Interference-Aware Video Streaming Over Crowded Unlicensed Spectrum

Ching-Ling Fan, Daniel Huang, Pin-Chun Wang, and Cheng-Hsin Hsu
Department of Computer Science, National Tsing Hua University, Taiwan

***Abstract*—Video conferences over the Internet with multiple participants have to be wirelessly connected to improve the communication efficiency. However, video conferences are often held in places suffering from crowded wireless medium, such as offices and schools. This leads to many challenges on providing high video conferencing experience. In this paper, we study the problem of video streaming over crowded wireless networks considering the interference. We conducted extensive experiments to model the unlicensed spectrum activity and design, implement, and evaluate an interference-aware bandwidth estimation and rate adaptation algorithm. The experiment results show that our proposed solution (i) reduces the retransmission ratio to lower than 6.5%, (ii) reduces the packet loss rate to 0.5% on average, (iii) achieves higher throughput, and (iv) leads to higher video quality than others by at least 8.3 dB in video quality.**

Fig. 1. A typical multi-site video conferencing setup.

I. Introduction

Modern workspace has been digitalized with various real-time collaboration systems, such as video conferencing. It is projected that 50% of conference rooms will be video-enabled by 2020 [8], which enables the possibility of multi-site video conferencing sessions with several participants in each conference room. Although, today, we may still connect our computers to video conferencing peripherals, like projectors, large displays, and cameras via cables, such as HDMI, VGA, USB, and Ethernet cables. Doing so in modern conference rooms with many participants is becoming increasingly more inefficient, e.g., participants may suffer from high *cabling* overhead, or even inflexibility, e.g., remote participants may not concurrently see documents opened on *different* local computers. To address these limitations, the video conferencing peripherals have to be *wirelessly* connected to the computers. Among the legacy cables, replacing HDMI/VGA cables with wireless technologies is most challenging, due to the bulky and delay-sensitive nature of video streaming.

The industry sees such market opportunity, and several wireless video streaming standards have been recently proposed to replace HDMI/VGA cables, including WiGig [5], WirelessHD [7], WHDI [6], and WiDi [2]. In fact, video streaming is the main driving force of next generation wireless technologies: for instance, the gigabit wireless market is expected to grow from 0.3 billion USD in 2015 to 2.73 billion in 2022, with an annual growth rate of 32.9% [9]. Although these new wireless standards are emerging, they are all vulnerable to *interference*, because all these standards work over *unlicensed* spectrum, which is usually crowded in offices, schools, and residences. The core challenge of high-quality video streaming is to *estimate* the available bandwidth, in order to *control* the video encoders for the highest possible video streams without resulting in network congestion. Existing solutions rely on either: (i) reactive bitrate controls, e.g., via monitoring packet loss rate [20] or queue status [12], or (ii) proactive bitrate controls, e.g., over the WiFi networks [10], [17] or the Internet [18], [13]. These studies do not take wireless interference into considerations, leading to less accurate bandwidth estimation [11], [24], and hence inferior video quality.

In this paper, we study the problem of streaming multiple videos over crowded unlicensed spectrum in an interference-aware fashion. To do that, we propose to deploy monitoring nodes, which are essentially wireless access points, but responsible for *sniffing* wireless frames sent to, or received from others; this monitoring node are used to collect statistics such as modulation and coding schemes, air-time ratio, and interference level. With the collected statistics, we develop an interference-aware approach to estimate the available bandwidth that can be used by the video stream. In the future, we plan to connect all wireless access points to a centralized controller, so that individual access points report the air medium statistics to the controller for overall bandwidth estimation and scheduling.

The paper is organized as follows. Section II gives the usage scenario. The literature survey is in Section III. We monitor and model the throughput in Section IV. Our solution is proposed in Section V and evaluated in Section VI. Section VII concludes this paper and describes the future work.

II. Usage Scenario

Fig. 1 shows a typical video conferencing setup. Several participants use their laptops to join a video conferencing session from a conference room at the local site (on the right of this figure). A wireless projector is used to display one or multiple laptop screens, while an IP camera is used to capture

the appearance of all the participants. The laptop screens and/or IP camera videos may be streamed to participants at the remote sites (on the left of this figure) over the Internet.

