On TCP-Aware Uplink Scheduling in IEEE 802.16 Networks

Hemant Kumar Rath and Abhay Karandikar
Information Networks Lab, Department of Electrical Engineering
Indian Institute of Technology Bombay, Mumbai 400 076, India
Email: {hemantr, karandi}@ee.iitb.ac.in

***Abstract*—In this paper we propose two polling based scheduling schemes for applications based on TCP in a multipoint-to-point IEEE 802.16 network. The first scheme uses TCP congestion window ($cwnd$) size, whereas the second one uses $cwnd$ size and TCP timeouts to allocate time slots among the contending TCP flows. We ensure fairness among the users by using a credit-based approach in which the user which misses the chance to transmit due to bad channel condition gets more weightage when its channel condition favors scheduling. We also propose a method to compute an optimal polling interval which maximizes the slot use and TCP throughput. Implementation of the proposed schemes requires a cross-layer based feedback protocol stack at the base station (BS) and at the subscriber stations (SS). Through exhaustive simulations, we demonstrate that the proposed schedulers provide fairness in terms of slot assignment and amount of data transmission.**

I. Introduction

Due to recent technological developments, Broadband Wireless Access (BWA) [1] based services turn out to be more advantageous than the traditional wired services in terms of fast deployment, dynamic sharing of radio resources and low cost. BWA systems are expected to support Quality of Service (QoS) for real time applications, such-as Video Conferencing, Video Streaming and Voice-over-IP (VoIP). IEEE 802.16 [1], [2] is a BWA standard for both multipoint-to-point and mesh mode of operation, sponsored by the IEEE LAN/MAN society. The IEEE 802.16 standard defines the Physical (PHY) and Medium Access Control (MAC) layers for fixed broadband wireless access networks. The MAC layer of IEEE 802.16 is connection oriented [2], i.e., at the MAC layer communication between the Base Station (BS) and Subscriber Station (SS) is based on a Connection IDentifier (CID). Each traffic flow between SS and the BS can be identified by an unique CID. Each flow can fall into four different categories of services; Unsolicited Grant Service (UGS), Real Time Polling Service ($rtPS$), Non Real Time Polling Service ($nrtPS$) and Best Effort (BE) service as defined by the standard. Guaranteed bandwidth in terms of minimum reserved traffic rate is the basic QoS parameter defined at the MAC layer of IEEE 802.16 for UGS, $rtPS$ and $nrtPS$ services, whereas it is not so for BE service.

Currently, many Internet applications are based on Transmission Control Protocol (TCP) that belong to BE services of IEEE 802.16. For BE service, there is a need of a fair resource allocation scheme to assign time slots among the contending TCP flows. We propose two uplink scheduling schemes for applications based on TCP in a multipoint-to-point IEEE 802.16 network. The first scheduler uses congestion window ($cwnd$) size, whereas the second scheduler uses $cwnd$ size and TCP timeouts of the contending flows. The proposed uplink scheduler operates at the BS which assigns time slots to TCP sources (based on their requirements) and attempts to maximize the use of allocated time slots taking the random nature of the wireless channel into consideration. We consider PHY layer characteristics such as Signal to Noise Ratio (SNR) during scheduling and do not assign slots to a user whose SNR falls below a certain threshold level (SNR_{th}) as per the PHY layer requirement. We introduce a credit-based approach using deficit counters to ensure fairness among the users.

A. Related Work

IEEE 802.16 network elements are permitted to implement their own scheduling algorithms in both uplink and downlink as the standard does not specify any specific algorithm to be implemented. Both uplink and downlink scheduling are performed at the BS. Scheduling in the downlink is simpler than in the uplink. This is because, the BS has the knowledge of all queues assigned to SSs and packet arrival time in the downlink, whereas it does not have this information in the uplink. In downlink scheduling, the BS can use a scheduler similar to that used in traditional wired networks like Weighted Fair Queuing (WFQ) [3], Self-Clocked Fair Queueing (SCFQ) [4], Worst-case Fair Weighted Fair Queuing (WF^2Q) [5] and wireless networks like Token Bank Fair Queuing (TBFQ) [6]. In IEEE 802.16, schemes like WFQ, SCFQ and WF^2Q would require computation of virtual start time and finish time (at the BS) for each packet arriving at SS for uplink scheduling. Since the packet arrival information is not available at the BS, such schemes are not suitable for uplink scheduling. Instead variants of Round Robin Scheduler are the candidates for uplink scheduling.

Most existing schedulers for IEEE 802.16 networks have been designed for $rtPS$ and $nrtPS$ services rather than for BE services. In [7], [8], the authors have analyzed the QoS support at the MAC layer of IEEE 802.16 by providing differentiated services to applications with different QoS requirements such as VoIP and web services. They have used

The research of Hemant Kumar Rath is supported by Bharti Centre for Communication, IIT Bombay and that of Abhay Karandikar is supported by Department of Science and Technology (Project Code 05DS025).

