Distributed Protocols for Scheduling and Rate Control to Achieve Max-Min Fairness in Wireless Mesh Networks

Shweta Jain, Samir R. Das and Himanshu Gupta
Computer Science Department
Stony Brook University
Stony Brook, NY 11794, U.S.A.

Abstract

The goal in this paper is to develop comprehensive protocol support in all layers to provide max-min fairness for multihop flows in a wireless mesh network. Our approach has three parts. First, we estimate the max-min fair rate of all multihop flows in the network using a distributed protocol. This estimation uses the knowledge of the flow contention graph that the network nodes learn by exchanging local information. Second, the nodes enforce this rate by controlling the rate at which a flow is scheduled to the link layer. Third, a back pressure flow control is used to reduce the transmission rate of a flow if it has been exceeding its fair rate. Finally, we argue that the fair rate estimation can at best be approximated in an 802.11 based MAC protocol. Thus, to complement our fair rate estimation and scheduling procedures, we develop a virtual time based MAC protocol. We demonstrate via extensive simulations the benefit of all these approaches for ensuring fairness relative to the base case that uses 802.11 MAC and FIFO scheduling.

1 Introduction

A common problem observed in wireless multihop networks is a situation where externally offered load entering the network exceeds the network capacity. If the network capacity is exceeded, packets are queued en-route to the receiver resulting in higher end-to-end packet delays, and wastage of bandwidth when packets are dropped at intermediate nodes. Unfair distribution of bandwidth among users is another challenge that a network designer needs to address specially in distributed ad-hoc and mesh networks. In this context, an appropriate and viable solution is a maxmin fair rate allocation[5] in which resources are allocated in order of increasing demand such that no user gets a resource share larger than its demand and users with unsatisfied demands get an equal share of the resource. Also a user with unsatisfied demands cannot increase its resource share without reducing the share of others who are already using equal or lesser amount of the resource.

Our goal in this paper is to develop a distributed max-min fair queuing mechanism that enforces this notion of fairness for multihop flows in wireless mesh networks. We compute the maxmin fair rate of a multihop flow by computing the maxmin fair rate at each hop along its path and finally enforcing the rate offered to the flow at the most constrained hop in the path. This approach provides the framework for a multihop maxmin fair rate allocation as well as bounds the rate at which packets are injected in the network to the maximum rate at which it can be delivered to the destination. Although our queuing mechanism can work with any reasonable MAC protocol, we find that the IEEE 802.11 MAC seriously deviates from fairness principles in certain scenarios [4],[11],[2]. In order to reduce MAC layer unfairness, we replace the exponential backoff mechanism in 802.11, with virtual time based CSMA (VTCSMA) which is a backoff scheme based upon packet arrival time.

VTCSMA [7] provides a distributed first come first serve medium access to contending nodes. This approach ensures that the scheduling order computed at the upper layer is also enforced in the MAC layer. The VTCSMA protocol was designed for single hop networks, and our work extends it for multihop networks. This is nontrivial as problems such as hidden terminals and starvation must be addressed. Our queuing method and the MAC layer protocol together form a complete protocol suite that computes and enforces max-min fair scheduling in wireless mesh networks in a distributed manner.

The rest of the paper is organized as follows. In section 2, we will explain the background, theory and definition of max-min flow control in the context of wireless multihop networks. We will then describe our upper layer protocol in section 3 followed by the MAC layer solution in section 4. We present performance evaluation in section 5 and related work and conclusions in sections 6 and 7.

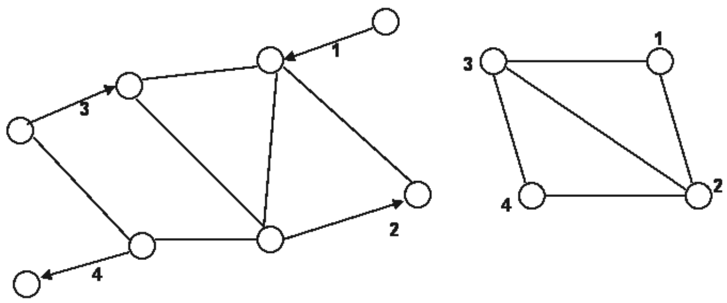


Figure 1. Network graph and the corresponding flow contention graph.