In this usage scenario, we solve the problem of streaming multiple videos through a single access point over crowded unlicensed spectrum. In particular, there are at least four video streams: (i) the IP camera to laptops(s), (ii) laptop(s) to the projector, (iii) laptop(s) to the Internet, and (iv) the Internet to laptop(s). Streaming so many videos over the same access point and air medium results in potential network congestion, which in turn leads to high packet loss and low throughput. We notice that these videos have different characteristics, e.g., a presentation document requires higher resolutions, while an animation movie requires higher frame rates. Therefore, by: (i) accurately estimating the available wireless bandwidth, and (ii) splitting the bandwidth among video streams in a content-dependent way, we can maximize the user experience. In the rest of this paper, we focus more on the former problem (interference-aware bandwidth estimation) over crowd unlicensed spectrum. We could not solve the latter problem (bandwidth allocation) due to the space limitation. That is, we only consider a single video stream in the current paper.

III. Related Work

Estimating available network bandwidth in crowded unlicensed spectrum has been studied in the literature. It is pointed out that the estimated bandwidth is often off, if the estimators fail to consider wireless interference [11] [24]. Therefore, we only survey related work that explicitly takes interference into considerations.

Estimating the interference level. Shrivastava et al. [25] propose to use a controller to monitor the frame transmission timestamps and successfully-transmitted frames from/to individual APs (Access Points). Therefore, the controller can infer the interference by calculating the overlapping portions and the loss rates of frames. They also study several interference mitigation mechanisms, such as channel assignments and transmission power control. Zhao et al. [26] adopt the packet reception rate models [19] to capture the interference level. Based on this, they propose to use a Software-Defined Networking (SDN) controller to mitigate the interference by scheduling the downloading packets to achieve high packet reception rate. The above studies assume that all APs are managed by a controller, which is less common in real deployments. There are also studies that use distributed monitoring nodes to sniff the wireless frames without a centralized controller. Sheth et al. [23] propose a system that detects several PHY layer statistics including interference. They make observations on the relationship among the signal strength, the corrupted frames ratio, the number of retransmissions, and the transmitted frame sizes, and interference levels. Riggio et al. [21] also propose to detect the interference with monitoring nodes in wireless mesh networks and further propose a channel assignment scheme to lower the interference level. The aforementioned work [25], [26], [19], [23], [21] only quantifies the interference levels.

Estimating the link capacity. The problem of estimating the wireless link capacity has also been studied. Kashyap et al. [16] propose a PHY-layer model that estimates the carrier busy ratio using the received interference power, and the packet delivery ratio using the Signal-to-Interference-plus-Noise-Ratio (SINR). They then propose sender/receiver side MAC-layer models to calculate the link capacity under multiple interfering nodes. Jindal et al. [15] propose to monitor the service time of data packets at each link and estimate the residual capacity among interfering links. After several iterative calculations, the estimated capacity will be converged. Rossi et al. [22] propose to derive/measure the expected/actual transmission intervals of individual data frames to estimate the fraction of time taken by interference. Besides, they compute the saturated link capacity of an access point (rather than a mobile client). None of these studies [16], [15], [22] estimate the available bandwidth, which is the highest sending rate that would not lead to wireless network congestion. The current paper strives to estimate the available bandwidth for video streaming traffic.

IV. Monitoring the Unlicensed Spectrum

In real deployments, not all APs are connected to the controller. Therefore, the crux of our solution is a monitoring node, which is developed and exercised in this section.

A. Setup

We build a testbed using off-the-shelf components, as illustrated in Fig. 2. We deploy the testbed in our lab, which is covered by more than two dozens of APs with intensive background traffic and interference. The AP is built on a Linux box with two wireless cards. One of them is configured into an AP using hostapd [1]. The other one is configured into the monitoring mode to capture all the frames on a specific channel with libpcap [4] for real-time analysis. These two wireless cards are installed on the same Linux box, so that the monitoring node captures all frames transmitted through the AP and those frames that interfere with the aforementioned ones. There are two laptops connected to the APs. We use iPerf [3] to generate a constant bitrate video stream over UDP, where the video bitrate b is varied among $\{1, 2, 4, 8, 16\}$ Mbps. We consider three disjoint IEEE 802.11n channels: 1, 6, and 11. We run each test for 60 seconds, and repeat it 5 times. We report the average results, along with the minimum and maximum values among 5 runs whenever applicable.

