

Performance Analysis of Best Effort Traffic in IEEE 802.16 Networks

Seungwoon Kim and Ikjun Yeom, *Member, IEEE*

Abstract—In IEEE 802.16 networks, a bandwidth request-grant mechanism is employed for reducing data collision and supporting various QoS requirements. In this paper, we investigate the effectiveness of the mechanism on Best-Effort (BE) traffic. We first derive two candidate schemes for bandwidth request based on a simple analysis. Then, we present a comprehensive study on the performance of BE traffic associated with those schemes through extensive simulations. In this study, we include a scheme proposed for bandwidth allocation without request. This study shows that IEEE 802.16 networks can be effectively managed through appropriate bandwidth request schemes. It is also shown that bandwidth allocation without request can be an alternative for increasing utilization. Finally, we show that the proposed schemes can be easily extended for non-real-time polling service without additional cost.

I. INTRODUCTION

Wireless network technologies have been evolved to provide the flexible access to the Internet while moving. Unlike wired networks which provide the fixed point of network attachment, wireless networks enable users to access the Internet from any place. In several years, there have been many attempts to replace the Internet access with WLAN and cellular networks. With increasing bandwidth, WLANs are successfully replacing wired networks in home and office environments. Due to the narrow coverage and lack of mobility support, however, they are not suitable for mobile users. Oppositely, cellular networks provide wider coverage and mobility support, and thus suitable for mobile users, but the communication cost and narrow bandwidth are the major obstacles to be widely deployed for the Internet access.

IEEE 802.16 WMAN technology [1] has been proposed to overcome the drawbacks of WLANs and cellular networks. It has been paid wide attention since it covers much larger area than WLAN while supporting high transmission rate. In addition, it provides various QoS scheduling for supporting heterogeneous traffic including legacy voice traffic, VoIP (Voice over IP), multimedia streams and the Internet data traffic. In IEEE 802.16 networks, a bandwidth request-grant mechanism is used to accommodate various QoS requirements of heterogeneous traffic. When a subscriber station (SS) wants to send data, it first needs to send a bandwidth request message to the corresponding base station (BS). Upon receiving the request, the BS grants an appropriate amount of bandwidth to the SS based on an uplink scheduling scheme. There are

four service classes defined based on their bandwidth request-grant mechanisms as follows: unsolicited grant service (UGS), real-time polling service (rtPS), non-real-time polling service (nrtPS), and best-effort (BE) service.

Among those four service classes, it is expected that the BE class will be the first one to be serviced due to the following practical reasons: Implementation cost for all the four services is too expensive while, even if the access network supports the QoS, end-to-end QoS is not supported eventually since the Internet does not provide any QoS scheme currently. *WiBro* service in South Korea [2], which is the first commercial service of IEEE 802.16e [3], supports BE class only at this time.

The objective of this paper is to present a comprehensive performance analysis of BE traffic in IEEE 802.16 networks. There have been several studies on performance of IEEE 802.16 networks, but they mostly focus on the physical layer and the scheduling framework in the MAC protocol. In [4], [5], impact of modulation and channel coding schemes on the performance of IEEE 802.16 networks has been analyzed. The IEEE 802.16 MAC protocol and scheduling have been analyzed in [6], [7]. In [6], a hierarchical scheduling framework for IEEE 802.16 has been proposed and evaluated. In the framework, BE connections are simply allocated an equal amount of bandwidth. In [7], recently, performance of IEEE 802.16 networks has been analyzed, especially on QoS support. In this study, Weighted Round Robin (WRR) scheduler in [8] has been employed for uplink scheduling.

In this paper, we focus on the performance of BE traffic associated with more specific bandwidth allocation schemes. Since BE traffic does not have any specific delay or bandwidth requirement, high utilization and fair bandwidth sharing are the major concerns of BE scheduling. We derive two bandwidth allocation schemes for BE traffic. They are practical and conforming the IEEE 802.16 standard. Through performance evaluation of them, we observe behaviors of BE traffic in IEEE 802.16 networks. In this study, we include a scheduling scheme, which has been proposed in our earlier work [9] for bandwidth allocation without request. Through comparison with this scheme, we observe the impact of the bandwidth request-grant mechanism on BE traffic. Then, we extend those schemes for nrtPS class, and show how nrtPS and BE traffic can be differentiated. To facilitate this study, we have implemented simulation modules working in ns-2 [10].¹

The study presented in this paper reveals the following significant observations: (a) performance of BE traffic in

This work was supported by SK Telecom Co.

S. Kim and I. Yeom are with the Department of Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, South Korea (e-mail: swkim@cmlab.kaist.ac.kr; yeom@cs.kaist.ac.kr).

¹Our simulation modules are available via <http://cmlab.kaist.ac.kr>.

IEEE 802.16 networks is seriously impacted by the request collision rate; (b) for given numbers of request slots and SSs, the collision rate can be effectively reduced; (c) bandwidth allocation without request can be an alternative for BE traffic, especially in a large scale IEEE 802.16 network; and (d) nrtPS class can be implemented as a simple extension of BE class without additional efforts or cost.

The rest of this paper is organized as follows: In Section II, we present background knowledge for understanding this paper. We summarize the standard for IEEE 802.16 and present the motivation of this paper. In Section III, we present bandwidth allocation schemes for BE traffic which we use for performance analysis of IEEE 802.16 networks. We also provide simple mathematical analysis for them. In Section IV, we present a comprehensive study of the proposed schemes through extensive ns-2 simulations. In Section V, we extend the schemes for nrtPS class, and observe how nrtPS class can be differentiated from BE class. In Section VI, we conclude this paper.

II. BACKGROUND AND MOTIVATION

In IEEE 802.16 networks, data transmission is divided into frames in the time domain, and each frame consists of downlink and uplink subframes, which are duplexed using either Frequency Division Duplexing (FDD) or Time Division Duplexing (TDD). A subframe is again divided into several slots for actual data transmission. Each frame begins with DL-MAP and UL-MAP in its downlink subframe. They contain allocation information of subsequent slots, and are used for each SS to access slots assigned to it.

To accommodate various QoS requirements of heterogeneous traffic, the MAC protocol of IEEE 802.16 networks is connection-oriented and employs strict admission control. To establish a new connection, an SS sends a connection request message containing its class and traffic specifications. The connection is accepted when there is enough resource. Then, it can send or receive its data through the allocated bandwidth. Bandwidth allocation is performed by the BS with appropriate scheduling schemes. Even though the standard does not provide any specific scheduling scheme, it is obvious that the downlink scheduling is much simpler than the uplink scheduling since the BS has all the information of downlink traffic. In [7], it has been shown that a simple Deficit Round Robin (DRR) scheduler [11] works well for downlink scheduling. In this paper, we focus on uplink scheduling.

For uplink scheduling, each SS sends a bandwidth request, and the correspondent BS allocates bandwidth to them based on their requests. The bandwidth request-grant mechanism is different for each service class as follows: (a) UGS (Unsolicited Grant Service) has the highest priority, and the target is legacy voice and VoIP traffic. Since they generate fixed length packets with a fixed period, they are not required to request bandwidth for each packet. Instead, the BS regularly assigns slots according to the traffic specification once the connection is accepted; (b) rtPS (real-time Polling Service) has the next priority, and is targeted for multimedia streams. Most multimedia streams consist of variable size of packets sent regularly,

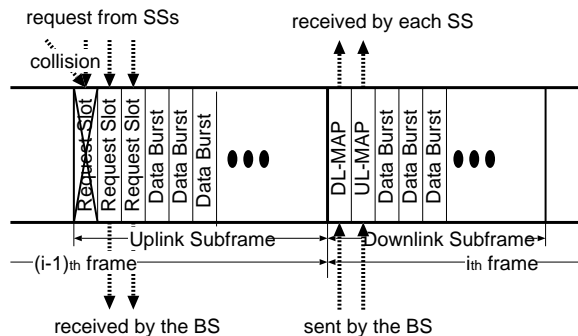


Fig. 1. Bandwidth request and grant for BE traffic

and the BS polls the connection periodically to ask how much bandwidth is needed; (c) nrtPS (non real-time Polling Service) is for better than Best-Effort service. Basically, it is the same as the BE class except that it may have additional bandwidth through non-periodic pollings; and (d) BE (Best-Effort) is allowed to use only contention-based request, that is, there are several shared slots for bandwidth request, and each BE connection contends for sending its request to the BS via the shared slots. Fig. 1 illustrates the bandwidth request and grant process for BE traffic.

