# An Adaptive Tail Time Adjustment Scheme Based on Inter-Packet Arrival Time for IEEE 802.11 WLAN

Si Young Jang, Byoungheon Shin and Dongman Lee
School of Computing, KAIST
Deajeon, Korea
Email: {sy.jang, bhshin, dlee}@kaist.ac.kr

***Abstract*— Power management of Wi-Fi interfaces can greatly impact the battery life time of a smart device. Thus, many commercial devices utilize the Power Save Mode - Adaptive (PSM-A) mechanism for energy saving by switching wireless radio between high and low power. Tail time is introduced in PSM-A to put the radio in high power state until the time expires, allowing near future packets to arrive without much delay. However, this fixed tail time may result in a considerable amount of energy drainage under various types of traffic. This paper proposes an adaptive tail time adjustment scheme, a simple yet efficient way to save energy wastage by adaptively resizing the tail time according to prediction of data packet arrival times. The simulation results show up to 28.4% energy savings and reduces packet delivery delay compared to existing schemes.**

***Keywords*— *IEEE 802.11; power saving mode; inter-packet arrival time; tail time;***

## I. INTRODUCTION

Wi-Fi communication on mobile devices is a major source of energy consumption due to the state changes of Wi-Fi interface. Regardless of what chipset the system uses, the energy gap between Constantly Awake Mode (CAM) and Power Saving Mode (PSM) [1] is an order of magnitude. For instance, as measured with Monsoon power monitor and shown in Fig. 1, HTC Amaze induces approximately 12 mW while PSM, and up to 402 mW when being in idle CAM. The consumption gets much higher during active transmission of packets. In an effort to increase the energy efficiency of Wi-Fi interface, there have been several approaches [3, 4, 5, 11] to alter pre-set tail time (awake time of the wireless interface awaiting for next packet arrival) before entering PSM. However, fixed tail time becomes problematic in terms of latency and power consumption because it causes network interface to keep awake even in the absence of traffic. Considering the trade-off between energy efficiency and delay sensitivity, tail time should be carefully and dynamically adjusted.

There have been approaches to enhance energy efficiency by dynamically adjusting tail time of the Wi-Fi interface [3, 4]. They mainly focus on cross-layer approaches where upper layer (e.g. application and transport layers) features are exploited. SAPSM [3] adjusts tail time by prioritizing application traffic flows with Type of Service (TOS) bit so that the MAC layer can decide whether to transit to CAM state or not. However, this approach over-simplifies various traffic patterns in an application. For instance, Facebook is simply categorized as a text message application though it allows users to exchange non-text messages including videos or images. PSM-AW [4] estimates next packet arrival time using an end-to-end delay between a sender and a receiver and wakes up the interface right after the packet arrives at AP. It reduces the amount of tail time as minimum as possible, which leads to high energy saving. However, it needs to measure the round trip time (RTT) of all senders which cannot be acquired only from MAC layer.

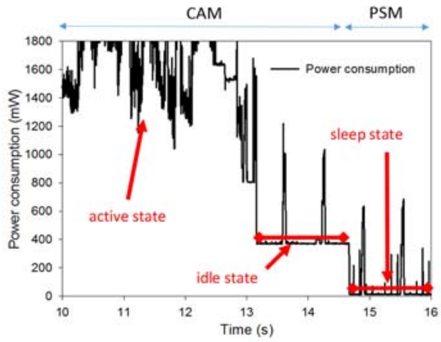


Fig. 1. Measurement of power consumption from the 802.11 device

In this paper, we propose an adaptive tail time adjustment scheme for IEEE 802.11 devices that dynamically adjusts tail time based on inter-packet arrival time with only MAC layer information. It exploits historical values of the inter-packet arrival time of on-going traffic obtained from the inter-packet arrival time monitor and determines how long tail time must be extended or shortened to decrease the duration of idle session of the interface. The proposed scheme adjusts tail time in relation with estimated packet arrival time from recent historical values of packet interval and beacon interval. To evaluate the proposed scheme, we implement a Qualnet [9] simulation with four different real-world traffic traces, i.e. web browsing, YouTube video streaming, and etc. As a performance metric, energy saving is compared to PSM-A with fixed tail time, SAPSM, and PSM-AW. The results show that the proposed scheme outperforms fixed PSM-A and SAPSM achieving up to 28.4% reduction of total awake time which is nearly close power savings of PSM-AW.