Weighted Round Robin (WRR) for uplink and Deficit Round Robin (DRR) for downlink scheduling. In [9], the authors propose an adaptive queue-aware uplink bandwidth allocation scheme for $rtPS$ and $nrtPS$ services. The bandwidth allocation is adjusted dynamically according to the variations in traffic load and/or the channel quality. In [10], we have proposed a credit-based scheduling scheme which polls SSs in an optimal manner to address the delay requirements of various classes of service.

B. Motivation and Primary Contribution

In the present study, we propose TCP-aware fair uplink scheduling schemes for BE services in IEEE 802.16. In the suggested uplink scheduling scheme for TCP flows, we use congestion window ($cwnd$) size and TCP timeouts of the contending flows or users. In TCP, the congestion window is an indication of the number of slots required per Round Trip Time (RTT) for transmission. This is because TCP window size changes only after one RTT. Hence, instead of assigning equal number of slots to all users, the BS should assign slots in proportion to their congestion window size, i.e., as per the flow's requirement. Assigning time slots based only on the $cwnd$ size will result in unfairness among TCP flows, since flows with smaller RTT will have larger $cwnd$ size as compared to the flows with larger RTT. To avoid this unfairness, we introduce a credit-based approach that ensures fairness among the flows. Higher number of slots are assigned to the flows which are close to TCP timeout, thereby preventing their $cwnd$ dropping to one due to timeout. The primary aim of this paper is to propose a new polling based uplink scheduling mechanism for applications based on TCP within the cross-layer framework of the protocol stack in IEEE 802.16.

In Section II, we discuss our system model for a multipoint-to-point IEEE 802.16 based network. In Section III, we propose two uplink scheduling schemes for applications based on TCP. We describe a feedback based cross-layer framework to implement these schemes and a method to compute the polling interval in Section IV. In Section V, we describe the experiments, discuss the results and performance of the proposed schemes. Finally, we provide the concluding remarks and future scope in Section VI.

II. System Model

We consider a multipoint-to-point IEEE 802.16 based network where multiple subscriber stations (SSs) are connected to a centralized base station (BS) as shown in Fig. 1. We consider WirelessMAN-SC air interface which is based on a single-carrier (SC) modulation. Though the standard supports both Time-Division Duplex (TDD) and Frequency-Division Duplex (FDD), we consider only TDD in this paper. In TDD, time is divided into frames, each of which in turn consists of an uplink subframe and downlink subframe. Each subframe is composed of a fixed number of slots. The standard supports a bandwidth request-grant mechanism in which the MAC of the BS assigns time slots to SSs. The uplink assignment is based on Demand Assigned Time Division Multiple Access (DAMA/TDMA), whereas the downlink assignment is Time Division Multiplexing (TDM). Bandwidth-requests are conveyed either in a contention mode or in a contention-free polling mode. We consider a contention-free polling mode in which the BS polls each SS for its bandwidth requirement. Though the standard supports optional variable data rates based on adaptive modulation schemes, fixed modulation scheme (QPSk) is considered here.

We consider uplink scheduling in a multipoint-to-point IEEE 802.16 network. In our framework SSs are the TCP traffic sources who transmit to the end users (TCP sinks) through the BS. Though there can be either single or multiple flows between each SS and the BS, we consider single TCP flow between each SS and the BS. A set I of TCP flows (also known as source-sink pairs) shares a network of L unidirectional links going through the BS. The links between SSs and the BS are considered to be the bottleneck links of the network. The downlink in the proposed scheme does not have any bandwidth constraint. The capacity of the individual link l is c_l, $l \in L$. For each TCP source-sink pair $i \in I$, there is an associated transmission rate x_i that is a function of the current congestion window ($cwnd_i$) size and RTT_i of the TCP flow: $x_i = \frac{cwnd_i}{RTT_i}$.

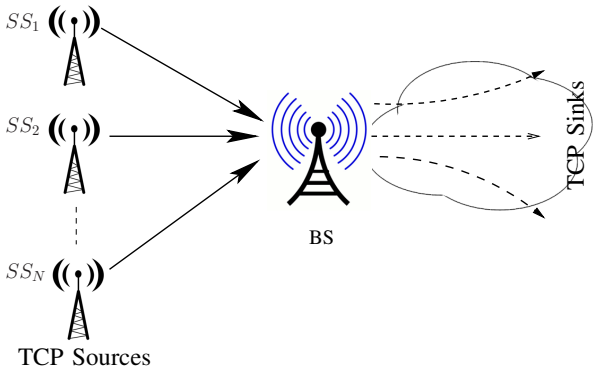

Fig. 1. Multipoint-to-Point Framework in IEEE 802.16 Networks

III. Scheduling of BE Service in IEEE 802.16

A. TCP Window-Aware Uplink Scheduler (TWUS)

The TCP Window-Aware Uplink Scheduler in its basic form is a polling based system wherein the BS polls each SS to determine its resource requirement in terms of the number of time slots required to transmit. Polling can be done once in every frame or once in multiple frames. In the proposed scheme, the BS polls each SS periodically once in every polling interval of k frames. The determination of value of k is explained in Section IV. An SS with non-zero congestion window size and having SNR greater than a certain given threshold SNR_{th} (which depends on the modulation scheme used) conveys its slot requirement to the BS. The list of SSs that responds to the polling with $cwnd$ parameters constitute a *schedulable set* (L_{sch}) at the BS. The BS does not alter the *schedulable set* till the next polling. In subsequent frames