Among the four service classes, UGS and rtPS have strict delay and bandwidth constraints. Therefore, scheduling of them is quite deterministic, and performance relies more on admission control rather than on scheduling. In [6], [7], it has been shown that existing real-time schedulers such as Earliest Deadline First (EDF) [12], Weighted Fair Queueing (WFQ) [13] and WRR [8] work well for UGS and rtPS. nrtPS and BE, however, do not have such constraint, and there are several choices in scheduling them. In this paper, we investigate the impact of scheduling schemes on BE traffic. nrtPS is a simple extension of BE, and we show that our study presented in this paper can be easily extended for that.

BE connections contend a limited number of shared slots for sending their requests, and if there is more than one SS attempting to send their requests via a slot, the requests are corrupted. When the number of request slots is too small to resolve the current number of connections, some of them may suffer from long delay and unfair bandwidth sharing due to repeated collisions. However, it may not be practical to arbitrarily increase the number of slots to reduce the request collision rate since uplink bandwidth for actual data transmission is also reduced.

Hence, it is necessary to manage the collision rate not too high with a given number of request slots. To reduce the collision rate, it might be effective to limit the number of requests attempted in a frame. On the other hand, frequent requests will help a connection respond the changes of network condition quickly and increase its throughput. In this paper, we look at the relation between the number of request slots and the collision rate, and the realized throughput as a result of the relation. We derive a simple analysis of them and propose a scheme to reduce the collision rate while maintaining high throughput.

Algorithm 1 Request Per Frame (RPF)

```

: Subscriber Station
1: for each frame do
2:   if  $q > 0$  then
3:     send  $q$  through a request slot randomly selected
4:   end if
5: end for

: Base Station
6: for each frame do
7:   send UL-MAP for this frame
8:   receive  $q$  from the  $i^{th}$  SS and update  $d[i]$  with it
9:   calculate  $b[i]$  using the Max-min Fair Scheduling with  $d[i]$  for
      $i = 0$  to  $n - 1$ 
10:   $d[i] = d[i] - b[i]$ 
11:  generate UL-MAP with  $b[i]$  for the next frame
12: end for

13:  $q$ : queue length of each SS
14:  $d[i]$ : demand of the  $i^{th}$  SS
15:  $b[i]$ : allocated bandwidth of the  $i^{th}$  SS

```

In this study, we include a scheme for bandwidth allocation without request, which was proposed in our earlier work in [9]. The motivation of this scheme has arisen from that BE traffic does not require any specific constraint, and thus the bandwidth request process might be unnecessary cost for scheduling BE traffic. In this scheme, to estimate the bandwidth demand without any explicit message for it, the BS measures the temporal sending rate of each connection and allocates bandwidth based on the measured sending rate. It has been shown that uplink bandwidth in IEEE 802.16 can be effectively allocated with the scheme. In this paper, we compare this scheme with the request-based schemes to investigate the impact of the bandwidth request process on the performance of BE traffic.

III. SCHEMES FOR BE UPLINK SCHEDULING

In this section, we present three bandwidth allocation schemes for BE uplink traffic in IEEE 802.16 networks. First, we propose two schemes based on the request-grant mechanism to conform the standard. Then, we present a scheme without the mechanism which was originally proposed in [9].

While we derive the schemes based on the request-grant mechanism, we assume that an SS is allowed to claim only the amount of bandwidth for packets which are already in its queue to avoid bandwidth wastage.²

A. A Primitive Scheme - Request per Frame

A simple and intuitive scheme is that each SS attempts to send its request in every frame as long as its queue is not empty. Then, the BS performs the max-min fair scheduling with the requests. We call this scheme as *Request Per Frame (RPF)*. The detailed procedure of *RPF* is presented in Algorithm 1.

²In this paper, we assume that an SS has one connection for simplicity. When multiple connections are in an SS, we treat the aggregation of them as a single connection.

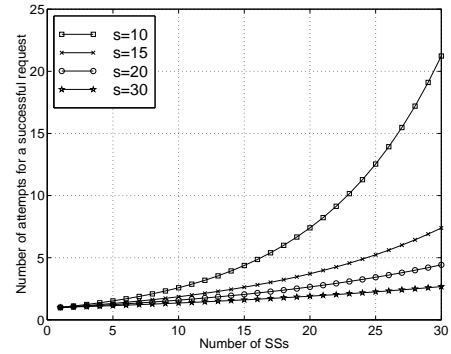


Fig. 2. The number of frames for a successful request in *RPF*

In *RPF*, each SS attempts to send its queue length as the bandwidth request through a request slot which is randomly selected. Here note that some of the requests may observe collision and not be delivered to the BS. The BS receives the requests transmitted without collision, and updates the demand of SSs with them in Line 8. Here also note that the demands of SSs observing collision are not changed. Then, the BS performs the max-min fair scheduling with the demands. Once some bandwidth is allocated to an SS, the demand of it is reduced by the allocated bandwidth in Line 10. Unless an SS succeeds to send its new request before receiving all the bandwidth in the previous request, it cannot be allocated bandwidth until sending the new request successfully. Any algorithm for the max-min fair scheduling such as in [15] can be applicable for the proposed scheduling, and we do not present it here.

This scheme will be effective when there is an enough number of slots for request to accommodate the current SSs. When the number of slots is not enough, however, the probability of request collision becomes high, and some of SSs may suffer from repeated collision of request. Suppose a network in which there are n number of connections, and the number of request slots is s . Then, the probability p_s that an SS successfully transmits its request in a frame is given by

$$p_s = \left(\frac{s-1}{s}\right)^{n-1} \quad (1)$$

Then, the expected number of frames, f , between two consecutive successful transmissions of request is calculated by

$$E(f) = \frac{1}{p_s} = \left(\frac{s}{s-1}\right)^{n-1} \quad (2)$$

The above equation shows that, with a given s , f exponentially increases as n increases. In Fig. 2, we present $E(f)$ with various n . It is observed that, when $n = 30$, each SS observes more than 20 collisions for a successful request with $s = 10$. As s increases, $E(f)$ decreases, but with $s = 20$ (which the two third of $n = 30$), five collisions still occur for a successful transmission.

Now, we apply the analysis of f for estimating the queue length in each SS. Suppose that the sending rate of a flow is λ , and it can transmit its request for every f frames. Then,

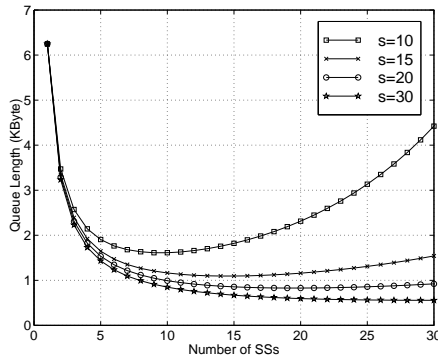


Fig. 3. Queue length in RPF

the amount of bandwidth requested for each request (which is the queue length), b , is given by

$$b = fl\lambda \quad (3)$$

where l is the frame length in time. In Fig. 3, we present the relation of queue length and the number of SSs with different s . In this figure, we assume that each flow equally shares the total bandwidth so that $\lambda = \frac{T}{n}$, where T is the total bandwidth. Then, (3) is rewritten as follows

$$b = \frac{lT}{n} \left(\frac{s}{s-1} \right)^{n-1} \quad (4)$$

In the figure, T and l are set to 10 Mbps and 5 msec., respectively. The queue length is reduced as more SSs share the link since the sending rate of each flow decreases. When $n > s$, however, the queue length increases as n increases since the probability of request collision drastically increases as shown in Fig. 2.