2 Background

In wireless networks, transmission between a pair of neighboring nodes (also called single hop flow) interferes with a transmission between another pair if either the two single hop flows have a common transmitter or receiver or if the transmitter or receiver of one is within two hop distance from the transmitter or receiver of the other. The two hop consideration is due to the assumption of an 802.11-like protocol where any transmission can interfere up to two hops. We model these interfering flows using a contention graph, henceforth called *flow contention graph*, where nodes are single hop flows on the network graph and edges are drawn between two nodes if the flows interfere. An example of the flow contention graph is shown in Figure 1.

Given this notion of flow contention graph, earlier work [2] has considered max-min fair rate of single hop flows. In our work, we consider end-to-end multihop flows as multiple single hop flows that can go over a sequence of links. We first treat these single hop segments as individual flows and then extend the idea of fairness to multihop flows. To demonstrate the technique let us first describe the notion of feasibility and max-min fair allocation.

A feasible rate allocation essentially constrains the rate allocation for each flow such that the sum total of the rates allocated to all flows belonging to a *clique* in the flow contention graph do not exceed the network capacity. A rate allocation is max-min fair if it is feasible and the only way a flow can get higher rate is by reducing the rate of some other flow that has been allocated equal or lower rate. Formal definitions are below.

Definition 1 (Feasible Rate Vector) *Assume that C is the link capacity in the wireless network. Let R represent a vector that represents transmission rates r_i allocated to each flow f_i in a "clique" in the flow contention graph. If F is the set of flows in the clique, then the vector R of rates r_i is feasible if*

$$r_i \geq 0, \sum_{\forall f_i} r_i \leq C.$$

Definition 2 (Max-min Fair Rate Allocation) *A feasible rate vector is max-min fair if for any flow f_i, the allocated rate r_i cannot be increased while maintaining feasibility without decreasing r_j for some flow f_j for which $r_j \leq r_i$ [1]. Flows f_i and f_j do not need to belong to the same clique.*

Prior work [2] has shown that a feasible rate vector R is max-min fair if and only if each flow has a bottleneck clique with respect to R. Bottleneck clique is defined as follows.

Definition 3 (Bottleneck Clique) *Given a max-min fair rate vector R, a bottleneck clique cl_i is that clique for which flow $f_i \in cl_i$, $\sum_{\forall f_k \in cl_i} r_k = C$, and allocated rate r_i of f_i is equal or greater than the allocated rate r_k of any other $f_k \in cl_i$. The largest clique in the network is the bottleneck clique for the flows it contains.*

2.1 Max-Min Rate Calculation

Based on the above, prior work [2] has provided a mechanism to compute max-min fair allocation of rates on single hop flows in the network. The technique simply determines all cliques in the flow contention graph. Since this can be computationally intractable, heuristics are used for the clique computation. Starting with the largest clique, each flow in the clique is allocated equal share of the remaining capacity except the ones that have already received an allocation. The remaining capacity is simply the capacity C minus the already allocated rates. The allocation is started with the largest clique, as this clique is always the bottleneck for the flows belonging to this clique and thus determines the fair rate allocation of these flows.

For the benefit of the reader, we illustrate the procedure using the example of Figure 2. Assume capacity $C = 1$. There are three cliques with 3, 4 and 7 nodes respectively with some common vertices's (A, B, C) corresponding to network flows. The procedure starts with clique 3, assigning a rate of $\frac{1}{7}$ to each vertex of clique 3. Then it turns to clique 2. Since B and C have already been allocated their rates, A and D are allocated the remaining capacity equally. Each of them gets $\frac{1}{2}(1 - \frac{2}{7}) = \frac{5}{14}$. But since the rate allocated to A by clique 1 is only $\frac{1}{3}$ which is less than the rate being offered by clique 2, it receives only $\frac{1}{3}$ rate, while node D finally gets $1 - 2 \times \frac{1}{7} - \frac{1}{3} = \frac{8}{21}$ part of the bandwidth.

3 Upper layer Protocol to achieve Max-min fair scheduling

In the prior section, we have described how to compute max-min fair rates for single hop flows in the network. In

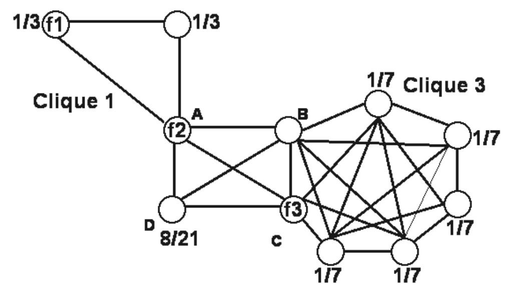


Figure 2. Illustrating computation of fair rates.

this section, we develop a queuing mechanism that computes and allocates max-min fair rates to multihop flows. The protocol has three components: "clique formation protocol" that computes the allocations locally on single hop segments of multihop flows; "back pressure protocol" that assigns fair rates to multihop flows; "rate enforcement protocol" which essentially controls the scheduling and enforces that no flow exceeds its allocated rate.

