Insights into the Design of Congestion Control Protocols for Multi-Hop Wireless Mesh Networks

Ihsan Ayyub Qazi, Daniel Mosse
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260, USA
Email: {ihsan,mosse}@cs.pitt.edu

***Abstract*—The widespread deployment of multi-hop wireless mesh networks will depend on the performance seen by the user. Unfortunately, the most predominant transport protocol, TCP, performs poorly over such networks, even leading to starvation in some topologies. In this work, we characterize the root causes of starvation in 802.11 scheduled multi-hop wireless networks via simulations. We analyze the performance of three categories of transport protocols. (1) End-to-end protocols that require implicit feedback (TCP SACK), (2) Explicit feedback based protocols (XCP and VCP) and (3) Open-loop protocol (UDP). We ask and answer the following questions in relation to these protocols: (a) Why does starvation occur in different topologies? Is it intrinsic to TCP or, in general, to feedback-based protocols? or does it also occur in the case of open-loop transfers such as CBR over UDP? (a) What is the role of application behavior on transport layer performance in multi-hop wireless mesh networks? (b) Is sharing congestion in the wireless neighborhood essential for avoiding starvation? (c) For explicit feedback based transport protocols, such as XCP and VCP, what performance can be expected when their capacity estimate is inaccurate? Based on the insights derived from the above analysis, we design a rate-based protocol called *VRate* that uses the two ECN bits for conveying load feedback information. VRate achieves near optimal rates when configured with the correct capacity estimate.**

I. Introduction

Multi-hop wireless mesh networks have been proposed as an alternative to wired connectivity in many settings such as in community wireless networks, disaster relief scenarios, etc [5], [11]. The widespread deployment of such networks will depend on the performance seen by the users in terms of response times and throughput. Unfortunately, TCP performs poorly over such networks. In many cases, some flows get completely starved due to lack of coordination among nodes. This situation doesn't improve if we employ MACs with higher bit rates such as 802.11a/g instead of 802.11b.

In wired networks, congestion at a link is *only* caused by flows passing through it. In such a case, flows compete for two shared resources, link bandwidth and buffer space. When congestion takes place, all flows are signalled to reduce their rates[1]. However, in wireless networks, congestion is not only caused by flows that pass through the congested link but also by those that do not traverse the congested link. This happens because in wireless networks, in addition to bandwidth and router buffers, space is also a shared resource. Flows sharing the same wireless medium interfere with one another and in many cases, nodes few hops away also cause contention and packet losses.

In this work, we investigate the following important questions related to multi-hop wireless networks:

- Why does starvation occur in different topologies? Is it intrinsic to TCP or, in general, to feedback-based protocols? or does it also occur in the case of open-loop transfers such as CBR over UDP?
- What performance can we expect from different transport protocols *e.g.*, TCP, XCP [3], and VCP [13] under different topologies?
- Many explicit-feedback based transport protocols such as XCP, VCP, MLCP [9], BMCC [7], [8] require an estimate of the capacity for feedback computation. What performance do we get when the capacity estimate is inaccurate?
- If we can determine the optimal max-min rates of flows, does that suffice for achieving fair rates? Does that ensure that no starvation takes place? Is congestion sharing essential in the case of feedback-based protocols?

The rest of the paper is organized as follows: Section II analyzes the performance of CBR traffic over UDP and FTP over TCP under the Stack topology. Section III analyzes the performance of some of the explicit feedback based congestion control protocols. In Section IV, we present *VRate*, a rate-based protocol for wireless mesh networks. We analyze more topologies in Section V. We discuss some issues in Section VI. We discuss related work in Section VII. Finally, we discuss future work in Section VIII and offer concluding remarks in Section IX.

II. Understanding Starvation

In this section, we use the Stack topology (see Figure 1) to understand starvation in multi-hop wireless mesh networks.

A. Simulation Setup

We use the Network Simulator [6] version 2.33 for our simulations. All the simulations are conducted using 802.11b MAC (DCF) unless specified otherwise. Auto-rate adaptation is not used at the MAC layer and the rate is fixed at 11 Mbps. All simulations are conducted with zero channel losses, although packet losses due to collisions do occur. In order to generate the above topologies, the interference range was made equal

[1]While this is not true in the case of droptail routers where only few unlucky flows that experience losses are signalled to reduce their rates but this is largely the case with ECN-enabled RED routers.

