Dynamic Bandwidth Management for Single-hop Ad Hoc Wireless Networks

Samarth H. Shah, Kai Chen and Klara Nahrstedt *
Department of Computer Science
University of Illinois at Urbana-Champaign
Email: {shshah, kaichen, klara}@cs.uiuc.edu

Abstract

Distributed weighted fair scheduling schemes for QoS support in wireless networks have not yet become standard. In this paper we propose an Admission Control and Dynamic Bandwidth Management scheme that provides fairness in the absence of distributed link level weighted fair scheduling. In case weighted fair scheduling becomes available, our system assists it by supplying the scheduler with weights and adjusting them dynamically as network and traffic characteristics vary. To obtain these weights, we convert the bandwidth requirement of the application into a channel time requirement. Our Bandwidth Manager then allots each flow a share of the channel time depending on its requirement relative to the requirements of other flows in the network. It uses a max-min fairness algorithm with minimum guarantees. The flow controls its packet transmission rate so it only occupies the channel for the fraction of time allotted to it by the Bandwidth Manager. As available bandwidth in the network and the traffic characteristics of various flows change, the channel time proportion allotted also dynamically varies. Our experiments show that, at the cost of a very low overhead, there is a high probability that every flow in the network will receive at least its minimum requested share of the network bandwidth.

1 Introduction and Motivation

The IEEE 802.11 Medium Access Control (MAC) protocol for wireless networks does not currently consider the actual requirements of flows competing for limited wireless bandwidth when scheduling them. As a result, some flows get far more throughput than minimally required while others cannot even satisfy their minimum requirements. Much research has gone into the area of distributed weighted fair scheduling (DWFS) [15, 3, 12, 13, 10] for the IEEE 802.11 MAC protocol operating in the Distributed Coordination Function (DCF) mode, which is a first step towards providing flows with their desired quality-of-service (QoS). In DWFS, each flow has a weight which defines its bandwidth requirement relative to that of other flows. A scheduler combined with the link level IEEE 802.11 protocol then schedules the flows so their received throughput is proportional to their weights.

Our goal is to determine the flow weights and adjust them as network and traffic characteristics change. In the absence of link level DWFS in current IEEE 802.11-based products, however, we use our own application level mechanism to limit the flows' throughput based on their respective weights. In other words, we use our own mechanism to enforce the flow weights we determined, in the absence of a link level enforcement mechanism. In case DWFS became available at the link level, our scheme would still be required to provide it with the flow weights. But in this case, the link level scheduler can enforce the flow weights.

The exact share of network bandwidth allotted to a flow depends on its requirements relative to the requirements of other flows. We propose the use of a centralized *Bandwidth Manager* (BM) which obtains from each flow its bandwidth requirements at the beginning of a connection. It uses this information to gauge what proportion of channel time each flow should be allotted. It returns the channel time proportions to the respective flows. The channel time proportion is defined as the fraction of unit time for which a flow can have the entire channel to itself. Since our network model allows only one node to transmit on the channel at a time, there is a direct correspondence between the channel time a flow uses and the share of the network bandwidth it receives. The BM may also *refuse* to admit a flow, i.e. allot 0% channel time. This can happen if the flow's requirement is so large that it either exceeds total network capacity or is likely to starve other flows of channel time.

The problem with the admission control solution described above is that it is a one-time procedure performed before the flow starts. It does not take into account the

*This work was supported by the ONR MURI program under grant NAVY CU 37515-6281, and the NSF EIA 99-72884 grant. Any opinions, findings and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

changes in the wireless network over the duration of the flow's operation. Not only can the overall throughput of the network vary over time due to changing system-wide interference, but the total network capacity as perceived by different nodes in the network at the *same* time can also be different. The latter phenomenon is due to errors and interference that are *location-dependent*. Furthermore, as shown in [6], the overall throughput of the network can also vary dynamically as flows arrive and depart, and the number of active stations changes.

The BM must therefore not just deal with admission control and teardown, but also perform *dynamic bandwidth management* when the total available bandwidth in the network, as estimated by the individual nodes, changes. Some bandwidth re-negotiation may also need to be done when a flow changes its packet transmission rate, as in the case of bursty VBR traffic.

The rest of the paper is organized as follows. The next section describes the network model, the architecture of our bandwidth management system, and the bandwidth management protocol. Section 3 presents our experimental results. This is followed by Section 4 which describes related work in the field. Section 5 concludes the paper.

2 Bandwidth Management System

In the previous section, we motivated the need for admission control coupled with dynamic bandwidth management in a wireless network. In this section, we describe the characteristics of the network we are concerned with, the architecture of the bandwidth management system, and the communication protocol.

2.1 Network Model

We design and implement our bandwidth management scheme for a wireless network consisting of heterogeneous computers and devices connected together over the IEEE 802.11 wireless MAC protocol. Unlike in [5], where a base-station determines the schedule of transmission for the entire network and all communication is via the base-station, in our network, transmission is distributed and peer-to-peer. The network we use consists of handheld PCs and laptop computers with their 802.11 interfaces configured in peer-to-peer ad hoc mode. Each node in our network is within the transmission range of every other node. Hence, only one node can transmit at a time over the channel, else a collision occurs. Since every receiver is within its corresponding sender's transmission radius, routing is single-hop. The IEEE 802.11 MAC protocol's DCF, which is relevant to our network model, does not have a provision for a fixed transmission schedule. A node can send any time it senses that the channel is not busy. A distributed binary exponential backoff mechanism resolves collisions that might occur as a result of nodes transmitting at random times. Moreover, any node in the network can transmit to any other node directly without using the base-station as an intermediary hop. The distributed, peer-to-peer and ad hoc nature of our wireless network model makes the bandwidth management problem significantly harder to solve than in the case of a base-station co-ordinated network where the base-station has full control of the contending flows.

