}}}{\sum_{j\in U} r_j} - \frac{K}{R_{1j}}. \tag{18}$$

□

From the structure of (18), we can see for a fixed integral solution, channel occupancy time allocation is proportional to fairness with respect to r_j if $K = 0$. The allocated channel occupancy times are not affected by handoff cost d for the clients that do not change AP. However, new associated clients will be affected by handoff delay.

When $|A| > 1$, NLP (5) is difficult to be solved because of the integral constraint on x_{ij}. This problem is similar to the general allocation problem: relax the constraint from $x_{ij} \in \{0, 1\}$ to fractional $x_{ij} \in [0, 1]$ can make the original problem solvable.

A. DISCRETIZED LINEAR FORMULATION

For each schedule epoch of NLP (5), we split it into D time slots and link e_{ij} using $z_{ij\tau}$ time slots; $y_{ij} = \tau/D$. A large $D = (1+\delta)\sum_{j\in U} r_j/min\{r_j|j \in U\}$ will be chosen to guarantee the solution of the discretized linear formulation to converge to the solution of nonlinear programming (5). By using these transformations, the origin NLP problem (5) changes to the following discretized linear formulation:

$$f_{dlp}(z_{ij\tau}) = \sum_{j\in U}\sum_{i\in A}\sum_{\tau=1}^{D} z_{ij\tau} r_j \log\left(\frac{R_{ij}\tau}{D} + K\right). \quad (19)$$

The whole discretized linear formulation with constraints is as follows:

$$\max\ f_{dlp}(z_{ij\tau}) \quad (20)$$

$$\text{s.t.}\ \sum_{i\in A}\sum_{\tau=1}^{D} z_{ij\tau} = 1, \quad \forall j \in U \quad (21)$$

$$\sum_{j\in U}\sum_{\tau=1}^{D} z_{ij\tau} = 1, \quad \forall i \in A \quad (22)$$

$$\sum_{j\in U}\sum_{\tau=1}^{D} \frac{\tau}{D} z_{ij\tau} \le 1, \quad \forall i \in A \quad (23)$$

$$\sum_{i\in A}\sum_{\tau=1}^{D} z_{ij\tau} R_{ij} \ge \beta, \quad \forall j \in U \quad (24)$$

$$z_{ij\tau} \in \{0,1\}, \quad \forall i \in A, \forall j \in U. \quad (25)$$

where the conditions are equivalent to conditions of nonlinear programming (5). Constraint (25) is different from (10) and (11). This is because we use $\sum_{\tau=1}^{D} z_{ij\tau}$ to instead of x_{ij} to relax the integral constraint of x_{ij}, *and* $\sum_{\tau=1}^{D} z_{ij\tau} \le 1$ corresponds to (11). Obviously $y_{ij} = \tau/D$ and $\tau/D \in [0,1]$, this corresponds to (10).

Discretized linear programming (20) is easily solved. Suppose the solution of NLP (5) is (x, y), and the solution of DLP is z; there should be some relationships between (x, y) and z. In following paragraphs we will analyze these relationships. Two lemmata will be proposed to show these relationships. These two lemmas are Lemma 2 and Lemma 35. First, we introduce Lemma 2.

Lemma 2: For every integer solution (x, y) of NLP (5), we have

$$f_{nlp}(x, y/2) \le f_{dlp}(z),$$

where z is a solution of DLP (20) that satisfies (32).

Proof: From the structure of (19) and (32), we can obtain that z is a solution of f_{dlp}.

To let Lemma 2 be valid, we need the following condition:

$$\frac{y_{ij}}{1+\delta} + K \le \lfloor\frac{\tau}{D}\rfloor + K. \quad (26)$$

Deducting K on both sides, and we have $\lfloor y_{ij}D\rfloor/D \ge y_{ij}/1+\delta$. Let $F = D * y_{ij}$ so that $\tau = \lfloor F\rfloor$, and we obtain the following:

$$\lfloor F\rfloor \ge \frac{F}{1+\delta}. \quad (27)$$

Thus, if we want Lemma 2 to be valid, inequality (27) should be valid.

Lemma 3: If $\delta \ge 1$, then inequality (27) is valid.

Proof: We assume (x, y) is an optimal solution. By Lemma 1, the optimal y_{ij} is as the following:

$$y_{ij} = \frac{r_j x_{ij} + r_j x_{ij}\sum_{j\in U}\frac{K}{R_{ij}}}{\sum_{j'\in U} r_{j'} x_{ij'}} - \frac{K}{R_{ij}}. \quad (28)$$

If $x_{ij} = 0$, then $F = 0$, and inequality (27) is valid.

If $x_{ij} = 1$, we will prove that inequality (27) is also valid. Because $K = d - x_{ij}(t)d$, we can obtain $K \le 0$. Substituting (28) into $F = D * y_{ij}$ and using the condition $K \le 0$, then we have the following:

$$F = \left(\frac{r_j(1+\sum_{j\in U}\frac{K}{R_{ij}})}{\sum_{j'\in U} r_{j'} x_{ij'}} - \frac{K}{R_{ij}}\right) * D.$$

Let $D = \frac{\sum_{j\in U} r_j}{min\{r_j + r_j\sum_{j\in U}\frac{K}{R_{ij}}|j\in U\}}$ and we have the following:

$$F \ge \left(\frac{r_j\left(1+\sum_{j\in U}\frac{K}{R_{ij}}\right)}{\sum_{j'\in U} r_{j'} x_{ij'}}\right) * D \quad (29)$$

$$\ge \frac{r_j(1+\sum_{j\in U}\frac{K}{R_{ij}})\sum_{j\in U} r_j}{min\{r_j + r_j\sum_{j\in U}\frac{K}{R_{ij}}|j\in U\}\sum_{j'\in U} r_{j'} x_{ij'}} \quad (30)$$

$$\ge 1. \quad (31)$$

Obviously, when $F \ge 1$, $\delta = 1$ can guarantee that Lemma 3 is valid. And the proof of Lemma 3 is complete. □

By using Lemma 3, we can prove that (26) is valid, and the proof of Lemma 2 is complete. □

Now, we have the lower-bound performance of DLP (20), then, we will compute the higher-bound performance of DLP (20) in following paragraphs.