(scheduling instants), the BS checks SNR of each SS among the *schedulable set* and schedules those SSs whose SNR is above SNR_{th}. The set of SSs which can be scheduled during a frame constitute an *active set* (L_{active}). Note that an *active set* is a subset of the *schedulable set* in the proposed scheme. The relationship between polling interval and scheduling instances is given in Fig. 2.

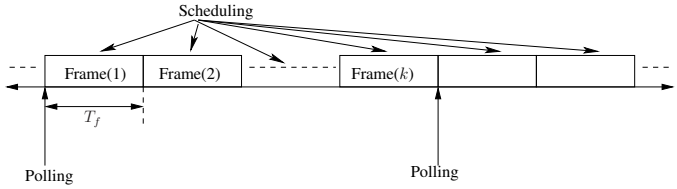

Fig. 2. Scheduling and Polling Instances in IEEE 802.16

In every frame the BS schedules SSs belonging to L_{active} based on a variant of Deficit Round Robin [11] scheduler. In this scheme the BS computes the weight $W_i(n)$ of each active SS_i in each frame n and then assigns slots in proportion to its weight. The weight of each SS is updated on a frame by frame basis and is computed in the following manner.

Let $cwnd_i$ be the congestion window size of SS_i which is conveyed to the BS at the time of polling. Let M be the number of subscriber stations in the *schedulable set*. Let N_s be the total number of uplink data slots in a frame of length T_f. We assume the number of uplink slots available in a frame is much larger than the number of subscriber stations connected to the BS. At the start of the polling interval, we compute quantum size $Q = \frac{N_s}{M}$, which is the average number of slots one schedulable user should get in every frame. The quantum size is fixed for every frame till the next polling and is updated at the start of every polling instant. To keep a track of the number of slots assigned to one SS as compared to its quantum size Q and to provide fairness among the subscriber stations, the BS maintains a deficit counter for each SS connected to it. At the start of a flow (or at the connection setup), the deficit counter DC_i of SS_i is initialized to zero. The deficit counter of SS_i is updated on a frame by frame basis as discussed below. The BS maintains an indicator variable $Flag_i$ for SS_i. If SS_i is scheduled in a frame n, then $Flag_i(n)$ is set to 1, otherwise it is set to 0.

At a scheduling instant n, the quantum assigned to an SS which is in the *schedulable set*, but not in the *active set* (i.e., $L_{sch} \setminus L_{active}$) is distributed among the remaining SSs (*active set* users) in proportion to their weights. The deficit counter $DC_j(n)$ of each $SS_j \in (L_{sch} \setminus L_{active})$ is *incremented* by Q, the amount of service it has missed. Likewise, the deficit counter $DC_i(n)$ of $SS_i \in L_{active}$, that has received more than its minimum share Q of the uplink slots is *decremented* by the amount of service that SS_i received over and above its quantum Q. Let $N_i(n)$ be the total number of slots assigned to SS_i in frame n. The deficit counter of SS_i, $\forall i \in L_{active}$ at the n^{th} frame is updated as:

$$DC_i(n) = DC_i(n-1) + Q - Flag_i(n-1)N_i(n-1). \quad (1)$$

From (1), we observe that the deficit counter can become positive or negative in a frame depending upon the number of slots assigned in that frame as well as the previous frames. We shift the deficit counter to obtain the scaled deficit counter $dc_i(n)$, a positive value for weight computation. The scaled deficit counter $dc_i(n)$ in a frame n is obtained by adding the magnitude of the minimum deficit counter value among the *active set* to the deficit counter $DC_i(n)$. In other words $\forall i \in L_{active}$,

$$dc_i(n) = DC_i(n) + \min_j |DC_j(n)|, \forall j \in L_{active}. \quad (2)$$

At the start of a flow (or at the connection setup), the scaled deficit counter dc_i of SS_i is initialized to one. The scaled deficit counters and the weights are computed only for users belonging to *active set*. For all other users, the weight is zero. The BS determines the weight $W_i(n)$ for SS_i in frame n using the following equation:

$$W_i(n) = \frac{cwnd_i \times dc_i(n)}{\sum_{j \in L_{active}} cwnd_j \times dc_j(n)}, \forall i \in L_{active}. \quad (3)$$

