Cross-layer Energy Optimization for Dynamic Video Streaming over Wi-Fi

Zhi Zhang*, Mehmet Karaca*, Farnaz Moradi*, Saeed Bastani*, Anders Nilsson Plymoth†, Rickard Ljung‡, Björn Landfeldt*
* Department of Electrical and Information Technologies, Lund University, Lund, Sweden.
Email: {zhi.zhang, mehmet.karaca, farnaz.moradi, saaed.bastani, bjorn.landfeldt}@eit.lth.se
†TelHoc AB, Stockholm, Sweden. Email: anders@telhoc.se
‡Sony Mobile, Lund, Sweden. Email: rickard.ljung@sonymobile.com

***Abstract*—Dynamic Adaptive Streaming over HTTP (DASH) constitutes a large fraction of traffic in the mobile Internet. Mobile devices often use video delivery over Wi-Fi, which is a significant energy drain. Dynamic Power Saving Mode (DPSM) is one of the most widely-used approaches for Wi-Fi devices to save power and shorten packet delay. However, DPSM uses a fixed timeout before a device goes to sleep, leading to excessive energy use in many cases. In this paper, we propose a cross-layer lightweight dynamic timeout adjustment algorithm. The application layer measures the Round-Trip Time (RTT) of video segments in the current timeout setup, determines an updated timeout, sets it to the medium access control layer and requests the next video quality based on the RTT and RTT change. We evaluate our algorithm via simulations in OMNeT++ and show that compared with the legacy Wi-Fi DPSM for DASH, our algorithm reduces the average power consumption in the radio front end while significantly improving the fairness among users. Moreover, the new algorithm retains video quality while achieving better energy efficiency. The improvement increases in significance with increasing number of users, which means that the new algorithm works well in dense scenarios.**

***Keywords*—cross-layer, energy optimization, round-trip time, dynamic adaptive streaming via HTTP, dynamic power saving mode**

I. Introduction

With the advance of mobile and smart device technology, the market is witnessing a tremendous penetration of consumer devices capable of supporting various video streaming applications. This is evidenced by the recent forecasts indicating that the global mobile traffic will reach 30.6 exabytes per month by 2020, and video traffic accounts for 75 percent of this traffic volume [1]. At the same time, the evolution towards faster cellular connectivity, e.g. 4G connectivity is forecast to experience an annual growth of 34 percent [1]. However, 55 percent of mobile traffic, including video, is predicted to be offloaded to Wi-Fi and small cell networks [1]. Moreover, the projected share of Wi-Fi for handling offload traffic is expected to be predominant since most mobile devices are already enabled with dual-mode cellular-Wi-Fi operations, whereas small cell technology is in its infancy.

The traffic offloading trend combined with the traffic produced by Wi-Fi-only devices bring Wi-Fi technology in the spotlight for further performance enhancement so as to overcome the challenges posed by the increasing traffic volume in general and delay sensitive video traffic in particular. Thus far, efforts have predominantly been focused on Wi-Fi data rate improvement, leading to the recently completed multi-gigabit IEEE 802.11ac (WiGig) standard [2]. While this is an important achievement, power consumption issues, particularly in face of high volume video traffic with stringent quality of service constraints, has not attracted much attention despite the increasing share of video traffic and the fact that a significant portion of such traffic is offloaded by battery-operated mobile devices. Among few efforts, the IEEE 802.11 standard proposed a static Power Saving Mode (PSM), implemented as a cooperative mechanism between wireless station (STA) and an access point (AP). In this mechanism, as soon as the last packet buffered in the AP is received by the STA, it immediately goes to sleep mode until the next beacon interval. Static PSM is efficient only when the traffic inter-arrival distribution is regular [3], whereas video traffic does not always exhibit this property. This potentially leads to an increased buffering delay in the AP, which in turn results in large Round-Trip Times (RTT) for packets received at the application layer. The drawbacks of the static PSM have led commercial Wi-Fi device manufacturers to implement Dynamic PSM (DPSM) which allows the wireless interface to stay active for an additional fixed amount of time, termed *timeout*, after the reception of traffic from the AP [4]. With DPSM, the experienced RTT is improved but the energy saving is reduced compared to the static PSM [4]. However, neither PSM nor DPSM takes into account the specific characteristics and requirements of video streaming applications including the application layer parameters and the underlying streaming protocol.