We can obtain a solution (x', y') from DLP's solution z.

$$x'_{ij} := \sum_{\tau=1}^{D} z_{ij\tau}, \quad (32)$$

$$y'_{ij} := \frac{\sum_{\tau=1}^{D}\frac{\tau}{D} z_{ij\tau}}{x'_{ij}}, \quad (33)$$

$$u_{ij} := r_j \log\left(R_{ij} y'_{ij} + d - d x'_{ij}(t)\right). \quad (34)$$

It is straightforward that (x', y') is a fractional solution of NLP (5). We prove that the solution of NLP (5) is no worse than the solution of DLP (20):

Lemma 4: Let (x', y') is a fractional solution for NLP (5) that is defined in (32) and (33); z can be derived from (32), then the following is valid:

$$f_{nlp}(x', y') \ge f_{dlp}(z).$$

Proof: Substituting (32) and (33) into (34), and we can obtain

$$u_{ij} = r_j \log\left(\frac{\sum_{\tau=1}^{D} z_{ij\tau}\left(\frac{\tau R_{ij}}{D} - d + x'_{ij}(t)d\right)}{\sum_{\tau=1}^{D} z_{ij\tau}}\right) \geq r_j \frac{\sum_{\tau=1}^{D} z_{ij\tau}\log\left(\frac{\tau R_{ij}}{D} - d + x'_{ij}(t)d\right)}{\sum_{\tau=1}^{D} z_{ij\tau}}.$$

Substituting u_{ij} into $f_{nlp}(x', y') = \sum_{i,j} x'_{ij}u_{ij}$, and we obtain,

$$f_{nlp}(x', y') = \sum_{j\in U}\sum_{i\in A} x'_{ij}u_{ij} \tag{35}$$

$$\geq \sum_{j\in U}\sum_{i\in A} x'_{ij}r_j \frac{\sum_{\tau=1}^{D} z_{ij\tau}\log\left(\frac{\tau R_{ij}}{D} - d + x'_{ij}(t)d\right)}{\sum_{\tau=1}^{D} z_{ij\tau}}. \tag{36}$$

By definition of x_{ij} and making use of the concavity of the logarithm function, we have the following:

$$f_{dlp}(z) = \sum_{j\in U}\sum_{i\in A}\sum_{\tau=1}^{D} r_j z_{ij\tau}\log\left(\frac{\tau R_{ij}}{D} - d + x'_{ij}(t)d\right) = \sum_{j\in U}\sum_{i\in A} x'_{ij}r_j \frac{\sum_{\tau=1}^{D} z_{ij\tau}\log\left(\frac{\tau R_{ij}}{D} - d + x'_{ij}(t)d\right)}{\sum_{\tau=1}^{D} z_{ij\tau}}.$$

Thus, by using (36) we can obtain the following:

$$f_{dlp}(z) \leq f_{nlp}(x', y')$$

The proof is complete. □

Solution (x', y') is recovered from DLP's solution z. x'_{ij} is still a decimal in $[0, 1]$. It should be rounded to a nearby binary value. We extend the rounding algorithm proposed in [29] to do this work.

B. ROUNDING

The GAP can be viewed as the following problems: each job is to be processed by exactly one machine, and processing job j on machine i requires time y_{ij} and incurs a cost of c_{ij}. Each machine i is available for T_i time units, and the final objective is to minimize the total cost incurred. A polynomial-time rounding algorithm proposed in [29] finds a schedule of cost of at most C, where each machine i is used at most $2T_i$ time units.

The GAP is as the following:

$$\sum_{j\in U}\sum_{i\in A} u_{ij}x_{ij} \leq C \tag{37}$$

$$\text{s.t.} \sum_{i\in A} x_{ij} = 1, \quad \forall j \in U \tag{38}$$

$$\sum_{j\in U} y_{ij}x_{ij} \leq T_i, \quad \forall i \in A. \tag{39}$$

Lemma 5: If GAP has a feasible solution, then there exists a schedule that has $\sum_{j\in U} y'_{ij}x_{ij} \leq 2T_i$ *and cost of at most* C.

Proof: The proof is in [29]. □

In this paper, we add a constraint (42) to guarantee the minimum throughput each client can obtain. We prove that this condition does not impact the time performance of the rounding algorithm proposed in [29]. This modified GAP (MGAP) formulation is as the following:

$$\max \sum_{j\in U}\sum_{i\in A} u_{ij}x_{ij} \tag{40}$$

$$\text{s.t.} \sum_{i\in A} x_{ij} = 1, \quad \forall j \in U \tag{41}$$

$$\sum_{i\in A} R_{ij}x_{ij} \geq \beta, \quad \forall j \in U \tag{42}$$

$$\sum_{j\in U} y_{ij}x_{ij} \leq T_i, \quad \forall i \in A. \tag{43}$$

Lemma 6: If MGAP has a feasible solution, then there exists a schedule that has $\sum_{j\in U} y_{ij}x_{ij} \leq 2T_i$.