Equation (3) essentially computes a weight $W_i(n)$ for an SS_i in frame n that is proportional to the normalized product of the scaled deficit counter and $cwnd$ size. In traditional TCP, if a certain flow has small RTT, its $cwnd$ value is large. Allocating the number of slots in proportion to $cwnd$ may result in allocation of even larger number of slots to such flows. The credit-based approach adopted in this paper ensures that the scaled deficit counter value is small for such flows and thereby ensures fairness. After the computation of weights, the BS assigns number of slots to SS_i in frame n using:

$$N_i(n) \quad = \quad \frac{W_i(n) \times N_s}{\sum_{j \in L_{active}} W_j(n)}, \forall i \in L_{active}. \quad (4)$$

The pseudo-code of the proposed TCP Window-Aware Uplink Scheduler (TWUS) is presented in Algorithm 1.

As noted before, TCP timeout is also an important parameter which affects TCP's performance. If a TCP source does not get an acknowledgment before the TCP timeout occurs, it drops its congestion window to one. TCP timeout occurs usually due to congestion in a link, but can also occur due to a TCP un-aware scheduling process. For example, the number of slots assigned to an SS may not be enough to transmit the window of data in one RTT resulting in TCP timeout. To avoid this scenario, we propose a Deadline based TCP Window-Aware Uplink Scheduler (DTWUS) in the next section.

B. Deadline based TCP Window-Aware Uplink Scheduler (DTWUS)

In this scheme, we use TCP timeout information along with congestion window and deficit counter value to compute the weights. An active SS whose TCP flow is approaching TCP timeout is scheduled with a larger weight than others[1]. We

[1]TCP flows generally start at random and hence different flows have different residual times to reach TCP timeout.

Algorithm 1 :TCP Window-Aware Uplink Scheduler (TWUS) for IEEE 802.16

$Flag_i(0) \leftarrow 0 \; \forall i$
$DC_i(0) \leftarrow 0 \; \forall i$
$dc_i(0) \leftarrow 1 \; \forall i$
$N_i(0) \leftarrow 0 \; \forall i$
Frame number $n \leftarrow 1$
while TRUE **do**
  Determine L_{sch} for the current polling interval
  $M \leftarrow |L_{sch}|$
  Quantum Size $Q \leftarrow \frac{N_s}{M}$
  $k \leftarrow \min_i\{RTT_i\}$
  $T \leftarrow kT_f$
  while $T > 0$ **do**
    $L_{active} \leftarrow \phi$
    for all $i \in L_{sch}$ **do**
      if $SNR_i(n) \geq SNR_{th}$ **then**
        $L_{active} \leftarrow L_{active} \cup \{i\}$
        $Flag_i(n) \leftarrow 1$
        $DC_i(n) \leftarrow DC_i(n-1) + Q - Flag_i(n-1)N_i(n-1)$
      else
        $Flag_i(n) \leftarrow 0$
        $DC_i(n) \leftarrow DC_i(n-1) + Q$
        $W_i(n) \leftarrow 0$
        $N_i(n) \leftarrow 0$
      end if
    end for
    for all $i \in L_{actve}$ **do**
      $dc_i(n) \leftarrow DC_i(n) + \min_j |DC_i(n)|, \forall j \in L_{active}$
      $W_i(n) \leftarrow \frac{cwnd_i \times dc_i(n)}{\sum_{j \in L_{active}} cwnd_j \times dc_j(n)}$
      $N_i(n) \leftarrow \frac{W_i(n) \times N_s}{\sum_{j \in L_{active}} W_j(n)}$
    end for
    $T \leftarrow T - T_f$
    $n \leftarrow n + 1$
  end while
end while

define deadline d_i for SS_i as the amount of time that it can wait before reaching TCP timeout since its last scheduling instant. At the start of a connection, d_i of SS_i is initialized to TTO_i (TCP timeout of SS_i). If SS_i is scheduled in a frame n, then the deadline $d_i(n)$ remains same as $d_i(n-1)$. Else, $d_i(n)$ is decremented by one frame duration from its previous value. In other words, at the n^{th} frame deadline is updated as:

$$d_i(n) = d_i(n-1) - T_f, \tag{5}$$

If T_f exceeds $d_i(n-1)$, then the deadline $d_i(n)$ of SS_i is initialized to TTO_i. In that case, TCP flow experiences a timeout before getting scheduled, resulting in $cwnd_i$ dropping to one. SS_i will start retransmitting again with $cwnd = 1$ and with a fresh timeout value. The deadline introduced here is a measure of how close a TCP flow is to a TCP timeout.