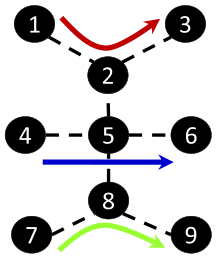


Fig. 1. Stack topology

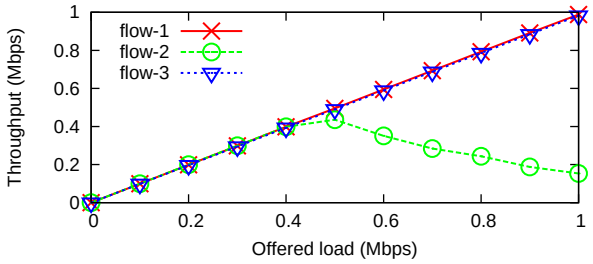


Fig. 2. Throughput of CBR flows (over UDP) with 802.11b MAC under the Stack topology

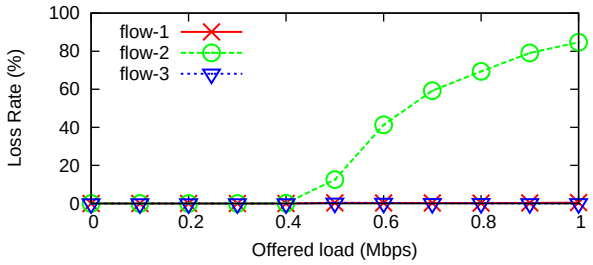


Fig. 3. Loss rate for CBR flows (over UDP) with 802.11b MAC under the Stack topology

to the transmission range as used in [10][2]. We consider bulk transfer flows which are made to run for 200 s. The Interface Queue (IFQ) size is set to 64 packets. We use static routing in all simulations. Each simulation scenario is run 10 times with starting times that are generated uniformly at random in [10, 20] s. TCP SACK is used with RED and ECN support. In case of UDP, we use Constant Bit-Rate (CBR) traffic with droptail routers.

B. CBR over UDP

In this section, we analyze the performance of CBR traffic over UDP under the Stack topology (see Figure 1). Figure 2 shows the throughput of three (top, middle and bottom) CBR flows as a function of the offered load. Observe that when the offered load is less than 0.4 Mbps, throughput of all flows increases linearly with the offered load (roughly equals the $y = x$ line). However, as we increase the offered load beyond 0.4 Mbps, throughput for the middle flow starts decreasing whereas the top and the bottom flows continue to achieve higher throughput. Figure 4 shows the throughput of flows under the 802.11a MAC which offers a higher maximum bit rate of 54 Mbps than 802.11b MAC. Observe that increasing the capacity doesn't prevent the middle flow from achieving low throughput during times of congestion.

This happens because when the offered load increases beyond 0.4 Mbps, the middle flow experiences congestion due to the spatial location of the nodes along its route. Note that while the middle flow contends with the top and the bottom flows, the other flows only contend with the middle flow. Since the top and the bottom flows are not aware of each other, their accesses are not coordinated. Hence, the middle flow does not get much opportunities to send traffic when the offered load increases beyond a certain threshold. Figure 3 shows the corresponding loss rate experienced by these flows. Observe that when the offered load is 1 Mbps, the middle flow experiences a loss rate of more than 80% causing the throughput to degrade.

C. FTP over TCP

In this section, we analyze the performance of FTP traffic over TCP. Figure 5 shows the throughput of the three flows. Observe that the middle flow gets starved while the two outer flows grab most of the bandwidth. The reason is that when link 4→5 gets congested, only the middle flow is signalled to reduce its congestion window. Thus, the middle flow reacts more aggressively to congestion, which causes it to achieve very low throughput. This is a consequence of the lack of coordination that exists between these flows. From the perspective of transmission opportunities, this can also be explained as follows: Nodes 2, 5 and 8 have different views of the underlying channel. At the MAC layer, node 5 can only transmit when nodes 2 and 8 are in their backoff phases. Under high load, node 5 gets very little air time which causes the queue length to grow at node 5. This either leads to packet losses or frequent congestion-experienced signals via ECN marks which causes TCP to timeout and in many cases to experience exponentially increasing timeouts due to the exponential backoff algorithm of TCP (see Figure 6).

III. Explicit Feedback-based Congestion Control Protocols that require the Capacity Parameter Estimate

In the precious section, we analyzed the performance of UDP and TCP flows under the Stack topology. In this section, we analyze the performance of XCP (eXplicit Congestion control Protocol) and VCP (Variable-structure Congestion control Protocol). The reason for considering XCP and VCP is that these protocols have been proposed as alternatives to TCP for high bandwidth-delay product networks and have been shown to be fair and efficient in wired networks. The widespread deployment of such protocols will depend on their performance over wireless networks such as multi-hop wireless mesh networks. Both these protocols require an estimate of the capacity for feedback computation. An inaccurate capacity estimate can either lead inefficiencies or

[2]This was done by reducing the carrier sensing threshold. The default carrier sensing and transmission ranges in NS are 550m and 250m, respectively.