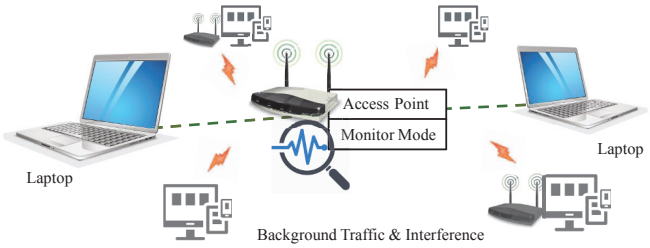


Fig. 2. Our WiFi testbed with an AP and a monitoring network interfaces.

B. Modeling Throughput

We may obtain the following values of each frame using libpcap [4]: (i) frame size, (ii) data rate determined by Modulation and Coding Scheme (MCS) mode, (iii) receiving time, and (iv) retry flag. Using these values, we derive the following metrics.

- **Air-time ratio** τ, which is the ratio of the occupied air-time to the total time.
- **Data rate** Λ, which is the number of transmitted bits per second.
- **Retransmission ratio** r, which is the ratio of the number of retransmitted bits to the number of total transmitted bits.
- **Throughput** θ, which is the number of successfully transmitted bits per second.

Next, we define some more symbols. Let s_f be the frame size, λ_f be the data rate, and t_f^r be the receiving time of frame f. We let t_f^s be the sending time of frame f. Since we consider short-range wireless networks with negligible propagation delay, we have:

$$t_f^s = t_f^r - \frac{s_f}{\lambda_f}. \tag{1}$$

We can then derive the air-time τ_f of frame f as:

$$\tau_f = \begin{cases} \frac{s_f}{\lambda_f}, & \text{if } t_f^s > t_{f-1}^r; \\ t_f^r - t_{f-1}^r, & \text{otherwise.} \end{cases} \tag{2}$$

Let τ be the air-time ratio within a time duration T. τ is written as:

$$\tau = \sum\nolimits_{f \in \mathbf{F}} \frac{\tau_f}{T}, \tag{3}$$

where $\mathbf{F}$ is the set of the transmitted frames within the time duration. We let Λ be the data rate, which is the total transmitted bits normalized to the time duration. We then have:

$$\Lambda = \sum\nolimits_{f \in \mathbf{F}} \frac{s_f}{T}. \tag{4}$$

Last, we derive the throughput θ as the data rate deducts the (wasted) rate due to retransmission. θ is written as:

$$\theta = \Lambda \times (1 - r), \tag{5}$$

where r is the retransmission ratio. Whether a frame is retransmitted can be determined by checking the retry flag. Nowadays, the wireless cards and drivers are mature, and the MCS mode and data rate selection are automatically done by a rate selection algorithm. Therefore, retransmissions, which are due to collisions and background noise, indicate the occurrence of interference. Hence, we use the retransmission ratio as the metric of the interference level.