After computing the scaled deficit counter as in (2) and deadline as in (5), the BS determines the weight $W_i(n)$ for SS_i in frame n using the following equation:

$$W_i(n) = \frac{\frac{cwnd_i \times dc_i(n)}{d_i(n)}}{\sum_{j \in L_{active}} \frac{cwnd_j \times dc_j(n)}{d_j(n)}}, \forall i \in L_{active}. \tag{6}$$

After the computation of weights, number of slots assigned by the BS to SS_i in frame n is computed as:

$$N_i(n) = \frac{W_i(n) \times N_s}{\sum_{j \in L_{active}} W_j(n)}, \forall i \in L_{active}. \tag{7}$$

The use of deadline in weight computation ensures that a higher number of time slots are assigned to an SS that has a smaller deadline. The pseudo-code of the proposed Deadline based TCP Window-Aware Uplink Scheduler (DTWUS) is presented in Algorithm 2.

IV. Discussions on Implementation

Implementations of the proposed scheduling scheme at the MAC layer can be done by using a cross-layer based feedback architecture similar to [12] instead of the traditional layered architecture. The cross-layer based feedback architecture would require a *tuning layer* for layer wise updates in the protocol stack. The *tuning layer* of TCP updates the MAC layer at the SS with the $cwnd$ size and RTT, whereas the *tuning layer* of PHY layer updates the SNR information at the MAC layer at the BS.

The polling interval k also needs to be chosen carefully. We argue that the polling interval should be the minimum RTT[2] among all TCP flows going through the BS. This is because, TCP timeout value is typically chosen to be four to five times the RTT in most TCP implementations. Therefore, if we choose the polling interval to be equal to two RTTs, then any SS (with an ongoing TCP flow) that misses polling needs to be polled in the next opportunity (as the TCP flow of that user might be reaching TCP timeout). Similarly, if the polling interval is more than two RTTs, and if the BS misses one SS with an active TCP flow, then congestion window reduction for that TCP flow will likely occur with high probability. This is because the chance of not getting scheduled in the next opportunity before TCP timeout is very high. If we poll very frequently, i.e., if the polling interval is less than one RTT, then we spend more number of control slots for polling. Moreover, one does not gain any advantage due to frequent polling since the congestion window changes only after one RTT (requirements of SSs will not change as such for one RTT). Hence, we choose the polling interval that equals to the minimum RTT of the active TCP flows.

V. Experimental Setup and Performance Evaluation

To evaluate the performance of the proposed schedulers, we have simulated a multipoint-to-point network in IEEE 802.16-2004 (WirlessMAN-SC) as shown in Fig. 1. We have

[2]Typical TCP RTTs are in the range of 100 msec - 200 msec, whereas the frame length T_f in IEEE 802.16 is either 0.5 msec or 1 msec or 2 msec.

Algorithm 2 :Deadline based TCP Window-Aware Uplink Scheduler (DTWUS) for IEEE 802.16

$Flag_i(0) \leftarrow 0 \; \forall i$
$DC_i(0) \leftarrow 0 \; \forall i$
$dc_i(0) \leftarrow 1 \; \forall i$
$N_i(0) \leftarrow 0 \; \forall i$
Frame number $n \leftarrow 1$
while TRUE **do**
  Determine L_{sch} for the current polling interval
  Update TTO_i
  if $n = 1$ **then**
    $d_i(0) \leftarrow TTO_i \; \forall i$
  end if
  $M \leftarrow |L_{sch}|$
  Quantum Size $Q \leftarrow \frac{N_s}{M}$
  $k \leftarrow \min_i\{RTT_i\}$
  $T \leftarrow kT_f$
  while $T > 0$ **do**
    $L_{active} \leftarrow \phi$
    for all $i \in L_{sch}$ **do**
      if $SNR_i(n) \geq SNR_{th}$ **then**
        $L_{active} \leftarrow L_{active} \cup \{i\}$
        $Flag_i(n) \leftarrow 1$
        $DC_i(n) \leftarrow DC_i(n-1) + Q - Flag_i(n-1)N_i(n-1)$
        $d_i(n) \leftarrow d_i(n-1)$
      else
        $Flag_i(n) \leftarrow 0$
        $DC_i(n) \leftarrow DC_i(n-1) + Q$
        $d_i(n) \leftarrow d_i(n-1) - T_f$
        if $d_i(n) \leq 0$ **then**
          $d_i(n) \leftarrow TTO_i$
        end if
        $W_i(n) \leftarrow 0$
        $N_i(n) \leftarrow 0$
      end if
    end for
    for all $i \in L_{actve}$ **do**
      $dc_i(n) \leftarrow DC_i(n) + \min_j |DC_j(n)|, \forall j \in L_{active}$
      $W_i(n) \leftarrow \frac{cwnd_i \times dc_i(n)/d_i(n)}{\sum_{j \in L_{active}} cwnd_j \times dc_j(n)/d_j(n)}$
      $N_i(n) \leftarrow \frac{W_i(n) \times N_s}{\sum_{j \in L_{active}} W_j(n)}$
    end for
    $T \leftarrow T - T_f$
    $n \leftarrow n + 1$
  end while
end while