Applications of a single-hop ad hoc wireless network include smart-rooms and "hot spot" networks. In such applications, wireless hosts in a small area share limited channel bandwidth. Since the area is small, the wireless hosts are all within each other's transmission range. Communication is *pervasive*, i.e. there are a large number of source-destination pairs distributed throughout the network. So, channeling all data through a single bottleneck, such as a base-station, is inefficient. A source must be able to talk directly to its corresponding destination. Furthermore, in military and disaster rescue environments, networks of small, handheld wireless devices need to be constructed quickly, with no time for planning and building a support infrastructure of base-stations. Thus, the infrastructure-less single-hop ad hoc wireless network accurately represents the network used in a pervasive computing environment.

We assume that the wireless network has one node selected to host the *Bandwidth Manager* (BM) entity in our approach. We choose one of the more resource-rich nodes in the network, i.e. one of the laptops, as the host system for the BM. The IP address and port number of the BM program are either well-known in the wireless network or are obtained through service discovery.

We assume a network has a set of flows F. Each flow $g \in F$ is uniquely identified by its source IP address, source port number, destination IP address and destination port number. We call this unique identifier the *flow-id* of the flow. A new flow f registers with the BM before beginning its transmission. The application initiating flow f has a minimum bandwidth requirement $B_{min}(f)$ and a maximum bandwidth requirement $B_{max}(f)$. The flow f also has an estimate of the total network bandwidth $B_p(f)$. It uses these values to obtain its minimum and maximum channel time proportion requirements, $p_{min}(f)$ and $p_{max}(f)$ respectively. It sends these values to the BM at registration time. In response, the BM adds flow f to set F and allots it a certain channel time $p_a(f)$, when the flow is admitted. Flow f then uses this allotted channel time proportion $p_a(f)$ to calculate its transmission rate. It transmits using this transmission rate until either it stops or until a new $p_a(f)$ value is allotted to it. A new $p_a(f)$ could be allotted to it when there is a change in the network characteristics or in the traffic characteristics of some other flow.

We assume that the flows in the wireless network are

well-behaved and co-operative, i.e. they will refrain from exceeding their allotted channel share and will release any channel share allotted to them when they stop. If the flows are not well-behaved and co-operative, then a policing mechanism must be used to detect the "rogue" flows and punish them.

2.2 Bandwidth Management System Architecture

The architecture of the bandwidth management system consists of three major components as shown in Figure 1: (a) the per-flow Rate Adaptor (RA) at the application level, (b) the per-node Total Bandwidth Estimator (TBE) at the link level and (c) the Bandwidth Manager (BM), which is unique in the entire wireless network. Our system takes advantage of *cross-layer interaction* between the application and link levels.

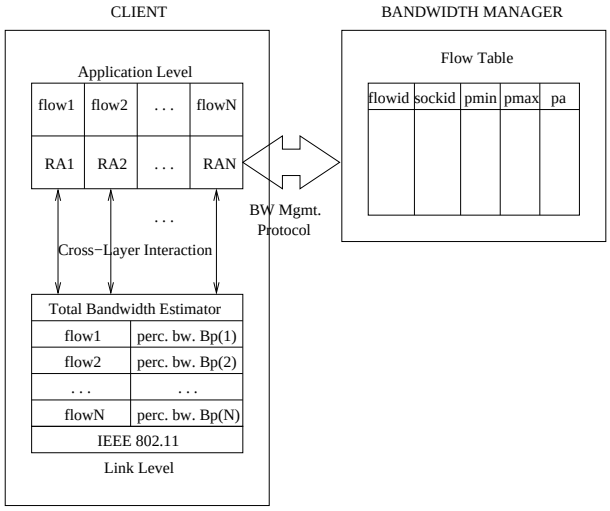

Figure 1. Bandwidth Management System.

Rate Adaptor (RA): In our design, we assume the absence of distributed weighted fair scheduling (DWFS) at the link level. Hence, a flow's bandwidth consumption in accordance with its allotted channel time proportion is regulated only by the per-flow Rate Adaptor (RA) at the application level; the link level scheduling scheme is unable to actively adjust the rate of the flow. The RA controls the packet transmission rate of the flow it is specific to, depending on the channel time proportion allotted to the flow by the BM. In our prototype implementation, for the sake of simplicity, we made the RA a part of the application itself.

Even if DWFS is provided at the link level, a Rate Adaptor at the application level is still required. However, its function is deprecated to merely communicating the flow's channel time requirements to the BM, receiving the allotted channel time proportion, and passing this weight to the link level weighted fair scheduler. The actual rate control will be done by the weighted fair scheduler at the link level.

Total Bandwidth Estimator (TBE): The per-node Total Bandwidth Estimator is co-located with the IEEE 802.11 protocol at the link level. It estimates the total network bandwidth $B_p(f)$ for each flow f sourced at the node it resides on. $B_p(f)$ is what flow f *perceives* to be the total bandwidth of the network, at a particular instant in time. It is equal to the *actual* total bandwidth of the network (2 Mbps, 5.5 Mbps or 11 Mbps for IEEE 802.11) minus the bandwidth lost due to interference and contention experienced by flow f's packets at that instant. The interference and contention at an instant in time is estimated from the interference and contention experienced in recent history. Flow f's packet size also affects its estimate $B_p(f)$ of the total bandwidth of the network. Details of the estimation method of $B_p(f)$ are in Section 2.4.