3.1 Clique Formation Protocol

In order to compute fair rates for all flows in the network in a distributed fashion, each network node needs to obtain the flow contention graph that represents its *local neighborhood.* The local neighborhood of a node consists of its neighbors that can be reached in up to two hops. A two-hop message exchange protocol gathers enough information to build the local flow contention graph. This can be done by sending "hello" messages and rebroadcasting the contents so that the two-hop neighbors of the original sender can receive the messages as well. These "hello" messages are similar to "hello" messages that many routing protocols (e.g., AODV [8]) employ to maintain neighborhood information; so we do not consider them to be additional overheads except the additional content. Frequency of such exchange for our protocol objective should be the granularity of any topology change or traffic changes (in terms of origination of a new flow or expiry of an existing flow).

Each node i maintains and includes in the "hello" messages, information about the single hop flows that a node originates, receives or routes. These single hop flows may be segments of multihop flows. This information includes the flow id (f_m), the nexthop receiver of the flow (node j) and the rate allocated to the flow ($r_{m,i}$) at node i. Thus, the "hello" messages contain a set of tuples $f_{m,i,j} =< f_m, j, r_{m,i} >$. We will refer to the set of $f_{m,i,j}$ tuples as the *local flow set* (L_i) for node i. Apart from L_i, node i also includes in the "hello" messages, the same information about the flows that interfere with its transmissions. We will refer to this set as the *interfering flow set* or (I_i). The I_i is the union of *local flow sets* L_j of all nodes within the two hop neighborhood of node i. Thus, if N_i is the set of one and two hop neighbors of node i then,

$$I_i = \bigcup_{\forall j \in N_i} L_j.^{1} \quad (1)$$

After receiving messages from all neighbors, node i is able to construct a *neighbors interfering set* or P_i such that,

$$P_i = \bigcup_{\forall j \in N_i} I_j. \quad (2)$$

This information is sufficient [2] for node i to compute the flow contention graph representing its neighborhood and calculate all cliques in this graph. The fair share of bandwidth of all members of the bottleneck clique in the network is simply the ratio of the bandwidth and the size of the clique [2].

We cannot obtain the size or content of the bottleneck clique in the entire network due to the hardness of the problem. But we can find all cliques and compute the bottleneck clique in the local neighborhood consisting of few nodes, in reasonable time. Thus, for every flow the node keeps track of the *local bottleneck clique* corresponding to that flow and computes rate, say S. If after subsequent "hello" message exchanges, the node sees that other flows in this clique insist on getting less than rate S, it redistributes the residual rate among other flows in the clique and recomputes the *local bottleneck clique.* Thus, we may claim that, at the steady state, the rate of each flow in the network is equal to that offered by the flow's *local bottleneck clique* which is the max-min fair rate of the flow.

Let us explain this with an example in *Figure 2.* This figure represents a flow contention graph of the network. Clique 3 is the largest clique in the network and thus is a *local bottleneck clique* for all member flows. Flow A in the graph is a member of both clique 1 and clique 2. The rates offered by the cliques to flow A are $\frac{1}{3}$ and $\frac{5}{14}$ respectively. Thus although clique 2 is the largest clique for flow A in terms of size, clique 1 is the bottleneck clique as it allows a rate lower than clique 2.

3.2 Back Pressure Protocol

In the previous section, we treated multi-hop flows as multiple single hop segments of the flow thereby assigning rates to each segment of the flow at the local bottleneck cliques. We now introduce the notion of a *global bottleneck*

[1]Here we would like to mention that when computing the union or intersect of sets, a node only considers the $< f_m, j >$ pair from the tuple while $r_{m,i}$ is used in rate computations at upstream and downstream nodes.

clique for multihop flows as the clique at which the flow receives the least rate along its path. A more formal definition is as follows.

