# An Energy Efficient Sleep Scheduling Considering QoS Diversity for IEEE 802.16e Wireless Networks

Jen-Jee Chen, Jia-Ming Liang, and Yu-Chee Tseng
Department of Computer Science
National Chiao Tung University, Hsin-Chu 30010, Taiwan
E-mail: {chencz,jmliang,yctseng}@cs.nctu.edu.tw

***Abstract*—Power management is one of the most important issues in IEEE 802.16e wireless networks. In the standard, it defines three types of *power saving classes (PSCs)* for flows with different QoS characteristics. It allows a mobile device to turn off its wireless radio when all its PSCs are in sleep states. In this paper, we consider the scheduling of power saving classes of type II in an IEEE 802.16e network with a BS and multiple MSSs (mobile subscriber stations). Previous work proposes to enforce all MSSs to have the same sleep cycle, thus leading to higher energy cost for those MSSs with less strict delay bounds. We observe that if the sleep cycles of MSSs can be assigned according to their delay bounds, MSSs can significantly reduce their duty cycles. We propose an efficient *tank-filling algorithm*, which is standard-compliant and can allocate resources to MSSs according to their QoS characteristics with the least number of active frames. Simulation results verify that our algorithm incurs less power consumption and leads to higher bandwidth utilization than the previous schemes.**

***Index Terms*—IEEE 802.16e, power management, power saving class (PSC), quality of service (QoS), WiMAX, wireless network.**

## I. Introduction

The IEEE 802.16e [1], [2] is a promising standard for providing broadband wireless access to *mobile subscriber stations (MSSs)* with high mobility. Like most other wireless mobile systems, how to conserve energy for battery-powered MSSs is a critical issue in IEEE 802.16e. In IEEE 802.16e, three types of PSCs (power saving classes) are defined. A PSC can be associated to one or more flows in an MSS. When a PSC is activated, it repeatedly switches between sleep and listening windows, where only during a listening window, can its member flows transmit/receive data. When all PSCs of an MSS are in their sleep windows, the MSS can turn off its radio transceiver to save energy.

The three types of PSC in IEEE 802.16e are reviewed below. In type I, the sizes of listening windows are fixed while the sizes of sleep windows grow exponentially when no data arrives. Once any traffic arrives, the PSC will be deactivated, until all queued traffics are delivered. So, PSCs of type I are more suitable for non-real-time traffic variable-rate (NRT-VR) and best-effort (BE) flows. In type II, the sizes of both listening and sleep windows are fixed. However, unlike type I, the arrival of traffics will not deactivate the PSC. This type II is more suitable for unsolicited grant service (UGS) and real-time traffic variable-rate (RT-VR) flows. In type III, it is only valid for one sleep window, after which the PSC is deactivated. This type is more suitable for multicast services and management operations. Among these three types, we are more interested in PSC of type II because one may dynamically adjust such PSCs' sleeping behaviors to maximize MSSs' energy efficiency.

In the literature, performance analyses for PSCs in an IEEE 802.16e network are conducted in [3]–[5]. For an MSS-BS pair, [6]–[8] focus on the design of PSCs of type I. How to adaptively adjust the initial sleep window is addressed in [6]. Assuming that the distribution of the response packet arrival time is known, [7] proposes a decision algorithm such that the MSS can stay asleep until response packets are expected to arrive. In [8], how to adjust the minimum and the maximum sleep windows is discussed. For type II, assuming that PSCs are already given, a Maximum Unavailability Interval scheme is proposed in [9] for selecting the optimal start frame for each PSC to maximize its unavailable duration. References [10], [11] propose to apply one single PSC to accommodate all real-time flows in an MSS; parameters of the PSC are selected to meet the flow with the strictest bandwidth and packet delay bound. Considering multiple MSSs under the same BS, [12] proposes a *Longest-Virtual-Burst-First (LVBF)* scheme, which always selects a primary MSS in the burst mode to serve and only gives the necessary bandwidth to the other MSSs to meet their requirements. However, it does not take delay constraints of flows into consideration. Reference [13] proposes to serve each MSS by a PSC of type II, but all of them share the same sleep cycle. This results in PSCs without overlapping in their active frames. However, since the common sleep cycle is bounded by the strictest delay bound of all MSSs, this way causes some MSSs to have too many active frames.