Proof: Changing step 2 of the algorithm of theorem 2.1 in [29] from "Find a minimum-cost (integral) matching M that exactly matches all job nodes in B(x)" to "Find a maximum-utility (integral) matching M that exactly matches all job nodes in $B(x)$ and $\sum_{i\in A} R_{ij} \geq \beta$"; other parts of the proof remain unchanged. Finding "minimum-cost" has the same complexity as finding "maximum-utility", and the total complexity remains unchanged. In addition, this "$\sum_{i\in A} R_{ij} \geq \beta$" will not change the complexity because it can be done in $O(|A|)$ time. □

Lemma 7: Let (x, y) *be a solution of NLP problem* (5)*, and* $\hat{x}$ *is a solution of MGAP* (40) *whose variable:* $\{x_{ij}\}$, $\{y_{ij}\}$ *and* $\{u_{ij}\}$ *come from* (32), (33) *and* (34) *. Then, we have* $f_{nlp}(x, y) = f_{mgap}(x)$*, and* $f_{mgap}(\hat{x}) \geq f_{mgap}(x)$.

Proof: Let u_{ij} in MGAP (40) be $r_j \log(R_{ij}y_{ij} - d + x_{ij}(t)d)$, and the aims of both NLP (5) and MGAP (40) are the same. Let $T_i = 1$ in MGAP (40), and all the constraints of NLP (5) and MGAP (40) are the same. Thus, a solution of NLP must be a solution of MGAP, and a solution of MGAP must also be a solution of NLP.

The step 2 of the proof of MGAP (40) is "Find a maximum-utility (integral) matching M that exactly matches all job nodes in $B(x)$ and $\sum_{i\in A} R_{ij} \geq \beta$." This corresponds to "Find a matching M that exactly matches all job nodes in $B(x)$ and $\sum_{i\in A} R_{ij} \geq \beta$." Thus, $f_{mgap}(\hat{x}) \geq f_{mgap}(x)$. □

We utilize MGAP as our rounding algorithm; as explained in previous paragraphs, it is an extended version of the general assignment problem in [29]. Therefore, the association control in this paper is as algorithm 2; we call it LR (Linearize and Rounding). The MGAP rounding algorithm is used in step 4 of algorithm LR.

In the following paragraphs, we compute the approximation ratio for LR.

Theorem 1: The approximation ratio of algorithm LR is 4.

Proof: Let x^*, y^* be an optimal solution for nonlinear formulation f_{nlp} (5). Let (x, y) be a solution for nonlinear formulation f_{nlp} (5), z is the solution of corresponding DLP (20). In addition, $\hat{x}$, $\hat{y}$ are the solution of f_{mgap} (40).

Algorithm 2 The Algorithm LR

Require: $\{RSSI\}, \{r_j\}, A, U, d$.
Ensure: Solution $(\hat{x}, \hat{y})$.

1. Input the RSSI into the Shannon-Hartley equation to obtain $\{R_{ij}\}$.

$$D = \sum_{j \in U} r_j / min\{r_j + r_j \sum_{j \in U} \frac{K}{R_{ij}} | j \in U\}$$

from real-time statistical data on the controller.
2. Solve discretized linear formulation (20), and obtain $z_{ij\tau}$.
3. Compute x'_{ij}, y'_{ij} and u_{ij} based on (32), (33) and (34) by using $z_{ij\tau}$.
4. Input $(x'_{ij}, y'_{ij}, u_{ij})$ to the rounding algorithm (40) and obtain $\hat{x}_{ij}$. Obtain $\hat{y}_{ij}$ by using (28).
5. Handover(x'_{ij}), where the Handover() function is provided by the LVAP module in the 5G-empower platform [13].

if $\sum \hat{y}_{ij} > 1$ **then**
6. Set the receive window to $\hat{y}_{ij} * CWND_{ij} / \sum \hat{y}_{ij}$, where the congestion window $CWND_{ij}$ is obtain by using the method provided in [49]. set_cwnd() function is implemented in the Open vSwitch.
else
7. Set the receive window to $\hat{y}_{ij} * CWND_{ij}$.
end if

By using Lemma 2, we have the following:

$$f_{nlp}(x, y/2) \leq f_{dlp}(z), \tag{44}$$

By using Lemma 35, we have the following:

$$f_{dlp}(z) \leq f_{nlp}(x, y). \tag{45}$$

Based on Lemma 7, we have the following three inequalities:

$$f_{nlp}(\hat{x}, \hat{y}) = f_{mgap}(\hat{x}); \tag{46}$$
$$f_{mgap}(x) = f_{nlp}(x, y); \tag{47}$$
$$f_{mgap}(\hat{x}) \geq f_{mgap}(x). \tag{48}$$

Combine (46), (47), and (48); we have the following:

$$f_{nlp}(\hat{x}, \hat{y}) \geq f_{nlp}(x, y). \tag{49}$$

Set $T_i = 1$ in this paper; by using (45) we have the following:

$$f_{nlp}(x, y) \geq f_{nlp}(x, \frac{y}{2}). \tag{50}$$

Substitute (49) into (50), and we have the following:

$$f_{nlp}(\hat{x}, \hat{y}) \geq f_{nlp}(x, \frac{y}{2}). \tag{51}$$

Combining (50), (51), and $\sum_{j \in U} y'_{ij} x_{ij} \leq 2T_i$ in Lemma 6, we have the following

$$f_{nlp}\left(x^*, \frac{y^*}{4}\right) \leq f_{nlp}(\hat{x}, \hat{y}). \tag{52}$$