B. A Scheme for Reducing Collision - Request and Wait

In the above, we have shown that RPF is simple, but may cause collisions frequently when the number of request slots is not enough for the current number of SSs. In this section, we present how to reduce collision with a given number of request slots.

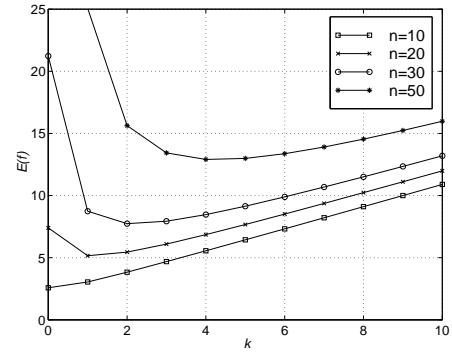
To reduce collision, the number of attempts to send a request in a frame should be limited. Instead of attempting to send the request in every frame, suppose that an SS waits m frames without sending the request after a successful transmission of a request.³ For a successful transmission, it takes $\frac{1}{p_s}$ trials in average. Then, the probability p_r that an SS attempts to send its request in a frame is given by

$$p_r = \frac{\frac{1}{p_s}}{\frac{1}{p_s} + m} = \frac{1}{1 + p_s m} \quad (5)$$

Then, the number of SSs requesting in a frame, n_r , is given by

$$n_r = p_r n = \frac{n}{1 + p_s m} \quad (6)$$

³In most packet networks such as Ethernet and WLAN, exponential backoff after collision is employed for reducing the collision rate. It may be effective when the number of hosts and the amount of traffic are not predictable. In IEEE 802.16 networks, however, the information on them is available with the BS, and a proactive approach can be more effective.


 Fig. 4. $E(f)$ with various k where $k = p_s m$

Eq. (1) is rewritten with (6) as follows,

$$p_s = \left(\frac{s-1}{s} \right)^{n_r-1} = \left(\frac{s-1}{s} \right)^{\frac{n}{1+p_s m}-1} \quad (7)$$

Eq. (7) can be solved using Lambert W function in [14], but it is too complicated and beyond the scope of this paper. To make it simple, we define m as follows

$$m = \frac{1}{p_s} k \quad (8)$$

The above equation means that an SS waits m frames for sending the next request, and m is proportional to the number of trials for a successful transmission. Now, (7) is simplified with k as follows

$$p_s = \left(\frac{s-1}{s} \right)^{\frac{n}{1+k}-1} \quad (9)$$

The expected number of frames between two consecutive successful requests is the sum of $\frac{1}{p_s}$ and m , and expressed by

$$E(f) = \frac{1}{p_s} + m = (k+1) \left(\frac{s}{s-1} \right)^{\frac{n}{1+k}-1} \quad (10)$$

In Fig. 4, we present the relation of k and f with various n . Here s is fixed as 10. Note that $k = 0$ corresponds to Fig. 2. It is observed that we can reduce f with appropriate k . When $n = 30$, it takes more than 20 frames for a successful transmission of a new request when each SS attempts to send its request for every frame ($k = 0$). With $k = 2$, it is reduced to eight frames.

To find the optimal k for minimizing f , from (10), we have

$$\frac{dg}{dk} = \left(\frac{s}{s-1} \right)^{\frac{n}{k+1}-1} \left(1 - \frac{n}{k+1} \ln \frac{s}{s-1} \right) \quad (11)$$

where $g(k) = E(f)$. Then, k for $g' = 0$ is calculated by

$$k = n \ln \frac{s}{s-1} - 1 \quad (12)$$

Since k cannot be negative, we rewrite (12) as follows

$$k = \begin{cases} n \ln \frac{s}{s-1} - 1 & \text{if } s \leq \frac{e^{1/n}}{e^{1/n} - 1} \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

Algorithm 2 Request and Wait (RW)

```

: Subscriber Station
1: for each frame do
2:   if wait > 0 then
3:     wait = wait - 1
4:   else
5:     if the previous request is successfully transmitted then
6:       wait = try × k
7:       try = 0
8:     else
9:       send q through a request slot randomly selected
10:      try = try + 1
11:    end if
12:  end if
13: end for
14: q: queue length
15: try: the number of attempts to send a request
16: wait: the number of frames waited for sending the next request
17: k: refer to Eq. (13)

```

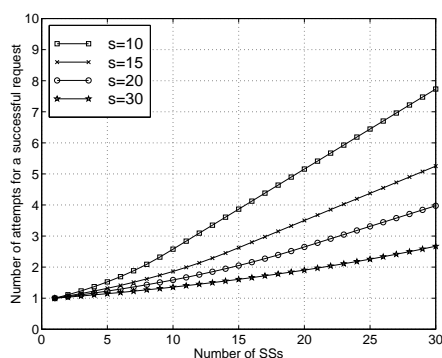


Fig. 5. The number of frames between two consecutive successful requests in RW

Using (13), we can find k for minimizing f with given n and s . Applying (13) to (10), we have

$$E(f) = \begin{cases} n \left(\ln \frac{s}{s-1} \right) \left(\frac{s}{s-1} \right)^{\frac{1}{\ln \frac{s}{s-1}} - 1} & \text{if } s \leq \frac{e^{1/n}}{e^{1/n} - 1} \\ \left(\frac{s}{s-1} \right)^{n-1} & \text{otherwise} \end{cases} \quad (14)$$

Eq. (14) reveals that, when s is enough to resolve n , RPF performs well, but when s is not enough, less than $\frac{e^{1/n}}{e^{1/n} - 1}$ precisely, f can be reduced by waiting m frames without sending a request. Based on the observations above, we derive a new scheme, called *Request and Wait (RW)*, for reducing collision and minimizing f . In Algorithm 2, we present an algorithm for RW. Here note that we do not present the operation of the BS since it is the same as in RPF except that it needs to inform each SS whether its previous request is successfully transmitted or not. It can be easily implemented with UL-MAP.

In Fig. 5, we present the relation of n and f in RW. It is observed that RW drastically reduces f when $s = 10$ compared to Fig. 2. The queue length of RW is given by

$$b = \begin{cases} lT \left(\ln \frac{s}{s-1} \right) \left(\frac{s}{s-1} \right)^{\frac{1}{\ln \frac{s}{s-1}} - 1} & \text{if } s \leq \frac{e^{1/n}}{e^{1/n} - 1} \\ \frac{lT}{n} \left(\frac{s}{s-1} \right)^{n-1} & \text{otherwise} \end{cases} \quad (15)$$

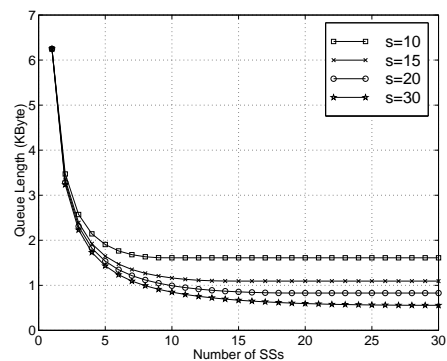


Fig. 6. Queue length in RW

We can observe that the queue length in RW is independent with n for $n \geq 1/\ln \frac{s}{s-1}$. Fig. 6 shows the queue length of RW. It is shown that RW stabilizes the queue length in its minimum in RPF .

C. A Scheme without Bandwidth Request

In the above, we have presented and analyzed two scheduling schemes based on bandwidth request. Now, we present a scheme without the bandwidth request, called *Grant Without Request (GWR)*. This scheme has been initially proposed in [9]. In this paper, we compare GWR with RPF and RW to observe the impact of the bandwidth request-grant mechanism on BE traffic.