In this paper, we propose a novel energy saving mechanism for Wi-Fi. The new scheme extends DPSM with a dynamic timeout period. More specifically, we propose a cross-layer algorithm which takes into account the measured RTT of video segments received at the receiver Application (APP) layer, and dynamically adjusts the next timeout period in the receiver medium access control (MAC) layer. We apply this new protocol in conjunction with Dynamic Adaptive Streaming over HTTP (DASH) to achieve increased energy efficiency. Our focus on DASH is justified by its strong technical features and its adoption in real world video streaming applications. DASH does not exhibit weaknesses of traditional statefull protocols such as Real-time Transport Protocol (RTP), Real Time Streaming Protocol (RTSP) and progressive download. Additionally, it is adaptive to changes in available link bitrates [5], [6], a feature that makes DASH suitable for wireless access

networks. Our proposed scheme departs from the existing solutions such as [7]–[9] which solely rely on MAC mechanisms and do not take into account the application layer parameters. It also differs from the cross-layer solutions such as [10] which either brings high computational complexity or require radical modifications to the standard PSM.

The remainder of this paper is organised as follows: Section II presents a literature survey. Section III describes the research problem in more details, followed by our proposed cross-layer algorithm in Section IV. Simulation results are presented in Section V, and the paper is concluded in Section VI.

II. Related Works

Energy optimization and management of multimedia delivery over Wi-Fi is an established research area. Existing research has either focused on the study of the timeout value in DPSM and its impact on network-related factors, the introduction of variations to the standard IEEE 802.11 PSM or entirely new approaches [7]–[15]. In [7], a practical PSM scheme, called PSM-AW for varying server delay is proposed. PSM-AW is a client-side solution that is designed to mitigate the long delays of the IEEE 802.11 PSM. This algorithm considers previous data exchanges to predict the next packet arrival time. It then calculates the next wake up time accordingly. It compares the energy saved by sleeping to energy cost of waking up and decides if it is better to sleep or stay awake. This algorithm also sets a threshold waiting time for the next packet, which means if the next packet does not arrive within the threshold time, the client will go to sleep mode. In [8], an energy-efficient sleep scheduling protocol is presented. In this approach, if there are several users, the AP coordinates the sleep time of each node to prevent overlapping active times. However, devices should send a sleep request to the AP before going into sleep mode after which a wake up time and duration will be determined. In [9], an Adaptive PSM (APSM), a traffic aware PSM, is proposed in which the AP grants different priorities to devices based on traffic load and channel conditions.

In [10] a cross-layer multimedia delivery system is proposed that turns input data into bursts and delivers bursts to the client. It uses a traffic shaper and traffic profiler to calculate maximum burst cycle and burst duration. The shaper can also determine the burst rate. In [11], a mobility aware PSM uses long term buffering to save energy. It is also adjusts Beacon Monitoring Interval (BMI) adaptively, which helps to reduce wakeup overheads. However it does not consider RTT increase due to immediate sleep and sleep/wake up delays and further requires some additional fields in the beacon frame, which is not desirable. [12] uses a proxy server behind the AP. This will allow users to have shorter PSM timeout and sleep earlier while the proxy is getting data from a remote server, a potentially lengthy process. In [13], an energy-efficient Predictive Green Streaming (PGS) optimization framework is proposed. The proposed framework minimizes the airtime of the transmission without causing interruptions to streaming, minimizes total power consumption, and allows a tradeoff between adaptive streaming (AS) quality and energy consumption. The main drawback of this scheme is the high computational complexity of the algorithms used. In [14], a dynamic Listen Interval (LI) scheme is proposed. In this method, if the device experiences an unnecessary wake up, i.e., the device wakes up to listen to beacons but there is no data buffered for this device, the LI is incremented by 1. On the other hand, if the device wakes up and realizes the presence of buffered data in the AP, then LI is set to 1, forcing it to wake up for every beacon. This is a simple scheme and easy to implement, but it does not consider energy consumed during the timeout period within every LI, which is deemed to have a significant impact on the overall energy consumption. [15] proposes an algorithm to dynamically determine devices' sleep and active times.

All aforementioned algorithms require complicated calculation and processing, and some of them require major modifications of the standard or significant computational overhead on both AP and client. In this paper, we propose a cross-layer lightweight algorithm that requires only minor modification to dynamically determine PSM timeout period.

III. Background and Research Problem

In this section, we first mention the DASH implementation to determine the video quality. Then, we give a detailed explanation of DPSM in Wi-Fi systems.

A. DASH

One of the major problems with DASH is to determine the video quality in such a way that the best Quality of Experience (QoE) can be achieved. In current implementations, the quality level is based on the measured network throughput. Let $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$ be the set of available quality levels, where l_c, l_1 and l_L represent the current, the lowest and highest video qualities, respectively. B_l denotes the bit rate of the video segment with quality l. Let $avgRTT$ be the average RTT value averaged over the last K packets, and $TotBit$ is the total amount of bits that are carried with the last K packets. Then, the average measured throughput $avgThr$ is given as:

$$avgThr = \frac{Totbit}{avgRTT} \tag{1}$$

After determining the average throughput, if $c \neq L$, then the client checks if the following condition holds:

$$avgThr \geq B_{l_{c+1}}. \tag{2}$$

If the condition in (2) is satisfied, then the client increases the current video quality as follows:

$$l_c = min(l_{c+1}, l_L) \tag{3}$$

The client decreases the video quality to a lower level, i.e., $l_c = max(l_{c-1}, l_1)$, if the following condition occurs:

$$avgThr < B_{l_c}. \tag{4}$$

Otherwise, the client keeps the same quality level.