The proof is complete. □

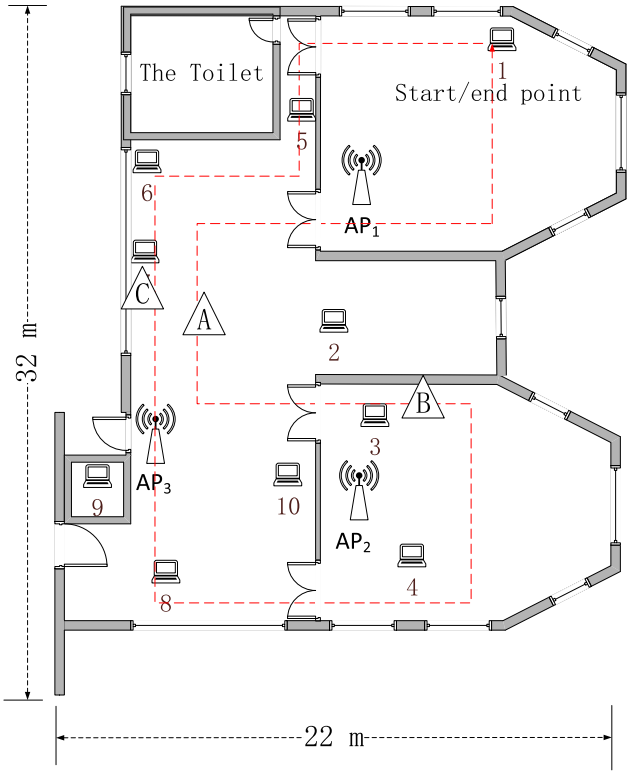


FIGURE 3. Deployment of APs and clients in the experiments.

VI. EVALUATION

Our evaluation aims to characterize the throughput, delay, and jitter improvements of our load balancing algorithm. There are transforming of non-linear programming to linear programming, rounding, clustering and classification in our algorithm, so, we named our algorithm LRCC. We compare our LRCC algorithm with the traditional RSSI-based method, and the Wi-Balance method proposed in [40]. The evaluating objects are throughput, retransmission ratios, and Jain's index.

A. EXPERIMENTAL SETUP

We carry out the performance evaluation on a real-world testbed. The deployment of the experiment is described in Fig. 3 and this deployment is similar to the one in [40]. Three NetGear 4300 routers which act as wireless termination points (WTPs) run OpenWRT 15.05.01(empower-lede) are deployed on the second floor of a teaching building. NetGear 4300 equips with the Atheros AR9344 chipset which support both 2.4 GHz and 5.8 GHz frequency. All the experiments are using the 5.8 GHz band frequency of the 802.11n protocol. These three WTPs are connected to a 5G-empower controller [13]. Ten laptops running Linux that act as the mobile stations are connected to the WTPs.

1) HANDOFF COST *d*

We measure the handoff delays of our testbed in IEEE 802.11n WLANs. The process is as follows: station 1 moves

from AP1 to AP2 and back, and we record its handoff events and the corresponding system time. We repeat this process ten times and get the average handoff time. The handoff time is about 50 ms, and this value is measured to be 59 ms in [53]. The handoff cost d is set to be the corresponding handoff delay multiple current data rate in our association control algorithm, also d is normalized to guarantee $\left(R_{ij}y_{ij} - d + x_{ij}(t)d\right) > 0$ in (3).

2) INTERVAL BETWEEN TWO HANDOVERS

The interval between two handovers is at least 1 s. When there is a load balancing event, the association control algorithm should be triggered. However, the average handoff delay in our evaluation is approximately 50 ms. Thus, we set the minimum interval between two handovers to 1 second.

Whether 1 second is suitable is not known from current research. The channel switching time is approximately 4.11/0.244 ms in [25], and the author set the interval to 6 ms. Mobilities of the stations and the channel processes might determine this interval. The mobilities are not considered in this article. The channel coherence time is the duration over which the channel impulse response is considered to be unvarying. The coherence time for 802.11 WLAN is approximately 25.3885 ms when the velocity of the client is 1 m/s [50]. From the aspect of channel coherence time, we should set the duration to several milliseconds, and from the perspective of handoff delay, the epoch duration should be longer than the handoff delay. Therefore, we set the minimum epoch duration to 1 second, and we will perform some research about this epoch duration in our future work.

3) EXPERIMENT PROCESSES

We do experiment for LRCC, Wi-balance and RSSI-based method. For each of the evaluated methods, we run the experiment for nine iterations; each test lasts 5 minutes. Table 9 shows the configurations of the nine tests. We reference the experiment design of Wi-balance [40]. All the stations stay still and are deployed randomly across the entire floor during each test except $station_1$. $station_1$ moves following the red dotted path in Fig. 3 in each test. The WTPs (NetGear 4300) run OpenWRT, Open vSwith and LVAP as described previously. The controller runs the 5G-empower runtime suit. The controller also acts as iperf3 servers, we open ten flows on it, and each flow corresponds to a client. During tests 1 to 6, the stations are divided into two groups. Stations 1 to 5 belong to group 1, and the others belong to group 2. Stations of group 1 transmit with a constant bitrate for 1 minute, the stations of group 2 transmit 40 seconds and stop for 20 seconds. Then the stations of group 2 transmit with constant bitrate for 1 minute, and the stations of group 1 transmit 40 seconds and stop for 20 seconds. During test 7 to 9, all the stations transmit with constant bitrate. The outputs of iperf3 are stored in both server sides and client sides to calculate throughput performance. We also implement a C program (running on the Laptop stations) that probes and parses information of data transmission from the kernel. It is similar to the "ss" command in the Linux system. From this data transmission information, we can obtain the throughput, fairness and retransmission performances of the clients.

