Algorithms for Routing and Centralized Scheduling to Provide QoS in IEEE 802.16 Mesh Networks *

Harish Shetiya
Dept of Electrical Communication Engineering
Indian Institute of Science
Bangalore, India. 560012
harish@pal.ece.iisc.ernet.in

Vinod Sharma
Dept of Electrical Communication Engineering
Indian Institute of Science
Bangalore, India. 560012
vinod@ece.iisc.ernet.in

ABSTRACT

We consider the problem of routing and centralized scheduling for IEEE 802.16 mesh networks. We first fix the routing, which reduces the network to a tree. We then present scheduling algorithms which provide per flow QoS (Quality of Service) guarantees to real and interactive data applications while utilizing the network resources efficiently. Our algorithms are also scalable: they do not require per flow processing and queueing and the computational requirements are minimal. We also discuss admission control policies which ensure that sufficient resources are available. We have verified our algorithms via extensive simulations.

Categories and Subject Descriptors: C.2.1 [Computer Communication Networks]: Network Architecture and Design - wireless communication

General Terms: Algorithms, Performance

Keywords: IEEE 802.16 networks, QoS, wireless multihop networks, routing and scheduling algorithms

1. INTRODUCTION

IEEE 802.16 standard [1], also known as WiMax has been specifically designed to provide wireless last-mile broadband access in the Metropolitan Area Network (MAN), delivering performance comparable to traditional cable, DSL or T1 offerings. In order to provide the coverage and data rates envisioned, even on uneven terrain, the use of multihop communication seems desirable. Hence WiMax supports a Mesh mode in which unlike the traditional cellular systems, the nodes can communicate without having a direct connection with the base station.

In a IEEE 802.16d Mesh network, a node that has a direct connection to backhaul services outside the Mesh network, is termed a Mesh Base Station (MBS). All other nodes of a Mesh network are termed Mesh Subscriber Stations (MSS). In IEEE 802.16d standards these nodes are stationary, i.e., the standards do not support mobility. The standard specifies a centralized scheduling scheme for mesh networks. Under this scheme, the MSSs notify the MBS their data transfer requirements and the quality of their links to their neighbours. The MBS uses the topology information along with the requirements of each MSS to decide the routing and the scheduling. The MAC scheme used is TDMA and the resource allocation is in terms of time slots within a frame. The standard does not specify an algorithm for scheduling of the slots to different MSSs; neither does it specify any routing algorithm. Scheduling and routing will have significant impact on the performance of the system and will largely decide the end to end QoS to different users.

Scheduling algorithms to provide QoS in single hop (Point to multipoint) IEEE 802.16 networks are considered in [3], [6], [14] and [16]. The problem of scheduling in multihop wireless networks has been considered in [2], [10], [13] and [15]. In fact [2] considers joint routing and scheduling. But in [2] and [10] spectral reuse is employed. Thus scheduling in these studies cannot be used in our system. Also some of the assumptions made in these studies do not hold in our system. In [15] (p 310), Klimov's algorithm is provided where a single server is shared by various queues. This algorithms can be modified to provide a useful scheduling algorithm in our setup. But all these studies limit themselves to using UDP connections and hence cannot be used to provide QoS to data applications. In addition they do not provide any explicit QoS to individual connections. This study is based on [13] which also contains algorithms that optimize the overall system performance (but those algorithms do not guarantee QoS to individual connections).

In this paper, we present algorithms for centralized scheduling of real and non real time traffic with the objective of providing QoS within the framework of the IEEE 802.16 mesh mode. We first fix the routing within the network. Then we develop scheduling algorithms which provide QoS to real time voice and video. These applications use UDP while data applications use TCP. TCP, being window flow controlled behaves very differently from UDP. Next we develop algorithms which can be used to provide QoS to interactive data which uses TCP. Finally we combine the algorithms to provide QoS in a network serving both real and nonreal time applications. To ensure sufficient resources we also dis-

*This work was partially supported by the DRDO-IISc program on Advanced Research in Mathematical Engineering

WMuNeP'05, October 13, 2005, Montreal, Quebec, Canada.

cuss an admission control algorithm that can be used in our setup. Our algorithms use the network resource efficiently and can be used in real time by the MBS.

The organization of the paper is as follows. Section 2 describes the system model. We obtain a good routing algorithm in Section 3. In Section 4 we develop scheduling algorithms to provide QoS to UDP connections. TCP connections will be studied in section 5. In Section 6 we handle both UDP and TCP traffic together. Section 7 provides an admission control policy. Section 8 concludes the paper.

2. SYSTEM MODEL

IEEE 802.16 supports two modes of operation: Point to Multipoint (PMP) and Mesh mode. In PMP the traffic is transmitted directly between the BS and an SS. This is the common mode and current implementation efforts are directed at PMP. In the Mesh mode, the overall area is divided into meshes. Each mesh has a Mesh BS. The other nodes in a mesh are called Mesh subscriber stations (MSSs). A transmission can take place two MSS within a mesh or between two different meshes. The transmission between two MSSs within a mesh can occur via other MSSs within the mesh which may or may not involve the MBS. Transmission between two MSSs in two different meshes involves transmission from the source MSS to its MBS (possibly via other MSSs in the mesh), from MBS to BS, from BS to MBS of the receiver mesh and finally from this MBS to the receiver MSS.

In this paper we consider the mesh mode. We provide a brief overview of the IEEE 802.16 mesh mode of operation and present the system model that we use in our work.

The mesh mode supports two different physical layers, WirelessMAN-OFDM and WirelessHUMAN. Both of these use 256 point FFT OFDM TDMA/TDM for channel access and operate in a frequency band below 11GHz. The first operates in the licensed band but the second uses the unlicensed band. The standards also support adaptive modulation and coding where the burst profile of the link (i.e., modulation scheme and the coding rate) and hence the link rate is changed depending upon the channel conditions.

The mesh mode supports only Time Division Duplex (TDD) to share the channel between the uplink and the downlink. A Mesh frame consists of a control and a data subframe. The control subframe serves two basic functions. One is the creation and maintenance of cohesion between different stations. The other is the coordinated scheduling of data transfers between stations. The data subframe consists of MAC PDUs transmitted by different users. A MAC PDU consists of a generic MAC header, a Mesh subheader and optional data. The standards support both centralized and distributed scheduling of slots. Centralized scheduling is mainly used to transfer data between the MBS and the MSSs. Since this is the usual scenario, centralized scheduling is the dominant mode. The MBS periodically collects the channel information and the resource (throughput) requests of all the nodes to draw up the schedule which it distributes to the nodes. Currently no spectral reuse is supported with this type of scheduling.

We consider the following scenario. Consider a Mesh network with M MSSs labeled 1,2,...,M. The MBS is labeled 0. We consider Uplink and Downlink *Centralized Scheduling* of the MSSs, which, according to the standards uses TDMA without spectral reuse. Also the data is directed either to or from the MBS. We assume each node transmits at the maximum allowed power, if needed. As the channel condition on a link changes, the data rate is also changed so as to meet the desired BER. Let r_{ij} denote the rate and E[r_{ij}] the average rate of the channel from node i to node j. Resource allocation is done by the MBS in units of time slots. One time slot consists of multiple OFDM symbols. Each allocation is valid for K frames consisting of N time slots (for simplicity of notation we will take K=1).