The rest of the paper is as follows. We analyze the related works in section II and present design considerations in section III. Section IV describes the proposed scheme and section V shows its performance evaluation. Lastly, in section VI, we conclude with further possibilities.

## II. RELATED WORK

The traditional 802.11 PSM standard accommodates PS-POLL method which transmits (polls) buffered packets in AP to

978-1-4799-6664-6/16/$31.00 ©2016 IEEE

client one by one until the more_bit has been set off. After more_bit has been set off, the clients' wireless interface transits to PSM. However, this method introduces additional latency (packets which will be buffered in AP after clients' interface switches to sleep state until next beacon) to packet delivery in interactive communication session due to quick transition to sleep state. PSM-A has been introduced to alleviate the problem of latency which occurs while using 802.11 PSM. PSM-A mechanism adopts tail timer. PSM-A is a mechanism which allows to switch from CAM to PSM adaptively so that the client can operate as CAM for a short time until the tail time expires. The tail timer extends upon another data packet arrival in this period allowing interactive performance to be as good as CAM while consuming less power by switching the interface to sleep state when necessary.

There has been several approaches to increase energy efficiency in mobile devices WiFi network by modifying the AP-side. NAPman [7] isolates PSM clients' traffic using an energy-aware fair scheduler, so to reduce unnecessary idle period caused by background traffic. SleepWell [8] further extends energy savings by taking other APs in vicinity into account in respect to staggering Beacon intervals. However, this approach could be burdensome to redeploy the software to already-fixed AP.

Increasing energy efficiency in mobile devices has been studied by modifying the client-side. [5, 11] uses traffic shaping on the client to add burstiness to TCP traffic which contributes to more efficient PSM behavior. Above two approaches target to minimize the number of tail time. In PSM-AW [4], the authors emphasizes the fact that packet delays are also affected by server delays which is critical in mobile environments. Thus, PSM-AW calculates possible delay between the client and the server to synchronize sleep time and minimizes the possible tail time.

SAPSM [3] implements an application classification module in Android which categories the traffic flow by machine learning and adds Type of Service (TOS) bit so that the MAC layer could decide when and when not to transit to CAM state. In scenarios where the traffic is classified as high priority, such as YouTube video application, the SAPSM system ensures that the system stays in CAM. On the other hand, the SAPSM system will stay in PSM when a low priority application such as Dropbox, transmits network data for maximum energy efficiency. While this approach may be feasible, it is not practical enough to be implemented in real world due to the need for cross layer information.

## III. DESGIN CONSIDERATIONS

A typical method for reducing energy consumption in Wi-Fi is to minimize the awake time. To overcome the limitations of the previous works, we propose a new scheme that enables energy saving using only MAC layer information. For this, we first analyze the factors that impact the awake time and find their correlation. Then we discuss the key parameters for reducing the awake time.

### *A. Factors Affecting Awake Time in Wi-Fi*

There are several factors affecting the awake time in Wi-Fi. Among them, tail time and inter-packet arrival time are key factors [10, 12]. Tail time is fixed in many commercial devices.

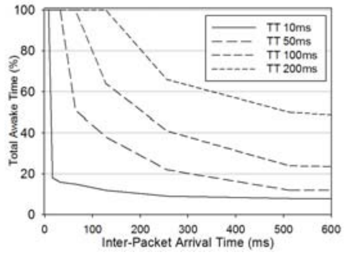


Fig. 2. Total awake time of the wireless interface by increasing inter-packet arrival time and varying tail time length

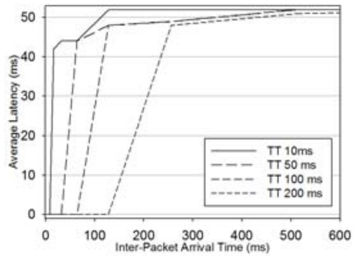


Fig. 3. Average delay time by increasing inter-packet arrival time and varying tail time

Inter-packet arrival time indirectly affects the awake time because it may induce additional tail time.