TABLE 9. Configuration of the measurement campaign.

Test	User groups traffic dist.	Bandwidth
1	Constant - Intermittent	10.
2	Constant - Intermittent	5.
3	Constant - Intermittent	-.
4	Intermittent - Constant	10.
5	Intermittent - Constant	5.
6	Intermittent - Constant	-.
7	Constant - Constant	10.
8	Constant - Constant	5.
9	Constant - Constant	-.

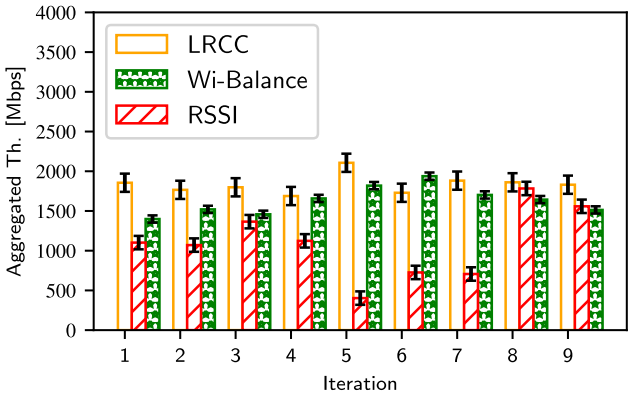


FIGURE 4. Network-wide aggregated throughput for the TCP traffic transmissions.

B. EXPERIMENTAL RESULTS

In this paper, we utilize the Jain's fairness index to quantify the fairness of throughput performances. The Jain's fairness index is denoted by the following equation:

$$Jain = \frac{\left(\sum_{i\in A} J_i\right)^2}{m\sum_{i\in A} J_i^2}, \tag{53}$$

where J_i represents the metric of AP a_i, and m is the amount of APs. The needless retransmissions and handoff costs caused by unbalanced load distribution make data transmission performance worse. Excellent load balancing algorithms should start necessary handovers and avoid unnecessary ones.

The throughput results are listed in Fig. 4 which shows that LRCC outperforms Wi-Balance in terms of throughput by up to 12.7%, and outperforms RSSI method by up to 28.13%. We observed the RSSI method's drawbacks during the iterations: handover events only happen on the moving stations, and these handovers do not change the unbalanced load distributions. As shown in Table 10, RSSI launches 9.66 handovers, these handovers happened when $station_1$ (laptop 1) was near to the APs. The Wi-Balance method performs well, and it starts handover when the difference between channel utility of an AP and the overall average value. We observed several handovers of $station_7$ and $station_{10}$ too except for

TABLE 10. Average number of handovers each iteration.

Algorithm	Average number of handovers
LRCC	8.22
Wi-Balance	8.88
RSSI	9.66

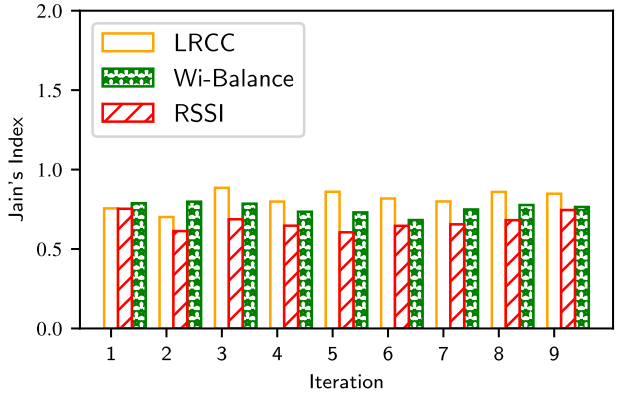


FIGURE 5. Jain's fairness index of the throughput achieved by all the wireless clients for the TCP traffic transmissions.

the ones of $station_1$. However, channel utilization itself is not throughput nor fairness, and this is why the throughput performance of Wi-Balance is not the best. When $station_1$ moved to location B, it still connected with AP_1 because the channel utilization kept almost the same when it moved. The handover should happen when its throughput decreased evidently at location A, and the communications with low RSSI has harmed the network performance. Moreover, when it moved to location C, it sometimes handover to AP_3 while its optimal AP is AP_1. The choices of AP_3 instead of AP_1 are the unnecessary handovers that can be avoided in LRCC.

The packet retransmission statistics of the algorithms are shown in Fig. 6. The sending rate decreases if a retransmission event happens in most of the TCP protocols. An algorithm causes a high retransmission rate will achieve a low TCP data rate. Statistics in Fig. 6 show that LRCC has the lowest retransmission rate, and this is not weird. The congestion and non-congestion retransmissions are monitored in our load monitoring algorithms. Furthermore, the monitored queue metrics are also signs of the coming retransmissions. The retransmissions can be predicted by other metric measurements, especially the queue statistic information. One of our current research projects is to use network metric measurements to predict future packet retransmissions.