IEEE 802.16 will support real and nonreal time applications. The real time applications, e.g., IP telephony and video conferencing use UDP while data applications use TCP. Real time applications and interactive data (file transfer and web browsing) require QoS. To provide QoS to these applications will require careful routing and scheduling of traffic through the mesh network. We will consider these problems for both types of traffic. Since UDP traffic and real time QoS requirements are very different from TCP traffic and interactive data QoS requirements, we will consider these problems separately and then show how to integrate them in the same system.

We will first provide a good routing algorithm to be used by all applications. Next we will provide algorithms to schedule the channel for real time and then data applications.

3. ROUTING

In this section we obtain a routing algorithm which may provide good performance. First we discuss the criteria that a good routing algorithm should satisfy. It will be desirable to have the same route for all the traffic at a node irrespective of its node of origin and whether it is real time or data traffic. This will considerably simplify the algorithm design complexity and implementation. Also it will simplify the development of scheduling algorithms to follow. However, the traffic generated by UDP and TCP connections behave differently and has different QoS requirements. Thus, an optimal route for one type of traffic may not be suitable for another. We would like to develop an algorithm which provides good performance for both type of traffic even if it is not optimal for either.

The next criterion is that the route should be fixed, i.e., not time varying even when the wireless channels are time varying. For one thing, even if the channels are time varying, since we are considering fixed nodes, the channel variation will be comparatively less. Also, time varying routing can cause loops and may require resequencing at the receiver which cause performance degradation. More importantly, in our context, to provide QoS guarantees, we will need to reserve resources along the route. This is possible only if we do not change the route of a connection unless it is absoultely essential. Thus, we will limit ourselves to fixed routing i.e. traffic originating at a node will follow the same path unless some node/link on it breaks down (or the statistics of some link on the route degrades considerably).

Based on the above arguments we develop a fixed routing algorithm which may work well for both real and data applications. To motivate the algorithm, we make assumptions which may be violated by both kind of traffic to varying degrees but otherwise are quite common in network literature. These assumptions are made only to motivate the development of the routing algorithm and will not be required in the rest of the paper and will also not be used in our simulations.

We assume that the channel states stay same during a frame. From frame to frame they change independently forming an i.i.d. (independent, identically distributed) sequence. The packets arriving in frame k at a node can be serviced in the next frame only. The external arrivals to each node form an i.i.d. sequence. We also assume that each node has an infinite buffer to store the packets.

We will comment later on to what extent these assumptions are satisfied by UDP and TCP connections. The assumptions of i.i.d. sequences can be replaced by stationarity and ergodicity of the sequences without any change in the arguments and conclusions in the following. Also even if the channel state changes during a frame it will not be known to the MBS and it fixes the routes and the channel schedule for the whole frame in the begining of it.

Assume we have fixed the routing. Let $r_k(i,j)$ be the assigned transmission rate, $X_k(i,j)$ the external arrivals and $Y_k(i,j)$ the arrivals from other nodes to node i for output link (i,j) during the frame k. Let $Q_k(i,j)$ be the queue length at node i for output link (i,j) in the beginning of the frame k. Also let $\lambda_{i,j} = \mathrm{E}[X_k(i,j)]$. Assume that the schedule is fixed and link $(i.j)$ is always assigned $n_{i,j}$ slots in a frame. Then

$$Q_{k+1}(i,j) = (Q_k(i,j) + Y_k(i,j) - n_{i,j} r_k(i,j))^+ + X_k(i,j)$$

where $(x)^+$ denotes max(0,x). For the queue to be stable (have a unique stationary distribution), we need

$$n_{i,j}\, E[r(i,j)] > E[X(i,j) + Y(i,j)] = \lambda_{i,j} + E[Y(i,j)]$$

where the expectation $\mathrm{E}[Y(i,j)]$ is under the stationary distribution.

For fixed routing, having loops is obviously not efficient. Also splitting traffic from a node along two paths to MBS is not optimal unless both the routes have the same cost and even then giving whole traffic to one path will not make it worse (because the channel schedule can be accordingly adjusted). Furthermore it will lead to resequencing delays at the receiver. Thus we will not split the traffic passing through a node. These together imply that we have a tree structure. In the following we will limit to such algorithms. Then corresponding to node i there will be a unique output link. Thus from now onwards we will index links also by a single index, i^{th} link being the output link of node i. Let $\lambda_i = \sum_{j=0}^{M} \lambda_{i,j}$. Then, $\mathrm{E}[Y_i] = \sum_{j=1}^{m_i} \lambda_{a_{i,j}}$, where $\{ a_{i,1}, a_{i,2}, \ldots a_{i,m_i}\}$ are the nodes whose data passes through node i. Hence, if

$$n_i > \frac{\lambda_i + \sum_{j=1}^{m_i} \lambda_{a_{i,j}}}{\mathrm{E}[r(i)]} \text{ for all } i = 1 \ldots M,$$

then the entire system is stable. Also, since $\sum_{i=1}^{M} n_i = N$, we get,

$$\sum_{i=1}^{M} \left(\frac{\lambda_i + \sum_{j=1}^{m_i} \lambda_{a_{i,j}}}{\mathrm{E}[r(i)] \cdot N} \right) < 1. \qquad (1)$$

In fact if (1) is satisfied, then we can find a fixed allocation scheme (by appropriately enlarging the channel frame, if necessary) that can stabilize the system. Rearranging the terms, we get

$$\sum_{i=1}^{M} \left(\lambda_i \cdot \sum_{j=1}^{h_i} \frac{1}{E[r(p_{i,j})]} \right) < N$$

where $\{p_{i,1}, \ldots p_{i,h_i}\}$ are the nodes through which the data of node i is routed. From the above equation it can be seen that for each node i, if we choose the route that minimizes $\sum_{j=1}^{h_i} \frac{1}{E[r(p_{i,j})]}$, the overall stability region can be maximized. Also, choosing this route minimizes the average work (transmission time) needed to transmit a packet from a node to the MBS (for uplink). This argument motivates the Shortest Path routing scheme, where the routing is fixed over all the frames for each node along the path that minimizes $\sum_{j=1}^{h_i} \frac{1}{E[r(p_{i,j})]}$. This route can be found by standard shortest path algorithms (Dijkstra's or Bellman-Ford) by assigning cost $\frac{1}{E[r(p_{i,j})]}$to link (i,j).

We will use this shortest path routing for both real time and data traffic. In the rest of the paper we develop scheduling algorithms to provide QoS to individual connections.

4. QOS FOR REAL TIME TRAFFIC

In this section we design scheduling algorithms to guarantee QoS to individual UDP connections. Two important real time applications are IP telephony and video conferencing. For these applications, the end to end delay of a packet should not exceed (say) 150 msec. If a packet exceeds this delay, it will be dropped. For satisfactory performance the fraction of packets dropped for an application should be less than (say) 2%. To satisfy these QoS requirements, we propose that at the end of a frame we drop the packets which could not be transmitted through the wireless network. This will ensure a maximum delay of about 40 msec (for 4 frames of 10 msec each) in the wireless network (the rest of the delay margin is left for the remaining part of the network that a packet may have to travel). We develop algorithms which will ensure that a particular user will not experience drop probability greater than (say) 2%.