To understand the impact of the tail time and the inter-packet arrival time on power consumption, we conduct a controlled experiment on the Qualnet simulator. In the simulation, 1024 byte sized Constant Bit Rate (CBR) data packets are constantly sent from AP to client, and latency and energy consumption are monitored by varying tail time and inter-packet arrival time.

Fig. 2 shows the total awake time incurred by various inter-packet arrival time and tail time. Shorter tail time length always results in smaller total awake time, which means lower power consumption. However, merely reducing tail time may increase latency which degrades service quality. Fig. 3 shows the average latency due to various inter-packet arrival time and tail time. Latency starts drastically increasing at some point along with inter-packet arrival time. If inter-packet arrival time is longer than the tail time, the delay time occurs because the client Wi-Fi interface transits to PSM state. So the next packet is buffered in AP which is then transmitted on next beacon interval. This implies that if the inter-packet arrival time of the packet is longer than the clients' tail time, the average latency is almost the same.

By taking two observations together, for saving power and keeping the quality of service at the same time, it is important to estimate the next possible packet arrival time to adjust the tail time.

### *B. Correlation with Factors Affecting Energy Consumption*

In designing the proposed scheme, we consider tail time, estimated packet arrival time and time left until next beacon message which affects energy consumption. Fig. 4 depicts three possible cases of the next packet arrival time.

If the next packet arrives before the current tail time expires (case 1 in Fig. 4), the tail time can be reduced to a certain value in the boundary of δ, time difference between estimated packet arrival (EPAT1) and tail time expiration. If the next packet

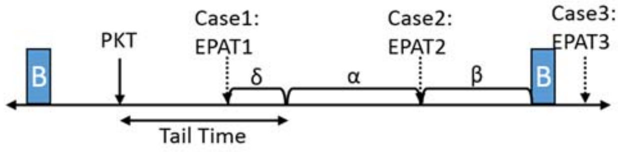


Fig. 4. Correlation between inter-packet arrival time and tail time

arrives in between the current tail time expiration and the next beacon message arrival (case 2 in Fig. 4), we should decide which option would be better for energy saving: either transit quickly into PSM state by expiring the current tail time or extend it to the estimated packet arrival (EPAT2). Conservatively extending the tail time would decrease the amount of sleep time by $\alpha$. We consider the clear trade-off between α and $\beta$ in our proposed scheme. If the next packet arrives after the next beacon message arrival (case 3 in Fig. 4), the next packet arrival will be delayed until the beacon message after the next beacon message. Thus it is beneficial to energy saving to expire the current tail time and transit to PSM state.

## IV. PROPOSED SCHEME

In this section, we describe the proposed algorithm in depth with the design consideration in mind. We propose a dynamic tail time adjustment scheme based on inter-packet arrival time which keeps low latency while reducing power consumption by resizing the tail time. The proposed scheme monitors recent *average inter-packet interval* (*AIPI*) and predicts the next *estimated packet arrival time* (*EPAT*). The tail time is reduced or extended based on the new prediction for energy saving. The proposed scheme consists of 'Inter-packet arrival time prediction ((1) in Fig. 5)' and 'Adaptive tail time decision adjustment ((2) in Fig. 5)'.