The TBE *continuously* measures the total perceived bandwidth for each flow. When the total bandwidth $B_p(f)$ perceived by flow f changes, the RA of f must re-negotiate the channel time proportion for f. This is because the channel time proportion allotted to flow f is directly related to its share of total network bandwidth. If flow f perceives the total network bandwidth as having decreased, its proportion will also correspondingly decrease, perhaps causing it to fail to meet its minimum bandwidth requirements.

Example: Assume a flow f in a 2 Mbps wireless network has minimum bandwidth requirement 300 Kbps and perceives total network bandwidth of 1.5 Mbps. (That is, the flow f perceives this to be the total capacity of the 2 Mbps channel.) Assume further that the channel time proportion allotted to it is 20%, thus ensuring it just meets its minimum bandwidth requirement. If the total network bandwidth, as perceived by f, decreases to 1.2 Mbps due to an increase in interference or contention, then the 20% channel time is no longer sufficient for the flow to meet its minimum bandwidth requirement. Its RA must then re-negotiate for at least 25% of the channel time. Similarly, if a flow perceives the total network bandwidth to have increased, it must release any excess share of the channel it has been allotted.

Bandwidth Manager (BM): The Bandwidth Manager performs admission control at the time of flow establishment and bandwidth redistribution at the time of flow teardown. Admission control involves revocation of some channel time from existing flows and re-allocation of this portion to the new flow. The BM also performs re-negotiation either when some flow detects a change in its perceived bandwidth or when its traffic characteristics change.

The BM rejects a flow if it cannot support even its minimum channel time requirement. Otherwise, the flow is admitted. The BM allots all the admitted flows *at least* their requested minimum channel time proportions. The re-

maining channel time as yet unallotted after all the admitted flows' minimum channel time requirements are satisfied, is allotted on a *max-min fair* basis.

2.3 Bandwidth Management Protocol

This section describes the protocol used in the interactions between the various components of the bandwidth management architecture and the details of the BM's operation. Figure 2 shows the events actions involved in the protocol. Table 1 clarifies the notation used in the description of the protocol.

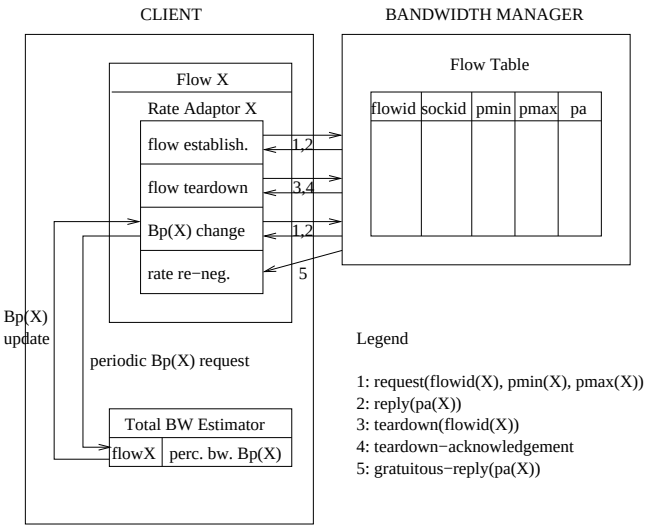

Figure 2. Bandwidth Management Protocol.

Flow Establishment: When an application requiring network access starts up, its RA starts up with it. At the time of initiating a flow f, the application specifies its required minimum bandwidth $B_{min}(f)$ and maximum bandwidth $B_{max}(f)$, both in bits per second, to its RA. These values have to be each divided by the flow f's perceived total network bandwidth $B_p(f)$ to obtain its requested minimum and maximum channel time proportion. The total network bandwidth $B_p(f)$ perceived by a flow f is estimated by the TBE at the local node. Note that a best-effort flow (e.g. FTP) will have $B_{min}(f) = 0$.

Both the channel time required by the flow f's packets in the forward direction as well as that required by the acknowledgements (if any) in the reverse direction must be included in f's channel time requirement. The forward and reverse bandwidth requirements are divided by the *same* value of $B_p(f)$, to give forward and reverse channel time proportion requirements respectively, and then added together. The underlying assumption is that total network bandwidth $B_p(f)$ perceived by flow f is approximately the same in both directions, be it the source doing the estimation using data packets or the destination doing estimation using acknowledgement packets. We normalize the TBE's estimate over different packet sizes so that this does not affect the $B_p(f)$ estimate. The assumption is valid because interference and contention experienced by data packets at their source is the same as that experienced by the acknowledgement packets at their destination and vice versa.

Notation	Meaning
F	Set of flows admitted by the BM.
$g \in F$	All flows previously admitted by the BM.
f	New flow requesting admission.
$B_{min}(f)$	Minimum bandwidth requirement of flow f.
$B_{max}(f)$	Maximum bandwidth requirement of flow f.
$B_p(f)$	Total network bandwidth perceived by flow f.
$p_{min}(f)$	Min. channel time proportion reqd. by flow f.
$p_{max}(f)$	Max. channel time proportion reqd. by flow f.
p_{rem}	$1 - \sum_{g \in F} p_{min}(g)$: Channel time remaining after $p_{min}(g), \forall g \in F$ is met.
$p_{newmax}(f)$	$p_{max}(f) - p_{min}(f)$: Maximum channel time proportion requirement for f input to max-min algorithm because $p_{min}(f)$ is already allotted.
$p_{mm}(f)$	Channel time proportion allotted to flow f by max-min algorithm. This is in addition to $p_{min}(f)$ already allotted before max-min algorithm began.
$p_a(f)$	$p_{min}(f) + p_{mm}(f)$: Total channel time proportion allotted to flow f.

