Distributed Flow-Based Scheduling in Multi-hop Ad Hoc Networks

Daiqin Yang and Victor O. K. Li
Department of Electrical and Electronic Engineering
The University of Hong Kong, Hong Kong, China
Email: {dqyang,vli}@eee.hku.hk

Abstract— **Shared channel contention-based MAC protocols, such as IEEE 802.11, are popular in ad hoc networks because of their ease of implementation. However, these contention-based MAC protocols do not coordinate between nodes at different hops within a multi-hop flow. This results in channel resource and node transmission power wastage and overall system throughput degradation. In this paper we present a novel distributed flow-based scheduling (DFBS) scheme that coordinates between neighbor links of a multi-hop flow. As demonstrated by the simulation results, DFBS achieves higher throughput and improves the transmission efficiency when traffic load is relatively high.**

I. INTRODUCTION

Ad hoc wireless network has been a most active research area for its flexibility in network construction. In ad hoc networks, contention-based MAC protocols, such as IEEE 802.11, are popular since they are easy to implement. However, these contention-based MAC protocols do not coordinate between nodes at different hops within a multi-hop flow. It has been pointed out that the 802.11 MAC can not discover the optimum schedule of multi-hop transmissions[1]. This non-coordination results in two main drawbacks. First, immediate neighbor links of the same multi-hop flow will become adversaries in channel contention. This will increase the collision rate and reduce throughput. The second drawback is that, when bottleneck links exist in a multi-hop flow, upstream non-bottleneck links with light contention can get more chances to transmit than downstream bottleneck links. The differences in channel access capability between bottleneck links and non-bottleneck links will result in packet dropping at bottleneck nodes due to limited buffer size. It is wasteful of both bandwidth resource and node transmission energy. Thus a flow-based packet scheduling scheme is needed to resolve this non-coordination problem.

In recent years, packet scheduling studies in contention-based shared channel ad hoc network have been mainly focused on achieving fairness and increasing spatial reuse [2][3][4][5]. However, most of these efforts address only one hop flows. In [5], a distributed priority scheduling scheme is proposed to achieve some limited coordination between different hops of a flow by exchanging between neighbor nodes priority information of pending packets, i.e. the next packet to be transmitted. But the coordination is rather frail since the packet priority is originally designed to arbitrate competition with other flows. It does not consider the problem of bottlenecks. In this paper, we propose a novel distributed flow-based scheduling (DFBS) scheme which coordinates between immediate neighbor links of the same multi-hop flow, and considers more general scenarios. DFBS is based on the 802.11 MAC protocol. It is also relatively easy to implement.

The paper is organized as follows. We will first present a detailed problem formulation in Section II. In Section III, the DFBS scheme will be described. Simulation results will be presented and analyzed in Section IV, and we will conclude in Section V.

II. PROBLEM FORMULATION

There are two major problems with scheduling multi-hop flows with a shared channel contention-based MAC protocol. The first problem is the contentions among links within one end-to-end flow. The second problem is the uneven distribution of contention along the end-to-end flow path. In this section, we will give some detailed analysis of these two problems.

A. Contention among successive hops of a multi-hop flow

In shared channel contention-based ad hoc networks, each link will contend for packet transmission independently. Contention occurs when two or more packet transmissions occur simultaneously within the receiving range of one receiving node. Nodes at different hops belonging to the same flow can be adversaries in contention, leading to lower throughput and channel utilization.

Fig. 1 gives a simple flow structure of a multi-hop flow. Each vertex represents a node in the network, each edge denotes the reachability between each pair of nodes, and the direction of the edge is the packet transmission direction.

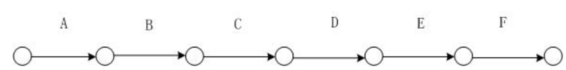

Fig. 1. A simple multi-hop flow structure.