Fairness is a crucial aspect of LRCC, and we consider fairness when problem formulation. Therefore, there are no reasons LRCC's fairness performance is worse than Wi-Balance and RSSI. The statistics in Fig. 5 also support this thesis. The global handovers in Wi-Balance and LRCC produce fairness enhancements. The operations of minimizing the difference between the channel utilizations are processes of fairness enhancements in Wi-Balance. LRCC outperforms Wi-Balance is also because of the necessary handoffs when

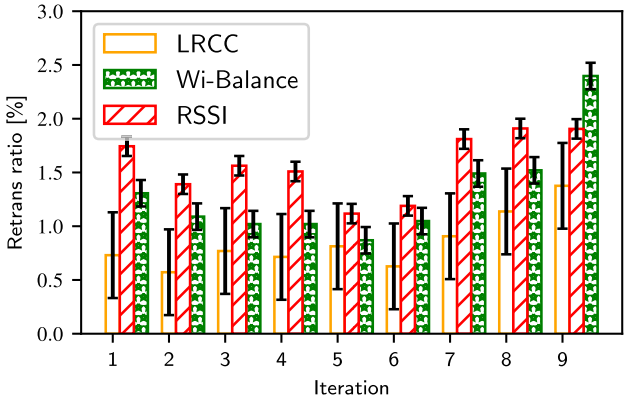


FIGURE 6. Network-wide average retransmission ratio for the TCP traffic transmissions.

$station_1$ arrives at location A and the wrong choice when it arrives at location C.

VII. CONCLUSION

We design and implement a real-world load balancing application for SDWN. The load monitoring and association control are two main parts for real-world load balancing applications. The previous research works point out three statements. Firstly, it is not accurate to use a single network metric as the AP load. Secondly, the AP load should be modelled with multiple network metrics. Thirdly, it is challenging to build a unique unified load model. The machine learning approaches are proven to be feasible for load monitoring. We implement sample modules on the APs to get related network metric measurements for learning. The learning results indicate that the throughput and queue backlogs have strong qualitative relationships with the other network metric measurements. It is well known that the queue backlogs determine the delay and jitter performance of data transmissions. The throughput, delay, and jitter determine the data transmission performance of an AP. So, as a sign of data transmission performance, the AP load must also have qualitative relationships with the network metric measurements. Moreover, when the AP is in "overloaded" or "light" state, some of the network metric measurements are either have significantly high values or significantly low values. Therefore, we cluster and label the training data by the K-means algorithm, and then we class the labeled data by the decision tree algorithm to get the load monitoring rules. The precision and recall performances of the algorithms are acceptable.

The association control problem which is modeled as the utilization maximization can be treated as a general assignment problem, and it is NP-hard in IEEE 802.11 WLAN. A general process to solve the general assignment problems contains two subprocesses. The first is to solve the original problem and obtaining a fractional solution. The second is to round the fractional solution back into binary values. Without handoff delay, the association control problem which maximizes network utility (defined by throughput gain and

fairness) can be solved by using convex methods and rounding methods. However, handoff loss changes the convex property of the utility maximization problem. In this paper, we reference existing methods of nonlinear programming and discrete linear programming to get fractional solutions. Then we extend existing GAP algorithms to round the fractional solutions back into binary ones. The approximation ratio for these algorithm processes is 4, 2 for the non-linear-to-linear transformation, and 2 for the rounding process.

We implement our load monitoring function in Open vSwitch that runs on the APs. And we integrate the extended Open vSwitch and the OpenWRT-based sampling modules into 5G-empower. Then, we evaluate our algorithms on the extended 5G-empower platform. The experiment results show that our LRCC algorithm outperforms Wi-balance method in terms of throughput by up to12.7%, and it outperforms the received signal strength indicator (RSSI) based method by up to 28.13%.

We also find that queue-related metrics have strong relations with AP load. Excellent performances are obtained when these metric measurements are used to predict the AP load. Also, as the developments SDN techniques in wireless networks, our works in this paper can be easily extended to enterprise WLAN and cellular networks.

REFERENCES

[1] W. K. Soo, T.-C. Ling, A. H. Maw, and S. T. Win, "Survey on load-balancing methods in 802.11 infrastructure mode wireless networks for improving quality of service," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–21, Jun. 2018.

[2] E. Luengo, "An openflow-based wireless user management system," Ph.D. dissertation, Dept. Elect. Comput. Eng., UOIT, Oshawa, ON, Canada, 2016.

[3] J. Schulz-Zander, L. Suresh, N. Sarrar, A. Feldmann, T. Hühn, and R. Merz, "Programmatic orchestration of WiFi networks," in *Proc. NSDI*, Philadelphia, PA, USA, 2014, pp. 347–358.

[4] M. Kawada, M. Tamai, and K. Yasumoto, "A trigger-based dynamic load balancing method for WLANs using virtualized network interfaces," in *Proc. IEEE WCNC*, Shanghai, China, Apr. 2013, pp. 1091–1096.

[5] A. Vlavianos, L. Law, I. Broustis, S. V. Krishnamurthy, and M. Faloutsos, "Assessing link quality in IEEE 802.11 wireless networks: Which is the right metric?" in *Proc. IEEE 19th Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2008, pp. 1–6.

[6] A. Guillen-Perez, R. Sanchez-Iborra, M.-D. Cano, J. C. Sanchez-Aarnoutse, and J. Garcia-Haro, "WiFi networks on drones," in *Proc. ITU Kaleidoscope, ICTs Sustain. World (ITU WT)*, Bangkok, Thailand, Nov. 2016, pp. 1–8.

[7] Y. Peng, H. Wu, K. Long, and S. Cheng, "Simulation analysis of TCP performance on IEEE 802.11 wireless LAN," in *Proc. IEEE ICII*, Beijing, China, vol. 2, Oct./Nov. 2001, pp. 520–525.

[8] K. Sui, M. Zhou, D. Liu, M. Ma, D. Pei, Y. Zhao, Z. Li, and T. Moscibroda, "Characterizing and improving WiFi latency in large-scale operational networks," in *Proc. ACM MobiSys*, Singapore, 2016, pp. 347–360.