C. Results and Observations

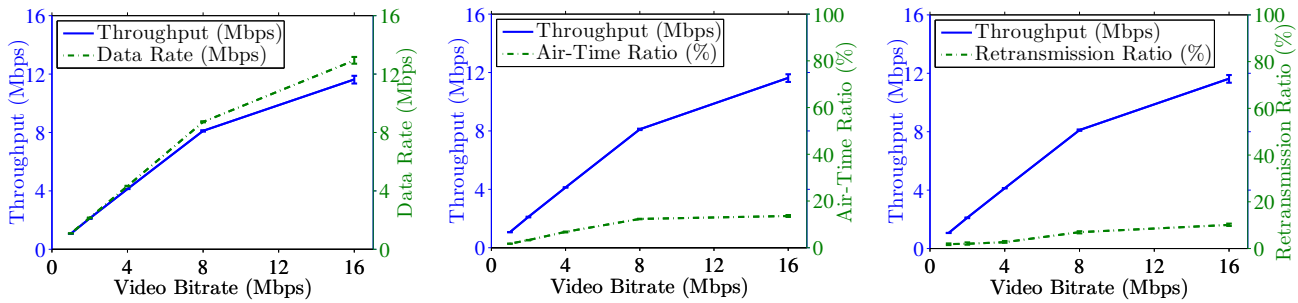
We plot the throughput with error bars of ranges (variance) under different video bitrates in Fig. 3. To understand the relation between the throughput and other metrics, we also present the air-time ratio, data rate, and retransmission rate in these figures. Sample results from channel 1 are presented for the sake of page limits. The figures show that the measurements have small range at each video bitrate, indicating that *the measurement results are consistent and reliable*. Fig. 3(a) shows the increasing trend of the throughput and data rate as the video bitrate increases. However, it is not linear, showing the nontrivial retransmission ratio r when bitrate is higher. Fig. 3(b) plots the air-time ratio with the throughput under different video bitrates. This figure shows that the air-time ratio increases as the throughput increases. However, the increasing trend diminishes when the video bitrate is close to 16 Mbps. This can be attributed to the fact that the throughput is close to the available bandwidth. This also leads to slightly higher variance among each measurements when video bitrate is set to 16 Mbps. *Therefore, the air-time ratio increases as the throughput increases. The air-time ratio does not increase proportionally to the throughput because of the existence of interference.* Fig. 3(c) plots the retransmission ratio with the throughput of the transmitted frames under different video bitrates. The frames are retransmitted because of failed transmission due to, e.g., collisions and noise. This figure shows that *high retransmission ratio indicates that the video bitrate is too high.* In the next section, we leverage the above observations to develop our solution.

V. Proposed Solution

A. Component Design

Fig. 4 gives a high-level overview of our proposed video streaming solution. We include several components to support the bandwidth estimation and rate adaptation mechanisms. They are highlighted in the bold font in this figure. These components are: (i) *monitoring node*, (ii) *bandwidth estimator*, (iii) *statistic collector*, (iv) *analyzer*, and (v) *bitrate reconfigurator*. The interactions among them are as follows. The monitoring node keeps capturing the frames on a particular channel and transmits the captured statistics, e.g., receiving times, data rates, and retry flags, to the statistic collector every T seconds. The statistic collector can also collect the statistics from other APs connected to the controller, combine the statistics, and report them to the analyzer. The analyzer estimates the throughput θ, the air-time ratio τ, and the retransmission ratio r using the statistics and then reports them to the bandwidth estimator. In the future, the analyzer may look into the transmission status of each connected AP to understand the overall network conditions or perform scheduling to improve the network performance.

After receiving the reports from the analyzer, the bandwidth estimator computes the available bandwidth by checking the retransmission ratio and the air-time ratio. If the retransmission ratio is high, the bandwidth estimator reduces the estimated available bandwidth. If the increasing rate of the air-time ratio is lower than it used to be, the bandwidth estimator keeps the current estimated bandwidth since this indicates that the throughput is close to the available bandwidth. Otherwise it increases the estimated bandwidth. The bandwidth estimator sets the target encoding bitrate as the same as the estimated available bandwidth and sends it to the video sender for the



(a) (b) (c)

Fig. 3. The metrics with throughput under different bitrates: (a) data rate, (b) air-time ratio, and (c) retransmission ratio. Sample results from channel 1 are shown.

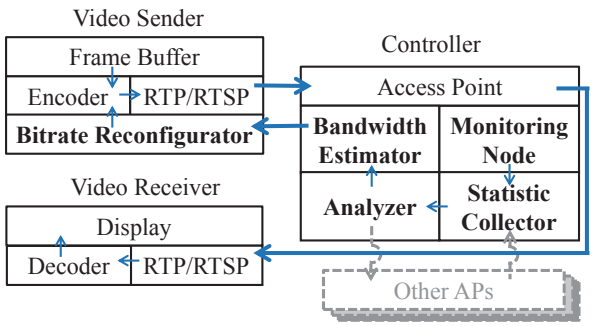


Fig. 4. The component diagram of our proposed controller.