From the flow structure in Fig. 1, we can easily draw the link contention graph of the flow as shown in Fig. 2, in which each vertex represents a certain link in the flow and each edge represents a contention relationship between the two links which are connected by it. The number following the link label is the number of contending links, called the contention number, of this link. From the link contention graph, we can conclude that each intermediate link will contend with the four closest neighbor links within the same end-to-end flow, except for the two links at each end of the flow. We can extend this to multi-hop multi-flow cases. In this paper, we assume, without loss of generality, each flow takes a different physical path. So the contention number of each link will increase at most by four for each additional flow passing through either of its end points.

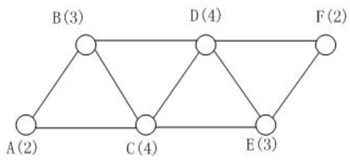

Fig. 2. Link contention graph within a multi-hop flow.

Also due to this non-coordinated contention among nodes at different hops of one flow, packets may be accumulated in some intermediate nodes. When the accumulation exceeds the buffer, packets will be dropped. Thus node transmission power will be wasted and channel utilization will be degraded.

B. Uneven contention along flow path

The second problem is the uneven distribution of contention along the end-to-end flow path. Since traffic loads of an ad hoc network are distributed arbitrarily, some links may encounter less contention and thus can get more chances to access the channel and send more packets. At the same time, some other links may suffer from more contentions, thus becoming bottlenecks. However, the overall throughput of an end-to-end flow will be constrained by the link access ability of the bottleneck.

As described in [3], we assume that with efficient contention resolution, each link can get its max-min share of channel resource according to its contention number. Under the basic principle that no downstream links can use more channel resources than upstream links, for a flow j, we assume the max-min channel share of each link is:

$$s_j^1, s_j^2, ..., s_j^{h_j} : s_j^i \geq s_j^{i+1}, 1 \leq i \leq h_j - 1 \tag{1}$$

where h_j is the link number of flow j.

Let bn_j be the hop sequence number of flow j's bottleneck link. The end-to-end throughput of flow j will be constrained by this smallest max-min channel share s_j^* of flow j:

$$s_j^* = s_j^{bn_j} \tag{2}$$

All the additional channel share of $s_j^i - s_j^*$ taken by the upstream nodes where $i \leq bn_j$ will result in both bandwidth resource wastage and node transmission power wastage. Ignoring retransmissions due to collisions, the transmission efficiency of flow j will be:

$$Te_j = \frac{h_j s_j^*}{\sum_{i=1}^{i=h_j} s_j^i} \tag{3}$$

III. Distributed flow-based scheduling scheme

To solve the above two problems, we develop a heuristic packet scheduling scheme, DFBS, based on the 802.11 MAC protocol. DFBS takes advantage of the broadcast nature of shared channel ad hoc networks, which allow each upstream node to overhear the packets forwarded by its immediate downstream neighbor. So each upstream node can automatically learn the packet accumulation status of its immediate downstream neighbor and adjust its behavior accordingly. The key idea of DFBS is that, in contentions between successive hops of one flow, a downstream node forwarding old packets will always have priority over an upstream node sending new packets, and this priority is gained through the upstream node's self regulation. With this approach, contention can be reduced between successive hops of one flow. At the same time, each link's channel access ability can automatically be limited to the bottleneck link's ability in a distributed manner without introducing too much system control overhead.

A. Information exchange

To make those packets forwarded by a downstream node recognizable by its immediate upstream neighbor, we should piggyback some flow information in the MAC header of data packets. Besides the source and destination addresses and the port number which are used to identify a certain flow, a packet sequence number is needed to indicate the sequence of this packet in the flow. Using one byte for sequence number, one byte for port number and 12 bytes for source and destination MAC addresses, totally 14 bytes are needed. Since the additional information is only piggybacked in data packets, and handshaking packets such as RTS, CTS and ACK are left unchanged, the overhead is rather small.

Besides this, each node j should keep a flow table T_j to record all the flows passing through it. The table contains a list of flow records f_i which include flow i's source address s_i, destination address d_i, port number p_i, next-hop address nh_i, last-sent sequence number ls_i and last-heard sequence number lh_i. The next-hop address nh_i is the node address of flow i's downstream immediate neighbor. The last-sent sequence number is flow i's latest sequence number of the packet that has been sent or is being sent by node j. The last-heard sequence number is the latest sequence number of the packet sent by nh_i which has been overheard by node j. The flow table in node j can be described as follow:

$$T_j = \{(s_i, d_i, p_i, nh_i, ls_i, lh_i) : ls_i \geq lh_i; 1 \leq i \leq nf_j\} \tag{4}$$

where nf_j is the total number of flows passing through node j. Whenever node j is about to send a packet or has overheard a packet, it should update its flow table.

