Implementing TCP Flow-Level Fairness Using 802.11e in a Multi-Radio Mesh Testbed

Venkataramana Badarla, David Malone, Douglas J. Leith

Abstract— **TCP is known to be subject to poor performance in multi-hop mesh networks. While most work has focused on interference and related issues, we note that cross-layer interactions can also induce considerable unfairness. In this paper, we propose a simple 802.11e scheme to mitigate these interactions and regulate TCP fairness in a flexible manner. The practical effectiveness of the approach is confirmed in a multi-hop, multi-radio testbed.**

Keywords— **802.11e, CSMA/CA, TCP, test-bed**

I. Introduction

The problems faced by TCP in 802.11 multi-hop mesh wireless networks have been the subject of considerable interest in recent years. However, almost all of this work has focused on interference and radio-noise related issues. Recently, [1] noted that the MAC layer can lead to undesirable bandwidth allocations for voice traffic in wireless mesh networks. Similar unfairness problems exist with TCP traffic in these networks. This unfairness exists independently of lower layer interference issues and can interact badly with TCP's congestion control mechanism. While it is a known issue in single-hop WLANs (see e.g. [2] and references therein), it has received little attention in a multi-hop context.

In this paper, we demonstrate that this is a significant issue for TCP in a multi-hop network and, moreover, by using simple 802.11e settings we can substantially improve TCP performance. The effectiveness of the proposed scheme is confirmed using an experimental testbed equipped with multi-radio relay stations built using commodity hardware. The proposed approach creates a straightforward framework for implementing a wide range of fairness policies within multi-hop networks using commodity hardware.

We begin by highlighting two issues, both caused by 802.11's tendency to give the same number of transmission opportunities to each wireless station when the network becomes busy. The first problem is that TCP expects the return path for TCP ACKs to be uncongested. Congestion in the reverse path has long been known to be a problem for wired TCP (e.g. [3]), and is also known to be an important problem for TCP in single-hop 802.11 WLANs (e.g. see [2] and references therein). This problem is also present whenever TCP flows are aggregated at a relay hop in a mesh network. This leads to queueing and loss of TCP ACKs and so to performance degradation due to disruption of TCP ACK clocking. The second problem is that at a relay hop a relay station, regardless of how many downstream stations/flows it is relaying packets for, will receive the same number of transmission opportunities as other contending stations. As a result, flows that travel through a relay station carrying many other flows may receive only a small share of the available bandwidth.

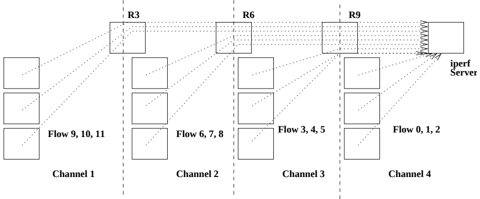


Fig. 1. Parking lot topology. **Rn** denotes a node relaying n flows.

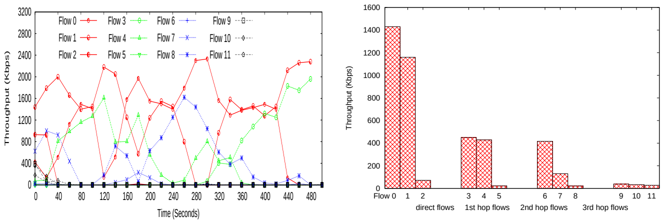


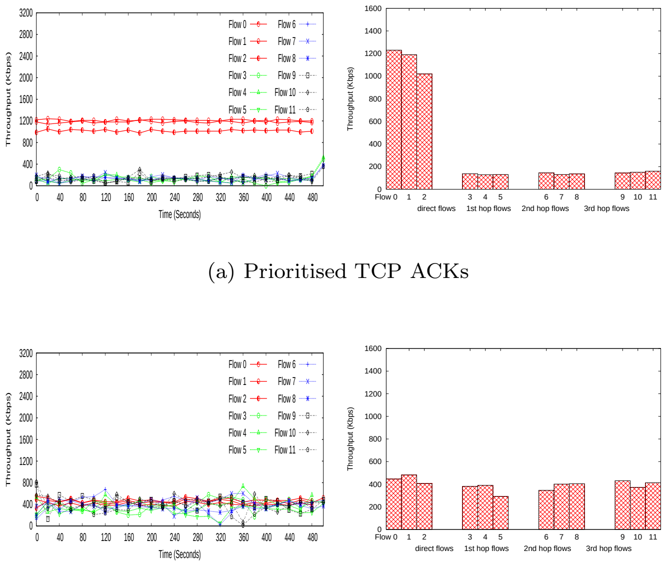
Fig. 2. TCP performance in 802.11 parking lot topology of Figure 1.

We illustrate the impact of these effects for the 4-hop parking lot topology shown in Figure 1. TCP flows travel from each station shown to the rightmost station. TCP ACKs are routed in the opposite direction. Each hop is on an orthogonal radio channel, and the relay stations are multi-radio, thus we avoid interference and hidden node problems and can focus on MAC/transport layer interactions. Figure 2 shows TCP goodput against time for 12 simultaneous TCP flows, one from each station. We see that the throughput is highly erratic and that flows travelling a smaller number of hops tend to starve flows travelling a larger number hops.