[9] M. Collotta, "FLBA: A fuzzy algorithm for load balancing in IEEE 802.11 networks," *J. Netw. Comput. Appl.*, vol. 53, pp. 183–192, Jul. 2015.

[10] C. Zhao and C. Hua, "Traffic-load aware user association in dense unsaturated wireless networks," in *Proc. IEEE WCSP*, Hefei, China, Oct. 2014, pp. 1–6.

[11] (Apr. 2019). *OpenWRT*. [Online]. Available: https://openwrt.org/

[12] M. Bernaschi, F. Cacace, G. Iannello, M. Vellucci, and L. Vollero, "OpenCAPWAP: An open source CAPWAP implementation for the management and configuration of WiFi hot-spots," *Comput. Netw.*, vol. 53, no. 2, pp. 217–230, Feb. 2009.

[13] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, "Joint mobility management and multicast rate adaptation in software–defined enterprise WLANs," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 2, pp. 625–637, Jun. 2018.

[14] L. Li, M. Pal, and Y. R. Yang, "Proportional fairness in multi-rate wireless LANs," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Phoenix, AZ, USA, Apr. 2008, pp. 1–9.

[15] V. Shrivastava, S. K. Rayanchu, S. Banerjee, and K. Papagiannaki, "PIE in the sky: Online passive interference estimation for enterprise WLANs," in *Proc. USENIX NSDI*, Boston, MA, USA, Apr. 2011, pp. 337–350.

[16] A. Patro, S. Govindan, and S. Banerjee, "Observing home wireless experience through WiFi APs," in *Proc. ACM MobiCom*, Miami, FL, USA, Oct. 2013, pp. 339–350.

[17] L. Yang, Y. Cui, H. Tang, and S. Xiao, "Demand-aware load balancing in wireless LANs using association control," in *Proc. IEEE Globecom*, San Diego, CA, USA, Dec. 2015, pp. 1–6.

[18] R. Krishan and V. Laxmi, "IEEE 802.11 WLAN load balancing for network performance enhancement," *Procedia Comput. Sci.*, vol. 57, pp. 493–499, Jan. 2015.

[19] D. Gong and Y. Yang, "On-line AP association algorithms for 802.11n WLANs with heterogeneous clients," *IEEE Trans. Comput.*, vol. 63, no. 11, pp. 2772–2786, Nov. 2014.

[20] A. Baran, "A new load balancing procedure in IEEE 802.11 WLANs," *Int. Res. J. Eng. Technol.*, vol. 2, no. 2, pp. 866–873, 2015.

[21] D. M. Ryan, "The solution of massive generalized set partitioning problems in aircrew rostering," *J. Oper. Res. Soc.*, vol. 43, no. 5, pp. 459–467, 1992.

[22] P. Samer, E. Cavalcante, S. Urrutia, and J. Oppen, "The matching relaxation for a class of generalized set partitioning problems," *Discrete Appl. Math.*, vol. 253, pp. 153–166, Jan. 2019.

[23] P. Brommesson, "Solving the generalized assignment problem by column enumeration based on Lagrangian reduced costs," Ph.D. dissertation, Dept. Math. Univ. Stockholm, Stockholm, Sweden, 2006.

[24] X. Chen, Y. Zhao, B. Peck, and D. Qiao, "SAP: Smart access point with seamless load balancing multiple interfaces," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 1458–1466.

[25] S. Rayanchu, V. Shrivastava, S. Banerjee, and R. Chandra, "FLUID: Improving throughputs in enterprise wireless LANs through flexible channelization," in *Proc. ACM MobiCom*, Las Vegas, NV, USA, Sep. 2011, pp. 1–12.

[26] G. D. Çelik and E. Modiano, "Scheduling in networks with time-varying channels and reconfiguration delay," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 990–998.

[27] J. Chen, B. Liu, H. Zhou, Q. Yu, G. Lin, and X. Shen, "QoS-driven efficient client association in high-density software-defined WLAN," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7372–7383, Aug. 2017.

[28] X. Zhe, F. Zhang, W. Wang, and S. He, "A polynomial time algorithm for minimizing a nondecreasing supermodular set function and its performance guarantee," *Int. J. Pure Appl. Math.*, vol. 52, no. 5, pp. 637–643, 2009.

[29] D. B. Shmoys and É. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, no. 1, pp. 461–474, Feb. 1993.

[30] Y. Bejerano, S.-J. Han, and L. Li, "Fairness and load balancing in wireless LANs using association control," *IEEE/ACM Trans. Netw.*, vol. 15, no. 3, pp. 560–573, Jun. 2007.

[31] I. Jabri, N. Krommenacker, T. Divoux, and A. Soudani, "IEEE 802.11 load balancing: An approach for Qos enhancement," *Int. J. Wireless Inf. Netw.*, vol. 15, no. 1, pp. 16–30, 2008.

[32] T. De Schepper, S. Latré, and J. Famaey, "A transparent load balancing algorithm for heterogeneous local area networks," in *Proc. Int. Symp. Integr. Netw. Manag. (IM)*, May 2017, pp. 160–168.

[33] E. Zeljković, J. M. Marquez-Barja, A. Kassler, R. Riggio, and S. Latré, "Proactive access point driven handovers in IEEE 802.11 networks," in *Proc. IEEE CNSM*, Rome, Italy, Nov. 2018, pp. 261–267.