B. MAC layer modification

Whenever node j sends a packet, it should look up the flow table to find this packet's corresponding flow record f_i. First, the last-sent sequence number ls_i should be updated to the current packet's sequence number. Then, the packet accumulation status of flow i's immediate downstream neighbor nh_i should be calculated:

$$b_i = ls_i - lh_i - 1 \quad (5)$$

where b_i represents the number of packets that have been blocked at node nh_i. Based on this information, MAC layer operation should be adjusted. In DFBS, we choose the initial contention window size CW as the MAC layer performance adjuster. That is, if there are b_i packets blocked in the immediate downstream node nh_i, the initial CW of the current packet should be doubled b_i times starting from CW_{min}:

$$CW = min\{CW_{max}, 2^{b_i} CW_{min} + 1\} \quad (6)$$

Obviously, the bigger the CW, the longer the average transmission waiting time. When there are more packets blocked in the immediate downstream node nh_i, the channel access ability of the current sending packet of node j is reduced. Even if there is only one packet of flow i left at node nh_i, node j will let node nh_i take priority to forward this single packet. By this means, the contention of node nh_i can be decreased. More time will be left for node nh_i to forward those blocked packets and packet accumulation in bottleneck nodes can be alleviated. Even when there are no bottleneck links in flow i, with DFBS, there is less chance for packet accumulation due to arbitrary contentions as in the original 802.11 system.

For each overheard data packet, node j should also look up its flow table to find out if this packet belongs to one of the flows in its flow table. In case it belongs to flow i, and its sender address is just the next-hop address nh_i of flow i, the last-heard sequence number lh_i of f_i should be updated to this packet's sequence number.

C. Packet scheduling at nodes

In the 802.11 protocol, each node only maintains one queue for all packets which may belong to different flows waiting for MAC layer processing. So, besides the MAC layer processing time, packets will also encounter a queuing delay. The MAC layer processing time includes all the handshaking time, backoff time and retransmission time of the packet. In the above MAC layer modifications, some packets may have longer MAC layer processing time due to packet blocking at downstream nodes. This longer MAC layer processing time benefits the packet's own flow by improving the transmission chance of downstream bottleneck links and alleviate the packet accumulation at bottleneck nodes. It also benefits other transit flows which can use the spare channel resource. However, it is unfair to those flows which also pass through this node and have packets waiting in the queue for MAC layer processing. To address this problem, we take the following steps to implement packet scheduling at each node, considering both system throughput and fairness.

Since those packets with worse packet accumulation status in their immediate downstream nodes will have longer average MAC layer processing time if they are passed to the MAC layer, it will be more efficient if we shift part of their MAC layer processing time to the queuing time. Based on this idea, the packet accumulation status of b_i in the downstream node of each packet in the queue should be checked. The packet with the smallest b_i according to Equation 5 will be passed to the MAC layer to contend for transmission. However, there may be cases in which some packets forwarded by downstream nodes are missed by this node j, or packets are dropped. In these cases, some packets in the queue may never get the chance to have the smallest b_i. These packets may be blocked forever and their corresponding flows are starved. So we introduce a packet checking window for each packet queue q_j at node j, which starts from the HOL packet for w_j consecutive packets. Instead of always sending the HOL packet to the MAC layer for transmission, all the packets in the packet checking window should be checked for their downstream nodes' packet accumulation status. The packet with the smallest b_i in the packet checking window should be sent to the MAC layer for transmission. However, each HOL packet should be blocked no more than $w_j - 1$ times. That means, if the HOL packet has been blocked for $w_j - 1$ times, it should be sent to the MAC layer next time without any checking. By this means, a balance is drawn between efficiency and fairness.

