AN OPPORTUNISTIC SCHEDULER FOR DENSE WLANS

Kwan-Wu Chin

School of Electrical, Computers, and Telecommunications Engineering
University of Wollongong, Northfields Ave, NSW, Australia 2522
kwanwu@uow.edu.au

ABSTRACT

The demand for bandwidth by multimedia applications remains unabated. This is particular critical given the growing number of devices with WiFi capability, and the ubiquity of Wireless Local Area Networks (WLANs). These trends have spurred researchers to develop low-cost and backward compatible solutions to increase the capacity of WLANs. One approach is to deploy additional Access Points (APs), and strategies to manage channel, user, and transmit power. As a result, stations are likely to be near one or more APs, and therefore are more likely to experience high data rates. In this paper, we take advantage of this fact to increase the capacity of a dense WLAN further by transmitting packets to a station via a neighboring AP if its associated AP is occupied by another station. Our approach is novel as prior works have not exploited the density and diversity of APs when scheduling downstream traffic. From extensive simulation studies, we show its viability in varying traffic scenarios, and in particular, WLANs with a high number of APs.s.

KEYWORDS

Scheduling, Wireless Local Area Networks, Access Points Diversity, Network Capacity

1. INTRODUCTION

The lack of capacity in current Wireless Local Area Networks (WLANs) is now a key concern as connectivity to the Internet via WLANs is now ubiquitous, as evident by the availability of hotspots in cities, universities and airports, and WiFi support in most portable devices. Users therefore expect access to the Internet anywhere anytime. Moreover, they require a minimum level of Quality of Service (QoS). Unfortunately, the popularity of WLANs has led to increased contention and ultimately low user satisfaction as multimedia applications require a minimum QoS in terms of delay and throughput. Hence, there is a need for solutions that increase the capacity of existing WLANs. One promising solution is to increase the number of APs. This has the benefit of providing better coverage and to load balance users amongst different APs. As an example, Intel Corporation has deployed 125 APs in its Oregon office to ensure adequate support for all voice, video and data traffic [14]. The advantages of dense WLANs have also attracted commercial interests. For example, Meru Networks sell dense, centrally controlled WLANs and software responsible for channel allocation, user association, and traffic management. Other vendors with similar products include Aruba Networks and Extricom.

Figure 1 shows a dense WLAN. A typical component that exists in dense WLANs is the use of a controller to manage user associations, air-time, frequency, and transmit power control. In addition, the controller is able to optimize the delivery of downstream and upstream traffic to/from stations given its intimate knowledge of AP coverage, and interference profile of stations and APs. In fact, as explained in Section 4.0, our scheduler/controller is able to use this information to increase the delivery of downstream packets to stations via its associated or neighbouring APs.

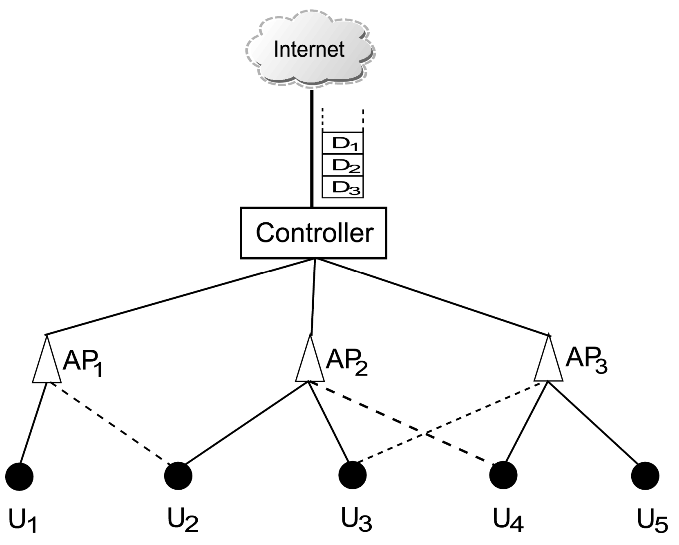


Fig. 1: An example enterprise WLAN with three APs. u_x denotes station *x*. Solid line indicates the associated AP of a given station. Dotted lines indicate alternate APs that can be used to deliver packets to stations. D_x corresponds to a data packet for station *x*.

To date, there are a number of related works concerning dense WLANs. These works, however, are mainly focused on the control plane. For example, Murty et al. [18] outlined the design and performance of dense APs in enterprises. A key contribution is a practical association strategy that works with unmodified stations. From a testbed comprising of 24 nodes in a 32m × 35m area, they recorded up to 1250% improvement in throughput. In a different work, Vasan et al. [21] co-located multiple APs per cell for load balancing purposes. They also adjusted each AP's carrier sense threshold to increase spatial reuse. In [5], the authors study the inter-dependencies between the following optimization strategies: channel allocation, load balancing and transmit power control. The key contribution is a framework that determines when each strategy, or their combination, should be used in a dense a WLAN. Finally, in [1] Ahmed et al. construct an annotated conflict graph, which is then used by the controller for channel assignment and transmit power control. Readers are referred to [19], and references therein, for the latest approaches on control plane management.