Packets generated by audio encoders (in IP telephony) usually generate a constant bit rate (CBR) traffic. But a video encoder (say MPEG) one may use in video conferencing generates variable bit rate (VBR) traffic. To satisfy the QoS requirements of these two types of applications efficiently we require different considerations. Therefore we consider these two cases separately. We consider scheduling of CBR traffic in Section 4.1 and VBR traffic in Section 4.2. Section 4.3 provides simulations to show the effectiveness of our QoS schemes.

We briefly comment on the validity of the assumptions we made in section 3 for the present scenario. The assumption of stationary, ergodic arrival stream holds for both voice (CBR) traffic and video (VBR) traffic. This is because a Markov modulated arrival stream is a common model for VBR traffic and it forms a stationary, ergodic sequence. Infinite buffer length of the queues is also realistic under the QoS requirement of less than 2% packet loss probability.

4.1 Scheduling of CBR traffic

Let X_i (a constant) be the amount of traffic generated by users at node i during a frame. As mentioned above, in order to provide delay guarantees to the flows, we propose dropping of data that cannot be transmitted at the end of the scheduling frame. Now, the scheduling has to ensure that the amount of data dropped conforms to the QoS requirements of the flow. Let the upper bound required on the drop probability of packets generated at node i be ϵ_i (for simplicity of notation we are taking this upper bound for all

CBR applications starting at node i to be same. If different flows have different requirements then ϵ_i is the minimum of those requirements).

The scheduling problem for CBR-UDP traffic is to calculate the number of slots $n_{i,p_{i,j}}, j = 1 \ldots h_i$ required at node $p_{i,j}$ such that X_i units of data can be transmitted to the MBS per scheduling frame and the end to end drop probability is bounded by ϵ_i. Here $p_{i,j}, j = 1 \ldots h_i$ denote the nodes through which the data of node i traverses.

We decompose the drop probability ϵ_i into $\{\epsilon_{i,j}, j = 1, \ldots, h_i\}$ such that $\prod_{j=1}^{h_i}(1-\epsilon_{i,j}) \geq (1-\epsilon_i)$. At node $p_{i,j}$ the number of slots allocated for flows from node i has to ensure that the drop probability is bounded by $\epsilon_{i,j}$. We use $n_{i,j}$ for $n_{i,p_{i,j}}$, $r(j)$ for $r(p_{i,j})$ and $X_{i,j}$ for $\prod_{k=1}^{j}(1-\epsilon_{i,k})\ X_i$ to simplify the notation. Then, $n_{i,j}$ the number of slots required , has to satisfy

$$\lim_{n\to\infty} \frac{\sum_{k=1}^{n}(X_{i,j} - n_{i,j}r_k(j))^+}{n\ X_{i,j}} \leq \epsilon_{i,j}. \qquad (2)$$

This reduces to

$$E((X_{i,j} - n_{i,j}r(j))^+) \leq \epsilon_{i,j}X_{i,j}.$$

We can rewrite it as

$$\int_0^{\frac{X_{i,j}}{n_{i,j}}} (X_{i,j} - n_{i,j}r)\, f_j(r)\, dr \leq \epsilon_{i,j}X_{i,j} \qquad (3)$$

where $f_j(\cdot)$ is the pdf of the link rate $r(j)$ which is assumed to be known. The quantity on the left in (3) is a non-increasing function of $n_{i,j}$ and can be easily used to compute the value of $n_{i,j}$ that satisfies the inequality.

Instead of arbitrarily choosing the values of $\{\epsilon_{i,j}, j = 1, \ldots, h_i\}$, we can consider the optimization problem

$$\min\Big\{\sum_{j=1}^{h_i} n_{i,j}\Big\}$$

subject to

$$\prod_{j=1}^{h_i}(1-\epsilon_{i,j}) \geq (1-\epsilon_i) \qquad \text{and}$$

$$\int_0^{\frac{X_{i,j}}{n_{i,j}}} (X_{i,j} - n_{i,j}r)\, f_j(r)\, dr \leq \epsilon_{i,j}X_{i,j}, \qquad j = 1, \ldots, h_i.$$

Once, for CBR traffic originating at each node i, we have computed the number of slots $n_{i,p_{i,j}}$ required along the path, we can compute the number of slots required by each node to satisfy the QoS requirement of total CBR traffic passing through it. The MBS can give that many slots to each node in a given frame. To ensure that the flow of traffic originating at each node gets its proper share of bandwidth along its route, we put this traffic in a different queue on each node along the path. The nodes provide the needed number of slots to its different queues via WRR (Weighted Round Robin). This will be particularly useful for VBR traffic.

4.2 Scheduling of VBR Traffic

Consider J VBR flows generated at node i. Let $D_k(i,j)$ be the amount of data generated by flow j in frame k. We assume that the arrival process $\{\, D_k(i,j), k \geq 0 \,\}$ for each $j = 1, \ldots, J$ is stationary and ergodic with known statistics.

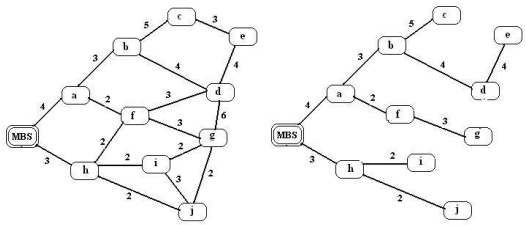

Figure 1: Network used in simulations and shortest path routing

We also assume that the arrival process from the various sources are mutually independent. As in the case of CBR traffic, we provide delay guarantees to the VBR flows by dropping the untransmitted data at the end of each frame. The problem is to calculate the number of slots required by this VBR traffic in order to bound the drop probability by ϵ_i.

The amount of resources required to accomodate the VBR traffic can be reduced by utilizing the statistics of the arrival process. Since the untransmitted data at the end of the frame is dropped we need to consider only the marginal distribution of $D_k(i,j)$ to calculate the amount of resources required at the first node to provide a statistical guarantee. Also, since the drop probability is typically small, we can assume that the statistics of the arrival process is not distorted after flowing through the first node. Hence we can use the same analysis for each of the nodes along the route. Choose ϵ_i^b and ϵ_i^d such that (1-ϵ_i^b)(1-ϵ_i^d) $\geq$ (1-ϵ_i). Now, find C_i ([9], Chapter 5) such that

$$P(\sum_{j=1}^{J} D(i,j) > C_i) \leq \epsilon_i^b. \qquad (4)$$

This C_i/J is called the equivalent bandwidth of the VBR source ([9]). The MBS can treat a VBR source as a CBR flow generating C_i/J units of data per frame and calculate the number of slots required to satisfy the drop probability requirement of ϵ_i^d. In practice, the exact statistics of a VBR arrival process may not be available. The statistics that is generally available is the maximum, the minimum and the average data rates. In order to satisfy the QoS requirements of the flows, we can calculate the value of C_i by using a source model that has all the known characteristics of the original source but has the worst case behaviour (i.e. gets the largest probability of loss). It is shown in [7] that for the case of J independent, homogeneous, stationary sources with arrivals in a slot taking values in a finite set (this class covers Markov modulated sources modulated by finite state Markov chains) the worst case drop probability is obtained by replacing these sources by i.i.d. ON-OFF sources having the same maximum, minimum and average rates.