The motivation of GWR is that the bandwidth request process can be redundant for BE TCP traffic since (a) it consumes additional uplink bandwidth for request. As we observe in the previous section, the amount of bandwidth for request slots should be also increased as the number of SSs increases; and (b) it is hard for each SS to estimate the amount of bandwidth requested for a TCP flow due to dynamic changes of its sending rate. In the previous schemes, we simply set the amount as the queue length. This conservative request is effective to increase utilization, but it is also observed that it requires a certain amount of packets in the queue.

GWR does not require any information from SSs for bandwidth allocation. Instead, it measures the sending rate of the current BE connections, and allocates bandwidth based on the measured sending rate. In Algorithm 3, we present a simplified scheduling algorithm for GWR.

GWR consists of two procedures, bandwidth change detection (Line 1-9) and bandwidth adjustment (Line 10-18). For detecting bandwidth changes, it measures the short-term sending rate of each flow and maintains the maximum and the minimum values of it. If the current short-term sending rate is detected to be out of the range between the minimum and the maximum values, bandwidth adjustment is triggered in Line 2-5. Otherwise, the current allocation is maintained. To resize the range, the maximum and the minimum values are periodically reset in Line 6-8.

Bandwidth adjustment consists of demand estimation and the max-min fair scheduling. In GWR, the demand of a flow is defined as the amount of access link (here IEEE 802.16

Algorithm 3 Grant Without Request (GWR)

```

: Upon receiving a packet from  $i^{th}$  flow
1: update  $s[i]$  and  $l[i]$ 
2: if  $s[i] > max[i]$  or  $< min[i]$  then
3:   update  $max[i]$  or  $min[i]$  with  $s[i]$ 
4:    $last\_update = now$ 
5:   do Demand Estimation and Max-min Fair Scheduling
6: else if  $now > last\_update + timeout$  then
7:    $max[i] = min[i] = s[i]$ 
8:   do Demand Estimation and Max-min Fair Scheduling
9: end if

: Demand Estimation
10: for  $i = 1$  to  $n$  do
11:   if  $l[i] \geq B/n$  then
12:      $d[i] = l[i]/\alpha$ 
13:   else if  $l[i] \geq \beta b[i]$  then
14:      $d[i] = l[i]/\delta$ 
15:   else
16:      $d[i] = l[i]/\alpha$ 
17:   end if
18: end for

19:  $s[i]$  and  $l[i]$ : short and long-term sending rates of the  $i^{th}$  flow
20:  $d[i]$  and  $b[i]$ : demand and allocated bandwidth of the  $i^{th}$  flow
21:  $B$ : the total amount of bandwidth for allocation
22:  $\alpha$ ,  $\beta$  and  $\delta$ : increasing rates,  $\delta < \alpha < \beta < 1$ 
23:  $n$ : the number of flows for allocation

```

link) bandwidth for achieving its maximum throughput so as not to be limited by the access link bandwidth. In the demand estimation procedure, note that we use the long-term sending rate rather than the short-term sending rate to avoid frequent fluctuations. It is simple to estimate the demand of a flow when the sending rate is measured less than the allocated bandwidth. In this case, the flow observes external congestion or bottleneck links out of the access link, and thus the demand is simply equal to the sending rate.

When the sending rate is equal to the allocated bandwidth, however, it is not straightforward to estimate the demand of the flow from its sending rate since it is hard to distinguish the following two cases: (a) the maximum throughput of the flow is equal to the amount of the current allocated bandwidth and is already achieved; or (b) the access link is the bottleneck of the path currently, and the sending rate is limited by the amount of the current allocated bandwidth.

To distinguish the two cases, the amount of allocated bandwidth is maintained to be slightly higher ($1/\alpha$ where $\alpha < 1$ in the algorithm) than the current sending rate. Then, we can expect that the sending rate will be maintained stably in case of (a) whereas it will increase to reach the maximum in case of (b). As a result, the proposed scheme can estimate the demand of each flow as either the current sending rate when it is less than the allocated bandwidth or an amount of bandwidth higher than the current allocated bandwidth when the flow fully utilizes the current bandwidth. In the latter case, it is hard to estimate the exact amount of it. *GWR* adaptively increases the bandwidth until the sending rate becomes stable.

As discussed in the above, the demand of the i^{th} flow, $d[i]$ is basically set to be $1/\alpha$ times of the current sending rate $l[i]$. In Line 11-12, the flows with $l[i] \geq B/n$ already achieve the equal share, and we simply apply the basic setting. Note that

since we perform the max-min fair scheduling after demand estimation, demand more than the equal share does not impact on the lower demand of other flows.

In Line 13, for flows with $l[i] < B/n$, we check if a flow is increasing its sending rate. In the scheme, since $l[i]$ is expected to be around $\alpha b[i]$ normally, we consider that the sending rate of the flow is increasing when $l[i] \geq \beta b[i]$ where β is set to be greater than α to ignore short-term variations. Then, the demand of the flow is set to be $1/\delta$ times of $l[i]$, which is larger than the basic setting with $\delta < \alpha$ in Line 14. For the flows observing external congestion, even though they do not achieve the equal share, we apply the basic setting to avoid bandwidth wastage.

The original algorithm for *GWR* in [9] employs several additional techniques for dealing with dynamic changes of TCP's sending rate and for fast recovery of the flows sharing less bandwidth than the equal share. However, those techniques are beyond the scope of this paper, and we do not present here. Once we get the demand for each flow, the max-min fair scheduling is performed based on the demand.

IV. PERFORMANCE ANALYSIS

In this section, we present performance analysis of the three bandwidth allocation schemes through ns-2 simulation. To facilitate the following study, we have implemented simulation modules for IEEE 802.16 networks. In the simulation modules, we simply use the two ray ground model for the physical propagation model. This model is generally used for wireless LAN simulation, and may not fit for IEEE 802.16 networks. However, this study is mainly focused on bandwidth allocation in the MAC layer, and we believe that this model is enough for the purpose.

Based on the standard of IEEE 802.16, downlink and uplink subframes are duplexed by TDD, and each frame is multiplexed by TDM. Actual bandwidth allocation to each SS is performed by assigning mini-slots in the time domain. The length of a mini-slot is two bytes. The MAC header length of IEEE 802.16 is six bytes, and one byte is added when the packet is fragmented. Each downlink subframe begins with DL and UL-MAP for the rest part of the frame, and the MAP size is varied by the number of concurrent connections. Each uplink subframe begins with several request slots for BE connections, and the slot size is 16 bytes. In the rest of this section, unless otherwise stated, the capacity of IEEE 802.16 link is set to 20 Mbps, and the link is equally shared by down and up links. The frame length is set to 5 msec.

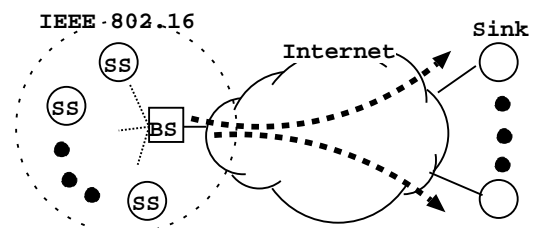
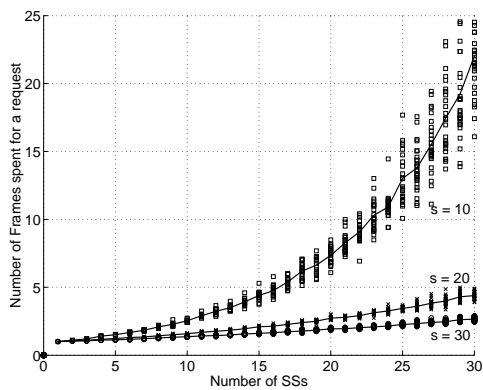
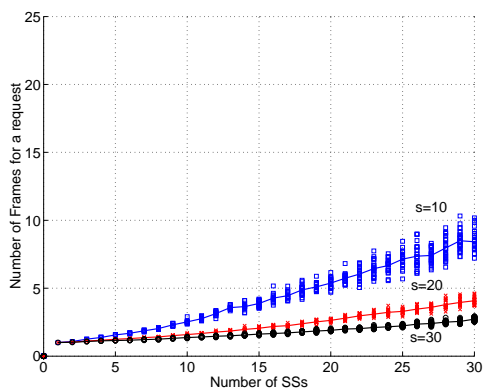


Fig. 7. Simulation topology



(a) RPF



(b) RW

Fig. 8. The number of frames between two consecutive successful requests

Fig. 7 shows the simulation topology. A set of SSs is connected to an IEEE 802.16 BS, and the BS is connected to wired networks. Each SS has one TCP source, and each TCP source is connected to the corresponding sink via the BS and wired links. We change the configuration of wired links to make different network conditions.