Definition 4 (Global Bottleneck Clique) *A global bottleneck clique for a multihop flow is the clique containing the single hop flow segment $f_{m,i,j}$ (flow id m, from node i to node j) of the multihop flow $F_{m,a,b}$ (flow id m, from source a to destination b), where the offered rate $S_{m,i,j}$ at node i is less than the rate offered at any other node k along the flow's path.*

Consider *Figure 2* again. A multihop flow F in the figure is represented by three single hop flow segments – $f1$, $f2$ and $f3$. The rate offered at each of these segments are $\frac{1}{3}$,$\frac{1}{3}$ and $\frac{1}{7}$ respectively. Thus clique 3 is the global bottleneck clique for flow F since it offers the least rate compared to other cliques along the path from source to destination.

If the rate provided at upstream nodes of a multihop flow is larger than the rate offered at the *global bottleneck clique*, packets may be queued and dropped at the forwarding nodes. Similarly, if the rate offered at downstream nodes is higher than the rate allocated at the global bottleneck clique, the allocated rate will remain unused instead of being utilized by other flows with unfulfilled demands. In order to prevent such wastage of bandwidth, we introduce a back pressure protocol in which each node limits a multihop flow's rate to the minimum of the rates provided at the next hop, at the previous hop and at the current hop. The source and destination of the multihop flow, limit the flow's rate to the minimum of the computed rate and that offered at the next or previous hop respectively. This scheme achieves what the authors in the paper [9] have tried to achieve by a more complex token generation process. Due to this *back pressure* mechanism, the rate offered by the global bottleneck clique for the flow is propagated to all nodes along the path from the source to the destination of the flow. The extra bandwidth available after applying the back pressure technique is distributed among other flows after the next hello message exchange and the *local and global bottleneck cliques* are recomputed. A detailed mathematical analysis of the token based back pressure technique is presented in [9] which also applies to our technique.

3.3 Rate Enforcement Protocol

In order to enforce the assigned rates, the protocol needs to ensure that the rate at which the packets are transmitted follows the rate computed by the *clique formation protocol* and the *back pressure protocol*. We employ a timer based mechanism to "release" packets at the computed rate. A flow may be served only if there is a packet that has been "released" for transmission. Every node that has packets to send, runs a timer, which we will refer to as the *release timer*. The interval of release timer is calculated dynamically and depends upon the number of contending flows in the local neighborhood. When the release timer fires, the node checks if there is a flow from which a packet can be "released". A packet can be "released" if the flow to which the packet belongs has used less than its allocated rate otherwise the next flow is considered. This scheme ensures that each flow receives no more that the rate computed by the clique formation and back pressure protocols, thereby enforcing the computed rates.

4 Virtual Time Based MAC Protocol

The three step upper layer protocol that we proposed in the previous section can be used in conjunction with any reasonable MAC layer protocol in wireless network. However, we know from [11],[4],[2] that the commonly used IEEE 802.11 MAC protocol suffers from several unfairness issues. This is due to several reasons including exposed terminals, hidden terminals and the backoff policy used in 802.11. We have developed a medium access protocol to complement our scheduling scheme. Our MAC protocol performs a packet arrival based backoff mechanism known as virtual time CSMA (VTCSMA) [7] rather than random exponential backoff mechanism used in 802.11.

The VTCSMA MAC protocol implements a first come, first serve access to the shared medium by emulating a single server multiple queue system. Only here the queues are maintained at different nodes in the network and the scheduling decision must be made in a distributed manner. In order to achieve this distributed scheduling process, each node in the network maintains two clocks, *real clock* and *virtual clock*, to measure the passage of *real time* and *virtual time* respectively. Both clocks may be initialized to zero and the real clock runs at a constant rate. The virtual clock runs η times faster than the real time clock while the medium is idle (unless the two clocks are in sync, in which case they run in lock steps). The virtual clock is stopped whenever the medium becomes busy and it resumes when the medium is idle again. When the virtual clock of a node passes the arrival time of the packet in the head of its queue, the packet is transmitted. If all nodes in the network share the same wireless medium and follow this transmission rule, the first-come first-serve scheduling is trivially achieved in a distributed manner. The analysis in [7] shows that this protocol can potentially provide a higher goodput as compared to random access CSMA.