### A. Inter-Packet Arrival Time Prediction

Estimating the next possible packet arrival is important in adjusting the tail time. We use recently received data packets intervals as prediction reference. Here, *average inter-packet interval* (*AIPI*) is computed to find the current trend of *previous inter-packet interval* (*PIPI*). For this, we use a moving average scheme as shown in formula (a). We set its window size to 25, which provides most accurate prediction in our experiment (Fig. 8(a)). In order to predict the expected packet arrival time (EPAT) accurately, we exploit the Jacobson/Karels principle which takes standard deviation of EPAT into calculation [6] as in formula (1) & (2). The next packet arrival time is estimated as follows:

$$AIPI = \frac{\sum_{i=0}^{i=window\_size} PIPI_i}{window\_size}, \quad (1)$$

$$dev = \frac{\sqrt{\sum_{i=0}^{i=window\_size}(PIPI_i - AIPI)^2}}{window\_size}, \quad (2)$$

$$\text{EPAT} = AIPI + dev\ . \quad (3)$$

### B. Adaptive Tail Time Adjustment

The proposed scheme decides the tail time based on three important factors: the estimated packet arrival time (EPAT), the time left until beacon message, and the current tail time. We build solutions for the three cases described in Fig. 6. The proposed algorithm handles three cases as shown in Fig. 6 and its pseudo code is listed in in Fig. 7.

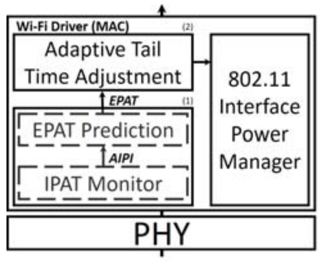


Fig. 5. Proposed scheme overview

**Case 1**: The estimated packet arrival time is less than the current tail time. In this situation, the tail time is reduced to the expected packet arrival time. Maximum energy savings can be expected if the tail time is reduced by $\delta$. Since an error boundary is adjusted in EPAT, it is safe to reduce by maximum $\delta$ value. Detailed parameter explanations are in Table 1. The tail time is adjusted as follows:

$$att = ctt - \delta \quad (4)$$

**Case 2**: The estimated packet arrival time exceeds the current tail time but before the next beacon reception. This situation divides into to two possibilities: $\alpha$ (difference between the tail time expiration and the expected packet arrival time) being smaller than $\beta$ (difference between the expected packet arrival time and the next beacon reception), or $\alpha$ being larger than $\beta$. Here we build a penalty model which combines energy penalty (extra energy consumption caused by extension of tail time - left part of (5)) and latency penalty (extra buffering delay caused by reduction of tail time - right part of (5)). The penalty model is as follows:

$$A = k\left(\frac{att}{ctt}\right) + (1-k)\left(\frac{L_a}{L_c}\right), \quad (5)$$

where a constant $k$ is the weighting factor ranging from 0 to 1. $L_a$ is the amount of packet delay time which could be assumed when adjusting the new tail time, while $L_c$ is the amount of packet delay time with the current tail time. *att* is a newly assumed tail time length, while *ctt* is a current tail time length. A constant $k$ allows user to give one parameter priority over the other. If user wants energy saving more than service quality, $k$ is set to a smaller than 0.5 and vice versa. $A$ is then used to determine which assumption (described in case 2a and 2b) produces smaller additional penalty.

Table 1. Parameter description

| Term | Description | Term | Description |
|---|---|---|---|
| $\alpha$ | Amount of time between EPAT and *ctt* | $att$ | Assumed (new) tail time length |
| $\beta$ | Amount of time between EPAT and $T_b$ | $ctt$ | Current tail time length |
| $\delta$ | Amount of reducible tail time | $T_{epat}$ | Expected packet arrival time |
| $L_C$ | Latency for currently set tail time | $T_b$ | Beacon time |
| $L_A$ | Latency for assumed (new) tail time | | |

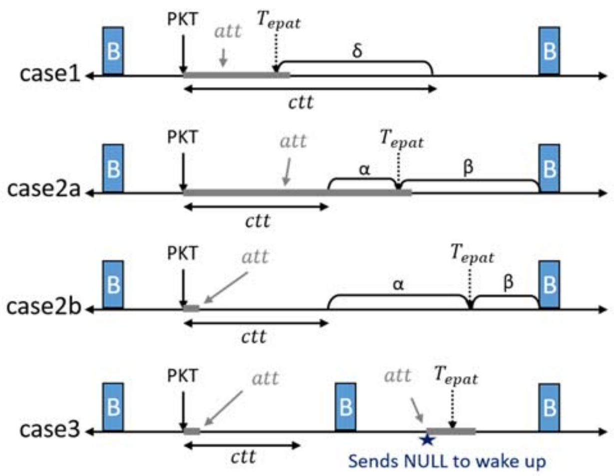


Fig. 6. Three possible scenarios of proposed scheme

Case 2a (α <β): We assume that it is energy efficient for the tail time to be extended until EPAT. Then we know $att$ will be $ctt$ + α. The penalty value for case2a is $A_1$.