In this work, we focus on PSCs of type II. Given multiple MSSs under a BS, we consider the arrangement of PSCs for these MSSs according to their delay bounds and bandwidth requirements. This involves not only the selection of each PSC's parameters, but also the selection of their listening windows to reduce the overall active frames of MSSs. We propose a tank-filling algorithm, which regards the resources of the BS as a sequence of periodical tanks, each being able to provide a fixed amount of bandwidth. The result outperforms that of [13] because we relax the constraint that all PSCs should share the same sleeping cycle. Simulation results are provided to verify these claims.

The rest of this paper is organized as follows. Section II gives some motivations and formally defines the problem. Our tank-filling algorithm is presented in Section III. Simulation results are shown in Section IV. Section V concludes this paper.

## II. Motivation and Problem Definition

In this section, we first motivate our work by discussing previous work [13]. Then we formally define our problem. In [13], assuming that there are multiple MSSs, each to be served by a PSC of type II, it enforces each MSS to adopt a PSC of the same sleep cycle length. The sleep cycle is selected to meet the MSS with the tightest delay bound. While the solution is easy to implement, this is too restricted and may incur too many active frames to some MSSs. Fig. 1 shows an example with two MSSs $M_1$ and $M_2$, which have data arrival rates of $\tau_1 = 0.2\Omega$/frame and $\tau_2 = 0.075\Omega$/frame and delay bounds of $D_1 = 4$ (frames) and $D_2 = 12$ (frames), respectively, where $\Omega$ is the capacity of a frame. Fig. 1(a) shows the schedule computed by [13]. Since $\min(D_1, D_2) = 4$, in every four frames, $M_1$ and $M_2$ will be active for one frame and be allocated of bandwidths $\tau_1 \times 4 = 0.8\Omega$ and $\tau_2 \times 4 = 0.3\Omega$, respectively, per frame. Also, their active frames are shifted to avoid overlapping. As Fig. 1(b) shows, by assigning each MSS a sleep cycle adaptive to its delay bound, $M_1$ and $M_2$ can have sleep cycles of 4 and 12 frames, respectively, where in each active frame, they receive $\tau_1 \times 4 = 0.8\Omega$ and $\tau_2 \times 12 = 0.9\Omega$ of bandwidths. Still we can manage to incur no overlapping among their active frames, so $M_2$'s duty cycle is significantly reduced.

The above observation motivates us to study a power management problem as follows. We consider a BS serving $n$ MSSs $M_i$, $i = 1..n$. Each $M_i$ has a data arrival rate of $\tau_i$ bits/frame and each data arrival has a delay bound of $D_i$ frames. Assuming the available bandwidth per frame is $\Omega$ bits and $\sum_{i=1..n} \tau_i \leq \Omega$, the goal is to assign each $M_i$ a PSC of type II with a sleep cycle of $T_i^S$, a listening window of $T_i^L$, and an offset of $T_i^O$, such that $T_i^S \leq D_i$ and the total number of active frames for all MSSs is minimized. Also, there is implicit requirement that whenever a listening window of an MSS arrives, the BS should be able to serve all its backlog data that would be overdue otherwise.

## III. The Proposed Tank-Filling Scheme

In an IEEE 802.16e wireless network, the BS is responsible for scheduling the sleep frames of the MSSs associated with it. Initially, each $M_i, i = 1..n$, will send a request to the BS containing its $D_i$. We propose a *tank-filling (TF)* algorithm for the BS to determine the following parameters for each $M_i$: (1) $(T_i^S, T_i^L, T_i^O)$ and (2) amount of bandwidth $B_{i,j}$ allocated to $M_i$ in the $j$-th active frame in each listening windows, $j = 1..T_i^L$ (noth that $B_{i,j}$ is a real number between 0 and 1). Then these parameters are sent to each $M_i$. Then these MSSs will behave accordingly.