VTCSMA as described above provides fair medium access when all nodes are within a single collision domain i.e., all nodes are within receive range of one another. Since in a single collision domain, nodes can "hear" transmissions from each other, the virtual clocks run almost in sync or atleast at the same average rate. The average rate is calculated as the rate at which the virtual time progresses with respect to progress of real time. The average rate of virtual clock at any node depends upon the contention level it experiences. Also since a packet is transmitted only when the virtual time reaches the packet arrival time, the throughput achieved by a node is also a function of the average rate of the virtual clock. In a multihop network, the contention experienced by nodes differ from one region to another. It is easy to construct scenarios where some nodes experience larger contention than their neighbors thereby getting fewer chances to transmit than other nodes. This phenomenon may lead to unfair share of bandwidth and even starvation. Figure 3(c) shows a typical scenario where this may happen. Here node 5 being in the carrier sensing range of both nodes 0 and 3, faces higher contention than either node 0 or node 3 which do not contend with one another. Therefore, the average rate of node 5's virtual clock is lower than that of 0 and 3. We suggest a two step approach to address this problem in the multihop extension of the VTCSMA protocol described in the next section.

4.1 VTCSMA in Wireless Multihop Networks

We have proposed a multihop VTCSMA MAC protocol that alleviates the starvation problem of VTCSMA. We borrow the virtual carrier sensing and solution to hidden terminal problem from IEEE 802.11 where nodes maintain "network allocation vectors (NAV)" and exchange RTS/CTS control packets to maintain channel state and to notify potential interferers of the impending transmission.

To solve the starvation problem in VTCSMA, we propose that every packet must carry the virtual time stamp of the transmitting node and every node in the network must follow a two step approach to prevent starvation. In the first step which we name "good neighbor approach", nodes reduce the possibility of starvation of their neighbors by adjusting their virtual clock to minimum of the virtual time stamp from overheard packets and the time measured by the local virtual clock. The second step which we name "bad neighbor approach" is invoked when a node that has packets to transmit, overhears another packet with a virtual time stamp that is ahead of its own virtual time by more than a fixed threshold (an indication of starvation). The starving node then sends a jamming message that conveys this situation to all receivers in its vicinity, forcing all nodes to invoke their collision recovery mechanism i.e setting the NAV and withholding all transmissions. Here we propose an additional network allocation vector called "soft NAV". When a node detects a jamming signal or a collision, it waits for the medium to become idle again and then sets a "soft NAV" in addition to the regular NAV. During this "soft NAV" state or "soft state", nodes do not run their virtual clock and do not initiate any transmission, but they may receive unicast transmissions and send acknowledgements. While neighboring nodes are in the "soft state", the starving node gets the opportunity to transmit its backlogged packets. At this time, nodes with faster virtual clocks adjust their clocks in the manner of the "good neighbor approach". This two step approach is instrumental in reducing the difference between average rate of virtual clocks in the network which prevents starvation in the network.

5 Results

We evaluated the performance of our queuing protocol and compared with a first-come-first-serve scheduling mechanism that schedules packets in the order they arrive in the queue at each node without consideration for the flow to which they belong. We have also compared the performance of the two MAC protocols in conjunction with each scheduling protocol. We used fairness index and goodput as the metrics to evaluate performance.

Definition 5 (Fairness Index) *If a system allocates resources to n contending users, such that the i^{th} user receives an allocation x_i, then fairness index is defined as*

$$f(x) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}, x_i \geq 0.$$

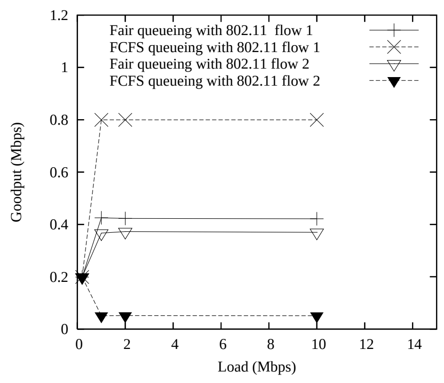
Definition 6 (Goodput) *Goodput is defined as the number of application layer data bits successfully received at the receiver over the total span of time for which the application layer sent data.*

We have used network simulator ns2 version 2.27 [3] for all simulations. We have experimented with both small scenarios that represent specific problems that arise in multihop networks as well as random scenarios with varying packet rates and number of traffic sources.

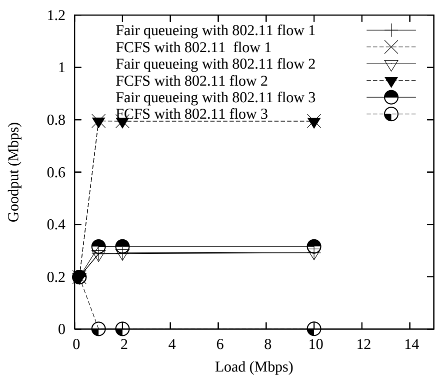
5.1 Max-min Fair vs FCFS Scheduling with IEEE 802.11

We placed 7 nodes in a network as shown in *Figure 3(a)*. We set up two TCP flows in the network, flow 1 from node 0 to node 6 and flow 2 from node 3 to node 6. We present the result of this experiment in table 1. We observe that the max-min fair scheduling protocol distributes the bandwidth more evenly between the two flows with flow 1 achieving a rate of 53kbps and flow 2 achieving 51kbps, but in FCFS scheduling, flow 1 receives a goodput of 169kbps while flow 2 is starved.