Case 2b (α >β): We assume that it is energy efficient for the tail time to be dropped instantly. We know that $att$ will be zero then. The penalty value for case2b is $A_2$.

By comparing those two penalty values, one with a lower penalty value will be set to *att*. Note that the delay average is bound to less than 50ms on average. In our simulation 0.3 was used to increase energy savings.

**Case 3**: The estimated packet arrival exceeds the next beacon message. This scenario may be when packet intervals are very sporadic. It is most energy efficient to set the tail time value to zero and change Wi-Fi interface to PSM state until the second beacon. However, the next packet has to be delayed until the third beacon message, thus the client sends a dummy NULL packet just before EPAT to wake up and receive.

Table 2. Simulation parameters

| Parameter | Value |
|---|---|
| Simulator | Qualnet |
| Traffic pattern | Real data trace |
| Beacon interval | 100 ms |
| Window size | 25 |
| k | 0.3 |
| Fixed tail time | 200, 1500ms |
| Distance from AP | ~100m |
| Radio model | IEEE 802.11b/g |
| Number of clients | 6 |
| Data rate | 11Mbps |
| Simulation time | 180 s |

## V. Performance Evaluation

We perform the evaluation on Qualnet simulator [9]. We first conduct a simulation for finding an optimal window size of average inter-packet arrival time used in the proposed scheme. We evaluate the proposed scheme in terms of the amount of energy saving and average packet delivery delay from AP. The amount of energy saving is represented by subtracting total energy consumption of each scheme by energy consumed by

*#Step1: get EPAT from previous packet arrivals*
**procedure** AcquireEPAT()
  Set window value to 25
  Compute *AIPI*  # moving average of PIPI
  Compute *Dev*  # standard deviation of AIPI
  EPAT = *AIPI* + *Dev*

*#Step2: Identify placement of* $T_{epat}$ *and fix* $att$
**procedure** FixTatt()
  **Case1**: $att = ctt$ - δ  # $T_{epat} < att$
  **Case2**:  # $ctt < T_{epat} < T_b$
    Set $att = ctt$ + α  # Assume case2a
    Compute $A_1$
    Set $att =$ 0  # Assume case2b
    Compute $A_2$
    **if** ($A_1 < A_2$)
      Take $att$ of $A_1$
    **else if** ($A_2 < A_1$)
      Take $att$ of $A_2$
  **Case3**: $att = 0$  # $T_b < T_{epat}$

Fig. 7. Proposed algorithm

always-on scheme. Average packet delivery delay from an AP to a receiver is obtained from how long the packet has been buffered at the AP due to target clients sleep state. Both evaluations are conducted varying traffic types and intensity of background traffic at AP. Then we compare the proposed scheme with legacy IEEE 802.11 MAC with two different fixed tail time lengths - TT200 and TT1500 (which are used in commercial smartphones), SAPSM, and PSM-AW. Lastly, we observe the performance with mixed traffic set where different traffics are flown together.

### A. Simulation Setup

We leverage the real world traffic data from [13] which is a collection from 33 different Android users: 17 knowledge workers and 16 high school students. The user traffic behavior dataset is comprised of web browsing 58% (including social media and e-mail access), light weight network intensive applications (including file synchronization, system updates, mobile gaming, etc.) 25%, instant messaging 10%, video streaming 7% for our calculation.