II. 802.11e Mesh Testbed Setup

We implemented the topologies shown in Figure 1 and Figure 5 using a testbed constructed from Soekris net4801 stations with Atheros 802.11a/b/g miniPCI cards. The NICs used support 802.11e EDCF functionality which makes adjustable the MAC parameters AIFS, CWmin and TXOP. All stations run Linux 2.6.21.1 with a version of the MADWiFi driver customised to allow the priorisations described in this paper. Otherwise, tests were performed in infrastructure mode using standard 802.11a parame-

Work supported by Science Foundation Ireland grant IN3/03/I346. The authors are with the Hamilton Institute, National University of Ireland, Maynooth, Co. Kildare, Ireland.



(a) Prioritised TCP ACKs

(b) Prioritised TCP ACKs and TXOP

Fig. 3. Improved 802.11e performance for parking lot topology.

ters and channels with PHY rate 6Mbps. To implement dual-radio mesh relay points, we joined two net4801s at 100Mbps with a cross-over cable to form a single logical mesh point. This avoided potential interference between network cards sharing the same PCI backplane. Routing in the network was statically configured. Iperf was used to generate TCP traffic and data was collected from both iperf and tcpdump. TCP flows used SACK enabled NewReno.

III. Prioritising TCP ACKs

First we address TCP ACK congestion. Similarly to [2], by increasing the AIFS of data packets by 4 slots and decreasing the CWmin of TCP ACKs to 2^2 we can effectively give ACKs priority over data packets. Allowing TCP ACKs prioritised access to the wireless channel does not lead to the channel being flooded. Instead, it ensures that the volume of TCP ACKs is regulated by the transport layer rather than the MAC layer. In this way the volume of TCP ACKs will be matched to the volume of TCP data packets, thereby restoring forward/reverse path symmetry at the transport layer[1]. To see the effect of this change, compare Figure 2 to Figure 3(a). We see that TCP's performance is now much more predictable, and even flows that travel many hops are not starved.

IV. Achieving Per Flow Fairness

While prioritising TCP ACKs ensures more predictable TCP performance, it is evident from Figure 3(a) that a wide range of throughputs are achieved by the flows, with flows 0–2 significantly out-performing the other flows. We can understand this by recalling that the DCF enforces per-station fairness, i.e. each station contending for access gains roughly the same number of transmission opportunities. Per station fairness takes no account of the number of flows being relayed by a station. Hence, when we have a station relaying n flows contending with a station relaying a single flow, each station is allocated a $1/2$ share of the channel capacity even though this means that the n flows are each allocated a $1/2n$ bandwidth share while the single flow gains a $1/2$ share. Over a number of cascaded hops, this type of behaviour can lead to the flow level unfairness potentially becoming large.

To address this unfairness, we consider how to achieve per flow rather than per station fairness. Here “flow” refers to any suitable administrative quantity and its precise definition is left as a policy decision for the network operator — we might for example choose to define a flow to consist of all packets originating from one IP address, or group of IP addresses, rather than the usual source and destination address/port tuple.

We propose that the TXOP packet bursting mechanism in 802.11e provides a straightforward and fine grained mechanism for controlling fairness. Let n_1^i denote the number of outgoing flows at a station on radio i (stations may have multiple radios). In order to achieve a per-flow fair allocation of bandwidth, at each station we choose TXOP equal to n_1^i for TCP traffic transmitted by radio i. That is, we transmit a number of packets equal to the number of flows carried.

To see the impact of this TXOP allocation, let n^i denote the number of stations contending for transmission opportunities on channel i. Let n_j^i denote the TXOP used by station j, $j = 1, 2, .., n$. The share of transmission opportunities allocated to station j is then approximately $n_j^i / \sum_{j=1}^{n^i} n_j^i$. Since station j transmits n_j^i flows, the per flow share is $1/\sum_{i=1}^{n^i} n_j^i$. In other words, transmissions are shared out equally amongst the flows. Note that even if a station has a certain allocated value of TXOP time, if during a transmission opportunity a node has no packets to send then that transmission opportunity is ended. Thus, if the offered load at a station is too low to make full use of its allocated TXOP share, the excess is not lost but rather becomes available on a best effort basis for use by other stations in the network.

We demonstrate the impact of this TXOP allocation strategy in the parking lot topology of Figure 1, setting the 802.11e TXOP parameter to 3, 6 and 9 packets for the left, middle and right relay stations. The impact on TCP flow throughput is shown in Figure 3(b). It can be seen that the per flow throughputs are now close to equal.

A. Avoiding large packet bursts

We note that increasing TXOP may result in a long period of uninterrupted transmission by one station. An alternative to a single long transmission is to use multiple shorter transmissions via a combination of TXOP and CWmin. For example, by halving the CWmin value at a station we can allow it to get roughly twice as many trans-

[1] Note, this also provides a basis for policing TCP ACKs, if needed.