4.3 Simulations

We consider a Mesh network of 10 nodes (Figure 1). The characteristics of the physical layer are summarized in Table I. We assume that the channel gain is Rayleigh distributed. Depending on the channel condition, the burst profile assigned to a node is changed by the MBS. The possible burst profiles and the associated data rates are shown in Table II.

Table 1: Physical Layer Parameters

Bandwidth	20 MHz
Number of Subcarriers	256
Frame Duration	10ms
No. of OFDM symbols / frame	844
No. of OFDM symbols / minislot	4
Total No. of minislots / frame	211
No. of minislots / frame for uplink Centr. Sched	194

Table 2: Burst Profiles

burst profile No.	Modulation	Coding Rate	Uncoded bytes per OFDM symbol	Uncoded bytes per minislot
1	QPSK	1/2	24	96
2	QPSK	3/4	36	144
3	16QAM	1/2	48	192
4	16QAM	3/4	72	288
5	64QAM	2/3	96	384
6	64QAM	3/4	108	432

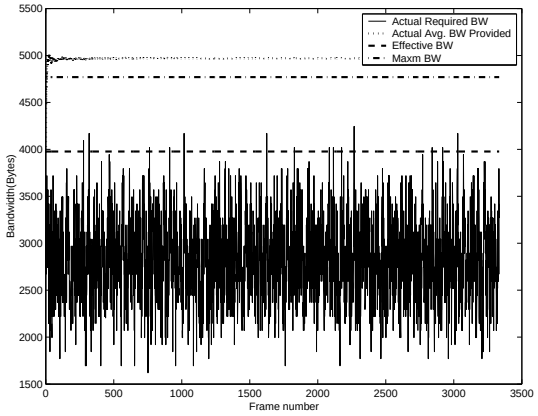

Figure 2: Comparison of Bandwidth required and Bandwidth provided

The link parameters shown in the figure are the mean data rates per slot expressed in terms of the burst profile as given in Table II.

The frame duration is 10 ms and scheduling is done over 3 frames. We consider a symmetric scenario with 3 CBR flows and 10 VBR flows at each node. The desired rate for a CBR source is 64 kbps and needs an upper bound of 60 ms and 2% on the delay and data drop probability respectively. The VBR source is characterized by a 4 state Markov chain with the rates 20 kbps, 40 kbps, 60 kbps and 80 kbps. The transition probability matrix of the Markov chain is

$$P = \begin{pmatrix} 0.4 & 0.3 & 0.2 & 0.1 \\ 0.2 & 0.4 & 0.2 & 0.2 \\ 0.2 & 0.3 & 0.2 & 0.3 \\ 0.1 & 0.4 & 0.2 & 0.3 \end{pmatrix}.$$

Each flow requires a delay bound of 60 ms and a drop-probability bound of 2%.

The routing provided by the shortest path algorithm of section 3 is also shown in Figure 1. The equivalent bandwidth of the VBR source is 66.9 kbps. The worst (maximum) average delays and drop probabilities of the CBR and VBR flows at each of the nodes is summarized in Table 4.3 (the maximum delay of 60 ms is ensured by dropping packets at the end of the frame). We can observe that all the flows

Table 3: Worst Average Delay and Drop Probability of CBR and VBR flows

Node Id	CBR Drop Prob(%)	CBR Avg. Delay(ms)	VBR Drop Prob(%)	VBR Avg. Delay(ms)
1	0	25.09	0.0048	25.40
2	0	26.00	0.0103	26.56
3	0	27.77	0.0026	28.08
4	0	27.76	0.0112	29.13
5	0	27.84	0.0144	29.69
6	0	26.49	0.0050	26.79
7	0	27.93	0.0117	28.72
8	0	24.39	0.0107	25.16
9	0	25.44	0.0116	26.08
10	0	25.63	0.0040	26.21

get their desired QoS. In Figure 2 we plot the bandwidth required and the average bandwidth provided for flows at node 4 (similar results were observed at other nodes). Also plotted is the required equivalent bandwidth and the bandwidth required if the VBR flows were provided with the maximum rates. From the plot it can be observed that the average bandwidth provided is greater than even the maximum required bandwidth. The reason for this is the large variance of the channel rate. In order to provide QoS assurances, we have to consider the worst case link rates for calculating the required number of slots. The extra bandwith although wasted here for a large fraction of time, can be used by TCP flows as will be discussed during the joint scheduling of UDP and TCP traffic.

5. QOS FOR TCP TRAFFIC

This section develops scheduling algorithms which can guarantee the QoS for TCP connections. Some TCP applications, e.g., email do not require any QoS. However web traffic and file transfer may require certain minimum throughput. Also unlike UDP connections, if there is more bandwidth available TCP connections can avail it also usefully. Therefore in addition to ensuring the minimum throughput requested, one problem we need to address is how to allocate excess bandwidth fairly to different TCP connections. Furthermore, if there is insufficient bandwidth to satisfy the minimum mean throughput of all the TCP connections, then again there is the question of how should we do the allocation in a fair way. In this case unlike the UDP connections where we recommend admission control, one other option is to give these connections less bandwidth than requested.

Initially we will consider the case of persistent TCP connections. These are long lived connections which need to send a large file. The QoS requirement for these connections is the minimum mean throughput. Later on we will also consider TCP-ON-OFF connections (see [5] for details on this model) which model the web traffic using HTTP 1.1. In this model, a TCP connection transfers multiple files. Between transfer of two files, a TCP connection may not have a file to transfer (OFF period) for sometime. This is the dominant traffic type in the current Internet. An appropriate QoS requirement for these connections in the mean file download time.

Let λ_i be the minimum throughput requirement of all TCP connections originating at node i. The routing of this traffic will be via the shortest path routing discussed in Section 3 (We will discuss the validity of the assumptions in Section 3 later in this section). Once the routing is fixed, we know the total minimum TCP throughput needed at each node (including the TCP traffic from other nodes). Thus we address the problem of slot allocation to different nodes to satisfy this requirement. As explained above, unlike in Section 4, we will consider this problem here, taking into the account the fairness issues. First we compute the (fixed) number of slots to be allocated at each node j for TCP traffic generated at node i. Later on, we also use the channel conditions to adaptively change the slot allocation.