To simulate the competing background traffic, we add five clients equipped with an IEEE 802.11b/g interface to the AP to observe how it affects the target node with our scheme. Amount of background traffic processed at the AP is represented in kbps in our evaluation. AP is set to schedule packets in first in first out manner. We declare packet delivery delay from AP as duration from packet arrival time at the AP to arrival time at the

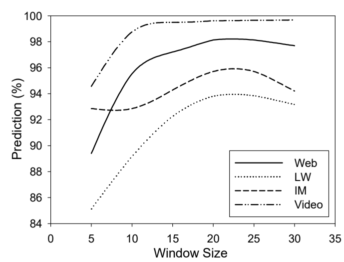


Fig. 8. Prediction rate with different window size (LW: light weight traffic, IM: instant messaging traffic).

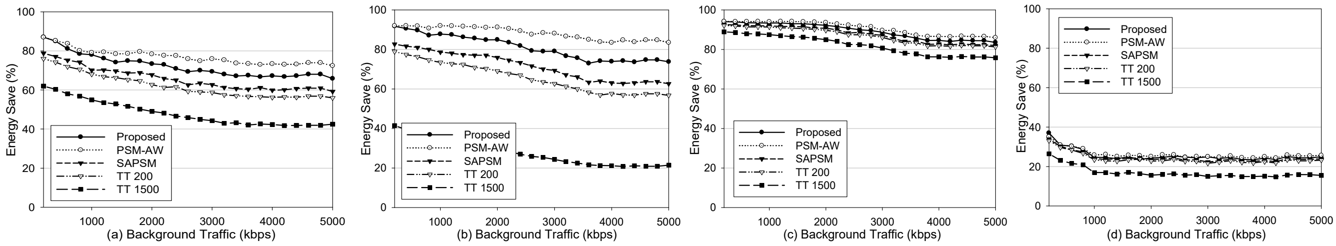

Fig. 9 (a). Energy saving with web traffic, (b). Energy saving with LW traffic, (c). Energy saving with IM traffic, (d) Energy saving with video traffic.

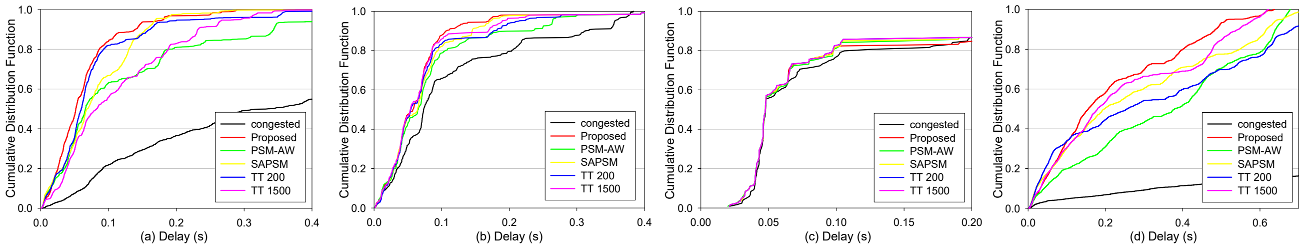

Fig. 10 (a). CDF with web traffic, (b). CDF with LW traffic, (c). CDF with IM traffic, (d) CDF with video traffic.

client to see how adjusting tail time affects proposed scheme. All plots are measured under 1000kbps of background traffic in average while 'congested' plot measures how much delay is occurred while delivering each packet to the target client under fully saturated AP regardless of scheme type.

In order to acquire the amount of energy saving, we use the power consumption measurements from a commercial Android smart device HTC Amaze, in Watts, of four different states: transmission (1417mW), reception (1319mW), idle (402mW) and sleep (12mW). Note that this values may vary a little by type of chipset. Total energy expenditure is obtained by multiplying duration of four states by each power consumption.

Inter-packet arrival time is calculated by the interval between each packet at the client node. The prediction of next estimated inter-packet arrival time is acquired by exploiting recent packet arrival time as described in section IV A. Parameters configured in this simulation are shown in Table 2. To normalize a bias from each single simulation, we run each session 20 times and average the results. For the power consumption metric, we use total amount of energy saving throughout the experiment.