[34] G. Bianchi and I. Tinnirello, "Kalman filter estimation of the number of competing terminals in an IEEE 802.11 network," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, vol. 2, Mar./Apr. 2003, pp. 844–852.

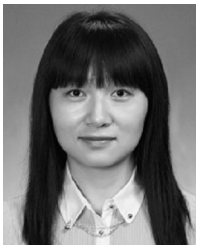
[35] G. Bianchi and I. Tinnirello, "Improving load balancing mechanisms in wireless packet networks," in *Proc. IEEE Int. Conf. Commun.*, New York, NY, USA, Apr./May 2002, pp. 891–895.

[36] I. Papanikos and M. Logothetis, "A study on dynamic load balance for IEEE 802.11b wireless LAN," in *Proc. Int. Conf. Adv. Commun. Control*, 2001, pp. 83–89.

[37] M. Collotta, G. Pau, V. M. Salerno, and G. Scatàă, "A distributed load balancing approach for industrial IEEE 802.11 wireless networks," in *Proc. IEEE 17th Int. Conf. Emerg. Technol. Factory Automat.*, Krakow, Poland, Sep. 2012, pp. 1–7.
[38] J. Geier. *How To: Define Minimum SNR Values for Signal Coverage*. Accessed: Mar. 2, 2017. [Online]. Available: https://www.wireless-nets.com/resources/tutorials/define_SNR_values.html
[39] H. Gong and J. Kim, "Dynamic load balancing through association control of mobile users in WiFi networks," *IEEE Trans. Consum. Electron.*, vol. 54, no. 2, pp. 342–348, May 2008.
[40] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, "Wi-Balance: Channel-aware user association in software-defined Wi-Fi networks," in *Proc. IEEE NOMS*, Taipei, Taiwan, Apr. 2018, pp. 1–9.
[41] G. Sawma, I. Aib, R. Ben-El-Kezadri, and G. Pujolle, "ALBA: An autonomic load balancing algorithm for IEEE 802.11 wireless networks," in *Proc. IEEE NOMS*, Salvador, Brazil, Apr. 2008, pp. 891–894.
[42] A. K. Rangisetti, H. B. Baldaniya, P. B. Kumar, and B. R. Tamma, "Load-aware hand-offs in software defined wireless LANs," in *Proc. IEEE WiMob*, Larnaca, Cyprus, Oct. 2014, pp. 685–690.
[43] A. Furtado, R. Oliveira, M. Luís, R. Dinis, L. Bernardo, and P. Montezuma, "The impact of transmission errors in MAC schemes for distributed wireless networks," in *Proc. IEEE Sarnoff Symp.*, Newark, NJ, USA, May 2012, pp. 1–5.
[44] J.-W. Jang, Y.-S. Lim, and C.-K. Kim, "Traffic-aware decentralized AP selection for multi-rate in WLANs," *IEEE ICACT*, Phoenix Park, South Korea, Feb. 2010, pp. 278–283.
[45] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Trans. Comp. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.
[46] 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 Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.
[47] (Apr. 2019). *OpenVswitch*. [Online]. Available: http://openvswitch.org/
[48] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Queue*, vol. 14, no. 5, p. 50, Oct. 2016.
[49] D. H. Hagos, P. E. Engelstad, A. Yazidi, and Ø. Kure, "A machine learning approach to TCP state monitoring from passive measurements," in *Proc. WD*, Dubai, United Arab Emirates, Apr. 2018, pp. 164–171.
[50] H. Jung, T. Kwon, K. Cho, and Y. Choi, "REACT: Rate adaptation using coherence time in 802.11 WLANs," *Comput. Commun.*, vol. 34, no. 11, pp. 1316–1327, Jul. 2011.
[51] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "Developing a predictive model of quality of experience for Internet video," in *Proc. ACM SIGCOMM*, Hong Kong, Aug. 2013, pp. 339–350.
[52] R. V. L. Hartley, "Transmission of information," *Bell Syst. Tech. J.*, vol. 7, no. 3, pp. 535–563, Jul. 1928.
[53] J. Saldana, R. Munilla, S. Eryigit, O. Topal, J. Ruiz-Mas, J. Fernández-Navajas, and L. Sequeira, "Unsticking the Wi-Fi client: Smarter decisions using a software defined wireless solution," *IEEE Access*, vol. 6, pp. 30917–30931, 2018.

SHIRONG LIN received the M.Sc. degree from the Institute of Engineering Mechanics, CEA, China, in 2012. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Harbin Institute of Technology University. His research interests include deep learning, data mining, and software-defined networking.

NAN CHE received the Ph.D. degree from the Harbin Institute of Technology University, China, in 2012. He is currently a Professor with the Department of Software, Harbin University of Science and Technology. His research interests include deep learning, software-defined radio, and WSN.



FEI YU received the M.Sc. degree from the Harbin Institute of Technology University, China, in 2014. She is currently pursuing the Ph.D. degree with the Department of Computer Science, Harbin Institute of Technology University. Her research work has been published in premier conferences (e.g., MDM, PAKDD, and Apweb-WAIM), and journals (e.g., *Journal of Software and Information Sciences*). Her research interests include deep learning, data mining, location-based services, and recommender systems.

SHOUXU JIANG received the M.S. and Ph.D. degrees from the Harbin Institute of Technology University, China, in 1995 and 2007, respectively, where he is currently a Professor with the Department of Computer Science. His research interests include software definition wireless, intelligent navigation, data management, GIS, mobile computing, location-based services, and data privacy. His research work has been published in VLDB, TKDE, ICDE, Sensys, MDM, PAKDD, and Apweb-WAIM.