Algorithm 1 Air-Time Based Rate Adaptation Algorithm

Phase 1 - Probing Estimation (Before Streaming)

1: Initialize b by iPerf Probing

Phase 2 - Monitoring Estimation (During Streaming)

2: **for** every T seconds **do**
3: Monitor and Compute θ, τ, r_r
4: **if** $r_r > r_r^H$ **then**
5: $b = b - \delta$
6: **else if** $\frac{\tau-\tau'}{\theta-\theta'} \geq \tau_p$ **then**
7: $b = b + \delta$
8: $\tau_p = \frac{\tau-\tau'}{\theta-\theta'}$
9: $\tau' = \tau$
10: $\theta' = \theta$
11: **end if**
12: **end for**

bitrate reconfigurator to adjust the encoding bitrate. The bitrate reconfigurator is able to not only adapt the encoding bitrate, but also select the best encoding parameters, e.g., resolution and frame rate, which is our future work.

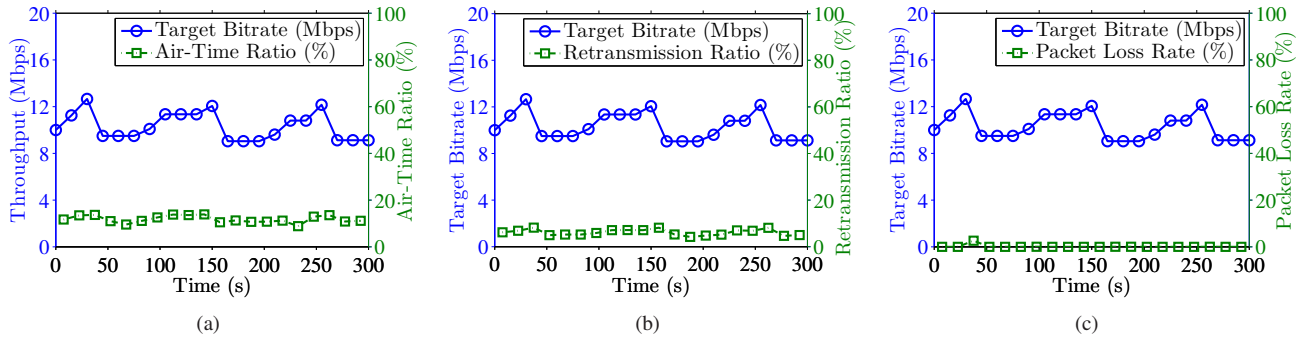
B. The Proposed Air-Time Based Rate Adaptation Algorithm

Algorithm 1 gives the pseudocode of our air-time based rate adaptation algorithm. Our algorithm consists of two phases: (i) before the streaming session and (ii) during the streaming session. We refer to them as phases 1 and 2, respectively. In phase 1, our algorithm first estimates the available bandwidth by sending probing packets before the video streaming starts. In particular, we use iPerf [3] to estimate the available bandwidth b and use it to initialize the encoding bitrate. In phase 2, we monitor our estimation during the video streaming. Line 3 monitors the channel conditions using the monitoring wireless card, and then computes the statistics of the throughput, air-time ratio, and retransmission ratio every T seconds. Line 4 checks whether the retransmission ratio is higher than a threshold r_r^H, which is a system parameter. If it is the case, line 5 reduces the estimated bandwidth b with step size δ, and reconfigures the encoding bitrate. Line 6 checks whether the increasing rate of the air-time ratio is higher or equal to a historical proportion value τ_p, which is updated once the air-time ratio increases proportionally to the throughput. If the current increasing rate is higher, line 7 increases the estimated bandwidth b with step size δ and reconfigures the encoding bitrate. δ is a system parameter. Line 8 updates τ_p, and lines 9–10 store the current air-time ratio and throughput for the next iteration. It is clear that the proposed algorithm terminates in $O(1)$ time.