*B. Optimal Window Size*

As described in Section IV, we use a moving average scheme for estimating next inter-packet arrival time. We vary the moving average window size from 5 to 30. As the window size increases, the prediction rate increases for more accurate prediction of the next inter-packet arrival time as shown in Fig. 8 for video traffic. However, further increase in window size over 25 degrades prediction rate for other traffic types. We set the optimal window size to 25 in our simulation.

*C. Energy Savings*

Fig. 9 shows how much energy is saved by each scheme for different traffic types under varying the amount of background traffic. The proposed scheme performs nearly as good as PSM-AW while outperforms SAPSM and the fixed tail time scheme. This proves that the proposed scheme can predict the arrival time of the next packet with much less overhead as accurately as PSM-AW which requires a cooperation with the IP layer. SAPSM saves less energy than the proposed scheme due to its high priority configuration for certain types of application traffic forcing Wi-Fi interface to stay in high power state for longer periods than necessary. TT 200 and TT 1500 schemes perform worst out of all schemes because of their fixed long tail time.

In overall, we can group energy saving patterns according to traffic types into two groups: web traffic and LW, and IM and video. Fig. 9 (a) and Fig. 9 (b) show the former. The proposed scheme saves up to 21.8% and 50.3% more energy than the existing schemes (beside PSM-AW) in case of web traffic and LW traffic, respectively. It is because these types of traffic have rather sporadic traffic patterns and energy saving is directly affected by the length of tail time. We can see that sporadic traffic patterns causes each scheme to be in idle state after receiving few packets and waiting for following packets to arrive.

Energy saving in case of IM and video traffic is shown in Fig 9 (c) and (d), respectively. The proposed scheme saves up to 11% more energy than the existing schemes in case of IM traffic. This is because the proposed scheme conserves more energy with varying background traffic at the AP due to a smaller amount of traffic generation for each session even though packets arrive sporadically. In case of video traffic, the proposed scheme saves up to 13.5% more energy due to longer state in transmission and reception state and shorter duration in sleep state. Hence, tail time does not affect much for this traffic pattern. Note that the transmission and reception states consume more energy than the idle or sleep state.

*D. Packet Delivery Delay from AP*

Fig. 10 shows packet delivery delay from AP to a receiver in a form of the cumulative distribution function, running different types of traffic. The proposed scheme performs better than PSM-AW because frequent changes in window size used for prediction reduce the accuracy of PSM-AW. The proposed scheme also performs better than SAPSM. This is because

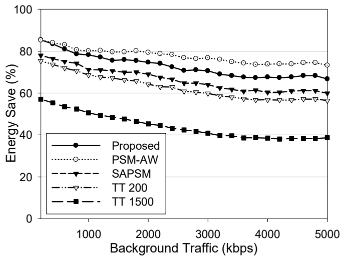

Fig. 11. Energy saving with mixed traffic.

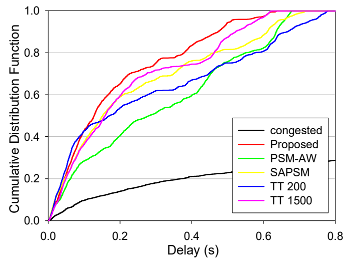

Fig. 12. Delay with mixed traffic.

SAPSM classifies certain applications as low priority, which preventing Wi-Fi interface from entering an active state.

The proposed scheme running web traffic induces 90% of packet delivery delay under 100ms while 350ms for PSM-AW which is worse than the fixed tail time schemes shown in Fig. 10 (a). A sporadic traffic pattern changes the prediction window size for PSM-AW, resulting in more delivery delay. This delay decreases to 200ms for PSM-AW in Fig. 10 (b) while the proposed scheme delivers more packets in a shorter time running LW traffic. In case of IM traffic, packet delivery delays are shown in Fig. 10 (c). For all schemes in running IM application, delay time is under 100ms. As shown in Fig. 10 (d), packet delivery for video traffic is slow due to a large amount of packets to be delivered for watching high resolution video. Packet transfer from a server to AP is faster than AP to a target client, thus packets get delayed in the AP buffer. The proposed scheme running video traffic induces 90% of packet delivery delay under 450ms while 630ms for PSM-AW. In summary, the proposed scheme shows lower delivery time from AP to a target client in all scenarios.