On the other hand, there has only been a handful of research on the data plane -- i.e., a scheduler/controller that manages the flow of packets to/from stations. Note that works such as [7][8] have shown that a large fraction, i.e., 70-80%, of traffic in WLANs are downstream in nature. Note, several companies claim to have software that controls the data plane, but their solutions are trade secrets. In [2][20], the authors showed how a central controller overcomes the following problems:

- *Hidden terminal.* The controller builds a Conflict Graph (CG) for a given WLAN; see [1]. Each vertex describes one of the following links: (i) AP-station, (ii), station--station, and (iii) AP-AP. Vertices are connected by an edge if the corresponding link interferes with one another. Given the CG, the problem is then to activate the maximum number of links, or in graph theoretic terms, find the maximum independent set -- an NP-hard optimization problem.
- *Exposed terminal.* They propose the following approaches: epoch scheduling, fixed backoff and packet staggering. These are needed due to the variable delays incurred by packets on the wired-path, and at their respective AP. As a result, APs tend to transmit packets at different times. In other words, they block each other from transmitting

concurrently. The proposed approaches, therefore, are practical solutions that ensure all exposed APs are able to transmit simultaneously.

The works of [2] and [20], however, are only focused on delivering packets to stations and APs that suffer from said problems. They, however, do not consider methods for improving the throughput of stations that are on non-interfering channels.

To this end, we investigate a scheduler that exploits the density and diversity of APs when scheduling downstream packets. Referring to Figure 1, assume the controller has packets for station u_2 and u_3. If AP_2 is busy transmitting to u_3, the controller can use AP_1 to deliver u_2's packets. This assumes all APs operate on a different frequency. Otherwise, AP_1's transmission will interfere with u_3 reception. Apart from that, u_2 needs to be informed the AP and time to receive its packet. Also, the scheduler will also need to consider fairness. For example, the scheduler cannot use AP_1 to transmit to u_2 at the expense of u_1. In other words, the scheduler needs to ensure the use of an AP to deliver packets to stations in a neighbouring Basic Service Set (BSS) does not starve stations. Lastly, the controller must be cognizant of varying downlink wireless conditions between APs and stations. From simulation studies, we find our scheduler to have a few orders of magnitude higher performance, in terms of delay and queue length, as compared to existing schedulers. Moreover, it is adaptive to varying burst sizes and non-uniform traffic.

In the next section, we first present the type of WLANs under consideration and our assumptions. After that in Section 3, we formulate the problem in hand and give examples. In Section 4, we outline our schedulers. This is then followed by our simulation methodology in Section 5. We present our results in Section 6, followed by a review of prior works in Section 7 before concluding in Section 8.

2. SYSTEM MODEL

We consider dense WLANs that use a central controller to manage both control and data plane operation. In addition, the controller buffers all downstream packets, and runs a scheduler that decides how packets are forwarded to stations. The scheduler, located at the controller, will maintain a single First-in-First-Out (FIFO) queue for each station. In this work, we assume all queues have unlimited capacity.

Similar to [2][20], all APs are connected by a switch to the controller. Moreover, these APs may be powered by the switch [14]. Each AP is assumed to be on an orthogonal channel. Note that, our schedulers work over IEEE 802.11a/b/g/n WLANs, and do not require any modifications to these standards. In addition, each AP informs the controller the success and failure of each packet transmission. As we will see in Section 4, this information allows the controller to track the state, i.e., busy or idle, of each AP.

We assume stations and APs support the IEEE 802.11k specification [13]. Hence, the controller is able to instruct APs and stations to collect data such as packet loss, signal quality and throughput. More importantly, it enables the scheduler to discover the set of APs that can be used to deliver packets to a station. Note, instead of IEEE 802.11k, stations can be installed with a software that exports to the controller all APs found during their scanning process.

We assume stations are able to switch or associate to multiple APs simultaneously. This can be achieved using MultiNet [6], where a station creates a virtual interface for each in-range AP. MultiNet, however, has a switching time in the tens of milliseconds. A better option is to use Kandula et al. [15]'s implementation, which has a 3ms switching delay. This is achieved by having the device driver maintain the transmission queue for each virtual interface.

Table 1 summarizes the information maintained by the controller, which are gathered via IEEE 802.11k or from association messages. The controller also monitors the free air time of each AP, which is determined by downstream and upstream traffic. Using the method presented in [18], APs first record the transmission time of a small, fix sized broadcast packet when the channel is idle, and store the resulting value as δ_{min}. The APs then broadcast the same packet periodically and record the number of times in which the transmission time exceeds δ_{min}. The controller then maintains the fraction of time in which these broadcast packets experienced a busy channel.

Table 1: Information maintained by the controller.

Information	*Description*
$\{STA_{mac}, AP_{id}\}$	This tuple maps a station to its AP.
$\{STA_{ip}, STA_{mac}\}$	Address Resolution Protocol (ARP) cache.
$\{STA_{mac},\quad , \ldots,\}$	The Received Signal Strength Indication (RSSI) observed by STA_{mac} to each in-range AP.
$\{AP_{id}, AirTime\}$	The free air time of each AP.
A_{sta}	The set of APs in range of station *sta*.
Q_i	The queue for station *i*.