Our TF algorithm considers the resources of the BS as a sequence of repetitive *tanks*, each being able to hold $\Omega$ amount of water per frame. It maintains an important property that $T_i^S$ of each $M_i$ is an integer multiple of its previous $T_{i-1}^S$ for each $i = 2..n$. So, we call $T_1^S$ as the basic cycle, or simply $T_{basic}$, of the network. Intuitively, this property helps make MSSs' sleeping behaviors regular and increase the overlapping of their listening windows. Assuming that $T_1^S, T_2^S, ..., T_n^S$, are known (recall that $T_1^S = T_{basic}$), the resources controlled by the BS are represented by an array $R[1 : \frac{T_n^S}{T_{basic}}, 1 : T_{basic}]$, where each $R[k, \ell]$, $k = 1..\frac{T_n^S}{T_{basic}}$ and $\ell = 1..T_{basic}$, is to record the amount of remaining resource in the $\ell$-th frame of the $k$-th basic cycle. Initially, $R[k, \ell] = \Omega$ is regarded as an empty tank. Gradually, we will fill in more data to each tank. Below, we present our TF algorithm in three steps. (A) Assuming that $T_{basic}$ is known, we will choose $T_i^S$ of each $M_i, i = 1..n$. (B) Determine $T_i^L$, $T_i^O$, and $B_{i,j}, j = 1..T_i^L$, of each $M_i, i = 1..n$. (C) In the end, we will come back and search for the most energy-efficient basic cycle $T_{basic}$.

### A. Determining $T_i^S$ of $M_i$

To decide $T_i^S$, we first sort MSSs by their delay bounds. Without loss of generality, let $D_1 \leq D_2 \leq \cdots \leq D_n$. Supposing that $T_1^S = T_{basic}$ is known and $T_1^S \leq D_1$, we set $T_i^S$, $i = 2..n$, as follows:

$$T_i^S = T_{i-1}^S \times \left\lfloor \frac{D_i}{T_{i-1}^S} \right\rfloor . \quad (1)$$

It is not hard to see that Eq. (1) implies $T_i^S \leq T_{i-1}^S \times \frac{D_i}{T_{i-1}^S} = D_i$. So, $T_i^S$ guarantees the delay bound of $M_i$. In fact, Eq. (1) also ensures that $T_i^S$ is an integer multiple of $T_{i-1}^S$.

*Lemma 3.1:* Eq. (1) guarantees that each $T_i^S$ is an integer multiple of $T_{i-1}^S$, $i = 2..n$, and $T_i^S \leq D_i$, $i = 1..n$.

### B. Scheduling $T_i^L$, $T_i^O$, and $B_{i,j}$ of $M_i$

Recall the array $R[\cdot, \cdot]$, which represents the resource of the BS. We will sequentially allocate resources for $M_i$, $i = 1..n$, by updating $R[\cdot, \cdot]$. Our algorithm is called 'tank-filling' when $M_i$ is being considered, we will test every 'starting' tank in $R$ by sequentially filling its data to the empty part of that tank and continuing to next tank in $R$, until all the data is drained. Note that here $R$ is regarded as $T_n^S$ tanks is a row-major way. Among these testing starting tank, the one resulting in the least active frames to $M_i$ is selected. The detail procedure for $M_i$ is follows, where $i$ starts from 1 and end at $n$:

a) Calculate the bandwidth requirement of $M_i$ per $T_i^S$ by $\gamma_i = \tau_i \times T_i^S$.

b) When $M_i$ enters the step, $R[\cdot, \cdot]$, if regarded in a row-major order, has a period of $T_{i-1}^S$ (see that note at the end of step (d)). So we let $j = 1..T_{i-1}^S$ as the potential indices of the starting tanks and run the following steps for each $j$.

   i) Starting from the $j$-th tank in $R[\cdot, \cdot]$, we fill in the bandwidth requirement $\gamma_i$ of $M_i$ into the empty part of the tank. If there is sufficient space for $\gamma_i$, we are done; otherwise we fill the $j$-th tank up proceed to the $(j+1)$-th tank. We continue the process until all $\gamma_i$ is satisfied.