D. Collision rate of bottleneck links

In this subsection we will give a simple analysis of collision rate reduction by our DFBS scheme. The first transmission attempt of a packet has the highest probability of collision, since the contention window size is the smallest at the first attempt. So we take the first attempt collision probability analysis as a performance bound. To simplify the analysis, we further assume that all the contending links start their backoff at the same time and contentions from links with larger contention window sizes are ignored. The collision rate is:

$$P_c(i) = 1 - P_{CW_{min}}^{n_i} / (CW_{min} + 1)^{n_i} \quad (7)$$

where n_i denotes the contention number of a link i.

With DFBS, for a link which can send packet with CW_{min}, the contention number can be reduced to half or even less. Taking 3/4 and 1/2 as two examples we can show that there are dramatic collision rate reductions (see Fig. 3).

With the simplified analysis model above, one can also get the average waiting time before the first transmission as:

$$t_\tau = \min\{b_j, j = 0, 1, 2, ..., n_i\} \times slot\_time \quad (8)$$

where b_j is the backoff counter of contending link j, and j denotes the sequence number of link i's contending links and $j = 0$ represents link i itself. b_j is uniformly drawn from $[0, CW_{min}]$.

To calculate t_τ, let $slot\_time = 1$, we have:

$$F(t_\tau) = 1 - [1 - F_0(b_0)][1 - F_1(b_1)] ... [1 - F_{n_i}(b_{n_i})] \quad (9)$$

where $F(\bullet)$ is the probability distribution function. Since all the backoff counters are drawn from the same interval with a

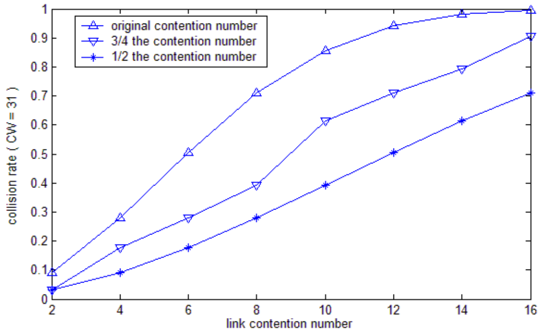

Fig. 3. Collision rate comparison with 3/4 the contention number and 1/2 the contention number

uniform distribution, we can get $F(t_\tau)$ as:

$$\begin{aligned} F(t_\tau) &= 1 - [1 - F(b_0)]\,[1 - F(b_1)] \ldots [1 - F(b_{n_i})] \\ &= 1 - [1 - F(b)]^{n_i+1} \\ &= \begin{cases} 1 - \left[1 - \frac{t_\tau + 1}{CW_{min}+1}\right]^{n_i+1} & 0 \le t_\tau \le CW_{min} \\ 0 & t_\tau < 0 \\ 1 & t_\tau > CW_{min} \end{cases} \end{aligned} \tag{10}$$

So the probability density function of t_τ, and the expectation of t_τ is:

$$f(t_\tau) = F(t_\tau) - F(t_\tau - 1) \tag{11}$$

$$E(t_\tau) = \sum t_\tau f(t_\tau) \tag{12}$$

From the above analysis, we can get the expected waiting time of the first transmission. As shown in Fig. 4, the average waiting time increases by no more than three timeslots when the contention number decreases to half. And, since each time a collision happens, CW will be doubled and the delay time will increase much faster than 3 timeslots per link, when we take both the collision rate and the total transmission delay into consideration, this additional waiting time is negligible. So the average transmission delay of the MAC layer can be reduced by DFBS, and consequently the throughput can be improved.