### *E. Mixed traffic scenario*

Fig. 11 and Fig. 12 depict energy savings and packet delivery time when all different types of traffic flow simultaneously. The proportion of mixed traffic is fixed as discussed in Section V. A. Due to the large proportion of web traffic, energy saving shows a similar pattern with that of web traffic while packet delivery delay graph shows a similar pattern with that of video traffic due to a large number of video packets. The proposed scheme saves energy up to 28.4 % in comparison with the existing schemes (beside PSM-AW) while minimizing overall packet delivery delay time.

## VI. Conclusion

In this paper, we present an adaptive tail time adjusting scheme based on inter-packet arrival time for IEEE 802.11 WLANs. As the main building blocks, inter-packet arrival time prediction module and adaptive tail time adjustment module are designed. The inter-packet arrival time prediction module keeps track of current inter-packet arrival times, averages the intervals and predicts the following packets' interval. Then the adaptive tail time adjustment module determines whether to reduce or extend the current tail time accordingly for both energy efficiency while reducing packet delivery delay.

For evaluation, we implement our scheme in Qualnet simulator and compare the total sleep time of the proposed scheme and existing schemes. The result shows the proposed scheme saves up to 28.4% of power consumption compared to PSM-A with fixed tail time length in mixed traffic scenario while minimizing packet delivery time. The proposed scheme saves as much as PSM-AW while having shortest packet delivery delay. To check the feasibility on real world devices, performance evaluation on Android smart devices are left for future work.

## VII. Acknoledgement

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (R0184-15-1037, Development of Data Mining Core Technologies for Real-time Intelligent Information Recommendation in Smart Spaces; B101-15-0334, Development of IoT-based Trustworthy and Smart Home Community Framework)

## VIII. References

[1] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, IEEE 802.11, June 1999.

[2] Krashinsky, Ronny, and Hari Balakrishnan. "Minimizing energy for wireless web access with bounded slowdown." ACM Wireless Networks. vol. 11, pp. 135-148, 2005.

[3] Pyles, Andrew J., et al., "SAPSM: Smart adaptive 802.11 PSM for smartphones." Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp 2012), pp. 11-20, 2012.

[4] Peck, B., and D. Qiao. "A Practical PSM Scheme for Varying Server Delay." IEEE Transaction on Vehicular Technology, vol. 64, pp. 303-314, 2014.

[5] Looga Vilen, et al.. "Exploiting traffic scheduling mechanisms to reduce transmission cost on mobile devices." Wireless Communications and Networking Conference (WCNC 2012),pp. 1766-1770, 2012.

[6] V. Jacobson, "Congestion avoidance and control." Symposium proceedings on Communications architectures and protocols (SIGCOMM 1988), pp. 314-329. 1988.

[7] Eric Rozner, et al., "NAPman: network-assisted power management for wifi devices," Proceedings of the 8th international conference on Mobile systems, applications, and service (MobiSys 2010). pp. 91-106, 2010.

[8] Manweiler Justin, and Romit Roy Choudhury. "Avoiding the rush hours: WiFi energy management via traffic isolation." Proceedings of the 9th international conference on Mobile systems, applications, and services (MobiSys 2011), pp. 253-266, 2011.

[9] Qualnet: http://web.scalable-networks.com/content/qualnet

[10] Liu Jiayang, and Lin Zhong. "Micro power management of active 802.11 interfaces." Proceedings of the 6th international conference on Mobile systems, applications, and services (MobiSys 2008), pp. 146-159. 2008.

[11] Enhua Tan, et al., "PSM-throttling: Minimizing energy consumption for bulk data communications in WLANs." Network Protocols, ICNP 2007. IEEE International Conference on. IEEE, pp. 123-132, 2007.

[12] Hao Han, et al., "Dozyap: power-efficient wi-fi tethering." Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys 2012), pp. 421-434, 2012.

[13] Falaki, Hossein, et al. "A first look at traffic on smartphones." Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (IMC 2010). Pp. 281-287, 2010.