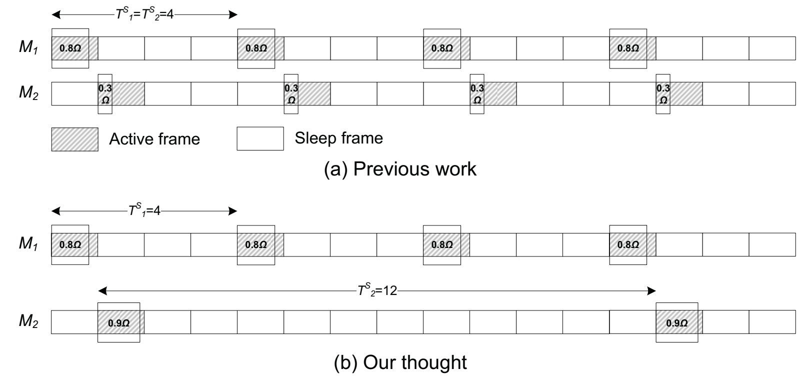



Fig. 1. Sleep scheduling for two MSSs $M_1$ and $M_2$ using (a) a common sleep cycle and (b) different sleep cycles.

ii) Let $f(j)$ be the number of tanks that have been used to serve $M_i$'s data. This is regarded as the cost function to start with the $j$-th tank.

c) Among all possible $j$s in step (b), let $j^*$ be the index which induces the smallest cost $f(\cdot)$. We then place $M_i$'s demand starting from the $j^*$-th tank according to above procedure. Note that in case of a tie, we will give priority to the one which leaves the least remaining resource in the last frame where $M_i$'s demand is placed.

d) Then we set $T_i^L = f(j^*)$ and $T_i^O = j^*$. Also, we set $B_{i,j}$ to the bandwidth allocated to $M_i$ in the $j$-th tank, $j = 1..T_n^S$, and subtract $B_{i,j}$ from the corresponding entry in $R$ (note that the allocation in step (b) should be repeated $\frac{T_n^S}{T_i^S}$ for array $R$, so $R$ has a period of $T_i^S$ at the end of this step).

As noted in step (d), after the allocation of $M_i$, array $R$ has a period of $T_i^S$. This would simplify our next allocation for $M_{i+1}$ since $T_{i+1}^S$ is an integer multiple of $T_i^S$.