5.1 Fixed Allocation Scheme

This scheme allocates a fixed number of slots per frame to each node depending upon the average data arrival rate and the estimated average channel rate. Let λ_i be the minimum throughput (in bytes per frame) required by TCP traffic generated at node i and let $E[r(i)]$ be the mean rate of link i. Let $n_{i,j}$ be the number of slots to be allocated at node j for traffic of node i. These should satisfy the constraint

$$\sum_{i=1}^{M}\sum_{j=1}^{M} n_{i,j} \;\leq\; N. \qquad (C1)$$

We will also require our solution to satisfy

$$n_{i,p_{i,j}}\; E[r(p_{i,j})] = n_{i,i}\; E[r(i)] \qquad \forall j = 1,\ldots,h_i \text{ and } i = 1,\ldots,M \qquad (C2)$$

where, the data of node i is routed through nodes $\{p_{i,1} = i, p_{i,2}, \ldots, p_{i,h_i}\}$. The set of constraints (C2) in the above optimization problem are required in order to ensure that the throughput (channel capacity) provided for the flows from a node at nodes along the route to the destination is the same. Since the end to end throughput is atmost equal to the minimum channel capacity (bandwidth) available for the flows along the route, the above constraint avoids wastage of resources (due to higher capacity available at certain nodes which will not be used).

Next we want our solution to satisfy the constraints of proportional fairness,

$$\frac{n_{i,i}\, E[r(i)]}{\lambda_i} = \frac{n_{1,1}\, E[r(1)]}{\lambda_1}, \qquad \text{for all } i = 1 \ldots M. \qquad (5)$$

This will ensure that the throughput provided to the traffic of node i is propotional to its minimum requirement λ_i. This seems to be a reasonable criterion of fairness whether we are providing the TCP traffic of different nodes more or less throughput than its minimum requirement. Combining (C1), (C2) and (5),

$$n_{1,1} = \frac{N}{(\sum_{i=1}^{M} \frac{\lambda_i}{\lambda_1} \sum_{j=1}^{h_i} \frac{E[r(1)]}{E[r(k_j)]})} \qquad (6)$$

with $n_{i,j}$ provided by (5) and (C2). It is a propotionally fair allocation which will provide the maximum throughput to different nodes.

Let $n(j) = \sum_{i=1}^{M} n_{i,j}$. This is the number of slots allocated by the algorithm to node j for the total TCP traffic passing through it. To provide QoS, it will not be sufficient to just provide n_j slots to node j in each frame. This will not ensure that the TCP traffic generated at node i will get its share of $n_{i,j}$ slots at node j. In fact it is known (see e.g. [13]) that if all the TCP packets at a node are served on an FCFS basis, then the TCP connections with fewer hops on their route get most of the throughput. Therefore for TCP traffic originating at node i, we form a separate queue at each node on its route. Via WRR out of the total allocation of n_j slots, we provide $n_{i,j}$ slots to this queue at node j.

The problem with the above Fixed Allocation Scheme is that, slots can be assigned to links which are in bad state or can be assigned to nodes that do not have enough data to transmit at a given time. In the Adaptive Fixed Allocation scheme that we develop now, we use the instantaneous channel state and queue length information to overcome these shortcomings (remember the MBS has this information).

5.2 Adaptive Fixed Allocation Scheme

The number of slots to be allotted to each node in a frame to satisfy its throughput requirement is calculated as in the fixed allocation scheme provided above. Next in each frame we compare the instantaneous link rate $r_k(i)$ with a predefined threshold parameter $R_{th}(i)$ and declare the link to be bad if $r_k(i) < R_{th}(i)$. In a scheduling frame, let $\mathcal{G}$, $\mathcal{B}$ denote the set of good and bad links respectively. The idea is to defer the allocation of slots to links which are bad or which do not have enough data to fully utilize the slots, while at the same time ensure that the node gets the required throughput in the long term. In order to achieve the latter objective, we maintain a counter $c(i)$ for the number of slots to be assigned to node i in order to compensate for the missed slots. We call this the credits for node i. In order to avoid starving nodes due to extended period of their channel being bad, we impose an upper bound $C_{Lim}(i)$ on the accumulated credits, beyond which a slot is assigned to the node irrespective of the channel condition.

More formally, let $c_k(i)$ be the accumulated credits, $r_k(i)$ the link rate and $Q_k(i)$ the queue length of node i in frame k. If $r_k(i) \geq R_{th}(i)$ then $i \in \mathcal{G}$, else $i \in \mathcal{B}$. The slot allocation process is done in two rounds. Let $n(i)$ be the total number of slots allocated to node i (for the overall traffic through it) by the fixed allocation scheme. Define

$$\widetilde{n}_{k,1}(i) = \begin{cases} n(i) & \text{if } i \in \mathcal{G} \\ (c_k(i) + n(i) - C_{Lim}(i))^+ & \text{if } i \in \mathcal{B} \end{cases}$$

Then the number of slots allotted to node i in frame k in the first round is

$$n_{k,1}(i) = \begin{cases} \min\,(n(i), \text{ceil}\,(\frac{Q_k(i)}{\widetilde{n}_{k,1}(i)})) & \text{if } \widetilde{n}_{k,1}(i) > 0 \\ 0 & otherwise \end{cases}$$

where ceil(x) is the smallest integer greater than x. The total number of slots allotted in the first round is $n_{k,1} = \sum_{i=1}^{M} n_{k,1}(i)$. The remaining $N - n_{k,1}$ slots are allotted in the second round by giving first preference to nodes with good links, positive credits and maximum data to transmit (in decreasing preference of these attributes). The second preference is given to good links with maximum data to transmit. If there are no nodes that satisfy the above conditions then bad links with maximum data to transmit are allotted the slots. The allocation is done one slot at a time, recalculating the credits and remaining data to transmit after each slot. Let the total number of slots allotted to node

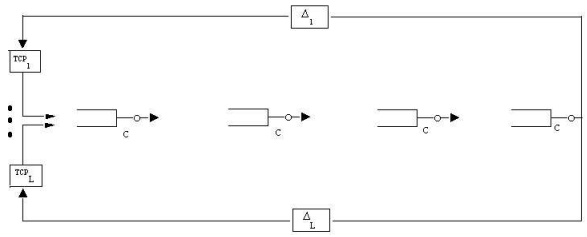

Figure 3: Multiple TCP flows through multiple queues with fixed rates

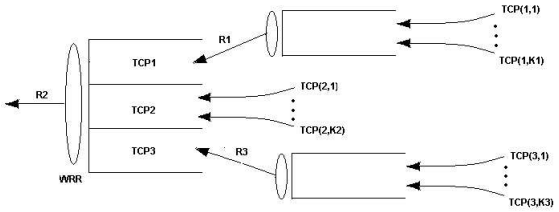

Figure 4: WRR used to provide the required throughput to the aggregate of TCP flows

i in this round be $n_{k,2}(i)$. Then the credit is updated as

$$c_{k+1}(i) = \min(c_k(i) + n(i) - (n_{k,1}(i) + n_{k,2}(i)), C_{Lim}(i)).$$

A slightly different approach is to not consider the queue lengths at the nodes and to set $n_{k,1}(i)$ to $\widetilde{n}_{k,1}(i)$ in the first round. In the second round we keep the same order of preference without the maximum data transmission condition. We call this scheme Channel Adaptive Fixed Allocation Scheme.

The parameters $(R_{th}(i), C_{Lim}(i))$ associated with each node i affect the performance of the above algorithm. We can optimize these parameters along with $n(i)$'s for some system performance (e.g. maximizing overall TCP throughput in the system). We have solved this optimization problem via Stochastic Approximation in [13].