B. DPSM

We recall that the most power hungry part of a Wi-Fi device is its radio front end, and DPSM is one of the most common practical approaches to save energy. In DPSM, each Wi-Fi device wakes up every Delivery Traffic Indication Map (DTIM) interval, after receiving a beacon frame and checks the Traffic Indication Map (TIM) field in the beacon packet. If

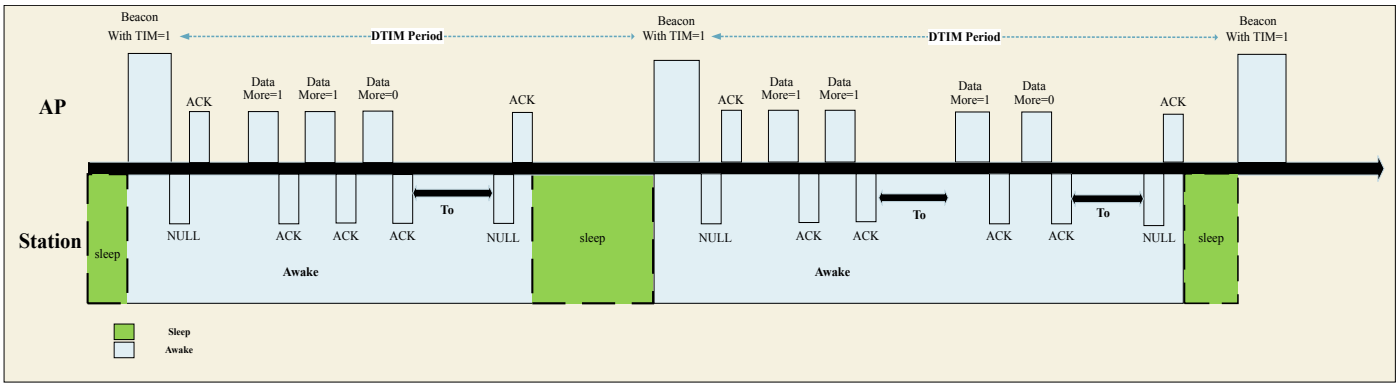

Fig. 1: DPSM with the latest packet

TIM is set to 0 then there is no data, and the user goes to sleep until the next listen interval. If TIM is set to 1 there is data for the device buffered at the AP. Then, the device switches to Continuously Aware Mode (CAM) to receive all the data in the buffer. During the transmission of the last packet in the buffer, the AP notifies the device that this is the last packet by setting More Data field in the beacon frame to 0. Unlike static PSM, in DPSM when a device receives its last packet it further stays in CAM for a predefined duration called PSM timeout instead of entering PSM immediately. By doing that, the device is able to receive any data that may arrive before the transmission of the last packet. Doing so, RTT decreases, and consequently the quality of the transmission will increase. Fig. 1 depicts a typical scenario when DPSM is enabled. During the first DTIM period, the device is informed that it has data, and it sends a NULL data frame to the AP to receive it. Then, unlike static PSM where the device has to send a PS-POLL frame for every packet, the device starts receiving all the buffered data without needing to send more NULL or PS-Poll frames.

As shown in Fig. 1, after receiving the last packet in the first DTIM period, the device remains in the awake mode for an additional PSM timeout instead of going to sleep mode immediately. We denote the timeout duration as T_o. The main motivation behind keeping the device in awake mode for the timeout duration is that due to RTT effects there may be newly arrived data right before the reception of the last packet. Hence, to prevent interruption to the communication and reduce delay, devices stay in awake during this timeout in practice [12]. For instance, in the example in Fig. 1, there is new data arriving before the last packet reception in the second DTIM period, the device stops the timeout timer, and receives the new data. Then, it starts another timeout timer, and since there is no new data it sends a Null Data Frame with the Power Management field set to 0, which indicates that the device goes to PSM.

The determination of the PSM timeout is implementation specific. For instance, the default timeout duration is set to 200 ms with Nokia N900 and Android Nexus One [12]. The reason for choosing such high timeout value is to reduce the RTT effect. However, in some scenarios RTT can be much shorter than the default value, which causes the device to spend unnecessary energy during timeout. In this work, we reduce the energy usage during timeout by dynamically adjusting the timeout value taking into account the RTT variation instead of fixing the value.