Table 1. Explanation of notation used in Bandwidth Management protocol.

The RA then registers the new flow with the TBE for the node. Initially, the TBE has no estimate of the total network bandwidth as perceived by the new flow. This is because it has to use the flow's packets themselves for obtaining an estimate of the total network bandwidth, based on the interference and contention these packets experience. But the flow has not sent out any data yet and is still in the process of establishment. So, when initially computing the flow's requested minimum and maximum channel time proportions, the RA has to use a hardcoded initial total bandwidth estimate [1]. Once the flow begins, a more accurate total bandwidth estimate will be available from the TBE. The requested minimum and maximum channel time proportions can then be modified using this more accurate estimate, and re-negotiation done with the modified values.

Alternatively, in the case of a connection-oriented flow, the first few flow-establishing packets (e.g. TCP three-way handshake messages or application-level handshake) can be used in the total bandwidth estimation instead of a hard-

[1] We use a 2 Mbps network and we set this hardcoded value to 1.5 Mbps.

coded estimate. A current estimate being used by *other* flows between the same end-points can also be used initially. In our prototype implementation we use UDP rather than TCP flows, so we use a hardcoded initial estimate at the start of the flow. Once the flow begins, the TBE uses the flow's packets to obtain a more accurate total network bandwidth estimate and re-negotiates based on this more accurate value.

Let the total bandwidth estimate, how ever it is obtained, for a new flow f be $B_p(f)$. The proportion $p_{min}(f)$ of channel time, required to satisfy the new flow f's minimum bandwidth requirement $B_{min}(f)$, is $p_{min}(f) = B_{min}(f)/B_p(f)$. (Zero for best-effort flows.) Similarly, the channel time proportion $p_{max}(f)$, required to satisfy flow f's maximum bandwidth requirement, is $p_{max}(f) = B_{max}(f)/B_p(f)$. The RA of the new flow f sends the BM a `request` message containing the flow-id of f, $p_{min}(f)$ and $p_{max}(f)$.

The BM checks whether, for all flows g in the set F of previously registered flows, $1 - \sum_{g \in F} p_{min}(g) \geq p_{min}(f)$. If this is true, the new flow f is admitted ($F = F \cup \{f\}$), else it is rejected and a `reply` message offering it zero channel time proportion is returned to its RA. A rejected flow may attempt again later to gain access to the channel. Note that a best-effort flow with $p_{min}(f) = 0$ is always admitted.

Once the new flow f is admitted, the BM must redistribute channel time within the new set of existing flows F. Since the original admission test was passed by flow f, accommodating it will not cause the channel time proportion allotted to any flow $g \in F$ to fall below its minimum requested proportion. Hence, the BM initially sets allotted channel time proportion $p_a(g) = p_{min}(g)$, $\forall g \in F$. The remaining channel time, $p_{rem} = 1 - \sum_{g \in F} p_{min}(g)$, is distributed among the flows $g \in F$ in *max-min fair* fashion. We thus deem our channel time allocation policy *max-min fair with minimum guarantees* [2]. The maximum requirement for each flow $g \in F$ in the max-min fair computation is set to $p_{newmax}(g) = p_{max}(g) - p_{min}(g)$. This is because $p_{min}(g)$ has already been allotted to it and it only needs $p_{newmax}(g)$ more to fulfill its maximum channel time proportion requirement. Thus, knowing p_{rem} and $p_{newmax}(g)$ $\forall g \in F$, the max-min algorithm can proceed.

In max-min fairness, flows with small channel time requests are granted their requests first; the remaining channel capacity is then *evenly* divided among the more demanding flows. The computation of the max-min allocation is as follows. Initially, the set of flows f, whose new maximum channel time requirement $p_{newmax}(f)$ has already been satisfied, is empty: $\mathcal{R} = \emptyset$. Then, we compute the first-level allotment as $CA_0 = p_{rem}/N$, where N is the total number of flows. Now we include all flows f with $p_{newmax}(f) < CA_0$ in set $\mathcal{R}$, and allot each of them $p_{mm}(f) = p_{newmax}(f)$. Next, we compute $CA_1 = \frac{p_{rem} - \sum_{f \in \mathcal{R}} p_{newmax}(f)}{N - \|\mathcal{R}\|}$. If for all flows $g \notin \mathcal{R}, p_{newmax}(g) \geq CA_1$, then we allot each of them $p_{mm}(g) = CA_1$ and stop. Otherwise, we include those flows g with $p_{newmax}(g) < CA_1$ in set $\mathcal{R}$, allot each of them $p_{mm}(g) = p_{newmax}(g)$, and re-compute the next level CA_2. When the algorithm terminates, the allocation $p_{mm}(f)$ for all the flows is max-min fair [3]. Now, $p_{mm}(g) \leq p_{newmax}(g)$ and $\sum_{g \in F} p_{mm}(g) = p_{rem}$. After every flow f's $p_{mm}(f)$ has been determined, the Bandwidth Manager sets total channel time allotted to f, $p_a(f) = p_{min}(f) + p_{mm}(f)$ and returns this value to flow f's RA. Note that for best-effort flows, since $p_{min}(g) = 0$, $p_a(f) = p_{mm}(g)$. In other words, channel time is allotted to best-effort flows only after the real-time flows are all allotted at least their minimum share.