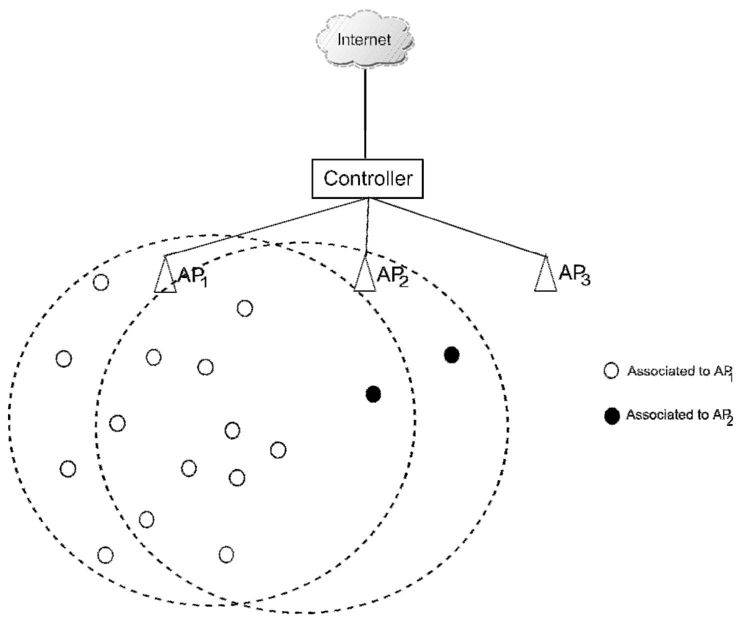


Figure 2: An example where most stations are associated to AP_1. This means AP_1 will have the highest load. A better option is to have AP_2 help forward packets to stations associated to AP_1.

3. The problem

Given a set of downstream packets, the goal is to schedule them as quickly as possible. A naive solution is to forward the packets to their respective AP in order of arrival. Hence, referring to Figure 1, the three data packets will be forwarded to AP_1 and AP_2 respectively. This solution, however, does not consider the following worst case scenario: see Figure 2. As we can see, all users are associated to the same AP -- as is the case when people use their laptop during meetings. This means all packets will be forwarded to AP_1, even though AP_2 could be used to deliver some of these packets. Advantageously, if the two stations associated to AP_2 are mainly idle, AP_2 can be recruited frequently to help alleviate the load of AP_1. We like to point out that current solutions have approached this problem by controlling how stations associate to APs. However, they are unable to take advantage of favourable channel condition offered by alternative APs, and also do not consider the traffic characteristics of stations; see Section 7.

A similar situation is when a subset of stations associated to an AP has a high volume of traffic, thereby, preventing other stations from receiving service promptly. This is particularly critical if some stations in the BSS are running real-time applications. Apart from that, the throughput of these stations can be exacerbated further by the performance anomaly problem [4]. That is, stations with low data rate will occupy a disproportionate amount of air-time as compared to those with better link condition. This means low rate stations with high traffic will severely reduce the throughput of a BSS.

4. PROPOSED SCHEDULER

Our scheduler exploits multiple APs to boost the capacity of a BSS by allowing stations to receive packets from the *best* AP, which may not correspond to their associated AP. In fact, it treats APs as *virtual antennas* that cooperatively deliver packets to stations. In addition, it preferentially schedules downstream traffic that has the longest waiting time. Lastly, it monitors delivery status via acknowledgments returned by APs, and makes use of IEEE 802.11k to learn the channel condition between APs and stations.

The details of our scheduler are shown in Algorithm 1. The goal is to determine the assignment of packets to APs whilst ensuring no starvation. To amortize the cost of switching overheads incurred by stations and variable wired delays, Algorithm 1 is only run every *epoch* -- defined as periods of time when packets are transmitted in batches [20]. As the value of epoch increases, efficiency improves at the expense of increased delay. In [20], based on experimentations, the authors used an epoch value of 10ms. In our simulation, however, we set an epoch to be equal to the beacon interval of all APs.

```
input  : Q
output : AP assigned to backlogged queues
/* Initialize set to ($Q_i$, $AP_{id}$) tuples */
1.    S = {}
      /* Set of queues with packets. */
2.    B = GetQueue(Q)
3.    while B ≠ {} do
      /* Get queue with max delay. */
4.    b = MaxDelay(B)
5.    if b ≠ NULL then
6.            ap = FindBestAP(b,S)
7.            if ap ≠ NULL then
8.               S ∪ (b, ap)
9.            end
10.   end
11.   B \ b
12.   end
13.   return S
```

Algorithm 1: Proposed Scheduler.

Our scheduler begins by initializing the variable S, which contains queue Q_i to AP mappings, to the empty set. After that, it retrieves the non-empty queues. From line 5--18, the scheduler starts by picking the queue in which its HOL packet has the largest delay. After that, it searches for the best AP assignment for said queue. It does this by calling the function *FindBestAP(b)*, see Algorithm 2, to determine the best AP that can be used to forward *b*'s packet. The function accesses A_{sta}, see Table 1, and S to determine an AP *a* that meets the following criteria: (i) it has not been scheduled; i.e., *a* is not in the set S, (ii) it has the highest free air time, and (iii) it has the best data rate. In addition, it returns the same AP as the previous epoch if the AP has not finished delivering the station's packets. This ensures the station is tuned to the correct BSS/channel in the next epoch. If an AP is found, its address is added to *S* along with the corresponding queue's address. The scheduler then removes *b* from B. Note that, at each epoch, due to line-11, only one AP is responsible for a station's packets.

```
input  : b, S
output : $AP_{id}$
/* Determine if an AP is already handling b. If yes, return that AP */
1.      if( ap=APq(b) ) $\neq$ NULL then
2.      return ap;
3.      end
4.      else
        /* Remove APs already assigned to other queues from $A_{sta}$. */
5.              RemoveAP($A_{sta}$, S)
6.              /* Return the AP with max data rate */
8.                 return MaxWeight($A_{sta}$,b)
10.     end
```

Algorithm 2: FindBestAP. The function APq(.) determines whether an AP is still busy transmitting packets to the station corresponding to queue b, whereas MaxWeight(.) returns the AP that has the highest r×T vaue, where r is the data rate to the given station, and T is an AP's free air-time.