VI. Evaluations

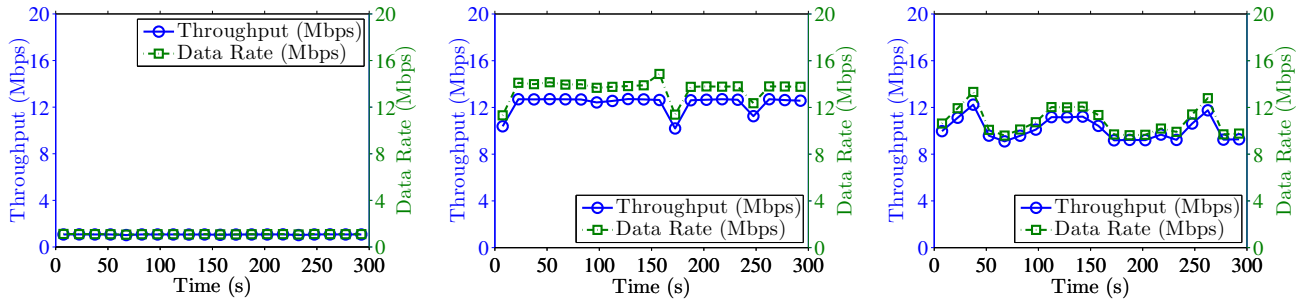
A. Evaluation setup

We have implemented a prototype system using C++ for evaluation. The testbed is generally the same as the testbed in Sec IV. The AP is built on a Linux box with two wireless cards. One of them is configured to be an AP and the other one is set to monitoring mode for capturing the frames transmitted through the specific channel. There are two laptops running iPerf [3] sender and receiver connected to the AP. The AP monitors, collects, analyzes the frames on the channel, estimates the available bandwidth using our proposed algorithm, and then sends the results to the iPerf sender for adapting the streaming bitrate. In our experiments, we compare our proposed solution against two baselines that stream with constant bitrates at: (i) 1 Mbps and (ii) 13 Mbps. We refer to them as Low-Bitrate (LB) and High-Bitrate (HB) in the figures, respectively. We conduct each streaming session 5 times and each lasts for 5 minutes. We look into the results including the target bitrate, air-time ratio, data rate, retransmission ratio, throughput, packet loss rate, and video



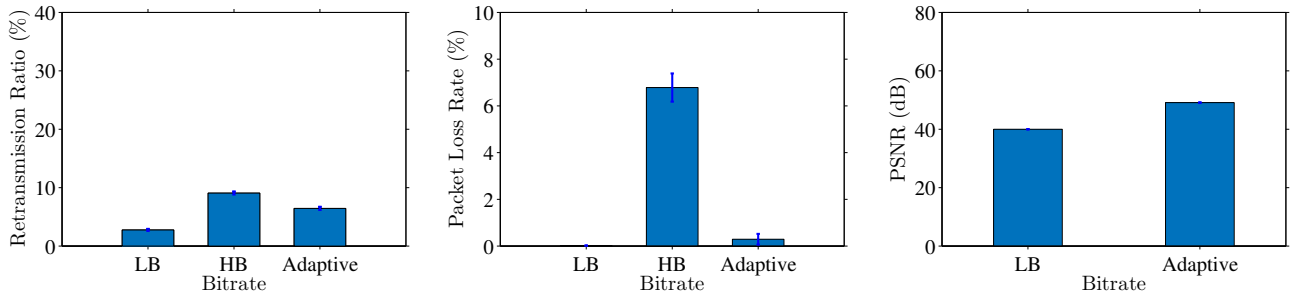
(a) (b) (c)

Fig. 5. Sample results of our proposed algorithm, target bitrate with: (a) air-time ratio, (b) retransmission ratio, and (c) packet loss rate.



(a) (b) (c)

Fig. 6. The data rate and throughput of: (a) LB, (b) HB, and (c) our proposed adaptive bitrate streaming.



(a) (b) (c)

Fig. 7. Our proposed algorithm outperforms others in terms of (a) retransmission ratio, (b) packet loss rate, and (c) video quality in PSNR.

quality. The packet loss rate is the fraction of lost packets and the video quality is the PSNR (Peak Signal to Noise Ratio) value of a talk show video (similar to the talking head scenes in video conferences) modeled by an empirical function of the bitrate using x265 with ultrafast preset, zerolatency tuning, and IPPP$\cdots$ structure.