After the new flow f is admitted, the BM registers an entry pertaining to it in its *flow table*. This entry consists of: (a) the new flow f's flow-id, (b) the socket descriptor of the socket used by the BM for communication with f's RA, (c) $p_{min}(f)$, (d) $p_{max}(f)$, and (e) $p_a(f)$. The socket descriptor is stored in the table so that if any re-negotiation needs to be done later with flow f's RA (for example, when the number of flows in the network changes), this socket can be used.

Finally, for every flow $g \in F$, the allotted channel time proportion $p_a(g)$ is then returned to flow g's RA using a `reply` message. (The name of the message is a misnomer in the case of all flows $g \in F$ except the new flow f because, in their case, the `reply` is gratuitous, not a response to any message they sent.) The RA of every flow $g \in F$ sets its packet transmission rate respectively to $p_a(g) \times B_p(g)/pktsize(g)$ packets per second where $pktsize(g)$ is the size of flow g's packets in bits. The new flow f can now begin operation. The older flows simply resume operation with their respective new rates.

Flow Teardown: When a flow f terminates, its RA sends a `teardown` message to the BM. The BM removes flow f from the set of existing flows F, i.e. $F = F - \{f\}$. It then redistributes flow f's allotted channel time proportion $p_a(f)$ among the other flows using the max-min fair algorithm with minimum guarantees. The RA of each flow $g \in F$ (the new set F) is told of its newly allotted channel time proportion by the BM. The socket descriptors in the flow table are used to send gratuitous `reply` messages for this purpose. The entry for the terminating flow f in the BM's flow table is expunged. A `teardown-`

[2] The max-min fair policy with minimum guarantees lends itself to an elegant two-tier pricing scheme. The guaranteed minimum channel time proportion $p_{min}(g)$ is valued at a substantial price, whereas any channel time $p_{mm}(g)$ allotted in excess of this is relatively very cheap. High minimum requirements are thus "punished" while high maximum requirements carry no penalty.

[3] The computational complexity of this algorithm is $O(N^2)$. A detailed pseudo-code is in [14].

`acknowledgement` message is sent to f's RA [4].

Change in Flow's Perception of Total Network Bandwidth: The RA of every flow periodically obtains from the TBE the flow's current perceived total bandwidth. The TBE updates the RA with the mean of the perceived total network bandwidth calculated for each packet transmitted by the flow since the previous update. The inter-update period could be set in terms of number of packets transmitted or in terms of time. We recommend using a hybrid scheme for determining period: it should be based on time when the packet transmission rate of the flow is low and based on number of packets transmitted when it is high. In our experiments, we use high packet transmission rates in order to determine the performance of our scheme under high network loads. Therefore, we use a perceived bandwidth update interval based on number of packets. We use a default interval of 100 transmitted packets but also measure the effect of other intervals on system performance.

In case a newly obtained perceived bandwidth value differs significantly from a previous one, the RA must re-negotiate its flow's channel time proportion with the BM, as indicated in the example in the previous section. It must also set the value of perceived bandwidth $B_p(f)$ to the newly obtained value. The RA only sets $B_p(f)$ to a new value when there is a significant change, not with every update.

In our experiments we assume a 15% deviation of a new update from the previous value as significant enough to warrant re-negotiation. We also measure how other deviation tolerance values affect system performance. If re-negotiation has to be done, the RA of flow f sends a `request` message to the BM with flow-id, $p_{min}(f)$ and $p_{max}(f)$. The values of $p_{min}(f)$ and $p_{max}(f)$ sent in the `request` message are calculated using the new value of $B_p(f)$. The rest of the re-negotiation procedure is almost identical to the one used for flow establishment, both at the BM as well as at the RA. (See Figure 2.) The only difference is that the BM does not have to add a new entry in its flow table for f; it only updates the already existing one.

Note that a flow f's re-negotiation request can be rejected by the BM [5], i.e., it receives a zero channel time proportion at re-negotiation time. This means that the flow has been cut-off in mid-operation. Unfortunately, the nature of the wireless network is inherently unreliable, and as network resources decrease, some flows necessarily have to be cut-off in mid-operation. Our scheme guarantees each flow its minimum requested QoS for almost 100% of its active duration. However, if even the minimum QoS cannot be supported, the flow is dropped altogether. Currently, we do not use any priority-based scheme to cut-off a particular flow; it is picked randomly.

Change in Flow's Traffic Characteristics: When a VBR flow f (e.g. MPEG video stream) needs to send a burst of traffic at a rate different from its normal rate, it must inform its RA. The RA will re-negotiate for a larger channel time proportion depending on the bandwidth of the burst. The re-negotiation procedure is the same as in the case of change in perceived bandwidth. At the end of the burst duration, the RA will again re-negotiate to release the excess channel time proportion. This solution is equivalent to splitting up a VBR stream in the time domain into multiple CBR streams, as has been done in [9] in the context of ATM networks. Since this scheme only involves re-organizing the traffic rather than the network, it can be directly applied from ATM networks to wireless networks. Our solution is also similar to that proposed in [17] for CDMA networks. Frequent bursts could result in an explosion in re-negotiation overhead. We deal with the problem of frequent bursts in one of two ways: (a) adjusting $B_{min}(f)$ for VBR flow f greater than its burst bandwidth and (b) having large buffering at the receiver to deal with the burst.

2.4 Total Bandwidth Estimation Procedure

To determine $p_{min}(f)$ and $p_{max}(f)$, the RA of a flow f needs to have an estimate of the actual bandwidth of the wireless link being used by the flow. To this end, we introduce a bandwidth measurement mechanism based on IEEE 802.11 MAC level, and demonstrate its robustness. This mechanism is also used in [7].