The scheduler uses the Information Element (IE) field in beacon messages to inform the respective station the AP, and hence channel, in which to receive a packet. The IE, which has a maximum size of 256 bytes, contains the tuple <BSSID, mbit>. Our scheduler uses the Bloom filter to indicate whether a given station is to receive packets from the AP with identity BSSID. Specifically, the scheduler hashes a station's MAC address onto the mbit. Upon receiving a beacon, a station accesses the IE field, iterates through each tuple, hashes its MAC address onto the mbit to determine its AP assignment. Once found, it switches to the designated BSS. Otherwise, it remains in its current BSS. Note that a station is aware of all nearby APs as part of its scanning process.

Figure 3 illustrates the key advantage of our scheduler. Assume the stations are within range of all APs, and are associated as follows: (i) AP_1:{A, B, C}, (ii) AP_2:{D, E}, and (iii) AP_3: {F}. In the controller's queue, there are six packets, each headed to the indicated station. Also shown is the transmission schedule determined by our scheduler, which takes a total of two epochs to deliver all packets.

If the controller were to send packets in order of arrival, AP_1 will take three epochs, whereas AP_2 and AP_3 will take two and one epoch respectively. In contrast, our scheduler delivers all packets in two epochs. We like to point out that our scheduler can be modified to assign a higher weight to a given station. For example, station F's packet can be scheduled in the first epoch by assigning a higher weight to its queue.

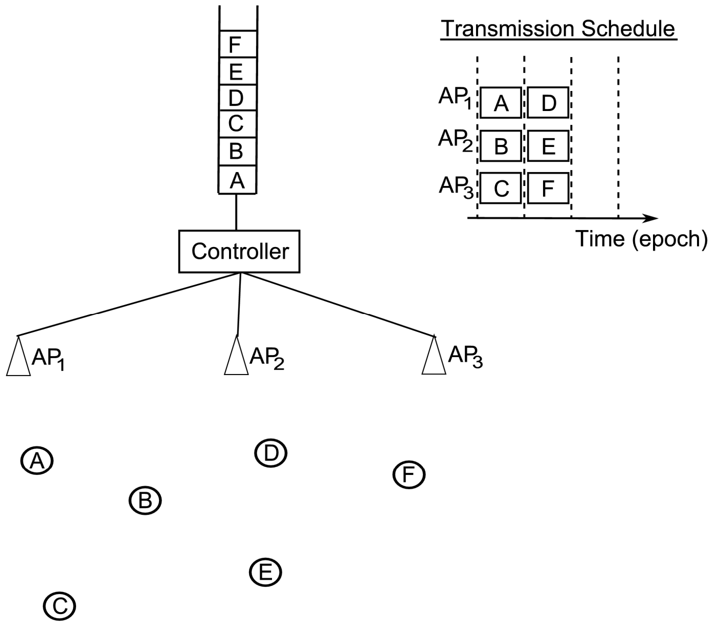


Figure 3: An example showing the scheduling of packets.

In summary, our scheduler, which enables the delivery of packets via multiple APs, has the following key features:

- It maintains *n* queues, corresponding to *n* stations.
- It schedules *K* longest queues, where *K* is the number of APs.
- It always selects the AP with the following properties: (i) free, (ii) highest amount of air-time, i.e., lowest load, and (iii) data rate. Moreover, stations with undelivered packets from the last epoch are assigned the same AP.
- It uses IE and Bloom filter to inform stations of the AP that will be delivering their packets.

5. SIMULATION METHODOLOGY

We used a custom, discrete-time simulator. There are 10 APs and 20 stations. Each AP is assumed to operate on a different channel. Moreover, it supports IEEE 802.11a [11]. Stations are randomly assigned to an AP, and their respective data rates are drawn randomly from the set r {6, 9, 12, 18, 36, 48, 54}. We assume the channel to be constant for the duration of an epoch, but varies over time. Specifically, we assign a station a new data rate at the start of each epoch. This means at each epoch, the number of packets delivered to a station will be different, as per the station's data rate and packet size.

The arrival process of each station is governed by a Bernoulli process. Specifically, with probability p_a, a station receives η burst of packets in a given epoch. Note, in Section 6, p_a is labeled as *load*. For uniform traffic experiments, all stations have the same p_a value. However, for non-uniform traffic scenarios, except for 10 stations with a fixed p_a value, we vary their p_a value in increments of 0.1.

The central controller maintains a queue for each station; i.e., a total of 20 queues. Note, we assume infinite buffer space for all queues. In all our experiments, we collected the following results, which are an average of 100 simulation runs: (i) delay, and (ii) queue length.

For comparison purposes, we also implemented the following schedulers:

- **FIFO**. In this approach, the controller sends a packet to the AP that is responsible for a given station. Packets are forwarded and transmitted according to the order of arrival.
- **APLongQ**. Each AP maintains a separate queue for each associated station. At each epoch, the AP serves the longest queue.
- **MaxRate**. At each epoch, the controller selects the station that has the best data rate -- either via its associated or neighbouring AP. After assigning an AP to the station's queue, it searches for the next station with the highest data rate. The process continues until all APs are assigned a station.

6. RESULTS

We conducted two sets of experiments. The first employs uniform traffic, where all stations have the same p_a value. After that, in Section 6.2, we investigate the impact of non-uniform traffic. In both sets, we study the performance of our scheduler with increasing load, number of APs, and varying burst sizes.