Once we have obtained the slot allocation for each node j via one of the above (fixed or adaptive) algorithms, we provide the $n_{i,j}$ slots to traffic of node i via WRR, as we explained at the end of Section 5.1.

We compare the performance of these algorithms in Section 5.4.

5.3 Providing QoS to TCP flows

The recipe provided so far will ensure that the TCP traffic originating at a node will get its share of bandwidth along the route. Next we need to ensure that the different TCPs sharing the same links get the throughput they want. Consider the system shown in Fig. 3. Let there be L TCP connections passing through (say) four queues. TCP_i has window size W_i (assume it is fixed) and propagation delay Δ_i (representing delays in the rest of the network). At each queue the link speed is c (ensured by WRR discussed above). In this scenario the packets/acks of different TCPs will be either at the first queue or propagating in the rest of the network (in propagation pipes Δ_is). If TCP_i gets a throughput of λ_i packets/sec, then by Little's law it has on the average (under stationarity - proved in [5]) $\lambda_i \Delta_i$ packets in the propagation pipe. Thus the average queue length in the first queue is $\sum_{i=1}^{L}(W_i - \lambda_i \Delta_i)$. If TCP_i has packet length s_i (in bytes) the mean queueing delay in the first queue is $\sum_{i=1}^{L}(W_i - \lambda_i \Delta_i)\, s_i/c$. Since there will be no queueing delays in the other queues, the total mean round trip time of TCP_i is approximately $\frac{1}{c}\sum_{j=1}^{L}(W_j - \lambda_j \Delta_j)\, s_j + \frac{3\, s_i}{c} + \Delta_i$. Thus the total throughput obtained by TCP_i is

$$\frac{W_i\, c}{\sum_{j=1}^{L}(W_j - \lambda_j \Delta_j)\, s_j + 3\, s_i + \Delta_i\, c} \text{ packets/sec.} \qquad (7)$$

To provide a desired throughput (or minimum throughput) to different TCPs, one needs to adjust c, W_j, s_j appropriately in (7) such that it becomes equal to the desired throughput for each i. The bandwidth c (in bytes/sec) should equal $\sum_{i=1}^{L} \lambda_i s_i$. It is possible that a service provider may not have the freedom to choose W_i and s_i for different TCP connections (these are selected by the receivers and the networks through which the connections are passing). However even though the maximum window size W_i may not be controllable, if it is large enough, its mean size can be reduced to an appropriate size by dropping its packets in a controlled way (say via RED [4]) such that the required throughput can be provided to the TCP. We explain this in the following.

Let us fix a desired queueing delay of d^* sec in the first queue. Define for each i, $\widetilde{\Delta_i} = \Delta_i + \frac{3\, s_i}{c}$. We fix the desired mean window size of $\mathrm{E}[W_i]$ such that

$$\frac{E[W_i]}{d^* + \widetilde{\Delta_i}} = \lambda_i \quad \text{for each } i \qquad (8)$$

Now we use RED control for each TCP connection i and specify its RED parameters such that at average queue length $d^*\, c$, it will drop the packets of TCP_i with probability p_i, where

$$p_i = \frac{8}{(3\,(E_\pi[W_i] + 4)^2 + 5)}$$

for each i. Then it can be shown (see [12] and [13]) that this system will operate under steady state such that the first queue will have the mean queue length $d^*\, c$ and each of the TCPs will have their mean window size $\mathrm{E}[W_i]$ satisfying the above requirements. Furthermore, TCP_i will get the throughput λ_i packets/sec.We will verify these claims via simulations in section 5.4.

It is also shown in [8] and [11], that the TCP connections can be grouped such that one needs only a few RED parameters to take care of the throughput requirements of different TCPs.

Using the above ideas, finally we present the overall scheme to provide the required throughputs to different TCP connections in the mesh network. We consider TCP persistent connections first. Let N_i^P TCP connections be entering the mesh network from node i. Let $\lambda_{i,j}$ be the minimum throughput requirements (in packets/sec) and $s_{i,j}$ the packet lengths (in bytes) of the j^{th} TCP connection originating at node i. Thus the total throughput requirement of the TCP connections originating at node i is $\lambda_i = \sum_{j=1}^{N_i^P} \lambda_{i,j} s_{i,j}$ bytes/sec. We use the shortest path routing developed in Section 3. Then the total throughput requirement of node i taking into account the TCP connections from other nodes is $\bar{\lambda}_i = \sum_{j=1}^{m_i} \lambda_{a_{i,j}}$. Now we use the Fixed and Adaptive Fixed

algorithms to compute the number of slots needed to ensure that each node gets the throughput $\bar{\lambda}_i$ it needs. To ensure that the TCP connections originating at different nodes get their share at each of the nodes along the route, we will have a separate queue at each node for the TCP traffic arriving from different nodes. Each of these queues is provided their share of bandwidth by a node via WRR (see Figure 4). Finally, at each node, for the TCP connections originating at it, there will be RED control to ensure that each individual TCP connection gets its required throughput.

We now consider TCP-ON-OFF traffic. Let N_i^O TCP-ON-OFF flows be entering the mesh at node i. For simplicity assume all of them to have the same mean download time requirement of T_i^{on} (this is the mean time that will be taken to download a file by such a connection) and the same mean number of packets D_i to be downloaded. Let the packet size of a connection be S_i. Let the mean time between two downloads be T_i^{off}. Then the throughput required by such a TCP flow to satisfy its QoS requirement is $\lambda_i^m = \frac{D_i S_i}{T_i^{on}}$. Also the long term average throughput required by this flow is $\lambda_i^a = \frac{D_i S_i}{(T_i^{on}+T_i^{off})}$. The probabilty of a connection being ON is $\frac{T_i^{ON}}{T_i^{on}+T_i^{off}}$. Making the assumption that the ON-OFF process of different connections are independent, the mean number of connections ON at anytime is $\frac{N_i^O T_i^{on}}{T_i^{on}+T_i^{off}}$. Now since each connection requires a throughput of λ_i^m when ON, the total throughput requirement of ON-OFF traffic at node i is

$$\lambda_i = \Big(\frac{N_i^O T_i^{on}}{T_i^{on}+T_i^{off}}\Big)\lambda_i^m = N_i^O \lambda_i^a.$$

The above approximation improves as the number of connections N_i^O increases. The RED parameters for individual flows are calculated such that each of the TCP-ON-OFF connections gets a throughput of λ_i^m during its ON time.

Once we have ensured that the overall TCP traffic originating at the different nodes gets the bandwidth it requires at each node on its route, to ensure that the different TCP connections in it get the throughput they want, we set the window sizes according to (7). To provide the needed mean window size, as explained above we can use RED at the bottleneck node along the route of the TCP connections. If the link rates are all fixed, then the node through which the TCP flows enter the mesh network is the bottleneck. However due to the random variation of the link rates in our case, any link along the route can momentarily turn into a bottleneck (whenever its channel state is poor). To make our scheme work, one method is to implement RED control at all nodes along the path of the TCP flows. However this involves difficulties in practical implementation since every node has to acquire the RED parameters of all the flows passing through it. The second method is to force the ingress node to be the bottleneck by providing about 3-5% extra bandwidth to the flows at the other nodes along the route. This extra bandwidth whenever not used by these TCP flows, can be provided to the best effort traffic.