B. Results

Effectiveness of our proposed solution. We configured the monitored period T as 15 seconds, the retransmission threshold r_r^H as 8, and the step size δ as 0.25 if not otherwise specified. Fig. 5 shows the sample results of our proposed solution. We first plot the target bitrate over time with the measured air-time ratio in Fig. 5(a). Our algorithm increases the target bitrate as the air-time ratio increases proportionally to the throughput and halts when the increasing rate of the air-time ratio diminishes. For example, the target bitrate increases from $t = 70$ s, $t = 210$ s and stops at $t = 100$ s, $t = 235$ s due to the saturated increasing trend on air-time ratio. Fig. 5(b) plots the target bitrate with the retransmission ratio over time. Our algorithm reduces the target bitrate when the retransmission ratio is high, for example, at about $t = 40$ s, $t = 165$ s, $t = 265$ s. High retransmission ratio indicates that there may be high fraction of packets are lost. Thus, adapting the bitrate considering the retransmission ratio prevents the application from suffering high packet loss rate as shown in Fig. 5(c). The packet loss rate are almost about 0% in the whole session.

Our proposed solution achieves high throughput without wasting the network resource. Fig. 6 plots the throughput and the data rate of a sample result from each solution. Fig 6(a) shows that there is extremely low retransmission ratio in LB. However, the throughput as expected is only 1 Mbps, which leads to lower video quality. Fig. 6(b) shows that there is about 1.3 Mbps bandwidth are used to retransmitting frames in HB. The high retransmission bitrate not only occupies the network resources but indicates the occurrence of the high packet loss rate, which leads to inferior video quality. Fig. 6(c) plots the throughput and data rate of our proposed solution. This figure shows that our proposed solution adapts to the network condition to fully utilize the network resources for streaming instead of retransmission. Thus, our proposed solution is suitable for video streaming application, since it achieves higher throughput with fewer retransmissions.

Our proposed solution leads to higher video quality and lower packet loss rate. We plot the average packet loss rate and video quality with 95% confidence intervals over all experiments in Fig. 7. Fig. 7(a) shows that the retransmission ratio of our algorithm is lower than that of HB by 3% on average. Fig. 7(b) shows that the average packet loss rate of our proposed solution is lower than 0.5%. However, HB has higher than 6% packet loss rate on average, which leads to inferior video quality. Thus, we only plot the video quality of LB and our proposed solution in Fig. 7(c). This figure shows that the video quality of our proposed solution is about 50 dB in PSNR, which is higher than LB by about 10 dB in average. In summary, our proposed solution can achieve higher video quality, and low packet loss rate in video streaming.

VII. Conclusion

In this paper, we conducted extensive experiments to monitor, model, and analyze the unlicensed spectrum activity. Based on the observations, we deign, implement, and evaluate an interference-aware bandwidth estimation and rate adaptation algorithm. Our algorithm leverages monitoring nodes to sniff wireless frames that are sent to or received from others to understand the impacts of interference. In particular, our algorithm keeps track of the air-time ratio, throughput, and retransmission ratio during video streaming to estimate the available bandwidth and perform the rate adaptation. The experiment results show that our proposed solution: (i) reduces the packet loss rate to lower than 0.5%, (ii) has averaged higher video quality than LB by 10 dB in PSNR, and (iii) has lower retransmission ratio than HB by 3%.

The presented work can be extended in several directions. First, we plan to implement the rate adaptation in a real video streaming system, GamingAnywhere [14], and look into the impacts from multiple video streaming. Second, we plan to connect APs to the controller for collecting the statistics to further improve the network performance. Last, we plan to adopt interference mitigation approaches to cope with the wireless interference.

References

[1] hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator. https://w1.fi/hostapd/.

[2] Intel widi. http://www.intel.com/go/widi.

[3] iPerf - The network bandwidth measurement tool. https://iperf.fr/.

[4] TCPDUMP&LIBPCAP. http://www.tcpdump.org/.

[5] Wireless gigabit alliance. http://wirelessgigabitalliance.org.

[6] Wireless home digital interface. http://www.whdi.org.

[7] Wirelesshd. http://www.wirelesshd.org.

[8] The Aragon research globe for web and video conferencing, 2015: Focus on platforms and outcomes, December 2015. http://www.cisco.com/c/dam/en/us/solutions/collaboration/analysts/downloads/aragon-research-web-video.pdf.