6.1. Uniform Traffic

Figure 4 shows that packets forwarded using our scheduler experience much lower delays, and corresponding in Figure 5, we see that the controller observes a much shorter queue overall. In both figures, our scheduler improves performance by a few orders of magnitude. The main reason is the use of multiple APs, and priority given to long queues. Interestingly, MaxRate performs better than both APLongQ and FIFO at high load. This is because at high loads, each queue has more than sufficient packets to fill up a epoch, and take advantage of the high data rate. However, as it does not consider queue length, its average queue length remains high.

6.2. Varying Burst Sizes

Figure 6 shows the impact of varying burst sizes. Our scheduler is particularly suited to this scenario because it will quickly drain a queue that has received multiple bursts quickly using the AP that has the highest data rate. In this respect, our approach is particularly attractive as the APs are used efficiently and appropriately to increase the QoS experienced by users.

6.3. Number of APs

Figure 7 shows how the number of APs helps improve the average delay experienced by stations. This is as expected as the set of stations are divided amongst APs. Note that when there is only a single queue, MaxRate provides a better performance as it targets stations that have the highest data rate; i.e., a service that drains the most packets from the queue. However, as we add more APs, MaxRate is unable to take advantage the additional APs, and is oblivious of queue lengths. On the other hand, our scheduler increases the service rate of the WLAN, and hence reduces packet delays by making full use of additional APs. In fact, each additional AP yields a 40-80% drop in delay.

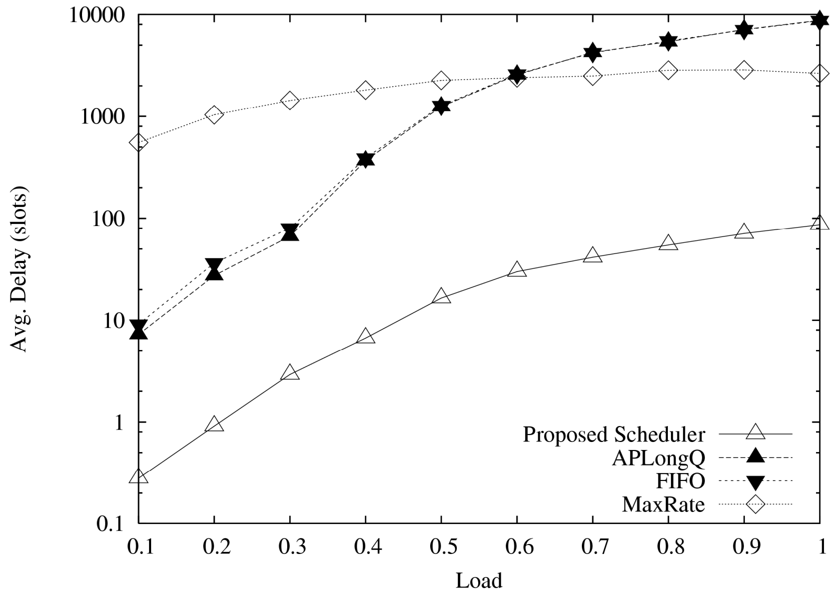


Figure 4: Load versus delay. 10 APs and 20 stations. Burst size fixed at five packets.

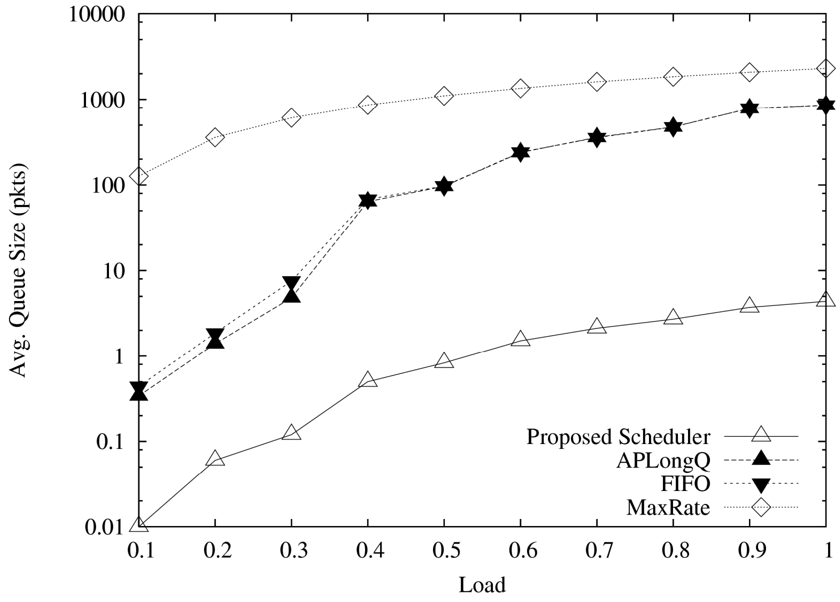


Figure 5: Queue size versus load. In this experiment, there are 10 APs and 20 stations. Each arrival contains a burst size of five packets.