First, we observe the impact of the number of request slots on the successful request in RPF and RW. Here note that we exclude GWR since it does not need any bandwidth request. To observe the relation between the number of slots and the number of SSs, we increase the number of SSs from one to thirty in every five second. We perform three runs of simulation for each scheme with different number of slots. To make the wireless link utilized fully, the wired link bandwidth is set to be large enough so that the flows do not observe congestion in wired links.

In Fig. 8, we present the number of frames between two consecutive successful requests (f), which is the direct inverse of the probability of a successful request in RPF (refer to (2)), and includes the number of frames intentionally waited for the next request, m as in (8), in RW (refer to (10)). In the figure, each marker represents the average value of each SS for a given interval, which is five seconds in this scenario. Note

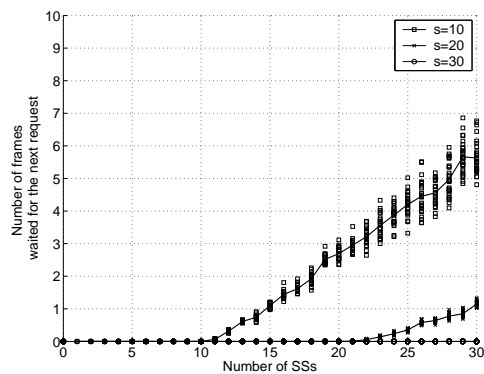


Fig. 9. The number of frames intentionally waited for the next request in RW

that the number of markers increases as more SSs join the network. Lines represent the average values of the markers. It is observed that f increases as the number of SSs increases for a given number of request slots, and the results are identical with our analysis presented in Fig. 2 and 5.

As expected, with RW, f is drastically reduced when $s = 10$. In Fig. 9, we show the changes of m as the number of SSs increases with a given number of request slots. When $s = 10$, m increases after more than ten SSs join the network. This results confirm our analysis in the previous section that appropriate regulation of the number of requests can be effective to reduce f when the number of SSs is much greater than the number of request slots. Here note that when $s = 30$, m is maintained to be zero, which means that RW works the same as RPF. In the rest part of this section, s in RW is configured much less than the number of SSs.

In Fig. 10, we present microscopic views of queue length, bandwidth request and allocated bandwidth. We randomly select an SS in the previous simulation and monitor them in each frame. In the figure, a circle represents the queue length. When a packet arrives the queue, the length increases by the packet length, which is 1 KByte in this simulation. In Fig. 10(a), for example, there are three packets arrived at around 21.28, 21.36 and 21.4 second, respectively. Thick lines show bandwidth requests. In Fig. 10(a), there are three successful requests at around 21.28, 21.32 and 21.44 second, and the amounts of requesting bandwidth in each request are around 3, 1.5 and 2.3 Kbyte, respectively. Squares represent the amount of bandwidth allocated in each frame. Once a request is successfully delivered to the BS, the BS allocates a certain amount of bandwidth in each frame to the SS until the SS gets the entire amount of requesting bandwidth. Here note that the amount of bandwidth allocated in each frame is determined by the max-min fair scheduling with other flows' requests. In Fig. 10(a), at around 21.44 second, a request is succeeded, and the SS is allocated around 250 byte bandwidth in each frame. As the SS gets more bandwidth, the queue length decreases.

In Fig. 10(a), a packet arrives the empty queue at 21.36 second. Then, the SS keeps attempting to send its request, but fails until 21.44 second. Meanwhile, one more packet arrives at 21.4 second, and the queue length increases more than 2

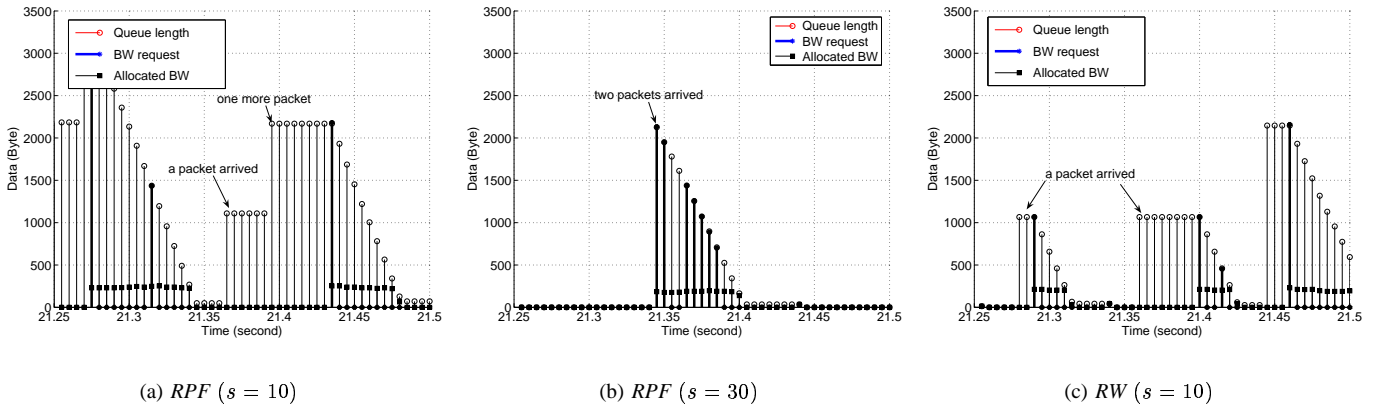


Fig. 10. Queue length, request and allocation

KByte. In Fig. 10(b), immediately after two packets arrival, the SS successfully requests the bandwidth, and starts sending the packets. As a consequence, the queue length is maintained smaller than in Fig. 10(a). In Fig. 10(c), *RW* is observed to perform better than *RPF* with the same number of request slots.

Now, we look at the impact of the probability of successful request on the queue length in each SS. In an IEEE 802.16 link, the link layer queue length of an SS is closely related to the sending rate. When the access link is the bottleneck of a flow, the queue length in the SS might be high, and vice versa. To make flows observe different levels of congestion in the access link, we divide 30 flows into three groups and assign different wired link bandwidth to each group. The total uplink bandwidth is 10 Mbps as described in the above, and some of bandwidth is consumed for transmitting control data including the MAC header, DL/UL-MAPs, and bandwidth requests. Through a number of simulations, we have observed that the amount of data bandwidth is around 8.5 Mbps when 30 SSs are in a 10 Mbps link. Then, the equal share of aggregated throughput of each group is around 2.8 Mbps. For each group to observe the different level of congestion in the access link, we set the wired link bandwidth as 2, 3 and 5 Mbps for each group, respectively. Then, throughput of the first group is not limited by the access link since the wired link bandwidth (2 Mbps) is much smaller than the access link (2.8 Mbps). The second group observes the similar bandwidth in both the access and the wired link. The third group observes the bottleneck in the access link.