IEEE 802.11 relies on the DCF method to coordinate the transmission of packets. The packet transmission sequence is illustrated in Figure 3. Similar to [11, 2], we measure the throughput of transmitting a packet as $TP = \frac{S}{t_r - t_s}$, where S is the size of the packet, t_s is the time-stamp that the packet is ready at the MAC layer, and t_r is the time-stamp that an ACK has been received. Note that the time interval $t_r - t_s$ includes the channel busy and contention time.

It is clear that the measured throughput of a packet depends on the size of the packet. A larger packet has higher

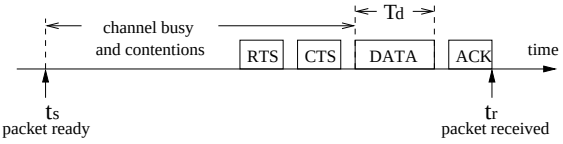

Figure 3. IEEE 802.11 unicast packet transmission sequence.

[4] If a flow dies abruptly without performing teardown, the bandwidth allotted to it is released after a time-out period. The entries in the BM's flow table are thus *soft-state* and need to be refreshed periodically.

[5] A flood of re-negotiation requests will occur if there is a decrease in available network bandwidth due to malicious flows "stealing" channel time. The BM can switch temporarily into a promiscuous *policing mode* to detect the culprits should this occur. This is a useful security-related extension of our scheme.

measured throughput because it sends more data once it grabs the channel. To make the throughput measurement *independent* of the packet size, we normalize the throughput of a packet to a pre-defined packet size. In Figure 3, $T_d = S/BW_{ch}$ is the actual time for the channel to transmit the data packet, where BW_{ch} is the channel's bit-rate. Here we assume channel's bit-rate is a pre-defined physical layer parameter. The transmission times of two packets should differ only in their times to transmit the DATA packets. Therefore, we have:

$$(t_{r1} - t_{s1}) - \frac{S_1}{BW_{ch}} = (t_{r2} - t_{s2}) - \frac{S_2}{BW_{ch}} \quad (1)$$

$$= \frac{S_2}{TP_2} - \frac{S_2}{BW_{ch}}. \quad (2)$$

S_1 is the actual data packet size and S_2 is a pre-defined standard packet size. We can use equation (2) to calculate normalized throughput TP_2 for the standard size packet.

To verify this bandwidth measurement mechanism, we simulate a group of mobile nodes within a one-hop wireless transmission range using *ns-2* [1]. In the simulation, we send CBR traffic from one node to another, and change the packet size from small (64 bytes) to large (640 bytes) during the simulation. The measured raw throughput is normalized against a standard size (picked here as 512 bytes). Figure 4 shows the result of the measured raw throughput and its corresponding normalized throughput during the course of the simulation. Obviously, the raw throughput depends on the packet size; larger packet size leads to higher measured throughput. The normalized throughput, on the other hand, does not depend on the data packet size. Therefore, we can use the normalized throughput to represent the bandwidth of a wireless link, to filter out the noise introduced by the measured raw throughput from packets with different sizes. We have also verified that using recent packets to estimate current channel bandwidth is feasible and robust [14].

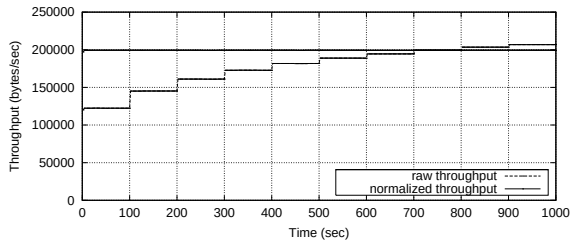

Figure 4. Raw throughput and normalized throughput at MAC level.

3 Experimental Results

We experimented with our Admission Control and Dynamic Bandwidth Management system using both our prototype implementation as well as simulations using the *ns-2* simulator. We evaluated its delay and throughput performance and also its overhead.

3.1 Request-Reply Delay

We used our prototype implementation in the experiment to obtain request-reply delay as this experiment could be performed using only a few network nodes. We used 2 IBM ThinkPad laptops and 3 handheld HP Jornada PCs, each equipped with an ORiNOCO PCMCIA wireless card configured in peer-to-peer ad hoc mode for a 2 Mbps network. We used a CBR audio streaming application over UDP for our experiments. All control messages had a 28-byte payload. The request-reply delay is the time delay between the sending of a `request` message and the receipt of a `reply` message. This exchange of messages occurs both during flow establishment as well as when perceived bandwidth changes significantly. We found that under heavy network load, each request-reply round-trip takes around 35ms. Flow establishment occurs only once per flow and if the perceived bandwidth does not change much, then the 35ms maximum request-reply delay is a small one.

3.2 Throughput Performance

For experiments with large numbers of nodes ($\geq$ 10 nodes) and connections, we used the *ns-2* simulator. We compared the performance of an IEEE 802.11 network enhanced with Dynamic Bandwidth Management with a base IEEE 802.11 network. We used a 25 node network in a 170m $\times$ 170m area with 10 CBR flows. The maximum theoretical bandwidth of the network was 2 Mbps. The transmission range of each node is 250m, so the entire network area falls within every node's transmission range. We used a random waypoint mobility model for the nodes. Our 10 flows each had a minimum bandwidth requirement of 100 Kbps and a maximum bandwidth requirement of 200 Kbps. These rates are typical of an audio streaming application. All 10 flows used 512 byte packets. The simulation ran for 600 seconds. The packet transmission rate used by our scheme at any instant was obviously determined by the value of $p_a(f) \times B_p(f)/pktsize(f)$ at that instant. The packet transmission rate used by the base IEEE 802.11 scheme was set to the maximum requested rate of the flow, as would be the case in a real application trying to capture as much network bandwidth as possible.