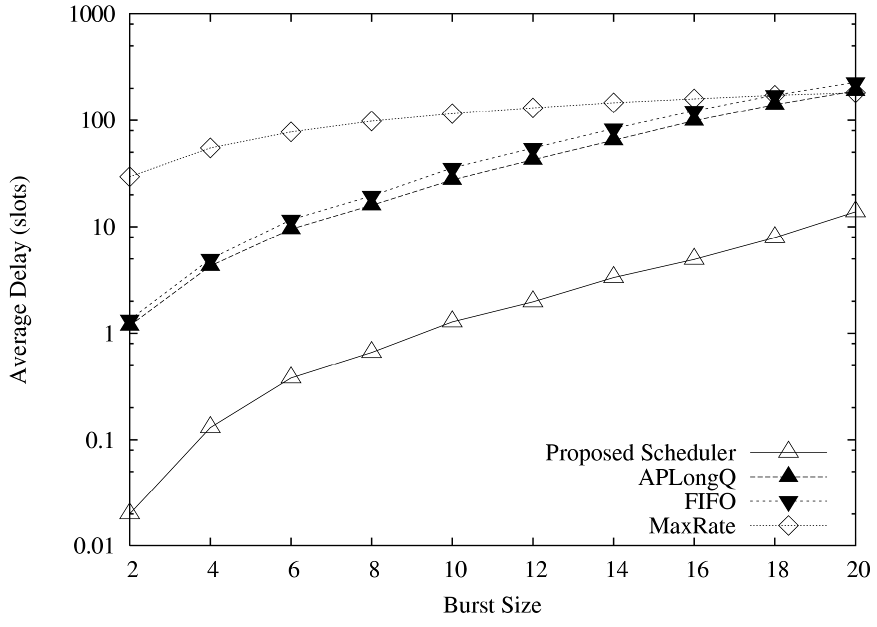


Figure 6: The impact of varying burst sizes. This experiment uses 10 APs and 20 stations, with each station injecting a fixed load of 0.1.

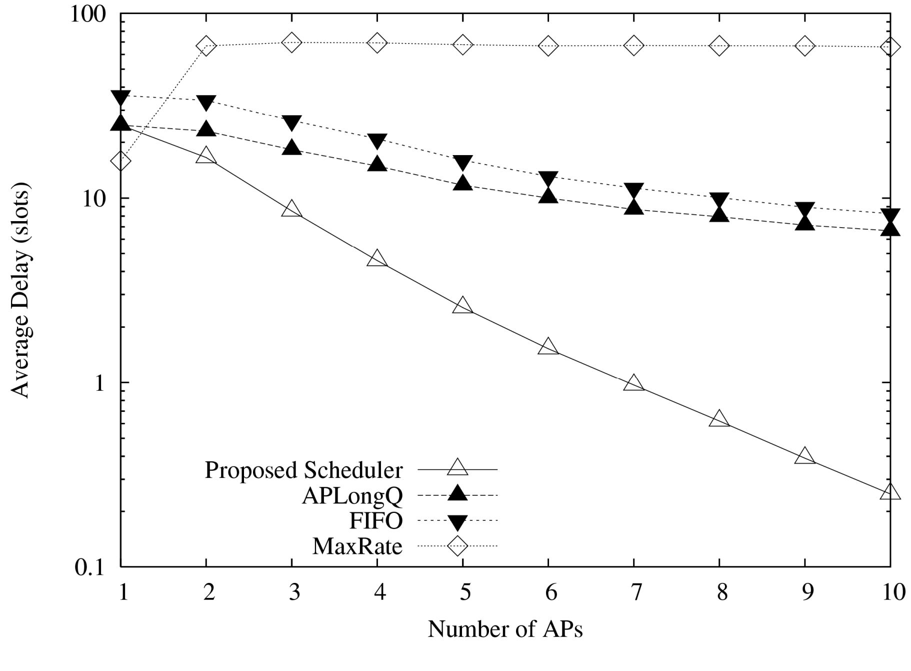


Figure 7: The benefits of additional APs. There are 20 stations, each has a load and burst size of 0.1 and 5 respectively.

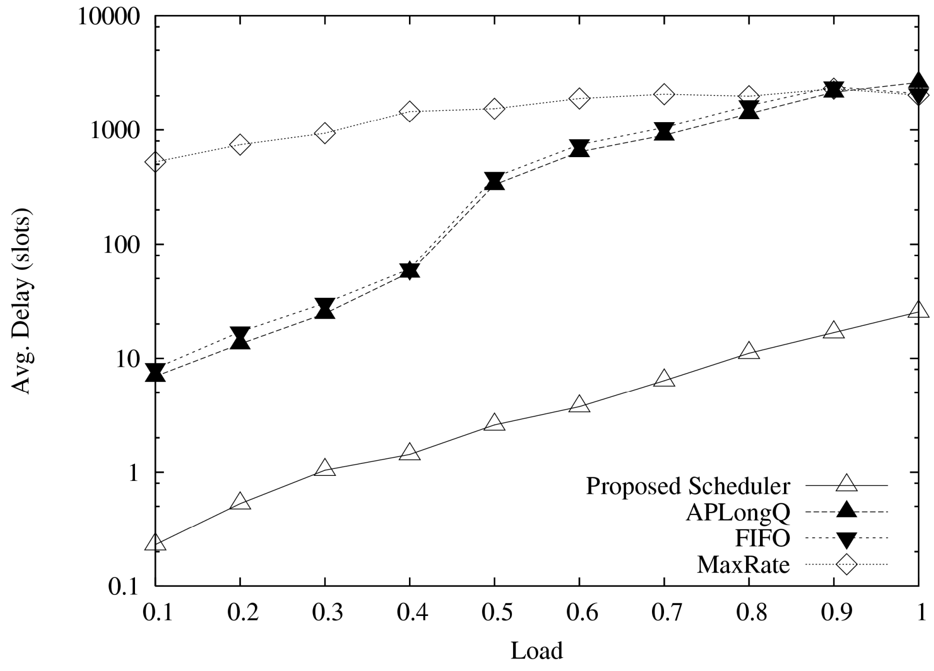


Figure 8: The impact of non-uniform traffic on delay. In this experiment, 10 stations have a fixed load of 0.1, whilst the load of the other stations are varied in increments of 0.1.