In Fig. 11, we present queue length and throughput with three allocation schemes. In the figure, Flow 1~10 (Group A), 11~20 (Group B), and 21~30 (Group C) are connected through 2, 3, and 5 Mbps wired links, respectively. We measure the link layer queue length and throughput in each SS, and represent them by individual markers in the figure. A horizontal line represents the average value of each group. In all the schemes, it is observed that the queue length of Group A is less than that of other groups since the access link is not the bottleneck of Group A. As we have analyzed in the previous section, in Fig. 11(a), it is observed that the queue

length can be reduced with an enough number of request slots when the access link bandwidth is not the bottleneck since the SS can request its bandwidth more frequently. Flows in Group B, however, observe higher queue length when $s = 30$, which is the opposite result of Group A. It is due to that more bandwidth is consumed for requests when $s = 30$, and the access link becomes the bottleneck. It is confirmed through Fig. 11(d). Throughput of Group B and C is the same when $s = 30$ while throughput of Group C is higher than that of Group B when $s = 10$. As a result, the queue length of the second and the third groups becomes the same when $s = 30$.

In Fig. 11(b) and 11(e), we present the results with *RW* when $s = 10$. Here note that we do not present the results when $s = 30$ since the results are the same as in *RPF* when $s = 30$. As shown in Fig. 8, *RW* with $s = 10$ is expected to allow more frequent requests than *RPF* with $s = 10$ while maintaining the same data bandwidth. Consequently, it is observed that (a) the queue length of Group A in *RW* is much less than that in *RPF* with the same S ; (b) the queue length of Group B in *RW* is equal to that in *RPF* with $s = 10$; and (c) aggregated throughput of Group A and B in *RW* is more close to 2 and 3 Mbps (which are the wired link bandwidth) than that in *RPF* since flows in *RW* can respond the changes of network condition more quickly through more frequent requests.

In Fig. 11(c) and 11(f), we present the results with *GWR*. We perform three runs of simulation with different values of α . α is used to allocate more bandwidth than the current sending rate so as to allow a flow to increase its sending rate when it is possible. The sending rate is maintained to be α of the allocated bandwidth. As a result, with a smaller α , a flow may increase its sending rate more quickly while the link utilization is reduced. β in each simulation is simply set to $\alpha + 0.5$. A small α in *GWR* has the same impact of a larger s in *RPF* since both help a flow respond the network change quickly with reduced utilization, and vice versa. We can confirm this from the results. In TABLE I, we summarize the link utilization in each scheme with different parameter settings.

Finally, we observe the achieved throughput of TCP flows in an IEEE 802.16 access link when they are competing a bottleneck link with wired TCP flows. In the topology, 30

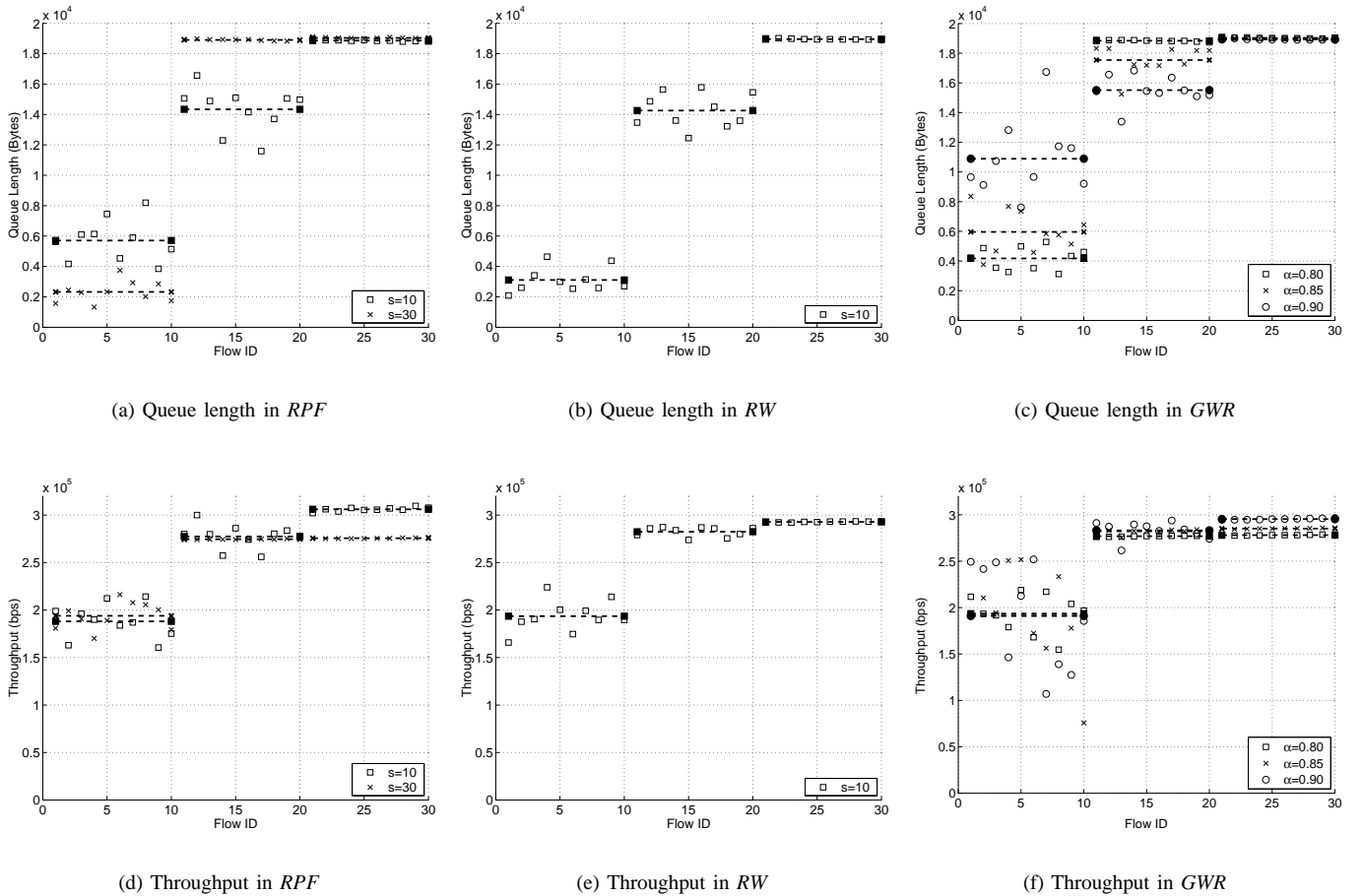


Fig. 11. Queue length and throughput

TABLE I
COMPARISON OF UTILIZATION

Scheme	<i>RPF</i>		<i>RW</i>		<i>GWR</i>	
Utilization (Mbps)	$s = 10$	8.50	$s = 10$	8.50	$\alpha = 0.90$	8.53
	$s = 30$	8.25			$\alpha = 0.85$	8.42
					$\alpha = 0.80$	8.30

flows share an IEEE 802.16 BS, and 20 flows among them are competing a 10 Mbps wired link with wired background TCP flows. Results are presented in Fig. 12. In Fig. 12(a), it is shown that flows with *RPF* ($s = 10$) achieve 0.16 Mbps in average, and this is only 70% of throughput with *RPF* ($s = 30$). Through these results, we can observe that frequent requests are important to increase TCP throughput. It is also observed that *RW* achieves better throughput than *RPF* with the same number of request slots. In Fig. 11(f), it is observed that throughput with *GWR* is similar to that with *RPF* or *RW*, and can be controlled with α and β .

V. EXTENSION TO NRTPS CLASS

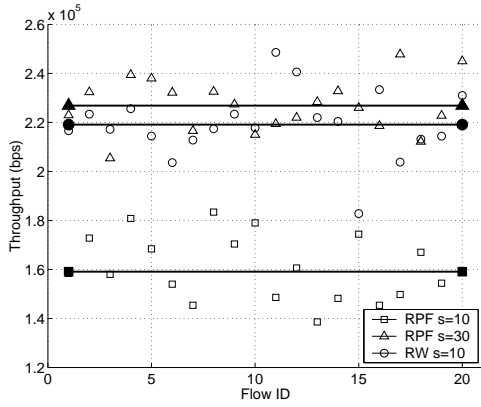
In the IEEE 802.16 standard, nrtPS class is defined as better than BE class. Both are supposed to deliver TCP traffic

without any delay or bandwidth constraint, and allowed to request bandwidth through the shared slots. In addition to the contention-based request, nrtPS class may get additional bandwidth through non-periodic individual pollings. In the previous sections, we have proposed bandwidth allocation schemes for BE class and evaluated them in various network topologies. In this section, we extend them for nrtPS class.