*Example 1:* Fig. 2 shows an example of step B. There are 5 MSSs $M_1$, $M_2$, $M_3$, $M_4$, and $M_5$ with sleeping cycles of $T_1^S = T_{basic}$, $T_2^S = 2T_{basic}$, $T_3^S = 2T_{basic}$, $T_4^S = 2T_{basic}$, and $T_5^S = 4T_{basic}$ and required resources per cycle of $\gamma_1 = 0.5\Omega$, $\gamma_2 = 1.25\Omega$, $\gamma_3 = 0.4\Omega$, $\gamma_4 = 0.4\Omega$, and $\gamma_5 = 2.5\Omega$, respectively, where $T_{basic} = 3$ frames. Initially, $R[k, \ell] = \Omega$ for $k = 1..4$ and $\ell = 1..3$. Then, each $M_i$ is scheduled as follows. For $M_1$, we can only set $j^* = 1$. Then, the BS reserves $\gamma_1 = 0.5\Omega$ resource for $M_1$ in every basic cycle as shown in Fig. 2(1) and set $B_{1,1} = B_{1,4} = B_{1,7} = B_{1,10} = 0.5\Omega$, $T_1^O = j^* = 1$, and $T_1^L = f(1) = 1$; so $R[1, 1] = R[2, 1] = R[3, 1] = R[4, 1] = 0.5\Omega$ and $R[k, \ell] = \Omega$ for $k = 1..4$ and $\ell = 2, 3$. For $M_2$, its $j^*$ can be 1 or 2 or 3 and allocating $\gamma_2$ by starting from any of the two basic cycles are the same. Since $\lceil 0.5\Omega + 1.25\Omega \rceil - (0.5\Omega + 1.25\Omega) = 0.25\Omega < \lceil 1.25\Omega \rceil - 1.25\Omega = 0.75\Omega$, setting $j^* = 1$ and 3 would create the least number of active frames and leave the least remaining resource in the last frame. So we select $j^* = 1$. After the allocation, shown as Fig. 2(2), we set $B_{2,1} = B_{2,7} = 0.5\Omega$, $B_{2,2} = B_{2,8} = 0.75\Omega$, $T_2^O = j^* = 1$, and $T_2^L = f(1) = 2$ and update $R[1, 1] = R[3, 1] = 0$ and $R[1, 2] = R[3, 2] = 0.25\Omega$. For $M_3$, choosing $j^* = 3$, 4, 5, and 6 would create the same and least number of active frames (i.e., $f(3) = f(4) = f(5) = f(6) = 1 < f(2) = 2 < \cdots$), but setting $j^* = 4$ would leave less remaining resource in the last allocated frame (i.e., $0.1\Omega$). So we set $j^* = 4$ and update $B_{3,4} = B_{3,10} = 0.4\Omega$, $T_3^O = j^* = 4$, and $T_3^L = f(4) = 1$, as shown in Fig. 2(3). Then update $R[2, 1] = R[4, 1] = 0.1\Omega$. For $M_4$, setting $j^* = 3$, 5, and 6 would create the same and least number of active frames (i.e., $f(3) = f(5) = f(6) = 1 < f(2) = f(4) = 2 < \cdots$) and leave the same remaining resource in the last frame. So we choose $j^* = 3$ and set $B_{4,3} = B_{4,9} = 0.4\Omega$, $T_4^O = j^* = 3$, and $T_4^L = f(3) = 1$, as shown in Fig. 2(4). Then we update $R[1, 3] = R[3, 3] = 0.6\Omega$. For $M_5$, choosing $j^* = 3$ would add the least number of active frames (i.e., $f(3) = 4 < f(2) = f(5) = 5 < f(4) = 6 < \cdots$). So we choose $j^* = 3$ and set $B_{5,3} = 0.6\Omega$, $B_{5,4} = 0.1\Omega$, $B_{5,5} = \Omega$, $B_{5,6} = 0.8\Omega$, $T_5^O = j^* = 3$, and $T_5^L = f(3) = 4$, as shown in Fig. 2(5). Then update $R[1, 3] = R[2, 1] = R[2, 2] = 0$ and $R[2, 3] = 0.2\Omega$.

### C. Selecting $T_{basic}$

Clearly, different values of $T_1^S$ will lead to different duty cycles for MSSs. Here we adopt an exhausted search by setting $T_1^S = 1..D_1$ and trying to find the sum of the total number of active frames of all MSSs over a windows of $T_n^S$ frames. Then $T_1^{S^*}$ leading to the least number of active frames is chosen as $T_{basic}$.

## IV. Performance Evaluation

We have developed a simulator by C++ to verify the effectiveness of our PMSS scheme. Unless otherwise stated, the following assumptions are made in our simulation. The number of MSSs is ranged from 5 to 45. Each MSS $M_i$ has a data rate $\tau_i$ of $1000 \sim 3000$ bits/frame and delay bound $D_i$ of $10 \sim 200$ frames, where 1000 is the minimum data rate, 3000 is the maximum data rate, 10 is the minimum delay bound, and 200 is the maximum delay bound of the MSS. The available bandwidth per frame of the system is $\Omega = 80000$ bits (16Mbps) and the length of an OFDM/OFDMA frame is set