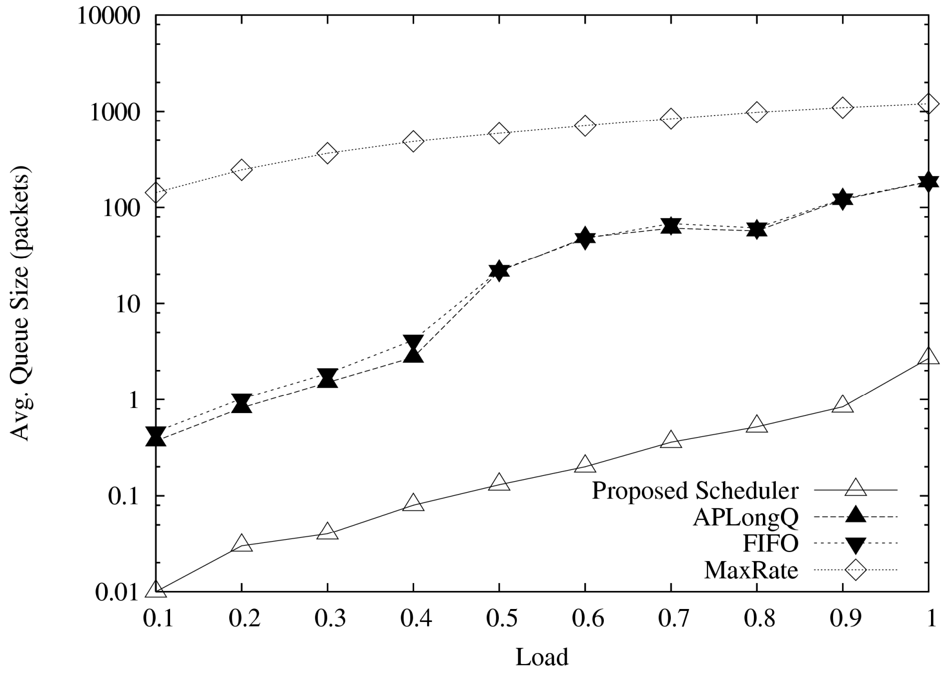


Figure 9: The average queue size of stations in non-uniform traffic scenarios.

6.4. Non-Uniform Traffic

As mentioned, we also studied the impact of non-uniform traffic, where we fixed the load of 10 stations at 0.1, and increase the rest by 0.1 after each experiment. In this section, we omit the result for varying burst sizes and number of APs as both of them bear similar characteristics to the uniform traffic case.

Figure 8 and 9 confirm the benefits of our scheduler in ensuring stations experience low delays and are serviced quickly. In non-uniform scenarios, MaxRate has a slight disadvantage at higher loads as a station with the best data rate may have a short queue. As for APLongQ and FIFO, stations are limited by their AP, unlike our scheduler, which exploits at least one AP to deliver packets.

7. RELATED WORKS

Our work resembles those concerned with designing stable schedulers for multi-queue multi-servers systems [10]. An example system is IEEE 802.16e [12], where queues and servers correspond to mobile stations and sub-carriers respectively. In these systems, the problem is to assign servers to queues with consideration for processes that govern traffic arrival and server connectivities. This means a scheduler may implement a policy whereby mobile stations are assigned servers/channels according to their queue length and channel condition. Halabian et al. [10] proposed and proved the stability of a Any Server/Longest Connected Queue (AS/LCQ) policy. They preferentially assign servers to the most backlogged queue. In [16], the authors show that maximizing throughput alone, where sub-carriers are allocated to the best users, lead to instability. Consequently, they propose a number of algorithms that consider queue lengths. These works, however, are fundamentally different to ours in the following manner. First, each queue can be assigned multiple servers. In IEEE 802.16e, this means multiple sub-carriers are assigned to a given station to reduce its queue length or delay, or to take advantage of superior channel condition. In contrast, the queues in our system can only be assigned one AP at a given time. This is because a station can only receive from one AP at any given point in time. Secondly, prior works assume all servers are available for assignment. However, in our case, each AP has a different coverage area. As a result, some APs are likely to be out of range or have very poor data rates to a station. This means not all APs can be used to deliver packets to a station. Thirdly, our scheduler considers the data rate from each AP. This is different to work such as [9], which only considers queue length.

We like to point out that there are also a number of prior works that focus solely on upstream traffic. For example, Kandula et al. [15] proposed a scheduler that allows a station to maintain multiple sessions to different APs. Their solution effectively overcomes the bandwidth limitation of its AP's link to the Internet. For example, if there are five APs, each with a 1 Mbps link to the Internet, a station will be able to exploit all five APs to gain a throughput of 5 Mbps. Another example is [17], where Miu et al. exploit multiple APs, and hence their different link condition, to recover from path dependent errors. Specifically, APs forward packet transmissions from a station to a server. If the station's packet is erroneous, the server uses the different copies to recover the original packet.