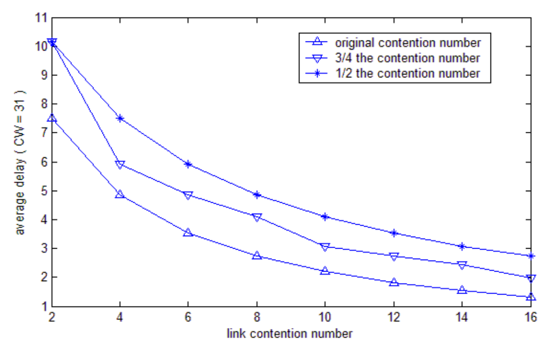

Fig. 4. First attempt transmission delay comparison with 3/4 the contention number and 1/2 the contention number

IV. Simulation Results

To evaluate the performance of DFBS, we implemented a simulation on ns2 [7]. We simulated both the pure IEEE 802.11 MAC protocol and our DFBS. Since the simulated buffer size was 50 packets for each node, we chose a packet checking window size of five in the packet queue to address both fairness and efficiency. We measured the one-hop throughput and the transmission efficiency of the network. As defined in [1], one-hop throughput measurements count all radio transmissions for data packets that successfully arrive at their final destinations, including packets forwarded by intermediate nodes. Transmission efficiency is the ratio of successful one-hop transmissions to total radio transmissions.

A. Single flow scenario

We first considered a simple scenario of a single flow with seven nodes as shown in Fig. 1. In this simulation, we used 40kbps, 80kbps, ..., 400kbps CBR as traffic loads, respectively. Figs. 5 and 6 provide the one-hop throughput and the transmission efficiency of this single flow scenario.

From Fig. 5, we observe that with the 802.11 protocol, when traffic load increases to more than 120kbps, the one-hop throughput of the system starts to decrease. However, our DFBS scheme can maintain a higher one-hop throughput even after traffic load exceeds the saturation point. Taking 400kbps traffic load as an example, the throughput of the 802.11 system is only 45.8% that of the DFBS. This can be attributed to two reasons. First, when traffic load is relatively high, with higher collision rate as analyzed in Section III, the MAC layer processing time of the ordinary 802.11 system is much larger than that of the DFBS system, and the throughput is consequently much less. Furthermore, packet accumulation due to non-coordinated packet forwarding in the 802.11 system results in packet dropping. This further decreases the end-to-end throughput. We also find that, when traffic load is light, the one-hop throughput of DFBS is slightly less than that of the 802.11 system. This can be attributed to the compulsive increase of CW, which results in a slightly longer MAC layer processing time when traffic load is light.

From Fig. 6, we find that, in the 802.11 system, the transmission efficiency starts to decrease when traffic load increases to more than 120kbps. When the traffic load is 400kbps, the transmission efficiency is only 58.1%. However, with our DFBS scheme, the transmission efficiency of the network is maintained at almost 100%, which means all the packets sent reach their final destinations. The decrease of transmission efficiency in the 802.11 system is due to the packet accumulation which results in packet dropping when the accumulation exceeds the buffer limit. In DFBS, packet forwarding at each hop of a multi-hop flow is coordinated and packet accumulation can be alleviated dramatically.

B. Random multi-flow scenario

We next simulated a multi-hop multi-flow scenario, where 50 nodes were randomly distributed in a square area of $800m \times 800m$. 10, 20, 30, 40 and 50 20kbps CBR flows were simulated. Figs. 7 and 8 show the one-hop throughput and the transmission efficiency of these random multi-flow scenarios.

From Fig. 7, we find that DFBS's one-hop throughput is always higher than that of the 802.11 system. When traffic load is high, the one-hop throughput improvement can be up to 43.0%. Besides the two reasons stated in the single flow scenario, these throughput improvements can also be attributed to constrained channel resource consumption along the whole flow path. When bottleneck links exist in multi-hop flows, the upstream links can automatically constrain their bandwidth consumptions according to the bandwidth share of the bottleneck links. By this means, the very limited bandwidth resources can be spared for other flows and thus the overall system throughput is improved.

Fig. 8 shows the transmission efficiency of the two systems. DFBS also outperforms the 802.11 system in all cases. The heavier the traffic load the greater the improvement. When traffic load is high, the transmission efficiency improvement can be up to 45.2%.

V. Conclusions

In this paper, we present a novel distributed flow-based scheduling scheme (DFBS) based on the 802.11 MAC protocol. DFBS takes advantage of the broadcast nature of shared channel ad hoc network and brings coordination to neighbor links of multi-hop flows. In contentions between successive hops of a multi-hop flow, a downstream node forwarding old packets is given higher priority over an upstream node sending new packets. Using the proposed approach, contention can be reduced between successive hops of a flow and each link's channel access ability is automatically limited to the bottleneck link's ability. As demonstrated in the simulation results, based on the proposed algorithm, both the system throughput and the transmission efficiency are improved.