considered one BS and 10 SSs in our network. We simulate one TCP flow per SS. The random channel gains between SSs and the BS are log-normally distributed with variance σ =8 dB. The path loss factor γ is assumed to be 4. Each SS has a single buffer of infinite size. The frame duration T_f is set equal to two msec[3]. The uplink subframe T_{ul} consists of 500 data slots (assuming negligible control slots). We consider both equal and unequal distances between SSs and the BS. For equal distances, the distances of all SSs from the BS are 1 Km each and for unequal distances the distances between SSs (SS_1 - SS_{10}) and the BS are 0.90 Km, 1.00 Km, 1.10 Km, 0.90 Km, 0.95 Km, 1.10 Km, 1.00 Km, 1.00 Km, 1.10 Km and 1.01 Km respectively. We have conducted four sets of experiments based on distances and the proposed scheduling algorithms TWUS and DTWUS. In Experiment-1 we simulate TWUS with equal distances. In Experiment-2 we simulate DTWUS with equal distances. In Experiment-3 we simulate TWUS with unequal distances. Finally, in Experiment-4 we simulate DTWUS with unequal distances. We have performed all four experiments for 25000 frames. For each experiment, we have performed 10 independent runs and averaged out the results. We have used discrete event simulator. The system parameters used in this paper are presented in Table I.

TABLE I
SYSTEM PARAMETERS

Type	Parameters
TCP Type	TCP Reno
Channel Bandwidth	25 MHz
Modulation Scheme	QPSK
No. of Frames Simulated	25000
No. of Iterations	10

A. Results

TCP congestion window variations of two users selected at random in Experiment-1 and Experiment-2 are shown in Fig. 3 and Fig. 4 respectively. We observe that the congestion window does not drop to one due to TCP timeout in Experiment-2 (Fig. 4), whereas the congestion window drops to one in Experiment-1 (Fig. 3). Congestion window variations of other users are similar to the congestion window variations of the two random users chosen here. By using the proposed scheduling schemes, one can avoid frequent congestion window dropping.

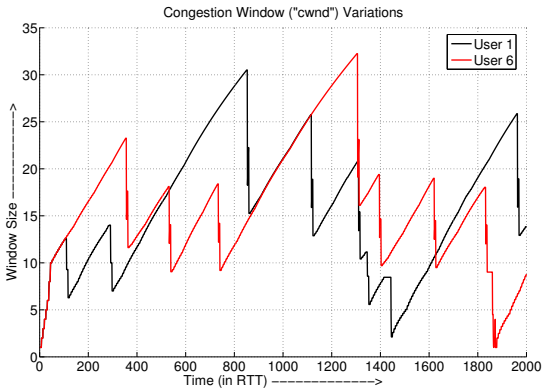

Fig. 3. *cwnd* Variations of Two Random Users in Experiment-1

[3]T_f is equally divided between uplink subframe T_{ul} and downlink subframe T_{dl}.

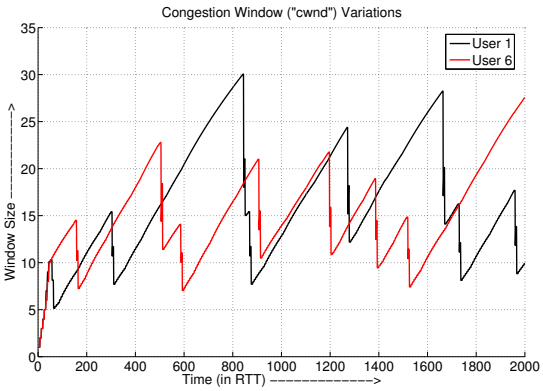

Fig. 4. *cwnd* Variations of Two Random Users in Experiment-2

The number of slots allocated to various subscriber stations in all four experiments are shown in Table II. We observe that the number of slots assigned for the equal distances experiments (Experiment 1 & 2) is more uniform as compared to unequal distances experiments (Experiment 3 & 4). We also observe that slot assignment using DTWUS (Experiments 2 & 4) is more fair as compared to the slot assignment using TWUS (Experiments 1 & 3).

Table III shows the window size variations among the SSs in all four experiments. We observe that the average window size achieved by DTWUS is larger as compared to the average window size achieved by TWUS. This is because, DTWUS considers TCP timeouts to assign time slots and hence the probability of congestion window dropping to one is small, resulting in larger average congestion window. As expected, the average window size achieved by the SSs in all four sets of experiments are uniform.

From Table IV, we observe that the average rate of transmission achieved by an SS depends upon the distance from the BS. So, to achieve fairness in terms of the amount of data transmission, SSs with lower rate of data transmission should get more number of slots as compared to SSs with higher rate of data transmission. From Table V, we observe that the total amount of data transmission by DTWUS is more as compared to TWUS. With unequal distances, the total amount of data transmission is lesser as compared to equal distances. This is because, our scheme is primarily designed for long-term fairness than for achieving high sum-capacity.