Another approach to the problems identified in Section 3 is to intelligently associate users according to AP load; see for example [3] and references therein. Hence, the stations shown in Figure 2 will be associated uniformly amongst the three APs. This approach, however, does not consider changing wireless conditions that allow a station to receive at a higher data rate from a non-associated AP. Moreover, such approaches do not exploit idle periods of nearby APs to help drain a backlogged queue.

8. CONCLUSIONS

We have proposed a scheduler that takes advantage of multiple APs, and their respective coverage and data rate to stations in order to increase the capacity of a WLAN. Our extensive simulation studies, involving both uniform and non-uniform traffic, confirm the efficacy of our scheduler in varying traffic scenarios. In particular, our results show packets scheduled using our algorithm receiving a few orders of magnitude lower delay and shorter queue length as compared to existing methods.

REFERENCES

[1] N. Ahmed and S. Keshav. SMARTA: A self-managing architecture for thin access points. In ACM CoNext, Lisbon, Portugal, Dec. 2006.

[2] N. ahmed, V. Shrivastava, A. Mishra, S. Banerjee, S. Keshav and K. Papagiannaki. Interference mitigation in enterprise WLANs through speculative scheduling (extended abstract). In ACM MOBICOM, Montreal, Quebec, Canada, Sept. 2007.

[3] Y. Bejerano and S.-J Han. Cell breathing techniques for load balancing in wireless LANs. IEEE Transactions on Mobile Computing, 8(6):735-745, June 2009.

[4] G. Berger-Sabbatel, F. Rousseau, M. Heusse, and A. Duda. Performance anomaly of 802.11b. In IEEE INFOCOM, San Francisco, CA, USA, 2003.

[5] I. Broustis, K. Papagiannaki, S. Krishnamurthy, M. Faloutsos and V. Mhatre. Measurement-drive guidelines for 802.11 WLAN design. IEEE/ACM Transactions on Networking, 18(3):722-735, June, 2010.

[6] R. Chandra and P. Bahl. Multinet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In IEEE INFOCOM, Hong Kong, China, 2004.

[7] Y. Cheng, J. Bellardo, P. Benkau, A.C. Snoeren, G. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In ACM SIGCOMM, Pisa, Italy, Sept. 2006.

[8] J. Erikson, S. Agarwal, P. Bahl, and J. Padhye. Feasibility study of mesh networks for all-wireless offices. In ACM MobiSys, Upssala, Sweden, June, 2006.

[9] A. Ganti, E. modiano, and J. Tsitsiklis. Optimal transmission scheduling in symmetric communication models with intermittent connectivity. IEEE Transactions on Information Theory, 53(3), March, 2007.

[10] H. Halabian, I. Lambadaris, and C.-H Lung. Network capacity region of multi-queue multi-server queuing system with time varying connectivities. In IEEE International Symposium on Information Theory, Texas, USA, June 2010.

[11] IEEE. Part 11a: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: High speed physical layer in the 5 GHz band, standard specification. IEEE Std 802.11a-1999, 1999.

[12] IEEE. Amendment 2: Physical and medium access control layers for combined fixed and mobile operation in licensed bands. IEEE 802.16e-2005, 2005.

[13] IEEE. Amendment 1: Radio resource measurement of wireless LANs. IEEE Std 802.11k-2008, 2008.

[14] A.P. Jardosh, K. Papagiannaki, E.M. Belding, K.C. Almeroth, G. Iannaccone, and B. Vinnakota: Green WLANs: On-demand WLAN Infrastructures. ACM/Springer MONET, 14(6):798-814, June, 2009.

[15] S. Kandula, K. C. Lin, T. Badirkhanli and D. Katabi. FatVAP: Aggregating AP backhaul capacity to maximize throughput. In the 5th USENIX Symposium on Networked Systems Design and Implementation, San Francisco, USA, April, 2008.

[16] S. Kittipiyakul and T. Javidi. Resource allocation in OFDMA with time-varying channels and bursty arrivals. IEEE Communications Letters, 11(9):708-711, 2007.

[17] A. Miu, H. Balakrishnan, and C. emre Klksal. Improving loss resilience with multi-radio diversity in wireless networks. In ACM MOBICOM, Cologne, Germany, September, 2005.

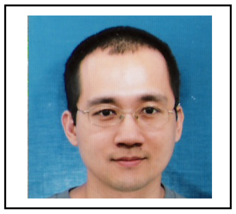
[18] R Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill. Designing high performance enterprise Wi-Fi networks. In the 5th USENIX Symposium on Networked Systems Design and Implementation, San Francisco, USA, April, 2008.

[19] R. Murty, J. Padhye, A. Wolman, M. Welsh. Dyson: An architecture for extensible wireless LANs. In ACM USENIX, Boston, USA, June, 2010.

[20] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, and K. Papaginnaki. CENTAUR: realizing the full potential of centralized WLANs through a hybrid data path. In ACM MOBICOM, Beijing, China, Sept., 2009.

[21] a. Vasan, R. Ramjee, and T. Woo. ECHOS: Enhanced capacity 802.11 hotspots. In IEEE INFOCOM, San Diego, USA, March, 2005.

Authors



A/Prof Kwan-Wu Chin obtained his Bachelor of Science with first class Honours, and PhD with the Vice-Chancellor's commendation from the Curtin University of Technology in Western Australia. He has published 47 papers on various aspects of computer networks, which include active/programmable networks, wireless networks, routing in mobile ad-hoc networks, and energy efficient protocols for wireless sensor networks. He also holds three United States patents.