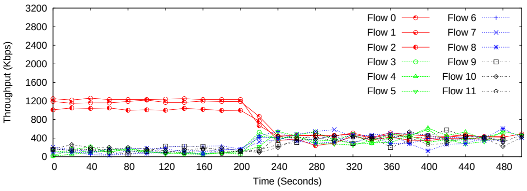


Fig. 4. Convergence when optimised parameters are applied.

mission chances, and then make each of these half as long i.e. reduce TXOP by half. Note that by using a combination of TXOP and CWmin in this way we gain fine grained control while respecting the practical constraint that CWmin can only be adjusted in powers of two. The mean MAC delay when using DCF at relaying nodes R9, R6 and R3 are 6.4ms, 2.5ms and 2.3ms, respectively. When using TXOP the delays at these relaying nodes are 3.3ms, 2.6ms and 2.8ms; for TXOP/2 and CWmin/2, these delays are as 3.3ms, 2.5ms, and 2.8ms. So, using TXOP actually improves the mean delay at R9, while having little impact at R6 and R3, where the network is less busy.

B. *Dynamic reconfiguration*

The number of flows in a network is typically not constant. Fortunately, it is usually straightforward for a station to determine the number of flows that its is transmitting. For example, if a flow is defined by its source and destination address/port tuple then inspection of the network interface queue allows direct measurement of the number of currently active flows. The TXOP settings can be readily adjusted on the fly — potentially at every transmission opportunity. This is illustrated, for example, in Figure 4 where TXOP at each station is initially at its default value of one (0s–200s) and then TXOP adaptation to the number of flows is enabled (200s–500s). We can see that following this major change there is a transient phase where the changes are applied from 200s–220s, and by the interval 220s–240s the performance is in line with what we expect from the new parameters.

C. *Other topologies*

Figure 5 shows an example of a hierarchical mesh topology. Here two flows come from the left hand side of the hierarchy and four from the right. Again, we consider the impact of prioritising the TCP ACKs and then setting TXOP to match the number of TCP flows relayed by a node. Figure 6 shows the performance of the stations with DCF alone, the network when we use the TCP ACK prioritisation scheme and TCP ACK prioritisation with TXOP. The results shown are an average of 10 runs and error bars show 95% confidence intervals. Again, we see that TCP ACK prioritisation had reduced the variability of the results substantially. Further, by adapting TXOP, we have been able to create almost exact fairness between flows.

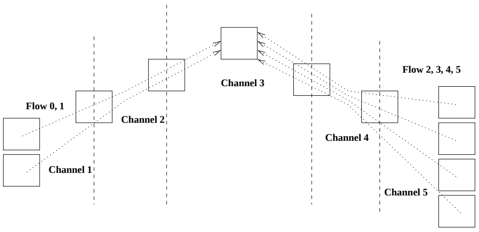


Fig. 5. Hierarchial topology.

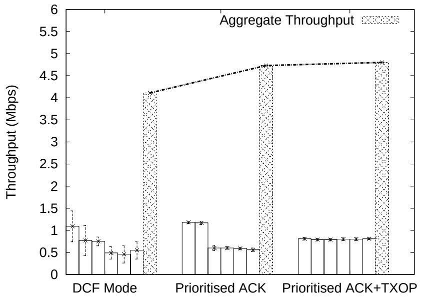


Fig. 6. Overall results from Hierarchial topology

D. *Impact on network capacity*

The overall network throughput is also shown in Figure 6. In addition to achieving fairness among the flows, the proposed scheme yields a ∼15% increase in network capacity over DCF. This gain is associated both with the increased efficiency (reduced MAC overhead) associated with the use of larger TXOP bursts and the positive impact of TCP ACK prioritisation on TCP performance. These benefits outweigh the cost of increasing AIFS for TCP data.

V. Discussion

We have demonstrated that the proposed scheme yields improved fairness and throughput, while being simple to implement on commodity hardware. These basic techniques can be readily extended in a number of directions. For example, differential prioritisation between TCP flows can be achieved by associating weight w_f with flow f, selecting TXOP=$\sum_{f=1}^{n_j} w_f$ and using a slightly modified queueing discipline at network interfaces. This 802.11e approach provides a simple and direct approach to distributed fair scheduling, avoiding complex message passing and custom scheduling schemes e.g. [4], [5].

References

[1] K. Duffy, D. Leith, T. Li, and D. Malone, “Modelling 802.11 mesh networks,” *IEEE Commun. Lett Letters*, vol. 10, no. 8, pp. 635–637, 2006.

[2] D. Leith, P. Clifford, D. Malone, and A. Ng, “TCP fairness in 802.11e WLANs,” *IEEE Commun. Lett*, vol. 9, no. 11, pp. 964–966, 2005.

[3] T. Lakshman, U. Madhow, and B. Suter, “Window-based error recovery and flow control with a slow acknowledgement channel: a study of TCP/IP performance,” in *Proc. INFOCOM*, 1997.

[4] 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. ACM MOBICOM*, 2001.

[5] M. Alam, M. Hamid, and C. Hong, “QoS-aware fair scheduling in multihop wireless ad hoc networks,” in *Proc. ICACT*, 2006.