IV. Cross-layer Algorithm

In this section, we present the design of the cross-layer, on-line algorithm that uses the timeout value from the MAC layer and RTT from the APP layer for the current video quality to determine the appropriate timeout and video quality for the next request. First, we study how the timeout value affects the performance in terms of RTT, power consumption, throughput and efficiency. Second, we present the new cross-layer dynamic timeout algorithm for DASH.

A. Study on Fixed Timeout for Fixed Video Quality

In this subsection, we investigate the system performance via simulations in OMNeT++ [16]. For each simulation, a fixed timeout is used for a fixed video quality. Both single-user and multi-user cases are evaluated. In all simulations we use IEEE 802.11n with DPSM and a bitrate of 150 Mbps. MAC Protocol Data Unit (MPDU) is used for frame aggregation. The power consumption values are taken from the IEEE 802.11ax Task group (TGax) simulation scenarios. For transmission, reception, listen and sleep, these values are 280, 100, 50 and 0.9 mA, respectively, with a voltage of 1.1 V [17] for all users. Four levels of video qualities are considered, 480i, 576i, 720p and 1080i with a bit rate of 1.5, 2.0, 4.6 and 5.4 Mbit/s [18], respectively. Each user requests a 2-minute video for playback. A video segment duration is 2 seconds. If a user does not receive a video segment within 3 seconds, it will again send a request. RTT is moving averaged with a window length of 5 samples. We set the timeout with a step of 1 ms when the timeout is less than 10 ms and a step of 10 ms when the timeout is from 10 ms to 100 ms. The confidence level of the simulations are high, i.e., the variance for the simulations with different seeds is small, and therefore, the variance is not plotted in the figures.

Fig. 2 depicts the average RTT for different video qualities. For a single user, as expected, RTT first decreases significantly when the timeout increases from 0 ms to 5 ms, and becomes almost flat when the timeout further increases. The reason is that when the timeout increases from a zero value, new arrival packets can be sent to the user during the extra timeout period and avoid being buffered at AP when the user goes to sleep mode. The capacity for processing new arrival packets grows until the timeout increases to a certain value, and after that there are no more packets arriving due to the bursty nature

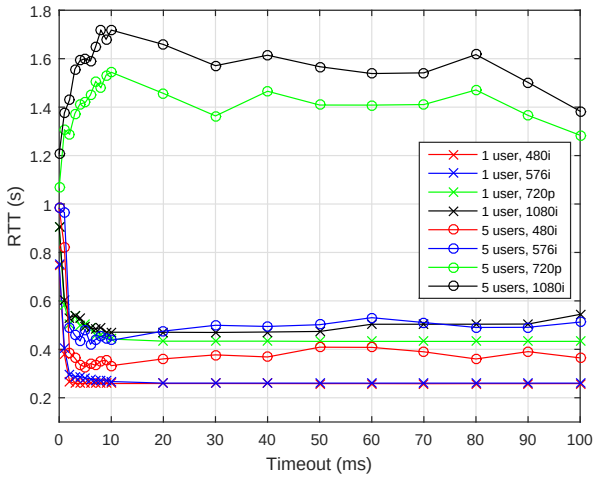

Fig. 2: Average RTT for Different Video Qualities

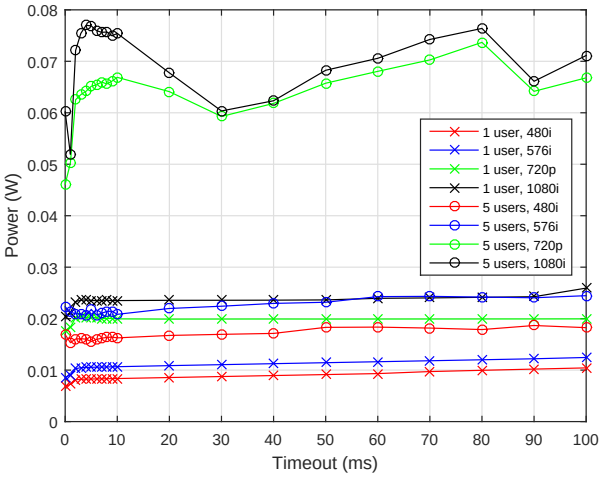

Fig. 3: Average Power Consumption for Different Video Qualities

of TCP traffic for one video request. The RTT is bigger for higher video quality because higher quality brings larger video segment sizes. For multiple users, the contention between users generally increases RTTs compared with those for the single-user case. Moreover, if a user increases timeout, there are two contradictory effects: it prompts to reduce its own RTT but in the meanwhile also increases the waiting time for other users. The two effects make the case complicated and cause fluctuation of the curves for multiple users. For low qualities, i.e., 480i and 576i, the burst packet arrival rates are low and therefore the first effect dominates so the increase on the timeout directly leads to a drop in RTT and also fluctuation. However, for high qualities, i.e., 720p and 1080i, when the timeout increases, many packets arrive during the timeout and a user has to cancel its timeout and issues a new one, which leads to a significant delay of the transmission for other users. The second effect dominates when the timeout increases from zero to a certain value, after which the first effect dominates.