Acknowledgment

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. HKU 7044/02E).

References

[1] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," *Proc. ACM SIGMOBILE*, Rome, Italy, July 2001.

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

[3] X. L. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation," *Proc. ACM MobiCom*, Long Beach, CA, Oct. 2001.

[4] V. Kanodia, A. Sabharwal, B. Sadeghi, and E. Knightly, "Ordered packet scheduling in wireless ad hoc networks: mechanisms and performance analysis," *Proc. ACM MobiCom*, Lausanne, Switzerland, June 2002.

[5] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly, "Distributed priority scheduling and medium access in ad hoc networks," *Wireless Networks*, vol. 8, no. 5, pp. 455–466, Sep. 2002.

[6] IEEE, IEEE standard 802.11, Wireless LAN medium access control (MAC) and physical layer (PHY) specifications (1999).

[7] ns-2 Network Simulator, www.isi.edu/nsnam/ns, (version 2.1b8a).

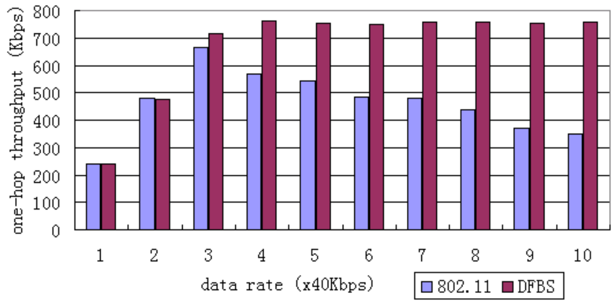

Fig. 5. One-hop throughput of six hop flow structure

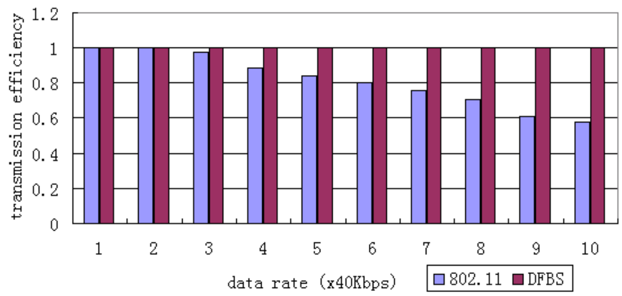

Fig. 6. Transmission efficiency of six hop flow structure

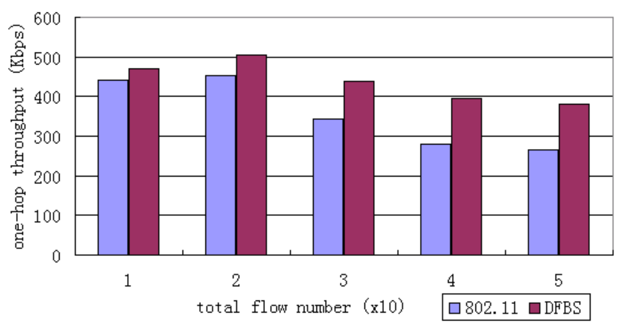

Fig. 7. One-hop throughput of 50 nodes in $800m \times 800m$ area

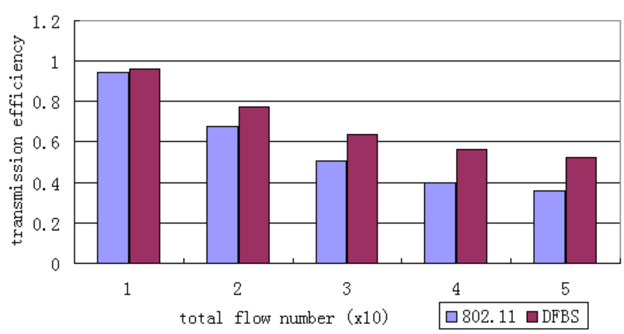

Fig. 8. Transmission efficiency of 50 nodes in $800m \times 800m$ area