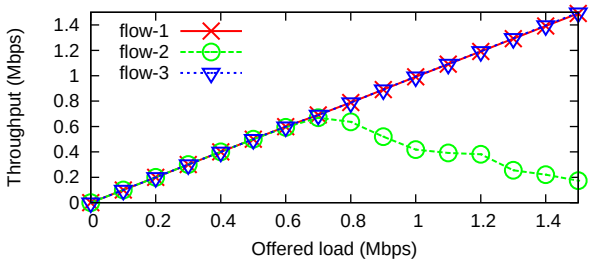


Fig. 4. Throughput of CBR flows (over UDP) with 802.11a MAC under the Stack topology

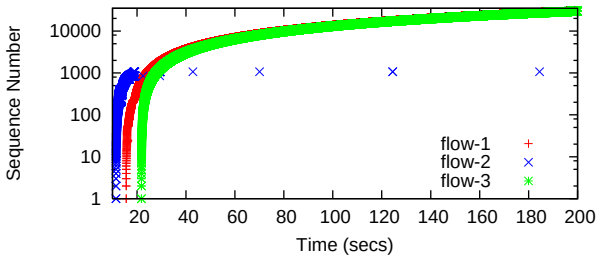


Fig. 6. Sequence plot of TCP flows under the Stack topology

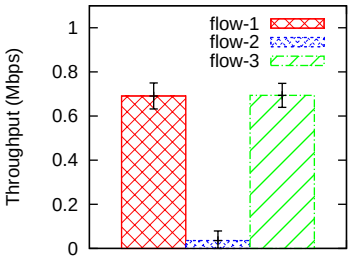
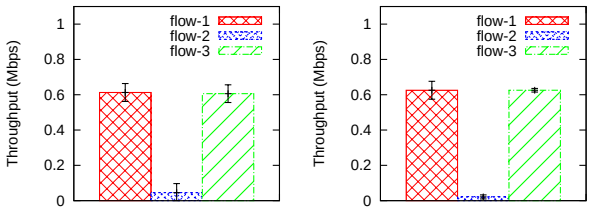


Fig. 5. Throughput of FTP flows under the Stack Topology



(a) XCP (b) VCP

Fig. 7. Throughput of FTP traffic over (a) XCP and (b) VCP under the Stack Topology

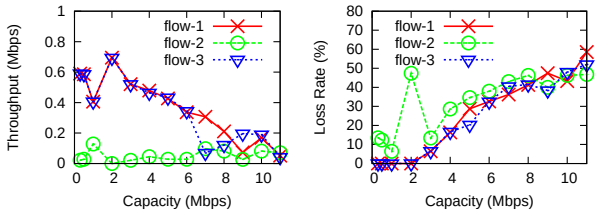
unfairness and large fluctuations in flow rates. Therefore, the performance of these protocol will depend on the accuracy and computation efficiency of the capacity estimate. We analyze and quantify the performance degradation due to different capacity estimates. We also investigate the need for congestion sharing in feedback-based congestion control protocols.

A. *FTP over XCP and VCP*

We now analyze the performance of FTP traffic over XCP and VCP under the Stack topology. We first consider the case where the optimal capacity is known. Figure 7 shows the throughput of flows for XCP and VCP. Observe that with XCP and VCP, the middle flow gets starved even when the routers are configured with the optimal capacity. The reason is as follows: The minimum congestion window size is one packet where the size of a packet is 512 bytes. The average RTT seen by the sources is ≈5.5 ms. This implies that sources cannot send at a lower rate than ≈0.75 Mbps. Some mechanism must be employed to reduce the rate of flows below this value. Note that rate based schemes would be able to handle this case naturally. Now, since the top and bottom flows are sending at much higher rates than their optimal rates, the middle flow starves. Note that reducing the packet size doesn't necessarily solve the problem. The reason is that small packets increase the overhead which also reduces the optimal max-min achievable rates.