Fig. 3 and Fig. 4 show the average power consumption and throughput per user, respectively. The power consumption is

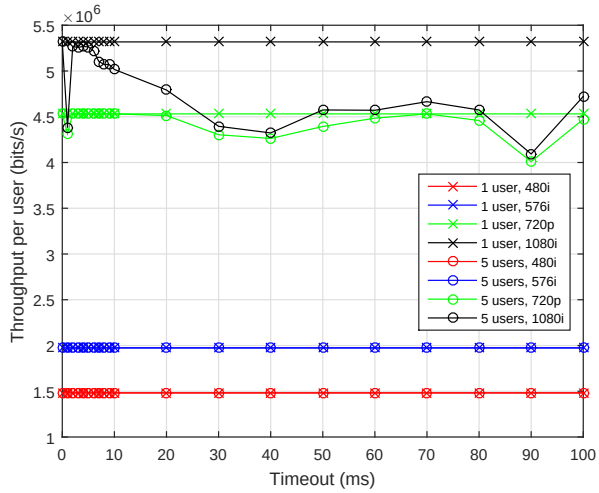

Fig. 4: Average Throughput per User for Different Video Qualities

defined as the average RF power during the whole simulation time, considering the states of transmission, reception, listen and sleep. The throughput is defined as the number of video bits successfully received on the APP layer over the whole simulation time and is flat if the required throughput can be fulfilled. For a single user, the power consumption increases fast and then grows slightly when the timeout increases. The reason is that there are more packets arriving when the timeout increases from zero, so the power consumption increases. After a certain timeout value, there are no more packets and the increase on power consumption is due to longer idle listening. The network can service the requested throughput and video playback is smooth for the single-user case. However, for multiple users, the power consumption varies due to the fluctuation of arrived packets caused by the RTT fluctuation. For high video qualities, i.e., 720p and 1080i, the power consumption varies significantly caused by the change of throughput, and the network cannot meet the throughput requirement at times, which leads to a playback halt for some users. It is worth mentioning that the video segment is 2 seconds long and average RTT is normally smaller than 2 seconds as shown in Fig. 2. Therefore, if DASH is enabled, when the estimated throughput of a certain user is good enough for requesting a higher video quality, such a request might lead to an increased delay for the requests from other users, which may cause playout-buffer underrun. This fact motivates us to improve the DASH algorithm by taking into account of the RTT. We will explain it in the next subsection.

Fig. 5 depicts the average energy efficiency. The efficiency is defined as received video bits over energy use. The efficiency tends to decrease with increasing timeout. For a single user, a higher video quality achieves better efficiency, whereas for multiple users, low qualities (480i and 576i) are more efficient than high qualities (720p and 1080i). Therefore, in a dense scenario, it is more efficient to request relatively low qualities.

In summary, there are two main guides to design the cross-layer algorithm. First, it is better to decrease the timeout to save energy and maintain the video quality until RTT has a significant change. Second, for a given timeout, it is necessary to also check the absolute value of RTT to roughly estimate if

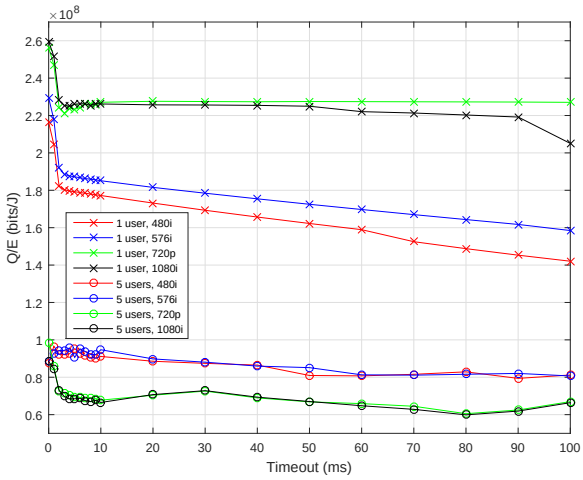

Fig. 5: Average Efficiency for Different Video Qualities

other users have ongoing traffics. If so, it is wise to request a lower quality to avoid affecting other users and thus achieve better efficiency, even though the estimated throughput might be higher than the current one. To do an accurate check, it is necessary to avoid timeout values close to zero, because RTT curves for different qualities and different numbers of users overlap and are hard to distinguish in this timeout range. In the next subsection, we will detail our new algorithm.