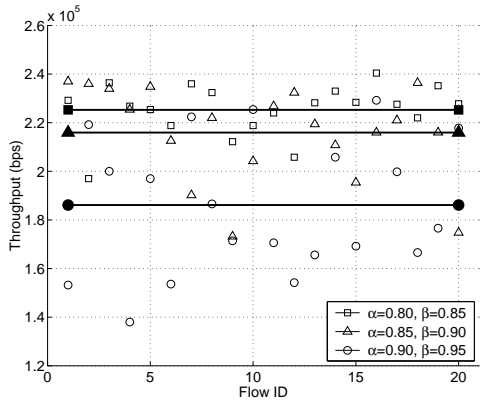
A. Extension of Request-based Bandwidth Allocation

Earlier in this paper, we have proposed two bandwidth allocation schemes, *RPF* and *RW*, based on the request-grant mechanism. Through extensive simulations, it has been observed that *RW* is effective to handle a large number of SSs with a limited number of request slots. In this section, we propose a bandwidth allocation scheme for both nrtPS and BE, called *RWn* (*RW with nrtPS*), which is an extension of *RW* to include nrtPS.

In *RW*, after a successful request, a SS needs to wait m slots to send the next request. In *RWn*, we apply different m for nrtPS and BE to provide differentiation between them. Suppose that there are n number of connections, and $n = n_n + n_b$, where n_n and n_b are the numbers of nrtPS and BE connections, respectively. Then, (5) is extended for nrtPS as



(a) RPF and RW



(b) GWR

Fig. 12. Throughput with background TCP traffic

follows

$$p_r = \begin{cases} \frac{\frac{1}{p_s}}{\frac{1}{p_s} + m_n} = \frac{1}{1 + p_s m_n} & \text{for nrtPS} \\ \frac{\frac{1}{p_s}}{\frac{1}{p_s} + m_b} = \frac{1}{1 + p_s m_b} & \text{for BE} \end{cases} \quad (16)$$

where m_n and m_b stand for m for nrtPS and BE. Then, the number of SSs requesting in a frame, n_r , is given by

$$n_r = p_r n = \frac{n_n}{1 + p_s m_n} + \frac{n_b}{1 + p_s m_b} \quad (17)$$

We also rewrite (8) as follows

$$m_n = \frac{1}{p_s} k_n, \quad m_b = \frac{1}{p_s} k_b \quad (18)$$

Then, (9) is rewritten by

$$p_s = \left(\frac{s-1}{s} \right)^{\frac{n_n}{1+k_n} + \frac{n_b}{1+k_b} - 1} \quad (19)$$

The number of frames between two consecutive successful requests is the sum of $\frac{1}{p_s}$ and m_n or m_b for nrtPS and BE, respectively, and expressed by

$$f_n = \frac{1}{p_s} + m_n = (k_n + 1) \left(\frac{s}{s-1} \right)^{\frac{n_n}{1+k_n} + \frac{n_b}{1+k_b} - 1} \quad (20)$$

$$f_b = \frac{1}{p_s} + m_b = (k_b + 1) \left(\frac{s}{s-1} \right)^{\frac{n_n}{1+k_n} + \frac{n_b}{1+k_b} - 1} \quad (21)$$

Now, we define the differentiation ratio γ as follows

$$\gamma = \frac{f_b}{f_n} > 1 \quad (22)$$

Then, using (22), $E(f)$ is calculated by

$$\begin{aligned} E(f) &= \frac{n_n f_n + n_b f_b}{n} \\ &= \frac{1}{n} (n_n + \gamma n_b) (1 + k_n) \left(\frac{s}{s-1} \right)^{\frac{n_n + n_b/\gamma}{1+k_n} - 1} \end{aligned} \quad (23)$$

To find the optimal k_n for minimizing $E(f)$, with $g(k_n) = E(f)$, we have

$$\begin{aligned} \frac{dg}{dk_n} &= \left(\frac{n_n + \gamma n_b}{n} \right) \left(\frac{s}{s-1} \right)^{\frac{n_n + n_b/\gamma}{1+k_n} - 1} \times \\ &\quad \left(1 - \left(n_n + \frac{n_b}{\gamma} \right) \left(\frac{1}{1+k_n} \right) \ln \frac{s}{s-1} \right) \end{aligned} \quad (24)$$

Then, k_n for $g' = 0$ is calculated by

$$k_n = \left(n_n + \frac{n_b}{\gamma} \right) \ln \frac{s}{s-1} - 1 \quad (25)$$

Since k_n cannot be negative, we rewrite (25) as follows

$$k_n = \begin{cases} \left(n_n + \frac{n_b}{\gamma} \right) \ln \frac{s}{s-1} - 1 & \text{if } s \leq \frac{e^{\gamma/(n_n + n_b)}}{e^{\gamma/(n_n + n_b)} - 1} \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

From (22) and (26), k_b is calculated by

$$k_b = \begin{cases} (\gamma n_n + n_b) \ln \frac{s}{s-1} - 1 & \text{if } s \leq \frac{e^{1/(\gamma n_n + n_b)}}{e^{1/(\gamma n_n + n_b)} - 1} \\ 0 & \text{otherwise} \end{cases} \quad (27)$$

In Fig. 13, we present $E(f)$ with various γ and n_n . First, we calculate the optimal k_n and k_b using (26) and (27) at a given set of $\{n, n_n, n_b, \gamma\}$. Then, we use this k_n and k_b to calculate (20), (21) and (23). In Fig. 13(a), we vary n_n from 0 to 30 with fixed $n = 30$. When $\gamma = 1$, there is no differentiation between nrtPS and BE, and $E(f)$ is constant. When $\gamma > 1$, $E(f)$ is maximized when $n_n = n_b$ to differentiate nrtPS and BE. Due to the same reason, $E(f)$ increases as γ increases. Note that $E(f)$ s with $n_n = 0$ and $n_n = 30$ are the same since all the connections belong to the same class. In Fig. 13(b), differentiation with different γ is presented. As n_n increases, both f_n and f_b increase, and the ratio of them is maintained. The algorithm for RWn is the exactly same as Algorithm 2 except using (26) and (27) for calculating k in Line 6.

B. Extension of Bandwidth Allocation Without Request

nrtPS can be implemented with GWR more simply than with RW. It has been observed that throughput in GWR can be differentiated by α in Algorithm 3. α is the ratio of the additional bandwidth in the allocated bandwidth, and defined as l_i/b_i where l_i and b_i are the sending rate and the allocated bandwidth of the i^{th} connection. We maintain α is less than one so that the sending rate of a flow is not limited by the access link and may have chance to increase its sending rate.