TABLE II
AVERAGE NUMBER OF SLOTS ASSIGNED ($\times 10^5$)

SS No.	No. of Slots Expt-1	No. of Slots Expt-2	No. of Slots Expt-3	No. of Slots Expt-4
1	12.18	12.39	12.92	12.76
2	12.24	12.22	12.30	12.34
3	12.16	12.44	11.46	11.63
4	12.38	12.37	12.95	12.85
5	12.28	12.34	12.67	12.59
6	12.28	12.22	11.49	11.92
7	12.26	12.32	12.38	12.52
8	12.29	12.33	12.39	12.37
9	12.21	12.30	11.63	11.84
10	12.32	12.33	12.30	12.42

TABLE III
AVERAGE WINDOW SIZE

SS No.	Window Size Expt-1	Window Size Expt-2	Window Size Expt-3	Window Size Expt-4
1	16.15	16.74	16.20	17.02
2	16.05	15.84	15.96	16.41
3	16.85	16.07	16.09	15.06
4	16.39	17.02	17.61	17.58
5	16.25	16.56	16.24	16.66
6	15.82	16.53	14.95	16.14
7	15.94	15.99	16.14	16.44
8	15.75	16.63	16.33	16.43
9	15.84	16.11	15.28	16.21
10	16.23	16.42	15.72	16.25

TABLE IV
AVERAGE DATA TRANSMISSION RATE ($Mbps$)

SS No.	Rate of Tx. Expt-1	Rate of Tx. Expt-2	Rate of Tx. Expt-3	Rate of Tx. Expt-4
1	32.17	32.12	34.97	34.97
2	32.12	32.13	32.09	32.13
3	32.14	32.11	28.93	28.86
4	32.09	32.13	34.94	34.97
5	32.09	32.14	33.58	33.61
6	32.07	32.10	28.91	28.87
7	32.17	32.13	32.13	32.13
8	32.18	32.14	32.11	32.14
9	32.14	32.20	28.91	28.95
10	32.14	32.16	31.87	31.85

TABLE V
AMOUNT OF DATA TRANSFERED (Mb)

SS No.	Amount Expt-1	Amount Expt-2	Amount Expt-3	Amount Expt-4
1	39.21	39.83	45.21	44.63
2	39.22	39.32	39.50	39.66
3	39.08	39.08	33.18	33.57
4	39.73	39.73	45.28	44.97
5	39.42	39.68	42.56	42.34
6	39.41	39.22	33.24	34.43
7	39.47	39.61	39.78	40.23
8	39.57	39.64	39.79	39.79
9	39.20	39.62	33.64	34.28
10	39.62	39.66	39.22	39.58
Total	394.00	395.79	391.40	393.58

B. Performance Evaluation

To assess the fairness of the proposed scheduling algorithms, we compute the Jain's Fairness Index (JFI) [13] for the amount of data transmission by the subscriber stations for all four experiments. This is shown in Table VI. We observe that JFI is more than 99% when the distances between the SSs and the BS are equal (Experiments 1 & 2) and more than 98% when the distances are unequal (Experiments 3 & 4). From the simulation results and the JFIs, we claim that our scheduling schemes are fair.

The usage of resources is an important factor for any type of scheduler. From Table VI, we observe that the usage of slots by the proposed scheduling schemes are more than 98%. Usage of slots can be further increased by adding different classes of traffic along with TCP traffic. Since, the proposed scheduling schemes consider the PHY layer characteristics like

SNR, the slots are assigned to users having better channel condition only. This maximizes the use of available slots.

TABLE VI
JAIN'S FAIRNESS INDEX (JFI) AND SLOT USE

	Expt-1	Expt-2	Expt-3	Expt-4
JFI	0.9999	0.9999	0.9877	0.9902
% Slot Use	98.26	98.60	98.13	98.61

C. Log-normal Fading and Performance Evaluation

To analyze the fairness of the proposed scheduling schemes with different log-normal fading we have simulated the schemes with five different σ (2, 4, 6, 8 and 10 dB). The results are plotted in Fig. 5 and Fig. 6. We observe from these figures that the proposed scheduling schemes are fair even for a large variation of fading in the channel.