B. Dynamic Timeout for DASH

In this section, we present our cross-layer, on-line algorithm, that determines the timeout dynamically without causing the RTT to increase. The key idea is to reduce the timeout as long as the measured RTT does not change significantly. If the RTT increases intolerably then we increase the timeout to compensate the effect of the increased RTT. The procedure is explained in detail in Algorithm 1.

The DASH procedure described in Section III-A may not be efficient since it may not be feasible to accurately measure network throughput due to high variations in the network. Hence, we modify the procedure to take into account not only the throughput but also the measured RTT values, and propose a RTT based DASH (R-DASH) as follows: Let RTT_{thr} be the minimum RTT threshold. If the RTT is greater than this threshold, i.e., $avgRTT \geq RTT_{thr}$, our DASH implementation selects a lower quality video even if the measured throughput can support higher quality. The determination of RTT_{thr} is based on a heuristic technique, historical RTT values and quality level can be utilized to determine this threshold.

Our dynamic timeout algorithm is a cross-layer algorithm in the sense that the optimization of timeout is performed by exchanging information between the APP and MAC layers. Specifically as shown in Fig. 6, at the application layer the average RTT is measured and by applying Algorithm 1 a new timeout is determined. Then, this information is passed to the MAC layer, where the new timeout becomes effective. We note that our algorithm is fully distributed and each device can determine its timeout values without requiring other network information.

Algorithm 1: Dynamic timeout

Given δ_1, δ_2, ϵ, T_o^l, T_o^u. For every new video content, do:

Step 0: Set the initial timeout to its maximum value,

$$to(t) = T_o^u$$

Step 1: Measure RTT over last K packets, denote it by $avgRTT$.

Step 2: Decrease timeout as follow;

$$to^{'}(t) = to(t)/\delta_1$$

If $to^{'}(t) < T_o^l$, then $to(t) = T_o^l$. Otherwise, $to(t) = to^{'}$.

Step 3: Measure RTT over K packets with new timeout, denote it by $newavgRTT$.

Step 4.1: Use $newavgRTT$ to determine video quality using R-DASH.

Step 4.2: **if**

$$|newavgRTT - avgRTT| < \epsilon$$

then set $avgRTT \leftarrow newavgRTT$, and go Step 2 to further decrease the timeout.

else

$$to^{''}(t) = to(t) + \delta_2$$

If $to^{''}(t) > T_o^u$, then $to(t) = T_o^u$. Otherwise, $to(t) = to^{''}(t)$. Set $avgRTT \leftarrow newavgRTT$, Go Step 3.