[9] Wireless gigabit market–global forecasts to 2022, May 2016. http://www.marketsandmarkets.com/Market-Reports/wireless-gigabit-wigig-market-812.html.

[10] M. Demircin and P. Beek. Bandwidth estimation and robust video streaming over 802.11e wireless lans. In *Proc.of IEEE International Conference on Multimedia and Expo (ICME'05)*, pages 1250–1253, Amsterdam, Netherlands, 2005.

[11] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. *ACM SIGCOMM Computer Communication Review*, 37(4):385–396, 2007.

[12] Y. Guo, C. Liang, and Y. Liu. AQCS: adaptive queue-based chunk scheduling for P2P live streaming. In *Proc. of International IFIP-TC6 Networking Conference*, pages 433–444, Singapore, 2008.

[13] H. Hong, C. Hsu, T. Tsai, C. Huang, K. Chen, and C. Hsu. Enabling adaptive cloud gaming in an open-source cloud gaming platform. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(12):2078–2091, 2015.

[14] C. Huang, K. Chen, D. Chen, H. Hsu, and C. Hsu. GamingAnywhere: The first open source cloud gaming system. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 10(1s):10:1–10:25, January 2014.

[15] A. Jindal, K. Psounis, and M. Liu. Capest: A measurement-based approach to estimating link capacity in wireless networks. *IEEE Transactions on Mobile Computing*, 11(12):2098–2108, November 2012.

[16] A. Kashyap, S. Ganguly, and S. Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proc. of ACM International Conference on Mobile Computing and Networking (MobiCom'07)*, pages 242–253, Montreal, QC, 2007.

[17] M. Li, M. Claypool, and R. Kinicki. WBest: A bandwidth estimation tool for ieee 802.11 wireless networks. In *Proc. of IEEE Conference on Local Computer Networks (LCN'08)*, pages 374–381, Montreal, QC, 2008.

[18] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran. Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE Journal on Selected Areas in Communications*, 32(4):719–733, 2014.

[19] R. Maheshwari, S. Jain, and S. Das. A measurement study of interference modeling and scheduling in low-power wireless networks. In *Proc. of ACM Conference on Embedded Network Sensor Systems (SenSys'08)*, pages 141–154, Raleigh, NC, 2008.

[20] Z. Qiao, L. Sun, N. Heilemann, and E. Ifeachor. A new method for voip quality of service control use combined adaptive sender rate and priority marking. In *Proc. of IEEE International Conference on Communications*, volume 3, pages 1473–1477, Paris, France, 2004.

[21] R. Riggioa, T. Rasheeda, S. Testib, F. Granellib, and I. Chlamtaca. Interference and traffic aware channel assignment in wifi-based wireless mesh networks. *Ad Hoc Networks*, 9(5):864–875, 2011.

[22] C. Rossi, C. Casetti, and C. Chiasserini. A passive solution for interference estimation in wifi networks. In *Proc. of International Conference on Ad-hoc, Mobile, and Wireless Networks (ADHOC-NOW'14)*, pages 156–168, Benidorm, Spain, 2014.

[23] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. Mojo: A distributed physical layer anomaly detection system for 802.11 wlans. In *Proc. of International Conference on Mobile Systems, Applications and Services*, pages 191–204, Uppsala, Sweden, 2006.

[24] A. Sheth, S. Nedevschi, R. Patra, S. Surana, E. Brewer, and L. Subramanian. Packet loss characterization in wifi-based long distance networks. In *Proc. of IEEE INFOCOM 2007*, pages 312–320, Anchorage, AK, 2007.

[25] V. Shrivastava, S. Rayanchu, S. Banerjee, and K. Papagiannaki. Pie in the sky: Online passive interference estimation for enterprise wlans. In *Proc. of the 8th USENIX conference on Networked systems design and implementation (NSDI'11)*, pages 337–350, Boston, MA, 2011.

[26] D. Zhao, M. Zhu, and M. Xu. Leveraging sdn and openflow to mitigate interference in enterprise wlan. *Journal of Networks*, 9(6):1526–1533, 2014.