Figure 5 is a plot of number of packets successfully transmitted over every 1 second interval versus time for each of the 10 flows using the base IEEE 802.11 protocol. Figure 6 is the same plot using the Dynamic Bandwidth Management scheme. Note that in our scheme two flows needed to be cut-off in mid-operation so that other

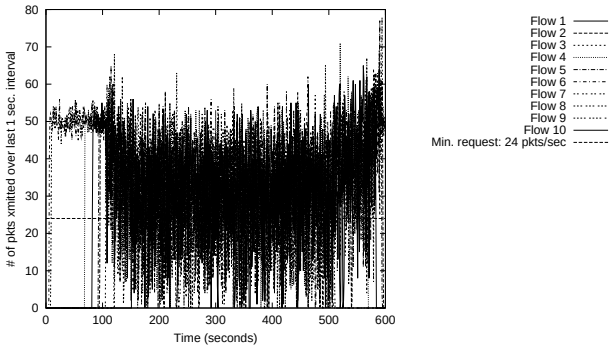

Figure 5. Performance of base IEEE 802.11.

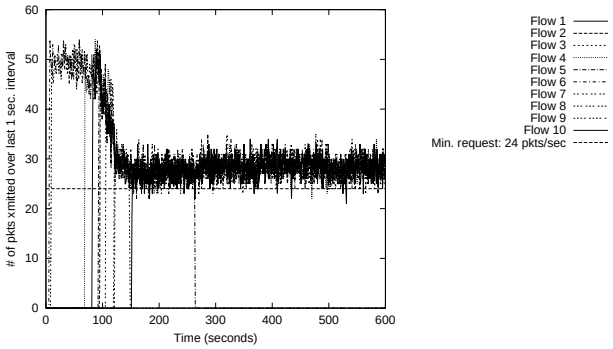

Figure 6. Performance of IEEE 802.11 with Dynamic Bandwidth Management.

flows were not starved. One of these is cut-off at time = 149 seconds and the other at time = 264 seconds.

It is clear from the plots that our protocol dramatically improves throughput fairness. No single flow takes up excess bandwidth during a particular interval thereby starving another flow of bandwidth. In the base IEEE 802.11 scheme, there are large throughput discrepancies between flows over the same time interval. Due to starvation, flows often fall far below their minimum bandwidth requirement over the 1 second measurement interval, resulting in a chaotic plot. Using our scheme, flows almost never fall below their minimum bandwidth requirement shown with the horizontal line at 24 packets per second. (100 Kbps / 4096 bit packets is approximately 24 packets per second.) Even when they do, it is only by a small amount. Our scheme thus ensures that the minimum bandwidth requirements of the flows are met with a far higher probability than in the base IEEE 802.11 scheme. Figure 7 is a 100-second snapshot from the combined plot of Figures 5 and 6 that shows the comparative behavior of a single flow (flow 1).

In addition to improving throughput fairness, our scheme

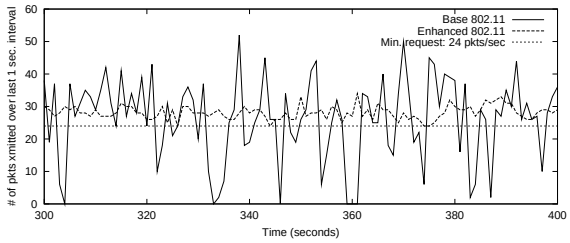

Figure 7. Comparative behavior of a single flow over base 802.11 versus 802.11 with Dynamic Bandwidth Management.

also reduces jitter in throughput as compared to base IEEE 802.11. Since we control the sending rate of the flows, we observe a negligible packet loss rate when using our scheme. With base IEEE 802.11, however, since the packet transmission rate is set to the maximum, a 33% packet loss rate results.

3.3 Overhead

In the simulations, when using our scheme, each flow re-negotiates its channel time proportion once every 14 seconds on average. Recall from Section 3.1 that each of these re-negotiations can cost a maximum of 35ms in terms of delay. This is the overhead of our scheme.

We also pay a penalty in the mean per-flow throughput. The mean throughput for an active flow in the base IEEE 802.11 case is 8% *higher* than in our scheme, for the scenario used. This is because of three reasons: (a) the flows are pumping data into the network as fast as possible in order to get as much throughput as they can in the base IEEE 802.11 scheme while we are using rate control, (b) our TBE is configured to return a conservative estimate for $B_p(f)$, and (c) in the Dynamic Bandwidth Management enhanced IEEE 802.11 scheme, re-negotiation messages between the RAs and the BM consume some network bandwidth. As mentioned previously, we might also have to drop some flows in order to ensure good performance of the others.

The conservative $B_p(f)$ estimate was used to minimize packet drop rate. The cost of using such a conservative estimate is that the IEEE 802.11 scheme enhanced with Dynamic Bandwidth Management *under-utilizes* the network. Mean per-flow throughput is less than it would be under full utilization. However, the TBE's estimate can be suitably tuned so that throughput of our scheme approaches that of the base IEEE 802.11 scheme and network utilization increases. On the other hand, this will also increase the packet drop rate of our scheme and thereby degrade performance as packets are dropped randomly from flows. So, there ex-

ists a trade-off between throughput and packet drop rate.