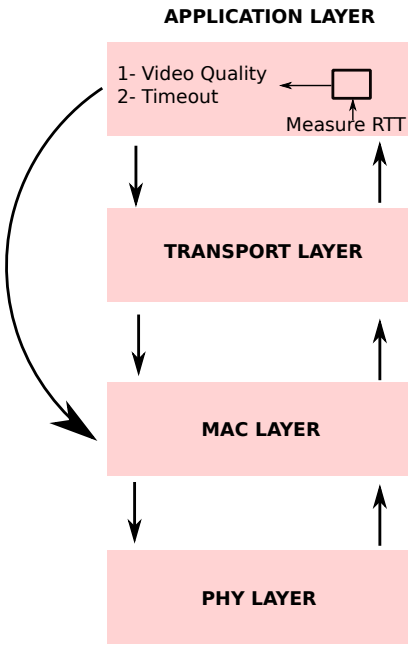

Fig. 6: DPSM with the latest packet

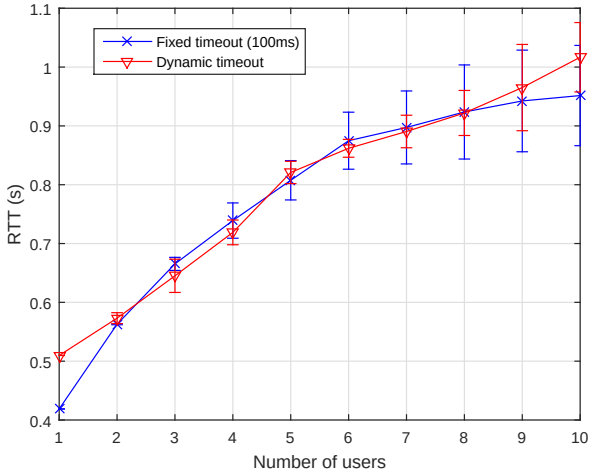

Fig. 7: Average RTT for DASH

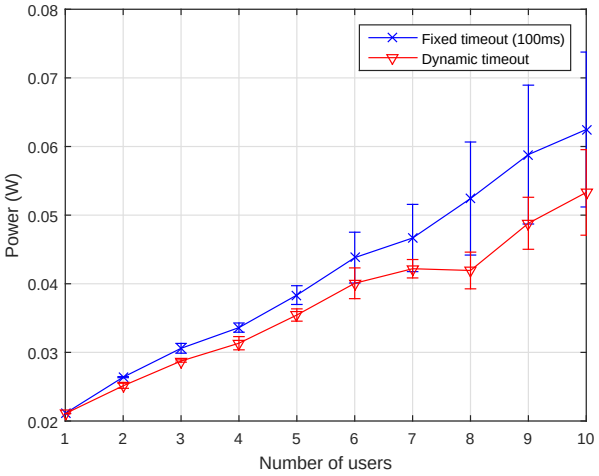

Fig. 8: Average Power Consumption for DASH

V. Performance Evaluation

Via simulations we compare the performance and fairness of our proposed algorithm with the legacy DPSM which uses a fixed timeout of 100 ms. The fairness is denoted by the variance among a given set of users. The simulation setup is the same as the one in Section IV-A with DASH enabled. The values for RTT change (ϵ), threshold (RTT_{thr}), lower bound (T_o^l), upper bound (T_o^u), incremental step (δ_2) and decremental radio (δ_1) are 0.3 s, 0.8 s, 3 ms, 100 ms, 10 ms and 3.0, respectively.

Fig. 7 compares RTT between the new algorithm and legacy DPSM. Although timeout is shortened in the new algorithm, RTT remains similar with a smaller variance. Fig. 8 compares power consumption between the new algorithm and legacy DPSM. For a single user, the power consumption is similar for the two algorithms, whereas the new algorithm reduces the power consumption by 8% for 5 users and by 15% for 10 users. The improvement tends to be more significant when the number of users further increases. Moreover, the new algorithm provides better fairness among the users, especially for dense scenarios, e.g., the power variance of the new algorithm for 10 users is only 55% of the legacy power variance. The new algorithm hence mitigates the case that some users run out of battery faster than the others.

The new algorithm achieves a throughput similar to the legacy DPSM as shown in Fig. 9, which means that the video quality does not worsen when the power consumption is reduced. Moreover, the users are served in a better way in terms of fairness. Fig. 10 shows that the new algorithm also achieves better efficiency than the legacy DPSM.

In summary, the new cross-layer algorithm reduces the power consumption significantly, especially for dense scenarios, and at the same time, retains the throughput, while achieving better efficiency and fairness.

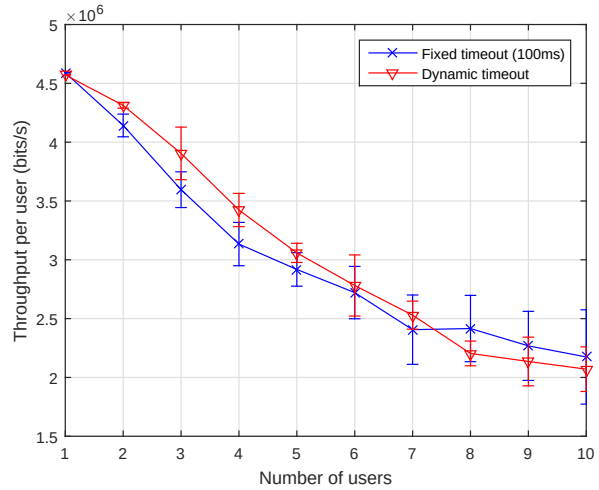

Fig. 9: Average Throughput per User for DASH

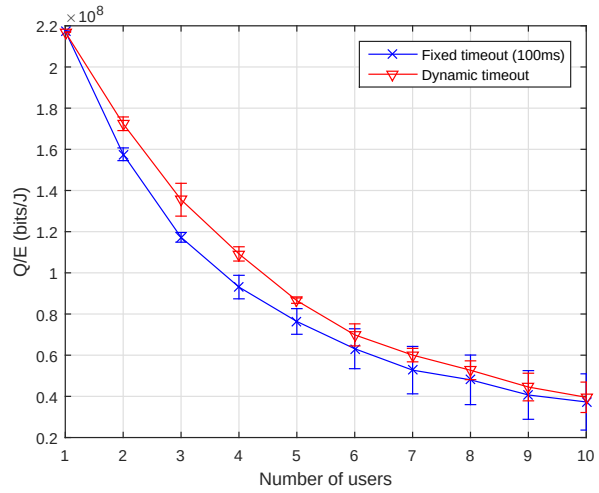

Fig. 10: Average Efficiency for DASH

VI. Conclusion

In this paper, we have proposed a cross-layer algorithm for Wi-Fi DPSM with DASH applications. In the new algorithm,