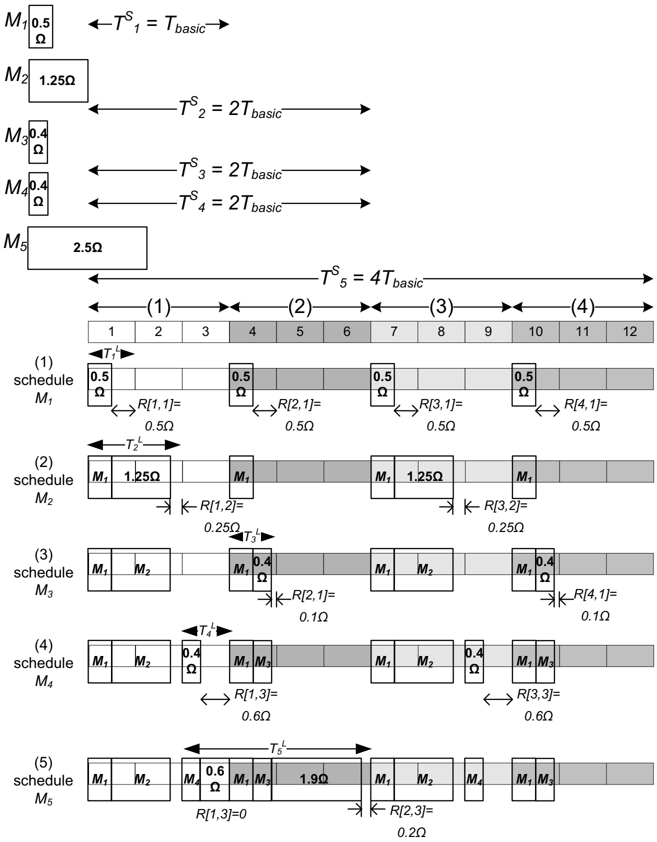


Fig. 2. Example of scheduling $B_{i,j}$, $T_i^L$, and $T_i^O$ for five MSSs $M_1$, $M_2$, $M_3$, $M_4$, and $M_5$.

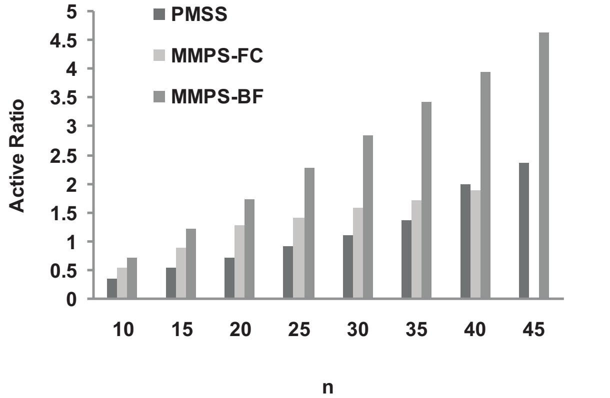
to 5 ms [14]. We consider two performance metrics: (i) *active ratio*: the ratio of active frames for the system and (ii) *fail-to-sleep probability*: the ratio of failure to schedule MSSs' sleep. We will compare our PMSS against the MMPS-FC (Multiple MSSs Power-saving Scheduler with Fragment Collection) and MMPS-BF (Multiple MSSs Power-saving Scheduler with Boundary Free) schemes in [13].

### A. *Effects of* $n$

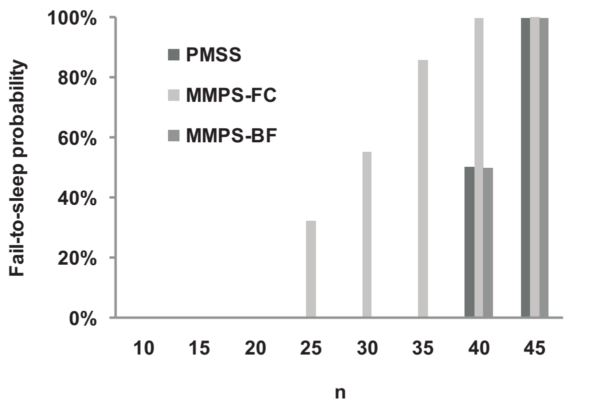
In this experiment, we study the effect of $n$ on the active ratio and fail-to-sleep probability. Fig. 3(a) shows the active ratio decreases when $n$ increases. Our PMSS almost always performs the best in all three schemes, except at $n = 40$, MMPS-FC performs better than our PMSS. However, when $n = 40$, the fail-to-sleep probability of MMPS-FC is almost $100\%$ (Fig. 3(b)). Fig. 3(b) shows the fail-to-sleep probability increases when $n$ increases. MMPS-BF and our PMSS schemes perform the same and the best in the fail-to-sleep probability. The fail-to-sleep probabilities of the two schemes is zero when $n < 40$. For MMPS-FC, it can $100\%$ successfully schedule MSSs into sleep when $n < 25$.