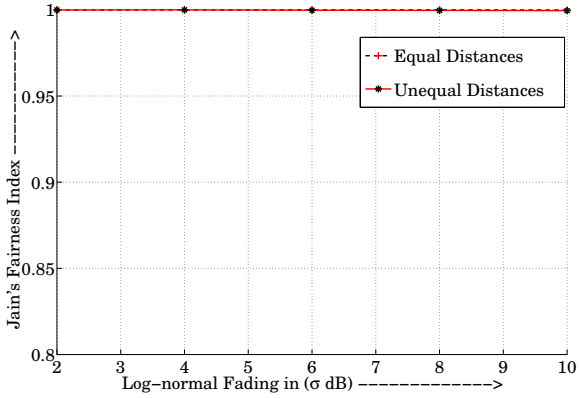

Fig. 5. JFI with Different log-normal Fading using TWUS

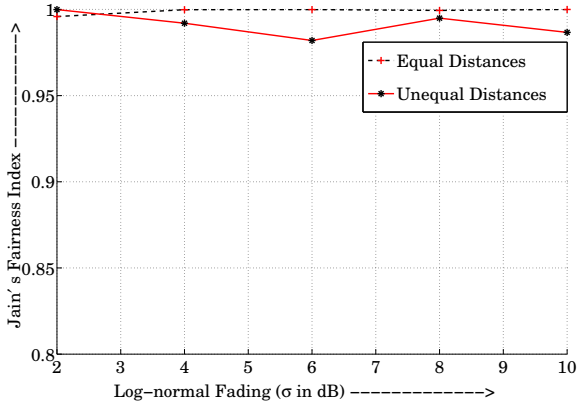

Fig. 6. JFI with Different Log-normal Fading using DTWUS

VI. CONCLUDING REMARKS

In this paper, we have introduced fair resource allocation schemes for BE service in the uplink of a multipoint-to-point IEEE 802.16 network. We have considered TCP congestion window and TCP timeout values for scheduling at the BS. We have achieved fairness in terms of slot assignment and amount of data transmission. The proposed schemes succeed in stabilizing the congestion window variation. We have attempted to avoid TCP timeouts occurring due to TCP un-aware scheduling at the MAC layer. Though our schemes are more fair, the sum-capacity suffers if the channel condition of some users become consistently bad. To achieve high sum-capacity, use of long-term fairness is not appropriate, instead a temporal fairness can be considered.

In these scheduling schemes, fixed data rates between the BS and SSs were considered. But, in practice, due to channel variation, the data rates between the BS and SSs vary. To accommodate the variable data rates, we can incorporate adaptive modulation and coding schemes defined by the IEEE 802.16 standard. Our scheme can also be used for services like $rtPS$ and $nrtPS$ in conjunction with BE services. We are currently investigating in this direction.

REFERENCES

[1] "Understanding Wi-Fi and WiMAX as Metro-Access Solutions," *Intel Corporation, White Paper*, August 2004. Available at: http://whitepapers.silicon.com. Accessed on:10/10/2005.

[2] LAN/MAN Standards Committee, *IEEE Standards for Local and Metropolitan Area Network: Part 16: Air Interface for Fixed Broadband Wireless Access Systems*. IEEE Computer Society and IEEE Microwave Theory and Techniques Society, May 2004.

[3] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queuing Algorithm," in *ACM SIGCOMM Symposium on Communications Architectures and Protocols*, September 1999.

[4] S. Golestani, "A Self-Clocked Fair Queuing Scheme for Broadband Applications," in *Proc. IEEE INFOCOM*, pp. 636–646, 1994.

[5] J. C. R. Bennett and H. Zhang, "WF^2Q: Worst-case Fair Weighted Fair Queueing," in *Proc. IEEE INFOCOM*, pp. 120–128, 1996.

[6] W. K. Wong, H. Tang, and V. C. M. Leung, "Token bank fair queuing: a new scheduling algorithm for wireless multimedia services," *International Journal of Communication Systems*, vol. 08, pp. 591–614, September 2004.

[7] C. Cicconetti, A. Erta, L. Lenzini, and E. Mingozzi, "Performance Evaluation of the IEEE 802.16 MAC for QoS Support," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 26–38, January 2007.

[8] C. Cicconetti, L. Lenzini, E. Mingozzi, and C. Eklund, "Quality of Service Support in IEEE 802.16 Networks," *IEEE Network*, vol. 20, pp. 50–55, March-April 2006.

[9] D. Niyato and E. Hossain, "Queue-Aware Uplink Bandwidth Allocation and Rate Control for Polling Service in IEEE 802.16 Broadband Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 668–679, June 2006.

[10] H. K. Rath, A. Bhorkar, and V. Sharma, "An Opportunistic Uplink Scheduling Scheme to Achieve Bandwidth Fairness and Delay for Multiclass Traffic in Wi-Max (IEEE 802.16) Broadband Wireless Networks," in *Proc. of IEEE Globecom*, November 2006.

[11] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 375–385, June 1996.

[12] V. T. Raisinghani and S. Iyer, "Cross-Layer Feedback Architecture for Mobile Device Protocol Stacks," *IEEE Communications Magazine*, vol. 44, pp. 85–92, January 2006.

[13] R. Jain, D.-M. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *Technical Report TR-301, DEC Research Report*, September 1984.