the APP layer uses the RTT of video segments and DPSM timeout value from the MAC layer to determine the video quality for the next request and adapt the timeout value dynamically. The new algorithm has low complexity and operates in a fully distributed way. We conducted simulations for different numbers of users. The new algorithm outperforms the legacy DPSM, especially in dense scenarios with multiple users. For example, it reduces the power consumption by 15% for 10 users and improves the fairness by 45% compared with the legacy DPSM, and in the meanwhile, retains the video quality with better efficiency. In the future, we will carry out a theoretical analysis and formulate the system, optimize the parameters, and then implement our algorithm in a real testbed. We will also study the buffering strategy, both on the user and AP side, to further reduce power consumption and improve the efficiency.

ACKNOWLEDGMENT

This work is supported by the European Celtic-Plus project CONVINcE and was partially funded by Finland, France, Sweden and Turkey. It is also partially funded by the EC FP7 Marie Curie IAPP Project 324515, MeshWise.

REFERENCES

[1] Cisco visual networking index: Global mobile data traffic forecast update, 20152020. Webpage, accessed February 2016. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf

[2] "IEEE P802.11ac. Specification Framework for TGac. IEEE 802.11-09/0992r21." *IEEE Std 802.11ac-2013*, Jan 2013.

[3] S. Chandra and A. Vahdat, "Application-specific Network Management for Energy-Aware Streaming of Popular Multimedia Formats," in *Proceedings of the General Track of the Annual Conference on USENIX Annual Technical Conference*, ser. ATEC '02. USENIX Association, 2002, pp. 329–342.

[4] E. Tan, L. Guo, S. Chen, and X. Zhang, "PSM-throttling: Minimizing Energy Consumption for Bulk Data Communications in WLANs," in *IEEE International Conference on Network Protocols (ICNP)*, 2007, pp. 123–132.

[5] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP: Standards and Design Principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11, 2011, pp. 133–144.

[6] T. Lohmar, T. Einarsson, P. Frojdh, F. Gabin, and M. Kampmann, "Dynamic adaptive HTTP streaming of live content," in *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2011, pp. 1–8.

[7] B. Peck and D. Qiao, "A practical psm scheme for varying server delay," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 303–314, 2015.

[8] L. Liu, X. Cao, Y. Cheng, and Z. Niu, "Energy-efficient sleep scheduling for delay-constrained applications over wlans," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2048–2058, 2014.

[9] Y. Xie, X. Sun, X. Chen, and Z. Jing, "An adaptive psm mechanism in wlan based on traffic awareness," in *10th IEEE International Conference on Networking Sensing and Control (ICNSC)*, 2013, pp. 568–573.

[10] M. A. Hoque, M. Siekkinen, J. K. Nurminen, S. Tarkoma, and M. Aalto, "Saving energy in mobile devices for on-demand multimedia streaming-a cross-layer approach," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 10, no. 3, pp. 1–23, 2014.

[11] X. Chen, S. Jin, and D. Qiao, "M-psm: mobility-aware power save mode for ieee 802.11 wlans," in *31st IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2011, pp. 77–86.

[12] N. Ding, A. Pathak, D. Koutsonikolas, C. Shepard, Y. C. Hu, and L. Zhong, "Realizing the full potential of psm using proxying," in *IEEE INFOCOM*, 2012, pp. 2821–2825.

[13] H. Abou zeid, H. S. Hassanein, and S. Valentin, "Energy-efficient adaptive video transmission: Exploiting rate predictions in wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2013–2026, 2014.

[14] Y. Li, X. Zhang, and K. L. Yeung, "Dli: A dynamic listen interval scheme for infrastructure-based ieee 802.11 wlans," in *26th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2015, pp. 1206–1210.

[15] H. Tabrizi, G. Farhadi, and J. Cioffi, "An intelligent power save mode mechanism for ieee 802.11 wlan," in *IEEE Global Communications Conference (GLOBECOM)*, 2012, pp. 3460–3464.

[16] Omnet++ discrete event simulator. Webpage, accessed February 2016. [Online]. Available: https://omnetpp.org/

[17] IEEE 802.11 Task Group AX. Status of Project IEEE 802.11ax High Efficiency Wlan (HEW). Webpage, accessed February 2016. [Online]. Available: http://www.ieee802.org/11/Reports/tgax_update.htm

[18] Convince: Coder-transcoder recommended settings. Webpage, accessed February 2016. [Online]. Available: https://bscw-convince.celticplus.eu/bscw/bscw.cgi/d12333/Convince%20-%20Coder-Transcoder-recommended%20settings%20-v1.0.docx