### B. *Effects of Maximum Delay Bound*

Then, we investigate the effect of maximum delay bound on the active ratio by fixing $n = 20$. Fig. 4 shows the active



(a)



(b)

Fig. 3. Effects of number of MSSs on (a) active ratio and (b) fail-to-sleep probability.

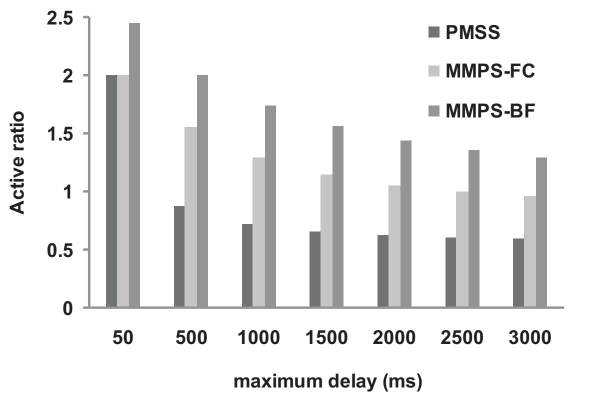
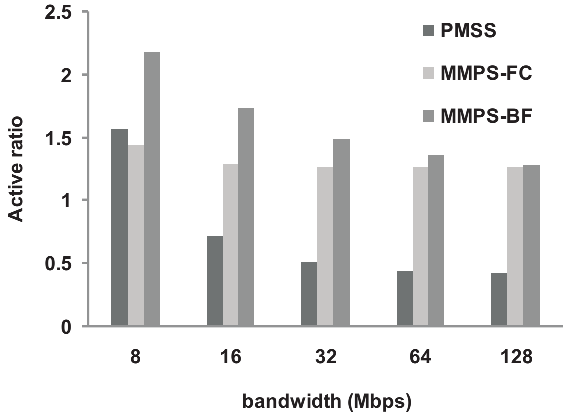
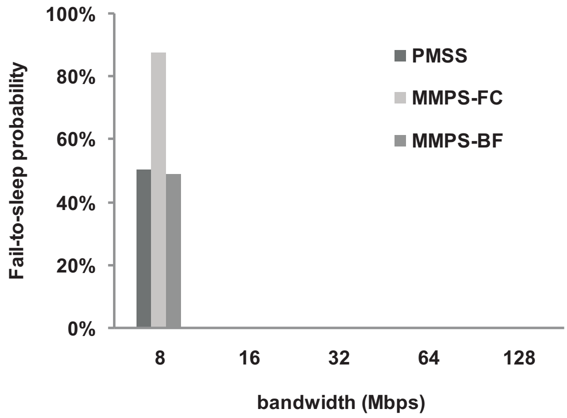


Fig. 4. Effects of maximum delay on active ratio.

ratio decreases when the maximum delay bound increases. Our PMSS performs the best in all three schemes. For the three schemes, our PMSS benefits the most when the maximum delay bound is increased from 50 ms to 3000 ms (70%); for MMPS-FC and MMPS-BF, the improvement is $52\%$ and $47\%$, respectively. This is because our scheme can more accurately capture the traffic characteristics of MSSs.




(a)




(b)

Fig. 5. Effects of system bandwidth on (a) active ratio and (b) fail-to-sleep probability.

### *C. Effects of System Bandwidth*

In this experiment, we investigate the effect of system bandwidth on the active ratio and fail-to-sleep probability by fixing $n = 20$. Fig. 5(a) shows the active ratio decreases when system bandwidth increases. Our PMSS outperforms other two schemes except when the system bandwidth is 8Mbps. When the system bandwidth is 8Mbps, MMPS-FC performs the best but its fail-to-sleep probability is much higher ($88\%$) than other two schemes (about $50\%$). For the three schemes, our PMSS benefits the most when system bandwidth is increased from 8Mbps to 128Mbps ($73\%$); for MMPS-FC and MMPS-BF, the improvement is $13\%$ and $42\%$, respectively.