In the network shown in Figure 3(b) two UDP flows represent the information asymmetry (IA) scenario [4]. Here,

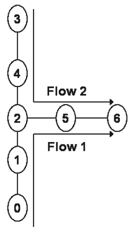


(a) Information Asymmetry

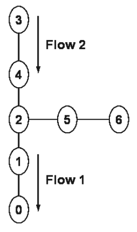


(b) Flow in the middle

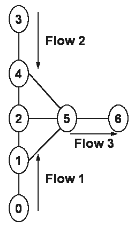
Figure 4. Goodput vs load for networks in Figure 3(a), 3(b) and 3(c)



(a) TCP flows from node 0 to node 6 and node 3 to node 6.



(b) Information Asymmetry: UDP flows from node 1 to node 0 and node 3 to node 4.



(c) Flow in the middle: UDP flows from node 0 to node 1, node 3 to node 4 and node 5 to node 6.

Figure 3. Network graphs of representative scenarios

Table 1. Goodput vs load for symmetric scenario of *Figure 3(a)* with two TCP flows from node 0 to node 6 and node 3 to node 6

Flow	FCFS Queue(Kbps)	Fair Queue(Kbps)
1	169.46579	52.94678
2	0.70691	51.1774

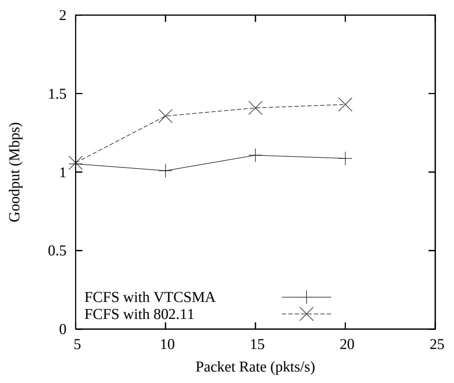
node 1 that originates flow 1 is within the carrier sensing range of node 4 which receives flow 2. On the other hand, node 3 that originates flow 2 does not have any information about flow 1 because it is beyond the transmission range of both node 1 and node 0. Since node 3 is unaware of transmissions by node 1, it is possible that node 3 attempts to transmit data while a transmission between nodes 1 and 0 is going on. These transmissions from node 3 may not be received correctly at node 4 due to interference with transmissions from node 1 causing multiple retransmission attempts by node 3. These retransmissions, in 802.11 based MAC protocols, lead to a larger contention window at the sender thus reducing its probability of acquiring the medium. This is reflected in the results shown in *Figure 4(a)*, where the goodput achieved by flow 1 is more than 75% larger than that achieved by flow 2.

In *Figure 3(c)*,we constructed a perceived collision [4] scenario with UDP flows from node 0 to node 1, node 3 to node 4 and node 5 to node 6. In a perceived collision scenario, three flows '1', '2' and '3' are such that flows '1' and '2' do not contend with one another but flow '3', contends with both flows '1' and '2'. Since the flow in the middle has to defer for the flows on each side, and therefore faces more contention compared to the neighboring flows, it gets fewer chances to transmit packets. Results in *Figure 4(b)* show that the middle flow receives very little share of the bandwidth while flows '1' and '2' each are able to receive 80% higher bandwidth share.

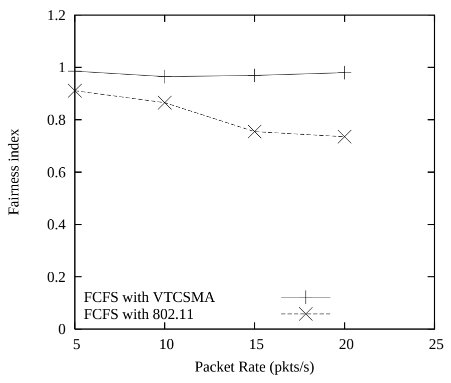
When maxmin fair scheduling is used in both information asymmetry and perceived collision scenarios, we observe that the contending flows form a clique in the network and thus equally divide the bandwidth among each other thereby achieving nearly equal goodputs as shown in *Figure 4(a)* and *Figure 4(b)*.