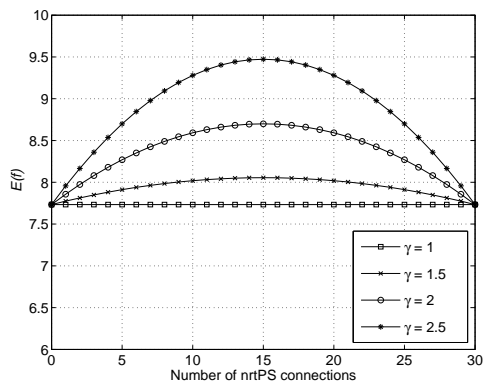
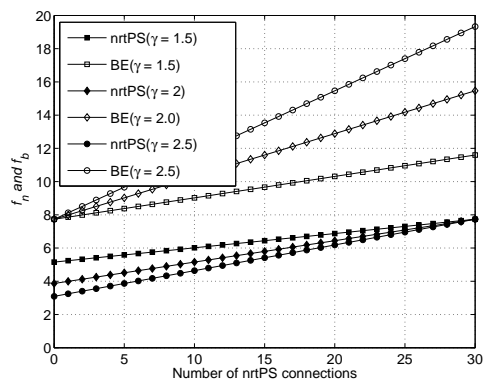
(a) $E(f)$ with various n_n when $n = 30$ (b) f_n and f_b with $n = 30$

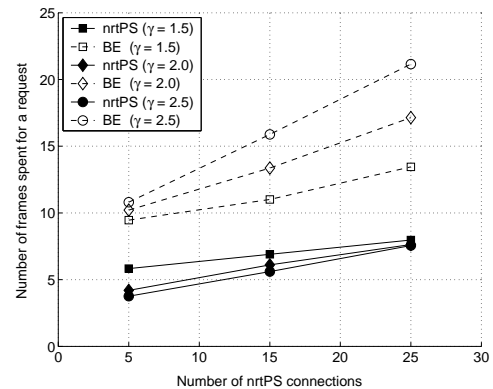
Fig. 13. Number of frames between two consecutive requests with nrtPS and BE

It has been shown that with a smaller α , a flow realizes higher throughput since it has more room for increasing sending rate. Using this property, *GWRn*, extension of *GWR* for nrtPS, can be simply implemented by applying different α to nrtPS and BE. In the rest of section, we observe the differentiation between nrtPS and BE with various parameter settings.

C. Performance Evaluation

In this section, we present performance evaluation of *RWn* and *GWRn*. First, we observe the differentiation between nrtPS and BE in the request rate with *RWn*. In *RWn*, we differentiate the number of frames between two consecutive successful requests for nrtPS and BE with a ratio γ . As a consequence, nrtPS can send its request more frequently than BE does.

In the first set of simulation, we vary the number of nrtPS connections and γ with a given number of SSs ($n = 30$). The results are presented in Fig. 14. We can confirm our analysis presented in Fig. 13(b) as follows: (a) as the number of nrtPS connections increases, both nrtPS and BE need more frames between two consecutive requests; (b) as γ increases, the differentiation between nrtPS and BE also increases; and (c) the differentiation is maintained with various numbers of

Fig. 14. f_n and f_b with various n_n

nrtPS connections.

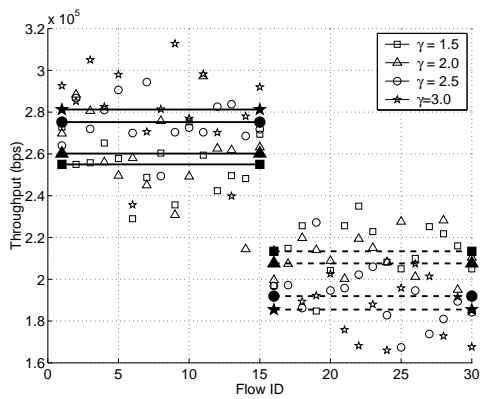
In Fig. 15, we present TCP throughput of nrtPS and BE. The numbers of nrtPS and BE connections are the same as 15. In this simulation, nrtPS and BE connections are competing a 7 Mbps bottleneck link. In Fig. 15(a), we vary γ and observe the impact of γ on the throughput differentiation. It is shown that differentiation in the achieved throughput between nrtPS and BE increases as we increase γ . Similarly, in Fig. 15(b), it is observed that we can provide differentiated throughput to nrtPS and BE with *GWRn*, and the differentiation ratio can be controlled by knobbing α .

VI. CONCLUSION

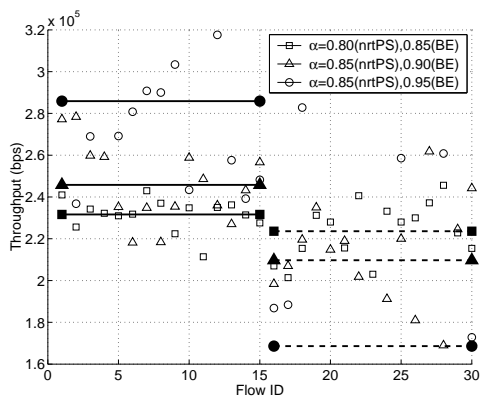
In this paper, we have presented a comprehensive performance analysis of BE traffic in IEEE 802.16 networks. First, we have derived two request-based bandwidth allocation schemes, *RPF* and *RW*. Both are simple and conform the IEEE 802.16 standard. Through a simple analysis, it has been shown that the collision rate of bandwidth request can be drastically reduced by limiting the number of requests with *RW*. Through extensive simulation, we have evaluated those schemes and confirmed the analysis results. We also have compared them with a scheme for bandwidth allocation without request. The results have shown that bandwidth allocation without request can be an alternative for large scale IEEE 802.16 networks since it does not need any additional bandwidth for request. Finally, it has been shown that the proposed schemes in this paper can be easily extended for providing service differentiation between nrtPS and BE traffic.

REFERENCES

- [1] IEEE 802.16-2004, "IEEE Standard for Local and Metropolitan Area Networks-Part 16: Air Interface for Fixed Broadband Wireless Access Systems," Oct. 2004.
- [2] "WiBro: Wireless Broadband," Available via <http://www.wibro.or.kr>.
- [3] IEEE 802.16e Task Group, "Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands," IEEE Std 802.16e-2005, Feb. 2006.
- [4] A. Ghosh et al., "Broadband Wireless Access with WiMax/802.16: Current Performance Benchmarks and Future Potential," *IEEE Communications Magazine*, Feb. 2005, pp. 129-136.
- [5] C. Hoymann, "Analysis and Performance Evaluation of the OFDM-based Metropolitan Area Network IEEE 802.16," *Computer Networks*, June 2005, pp. 341-363.



(a) RWn



(b) GWRn

Fig. 15. Throughput differentiation

[6] K. Wongthavarawat and A. Ganz, "Packet Scheduling for QoS Support in IEEE 802.16 Broadband Wireless Access Systems," *International Journal of Communication Systems*, vol. 16, 2003, pp. 81-96.

[7] C. Cicconetti et al., "Quality of Service Support in IEEE 802.16 Networks," *IEEE Networks*, Mar./Apr. 2006, pp. 50-55.

[8] M. Katevenis et al., "Weighted Round-Robin Cell Multiplexing in a General-Purpose ATM Switch Chip," *IEEE JSAC*, vol. 9, np. 8, Oct. 1991, pp. 1265-79.

[9] S. Kim and I. Yeom, "TCP-aware Uplink Scheduling in IEEE 802.16," accepted to publish in *IEEE Communications Letters*, Nov. 2006.

[10] L. Breslau et al., "Advances in Network Simulation," *IEEE Computer*, vol. 33, no. 5, May 2000, pp. 59-67.

[11] M. Shreedhar and G. Varghese, "Efficient Fair Queueing using Deficit Round Robin," *IEEE/ACM Trans. on Networking*, vol. 4, no. 3, June 1996, pp. 375-385.

[12] L. Georgiadis et al., "Optimal Multiplexing on a Single Link: Delay and Buffer Requirements," *IEEE INFOCOM*, 1994.

[13] A. Demers, "Analysis and Simulation of a Fair Queueing Algorithm," *ACM SIGCOMM Computer Communications Review*, 1989.

[14] R. Corless, "On the Lambert W Function," *Advanced Computational Maths*, vol. 5, 1996, pp. 329-359.

[15] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Englewood Cliffs, New Jersey, 1992.

[16] D. Clark and W. Fang, "Explicit Allocation of Best Effort Packet Delivery Service," *IEEE/ACM Trans. on Networking*, vol. 6, no. 4, Aug. 1998, pp. 362-373.