Let us again comment on the validity of the assumptions made in Section 3 to choose the shortest path routing. Although the traffic generated by a TCP connection is far from i.i.d., we claim that the superposition of a large number of TCP connections controlled by RED may be approximated by an i.i.d. (or stationary) sequence. This happens due to RED control. The packets of different TCPs are dropped randomly and asynchronously by RED. Thus, the window sizes of different TCPs will increase and decrease asynchronously, causing a more uniform overall traffic. Regarding the infinite buffer case, we have already ensured that there is no queueing of TCP traffic at later nodes on its route. On the first queue, due to RED control, buffer overflow can be minimized. Thus, effectively one sees infinite buffers at each queue (with packets dropped by RED only).

Table 4: Comparison of Scheduling schemes for TCP Traffic: Number of TCP flows with the difference in throughput (TCP-Perst) and Mean Download Time (TCP-ON-OFF)

Diff.	Fxd. Alltn.		Chan. Adapt.		Adapt. Fxd.	
	pers	on off	pers	on off	pers	on off
> -50	0	0	0	0	0	0
-50 to -25	0	10	0	14	0	15
-25 to -10	2	24	0	30	0	32
-10 to 0	6	4	7	5	5	10
0 to 10	60	19	47	14	46	13
10 to 25	49	30	45	29	48	26
25 to 50	3	13	21	8	17	4
50 to 100	0	0	0	0	4	0
> 100	0	0	0	0	0	0

5.4 Simulations

We consider the mesh network shown in Fig. 1. There are 12 TCP persistent flows originating at each node, 6 in the uplink and 6 in the downlink. There are three classes of traffic with 2 flows each at a node. The throughput requirement of the three classes of traffic are 40kbps, 80kbps and 120kbps. There are also 10 TCP-ON-OFF flows originating at each node, all in the uplink. There are two classes of TCP-ON-OFF flows with 5 flows in each class. Class 1 has $T^{on} = 2$s, $T^{off} = 4$s and $D = 25$ packets. Class 2 has $T^{on} = 3$s, $T^{off} = 5$s and $D = 50$ packets. The mean packet size of all flows is 1000 bytes. The delays Δ_i's experienced by the flows in the external network are arbitrarily fixed between 0 to 60ms. The average window sizes of different TCPs (persistent and ON-OFF) were computed according to (8) and controlled through RED. A comparison of the different scheduling schemes developed in Sections 5.1 and 5.2 is provided in Table 5.4 in terms of percentage difference between the required throughput and the throughput achieved for TCP persistent flows (-ve shows throughput achieved is less than the desired throughput) and the required download time and the download time achieved (-ve shows that the achieved mean download time is less than the required mean time) for TCP-ON-OFF flows.

We observe that all the fixed schemes satisfy QoS of most of the users. However, the adaptive fixed allocation scheme provides more than the minimum required throughput to most of the flows. This implies that more flows can be supported with this scheme.

6. JOINT SCHEDULING OF UDP AND TCP FLOWS

In this section we address the problem of scheduling in

presence of both UDP and TCP traffic. The requirements of UDP traffic were discussed in Section 4 and of TCP traffic in Section 5. From the arguments in these sections, in order to provide QoS to UDP we had to consider the worst case channel conditions whereas for TCP we had to consider the average channel rates. From Figure 2, we could see that there was a huge difference between the average bandwidth requested and the average bandwidth provided (to guarantee the QoS of the CBR and VBR connections). Here we utilize this bandwidth for scheduling of TCP flows.

We provide priority to UDP traffic over TCP traffic in the network. It has been observed in [5] that by doing this the delays experienced by UDP flows can be drastically reduced without affecting the throughput of the TCP flows. Also, in the present context, it will allow us to save resources by using (9) below.

The total number of slots alloted to the nodes of the TCP and UDP traffic is calculated as follows. Let $n_U(i)$ denote the total number of slots to be allotted to node i, to meet the QoS requirements of all the UDP flows passing through node i, as calculated in Section 4. Let $\lambda_U(i)$ denote the average throughput required by all UDP (CBR and VBR) flows from node i. Let $\lambda_T(i)$ denote the total throughput required to satisfy the QoS requirement of all TCP flows passing through node i. Then the average total throughput required by node i is $\lambda(i) = \lambda_T(i) + \lambda_U(i)$. We now use the Fixed Allocation Scheme discussed in Section 5 to calculate the number of slots $n_T(i)$, required to provide a mean throughput of $\lambda(i)$ to node i. Finally we allocate

$$n(i) = \max\left(n_T(i), n_U(i)\right) \tag{9}$$

slots to node i per frame. Since UDP has higher priority, this will ensure QoS to all UDP connections while TCP connections get their average throughput. Also the wastage of bandwidth is minimized.

Channel Adaptive Fixed Allocation scheme can also be used for scheduling as follows. In order to satisfy the requirements of the UDP flows, $n_U(i)$ slots have to be allotted to node i in every frame. Now, depending upon the channel conditions we can defer the allocation of the other $(n(i) - n_U(i))^+$ slots.

We comment on the scalability of our overall scheme. At each node we need two queues for each node whose traffic is passing through is passing through this node. In one mesh the number of nodes will not be large (it will be a few tens) and hence even in a large overall network this will not cause any problem. We also commented earlier that the RED control for TCP traffic can be used only at the node it enters. Also as in [8] and [11], we could classify TCP connections according to the mean window size (equivalently, their probability of loss) they need. Thus, even at the entry node we do not need to provide different RED parameters for each TCP but rather for each TCP class. The number of TCP classes needed will not be more than twelve (as shown in [8], [11] although one can work with even fewer). Furthermore, the computational complexity of our algorithms is rather low and hence these can be implemented easily in real time for a reasonably large network.

6.1 Simulations

We consider the network shown in Figure 1. The network characteristics are summarized in Section 4.3. The frame duration is 10ms and scheduling is done over 3 frames. We consider a symmetric scenario with 3 CBR flows and 3 VBR flows at each node. The desired rate for a CBR source is 64kbps and needs an upper bound of 60ms and 2% on the delay and fraction of data dropped. The VBR source is characterized by a 4 state Markov chain with the rates 20 kbps, 40 kbps, 60 kbps and 80kbps. The transition matrix of the Markov chain is the same as in Section 4.3. Each flow requires a delay bound of 60ms and drop probability bound of 2%. There are 12 TCP persistent flows, 6 in the uplink and 6 in the downlink. There are three classes of traffic with 2 flows each at a node. The throughput requirement of the three classes of traffic are 42kbps, 84kbps and 126kbps. The packet lengths of different connections are as before. The delays Δ_i experienced by the flows in the external network are arbitrarily fixed between 0 to 60ms.