There are two ways to minimize re-negotiation frequency and hence bandwidth consumed by re-negotiation messages. One method is to increase the inter-update period between successive perceived bandwidth updates from the TBE to the RA. The other is to increase the tolerance to changes in perceived bandwidth $B_p(f)$. The effect of these optimizations on overhead and performance are discussed in detail in [14]. Our results show that for a small price in terms of performance (we quantify fairness and throughput jitter, the key performance metrics, in [14]), we can obtain large gains in overhead reduction.

4 Related Work

Past research in wireless bandwidth management has mostly focused on flow scheduling at the *base-station* to achieve fairness between flows competing for the channel (e.g. [5]). In contrast, we use a peer-to-peer MAC layer transmission model rather than the base-station model.

In [16], the authors propose an admission control scheme for a peer-to-peer, single-hop, ad hoc wireless network model similar to the one we have used. Their scheme requires the use of special probe packets to obtain the service curve, which is an estimate of network load. Using the service curve, one-time admission control is performed. In contrast, our scheme estimates network load using the data packets of the connection itself. Moreover, we perform dynamic bandwidth re-negotiation over the course of the connection, in addition to admission control at flow startup.

Another area of related work is the QoS-aware link-level scheduling schemes in single-hop and multi-hop wireless networks [15, 3, 12, 13, 10, 8]. As mentioned before, our bandwidth management scheme is a co-operative link-middleware scheme that can work independently or assist a QoS-aware link-level scheduler when available. The ability of our bandwidth management scheme to work independent of a link-level QoS-aware scheduler makes it highly deployable in today's smart-rooms.

5 Conclusion

In this paper we present an Admission Control scheme to determine the fraction of channel time that each flow in a single-hop ad hoc wireless network receives. Since one-time admission control is not sufficient to handle the changes in network and flow characteristics, we also add a Dynamic Bandwidth Management system that adapts the flows' respective channel share proportions during the course of their operation. The simplicity and robustness of our system enables the future incorporation of elegant pricing and security features into it. We have a working prototype implementation of the system and we have used it in conjunction with extensive simulations to demonstrate the feasibility and utility of our scheme.

References

[1] The network simulator ns-2. *http://www.isi.edu/nsnam/ns/*, updated Oct 2001.

[2] G. Ahn, A. Campbell, A. Veres, and L. Sun. SWAN: Service differentiation in stateless wireless ad hoc networks. In *Proc. of IEEE InfoCom*, New York, NY, Jun 2002.

[3] B. Bensaou, Y. Wang, and C. Ko. Fair medium access in 802.11 based wireless ad hoc networks. In *Proc. of IEEE MobiHoc*, Boston, MA, Aug 2000.

[4] D. Bertsekas and R. Gallager. *Data Networks (2nd Ed.), Chapter 6*. Prentice-Hall, 1992.

[5] G. Bianchi, A. Campbell, and R. Liao. On utility-fair adaptive services in wireless networks. In *Proc. of IEEE/IFIP IWQoS*, Napa, CA, May 1998.

[6] F. Cali, M. Conti, and E. Gregori. Dynamic tuning of IEEE 802.11 protocol to achieve a theoretical throughput limit. *IEEE/ACM Trans. on Networking*, 8(6):785–799, Oct 2000.

[7] K. Chen and K. Nahrstedt. EXACT: An explicit rate-based flow control framework in MANET. Tech. Report UIUCDCS-R-2002-2286/UILU-ENG-2002-1730, Dept. of Computer Science, UIUC, Jul 2002.

[8] D. Eckhardt and P. Steenkiste. Effort-limited fair (ELF) scheduling for wireless networks. In *Proc. of IEEE InfoCom*, Tel Aviv, Israel, Mar 2000.

[9] M. Grossglauser, S. Keshav, and D. Tse. RCBR: A simple and efficient service for multiple time-scale traffic. In *Proc. of ACM SigComm*, Cambridge, MA, Aug 1995.

[10] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed multi-hop scheduling and medium access with delay and throughput constraints. In *Proc. of ACM MobiCom*, Rome, Italy, Jul 2001.

[11] M. Kazantzidis, M. Gerla, and S. Lee. Permissible throughput network feedback for adaptive multimedia in aodv manets. In *Proc. of IEEE ICC*, Helsinki, Finland, Jun 2001.

[12] H. Luo, S. Lu, and V. Bharghavan. A new model for packet scheduling in multihop wireless networks. In *Proc. of IEEE MobiCom*, Boston, MA, Aug 2000.

[13] H. Luo, P. Medvedev, J. Cheng, and S. Lu. A self-coordinating approach to distributed fair queuing in ad hoc wireless networks. In *Proc. of IEEE InfoCom*, Anchorage, AK, Apr 2001.

[14] S. Shah, K. Chen, and K. Nahrstedt. Dynamic bandwidth management in single-hop ad hoc wireless networks. Tech. report UIUCDCS-R-2003-2314/UILU-ENG-2003-1701, Dept. of Computer Science, UIUC, Jan 2003.

[15] N. Vaidya, P. Bahl, and S. Gupta. Distributed fair scheduling in a wireless LAN. In *Proc. of ACM MobiCom*, Boston, MA, Aug 2000.

[16] S. Valaee and B. Li. Distributed call admission control in wireless ad hoc networks. In *Proc. of IEEE VTC*, Vancouver, Canada, Sep 2002.

[17] J. Zhang, M. Hu, and N. Shroff. Bursty data over CDMA: MAI self similarity, rate control and admission control. In *Proc. of IEEE InfoCom*, New York, NY, Jun 2002.