5.1.1 Multihop VTCSMA vs IEEE 802.11

We performed some experiments to demonstrate the advantage of using multihop VTCSMA over IEEE 802.11. We randomly placed 50 nodes in a network of size 1500x300m. Each node in the network transmits packets to a randomly selected neighbor. The virtual clock rate in VTCSMA is 200 times the real clock rate. The packet size is 512 bytes and we vary packet rates and compare fairness index and goodput for multihop VTCSMA and IEEE 802.11 in *Figure 5(b)* and *Figure 5(a)* respectively. We observe



(a) Goodput vs packet rate.



(b) Fairness Index vs packet rate.

Figure 5. Multihop VTCSMA and IEEE 802.11 MAC and FCFS in 50 node random multihop networks.

that VTCSMA achieves nearly perfect fairness index but lower goodput compared to 802.11. Here 802.11 achieves a higher goodput compared to VTCSMA but the fairness index graph shows that this is at the cost of unfair distribution of bandwidth among flows. The lower bandwidth utilization in fair scheduling protocols is due to the conflicting nature of the two goals. In [6] the author explains the difficulty of simultaneously achieving both fairness and maximizing bandwidth usage.

5.1.2 Maxmin and FCFS Scheduling with Multihop VTCSMA and IEEE 802.11

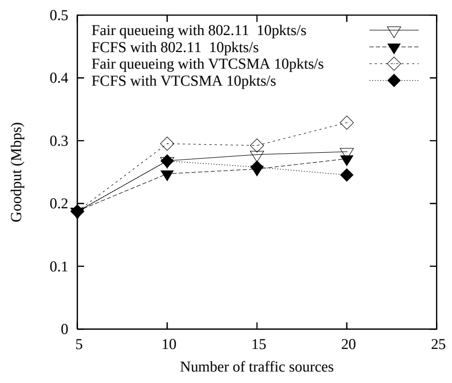
We randomly placed 50 nodes in a network of size 1500x300m and selected multihop flows between random pairs of nodes in the network. We experimented with 5,10,15 and 20 traffic connections that transmit UDP packets of size 1000 bytes at a rate of 10pkts/s. We compared the goodputs and fairness index5 of the two scheduling protocols under varying load conditions and the plots are shown in *Figure 6(a)* and *Figure 6(b)*. We observe that with 20 traffic sources, maxmin scheduling with VTCSMA MAC provides a fairness index above 0.9 while fairness index in maxmin scheduling with 802.11 MAC protocol drops to 0.8. FCFS with VTCSMA is more fair compared to FCFS with 802.11. Also note that max-min fair scheduling with VTCSMA in the MAC layer outperforms all combinations in terms of both fairness index and goodput. These results clearly demonstrate the advantages of the protocol suite that we have proposed in this work.

6 Related Work

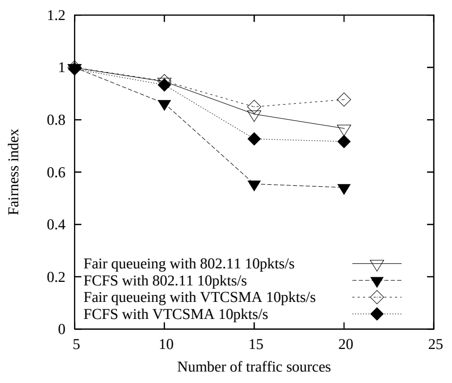
Fair scheduling of flows in a wireless multihop network has been a popular topic of research for several years. In some of the earlier works, researchers have focused on providing a MAC layer solution for fair bandwidth allocation. In [11] the authors have proposed a scheduling discipline to schedule packets on an arrival time and packet size basis with concepts similar to virtual time CSMA. We discussed earlier in this paper the drawbacks of using virtual time for scheduling in multihop networks. Since this scheme was suggested for wireless LAN, the authors did not discuss the problems that may arise in wireless multihop networks. Similarly the scheme suggested in [10] and [4] schedules packets on a priority order, where the priorities are learned from information piggy backed on control and data packets. These papers also provide MAC layer solutions and fairness is achieved by appropriate backoff policy.