## V. CONCLUSIONS

In this paper, we propose a per-MSS sleep scheduling scheme for multiple MSSs in IEEE 802.16e wireless networks such that the overall power consumption of the system is minimized while the QoS of each MSS can be guaranteed. Compared to the previous work, our approach assigns and schedules type II PSCs for each MSS by considering each of their QoS characteristics such that the sleep scheduling can more accurately capture each MSS's QoS requirement. This leads to each MSS can sleep more and the total power consumption of the system is significantly reduced. Also, the proposed scheme is easy to implement and compatible to the standard.

## ACKNOWLEDGEMENTS

Y.-C. Tseng's research is co-sponsored by MoE ATU Plan, by NSC grants 96-2218-E-009-004, 97-3114-E-009-001, 97-2221-E-009-142-MY3, and 98-2219-E-009-005, by MOEA 98-EC-17-A-02-S2-0048, and 98-EC-17-A-19-S2-0052, and by ITRI, Taiwan.

## REFERENCES

[1] IEEE Std 802.16e-2005, "IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1," Feb. 2006.

[2] IEEE Std 802.16-2009, "IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems," May 2009.

[3] Y. Xiao, "Energy saving mechanism in the IEEE 802.16e wireless MAN," *IEEE Communications Letters*, vol. 9, no. 7, pp. 595–597, 2005.

[4] Y. Zhang and M. Fujise, "Energy management in the IEEE 802.16e MAC," *IEEE Communications Letters*, vol. 10, no. 4, pp. 311–313, 2006.

[5] Y. Zhang, "Performance Modeling of Energy Management Mechanism in IEEE 802.16e Mobile WiMAX," in *Proc. of IEEE Wireless Communications and Networking Conference*, 2007, pp. 3205–3209.

[6] J. Xiao and S. Zou and B. Ren and S. Cheng, "An Enhanced Energy Saving Mechanism in IEEE 802.16e," in *Proc. of IEEE GLOBECOM*, 2006.

[7] J.-R. Lee and D.-H. Cho, "Performance Evaluation of Energy-Saving Mechanism Based on Probabilistic Sleep Interval Decision Algorithm in IEEE 802.16e," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 4, pp. 1773–1780, July 2007.

[8] M.-G. Kim and J.-Y. Choi and M. Kang, "Adaptive power saving mechanism considering the request period of each initiation of awakening in the IEEE 802.16e system," *IEEE Communications Letters*, vol. 12, no. 2, pp. 106–108, 2008.

[9] T.-C. Chen, J.-C. Chen, and Y.-Y. Chen, "Maximizing Unavailability Interval for Energy Saving in IEEE 802.16e Wireless MANs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 4, pp. 475–487, Apr. 2009.

[10] S.-L. Tsao and Y.-L. Chen, "Energy-efficient packet scheduling algorithms for real-time communications in a mobile WiMAX system," *Computer Communications*, vol. 31, no. 10, pp. 2350–2359, June 2008.

[11] H.-L. Tseng, Y.-P. Hsu, C.-H. Hsu, P.-H. Tseng, and K.-T. Feng, "A Maximal Power-Conserving Scheduling Algorithm for Broadband Wireless Networks," in *Proc. of IEEE WCNC*, 2008, pp. 1877–1882.

[12] J. Shi, G. Fang, Y. Sun, J. Zhou, Z. Li, and E. Dutkiewicz, "Improving Mobile Station Energy Efficiency in IEEE 802.16e WMAN by Burst Scheduling," in *Proc. of IEEE GLOBECOM*, 2006.

[13] S.-C. Huang, R.-H. Jan, and C. Chen, "Energy Efficient Scheduling with QoS Guarantee for IEEE 802.16e Broadband Wireless Access Networks," in *Proc. of the International conference on Wireless Communications and Mobile Computing (IWCMC)*, pp. 547–552.

[14] H. S. Kim and S. Yang, "Tiny MAP: an efficient MAP in IEEE 802.16/WiMAX broadband wireless access systems," *Computer Communications*, vol. 30, no. 9, pp. 2122–2128, 2007.