Fixed Allocation scheduling scheme is used in the simulations. The worst (maximum) average delays and drop probabilities of all the CBR and VBR flows in the network are summarized in Table V. The performance of the TCP flows is presented in Table VI. We can observe that almost

Table 5: Performance of UDP Flows

	CBR Flows	VBR Flows
Max Avg Delay	9.16 ms	9.48 ms
Max Drop Prob	0.0002%	0.0002%

Table 6: Performance of TCP Flows

Percent Error	Number of Flows
< -25	0
-20 to -10	2
-10 to 0	31
0 to 10	54
10 to 20	19
20 to 50	14
> 50	0

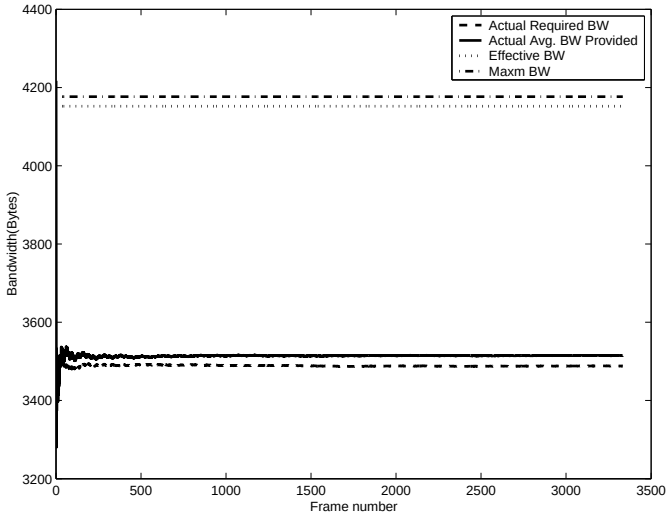

Figure 5: Comparison of Bandwidth required and Bandwidth provided

all the flows get their desired QoS. In Figure 5 we plot the average bandwidth required and the average bandwidth provided for flows from node 4 (similar results were observed at other nodes). Also plotted is the required bandwidth as obtained from the equivalent bandwidth analysis and also the bandwidth required if the VBR flows were provided with the maximum rates. As compared to Figure 2, we observe that the available resources are almost fully utilized here.

We have simulated several other networks with different traffic mixes and have made similar observations.

7. ADMISSION CONTROL

With centralized scheduling, all the connection admission control decisions are made at the MBS. Each node has to convey to the MBS its requirements for UDP and TCP traffic separately (currently the standards support only a single request. This limitation can be overcome by using some form of coding in the request message). Let $n_T(i,j)$ and $n_U(i,j)$ be the number of slots required to satisfy the total average data rate (TCP and UDP) and meet the QoS requirements of the UDP flows respectively, of node i at node j. Then, as discussed in the previous section, the number of slots allotted for node i at node j, is $n(i,j) = \max(\, n_T(i,j), n_U(i,j)\,)$. Suppose now a new TCP connection request for a bandwidth of $\lambda_n(i)$ arrives. In order to accomodate this connection, the MBS has to allocate resources to all nodes along the route to the MBS. Since implementation of the fixed allocation scheme is easy, the MBS applies this scheme first. The number of slots required at the node $p_{i,j}$ for the new request is

$$n_n(i,p_{i,j}) = \Big(\frac{\lambda_n(i)}{E[\,r(p_{i,j})\,]} - (n_U(i,p_{i,j}) - n_T(i,p_{i,j}))^+\Big)^+.$$

The connection is accepted if

$$\sum_{j=1}^{h_i} n_n(i,p_{i,j}) \leq (\, N - \sum_{i=1}^{M} n(i)\,).$$

If the above condition is not met, then the Adaptive Allocation scheme discussed in Section 5 is used (since it has a larger stability region). If we do not get additional resources even with this algorithm, the new request is rejected.

The arrival of a new UDP connection along with its QoS requirements is conveyed to the MBS. If the arriving connection generates VBR traffic, then the MSS determines the class to which it belongs, recomputes the effective bandwidth required for that class and conveys it to the MBS. The MBS calculates the number of slots required to satisfy its QoS requirements as in Section 4. If the total number of slots required is less than the number of slots available then the connection is accepted; otherwise not.

8. CONCLUSIONS

In this paper we have designed efficient and practically implementable algorithms for routing and centralized scheduling in IEEE 802.16 mesh networks. We have considered providing end to end QoS to different flows in the network. We have handled UDP and TCP traffic separately at first and then considered them jointly. Our algorithms are able to provide QoS to real and nonreal time individual flows efficiently. We have also provided an admission control policy which is an important part of any QoS framework.

9. REFERENCES

[1] 802.16d-2004, "Draft IEEE Standard for Local and Metropolitan area networks – Part 16: Air Interface for Fixed Broadband Wireless Access Systems", May 2004.

[2] M. Andrews and L. Zhang, " Routing and scheduling in multihop wireless networks with time-varying channels", *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan 2004.

[3] M. Ergen, S. Coleri and P. Varaiya, "QoS aware adaptive resource allocation techniques for fair scheduling in OFDMA based broadband wireless access", *IEEE Transactions on Broadcasting*, Vol. 49, No. 4, Dec 2003.

[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance", *IEEE/ACM Trans. Networking*, Vol. 1, 1993, 397-413.

[5] A. Gupta, " A unified approach for analyzing persistent, non-persistent and ON-OFF TCP sessions with RED control and exogenous traffic ", M.Sc. Thesis, Dept. of ECE, IISc, Bangalore, 2002.

[6] M. Hawa and D. W. Petr, "Quality of service scheduling in cable and broadband wireless access systems", *10th IEEE International Workshop on Quality of Service*, May 2002, pp. 247-255.

[7] I. Hsu and J. Walrand, " Admission control for multiclass ATM traffic with overflow constraints", Computer Networks and ISDN Systems, Vol. 28, 1996, 1739-1751.

[8] V. Kamble and V Sharma, " A simple approach to provide QoS and fairness in Internet", in *Proc. International Conf. on Signal Processing and Communications (SPCOM 2004)*, Bangalore, Dec. 2004.

[9] A. Kumar, D. Manjunath and J. Kuri, "Communication Networking: An Analytical Approach", Morgan Kaufmann Publishers, 2004.

[10] S. Ramanathan and E. L. Llyod, "Scheduling algorithms for multihop radio networks", *IEEE/ACM Transactions on Networking*, Vol. 1, Apr. 1993, pp. 166-177.

[11] V. Reddy, V. Sharma and M B Suma, "Providing QoS to TCP and real time connections in the Internet", *Queueing Systems*, Vol. 46, 2004, 461-480.

[12] V. Sharma and P. Punyaslok, " Stability and analysis of TCP connections with RED control and exogenous traffic"', in *Queuing Systems*, Vol 48, 193-235, 2004.

[13] H. Shetiya, " Efficient routing and scheduling algorithms for IEEE 802.16 mesh networks ", M.E. Thesis, Dept. of ECE, IISc, Bangalore, 2005.

[14] V. Singh, "Efficient Scheduling of Uplink and Downlink in WiMaX networks ", M.E. Thesis, Dept. of ECE, IISc, Bangalore, 2005.

[15] Jean Walrand, "An Introduction to Queueing Networks", Prentice Hall, 1988.

[16] I. C. Wong, Z. Shen, B. L. Evans and J. G. Andrews,"A low complexity algorithm for proportional resource allocation in OFDMA systems", SIPS, 2004.