In [6], the authors have provided a two tier solution to provide maxmin fair allocation for local flows and to maximize the network throughput. In the first step, the protocol achieves the fairness model by selecting a set of flows and then in the second step, the protocol tries to maximize the bandwidth utilization by scheduling the maximum independent set subject to the selection of the flows in the first phase. Since the problem of finding the maximum independent set is NP-complete, the authors implement a minimum degree greedy algorithm. The distributed implementation of the global model proposed in the paper requires that each time there is a change, the new information must be disseminated throughout the network in order to maximize network throughput. A backoff based protocol is used to achieve the local fairness model and to implement the minimum degree greedy algorithm for maximizing bandwidth utilization.

In [2] the authors allocate maxmin fair rate to single hop flows in a multihop network and the fair rate of each flow is limited by the share provided by the bottleneck clique. The fair rate of a flow is calculated by computing the rate provided by the largest clique in the flow's flow contention graph and the fair rates are achieved by a backoff based MAC protocol. The authors in [9] present an algorithmic



(a) Goodput vs packet rate.



(b) Fairness Index vs packet rate.

Figure 6. Fair queuing with multihop VTCSMA and IEEE 802.11 in 50 node random multihop networks.

perspective of max-min fair allocation in wireless multihop networks. The network model used in this work is different from what we used in our work. Here each node in the network has a locally unique frequency, thus there is no location dependent contention. Unlike [2], flows are multihop flows and the fair rate of a flow in the network is limited by the share provided by the bottleneck link along the path of the flow.

7 Conclusion

We have defined max-min fairness in terms applicable to multihop flows in wireless mesh networks. We have then developed a protocol suite to achieve max-min fairness in a distributed manner in the network. Our solution consists of an upper layer protocol for achieving max-min fairness that can be used with any MAC protocol. This protocol suite also consists of a fair MAC protocol that schedules flows on a first in first out basis. This MAC protocol truly complements our upper layer protocol to provide a complete implementation of max-min fair scheduling in mesh networks. We have presented a comprehensive performance evaluation of the protocols and compared performances with IEEE 802.11 and FCFS scheduling protocols.

References

[1] D. Bertsekas and R. Gallager. *Data Networks*, chapter 6. Prentice-Hall, 1992.

[2] X. L. Huang and B. Bensaou. On Max-Min Fairness and Scheduling in Wireless Ad-hoc Networks: Analytical Framework and Implementation. In *MobiHoc '01: Proceedings of the 2nd ACM International Symposium on Mobile Ad-hoc Networking & Computing*, pages 221–231, New York, NY, USA, 2001. ACM Press.

[3] K. Fall and K. Varadhan. NS Notes and Documentation, 1999. The VINT Project, UC Berkeley, USC/ISI, LBL and Xerox Parc http://www-mash.cs.berkeley.edu/ns/.

[4] V. Kanodia, A. Sabharwal, B. Sadeghi, and E. Knightly. Ordered Packet Scheduling in Wireless Ad Hoc Networks: Mechanisms and Performance Analysis. In *Mobihoc '02: Proceedings of the 3rd Acm International Symposium on Mobile Ad-hoc Networking & Computing*, pages 58–70, New York, NY, USA, 2002. ACM Press.

[5] S. Keshav. *An Engineering Approach to Computer Networking : ATM Networks, the Internet, and the Telephone Network*, chapter 9. Addison-Wesley, 1998.

[6] H. Luo, S. Lu, and V. Bharghavan. A New Model for Packet Scheduling in Multihop Wireless Networks. In *MobiCom '00: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 76–86, New York, NY, USA, 2000. ACM Press.

[7] M. L. Molle and L. Kleinrock. Virtual Time CSMA: Why Two Clocks Are Better than One, 1985.

[8] C. Perkins, E. Royer, and S. R. Das. Ad Hoc On Demand Distance Vector (AODV) Routing. http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-13.txt, February 2003. IETF Internet Draft (work in progress).

[9] L. Tassiulas and S. Sarkar. Maxmin Fair Scheduling in Wireless Ad-hoc Networks. In *IEEE Journal for Selected Areas in Communications Special Issue on Ad Hoc Networks, Part I*, volume 23, pages 163–173, January 2005.

[10] V. Kanodia and C. Li and A. Sabharwal and B. Sadeghi and E. Knightly. Distributed Multi-hop Scheduling and Medium Access with Delay and Throughput Constraints. In *MobiCom '01: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pages 200–209, New York, NY, USA, 2001. ACM Press.

[11] N. H. Vaidya, P. Bahl, and S. Gupta. Distributed Fair Scheduling in a Wireless LAN. In *MobiCom '00: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 167–178, New York, NY, USA, 2000. ACM Press.