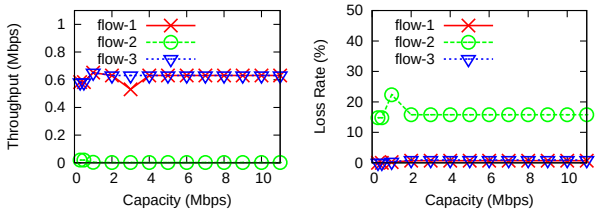
Varying the Capacity Parameter in XCP and VCP We now vary the value of the capacity parameter from 0.3 Mbps to 11 Mbps for XCP and VCP and quantify impact on their performance. Figure 8 shows the throughput and loss rate of the flows under the Stack topology. Observe that the middle flow achieves very low throughput across the entire range of the capacity parameter. However, as the capacity parameter value is increased, the throughput of the top and the bottom flow decreases due to increase in loss rates. This happens due to the following reason: The optimal max-min rate is 0.3 Mbps. Assigning a higher value to the capacity parameter generates a larger amount of positive feedback than desired. This causes sources to generate traffic that cannot be handled by the network and thus gets dropped causing the throughput to degrade. Note that higher the value of the parameter, higher is the performance loss.

Figure 9 shows the peformance VCP as a function of the value of the capacity parameter. Observe that while the middle flow completely staves with increasing capacity, however, it doesn't affect the throughput of the top and the bottom flows. The loss rate also remains constant beyond 2 Mbps which is unlike XCP's performance. The reason is that VCP uses a coarse-grained 2-bit feedback. Beyond 2 Mbps, the load factor (or congestion level) remains below 80% which causes the sources to apply MI. Since sources respond in the same way as they would with higher capacity values, they generate the same behaviour. With XCP, a higher capacity parameter value results in a larger positive feedback which causes the congestion window sizes of flows to assume large values that are proportional to the capacity parameter value. This can be seen in Figure 10 which plots the congestion window sizes for XCP and VCP flows when the capacity parameter is set to 11 Mbps. Observe that while XCP flows achieve window sizes that are as high as 1000 pkts, window sizes for VCP remains below 100 pkts.



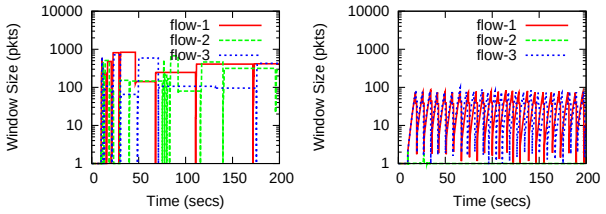
(a) Throughput (b) Loss Rate

Fig. 8. Throughput of FTP traffic over XCP as a function of capacity under the Stack Topology



(a) Throughput (b) Loss Rate

Fig. 9. Throughput of FTP traffic over VCP as a function of capacity under the Stack Topology



(a) XCP (b) VCP

Fig. 10. Congestion window size of XCP and VCP flows under the Stack Topology when the capacity is set to 11 Mbps

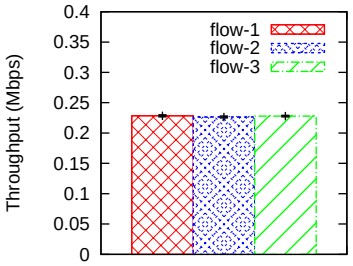


Fig. 11. Throughput of FTP traffic over VRate under the Stack Topology

IV. VRate: A Rate-based Protocol for Multi-hop Wireless Mesh Networks

In order to see the impact of removing the discrepancy with XCP and VCP, we implemented a rate-based protocol called *VRate* that uses one bits of explicit feedback which indicates if the rate is greater or less than 100%.[3]. The feedback computation uses the capacity estimate. VRate sources employ the AIMD control laws to adapt their sending rates. VRate works as follows: Each source starts with an initial sending rate of 25 Kbps. When the load factor, f is less than 100%, sources increase their rates according to the following policy:

$$r(t+T) = r(t) + \alpha \quad (1)$$

where $T = 50ms$ is the rate update interval and $\alpha = 10\,Kbps$. This corresponds to a rate of increase of 200 Kbps every second[4].

When $f > 100\%$, sources apply the Multiplicative Decrease (MD) policy. This translates into the following rate adjustment strategy:

$$r(t+\Delta) = r(t) \cdot \beta \quad (2)$$

[3]We could have employed some other strategy of reducing the rate of flows such as such using hybrid rate-based and window-based protocol or simply decreasing the frequency of packet transmission for this special case. We preferred to implement a complete protocol for convenience and simplicity of design.

[4]Ideally, T should be related to the round-trip of a flow and so does $r(t)$. Intuitively, if T is much smaller than the RTT of a flow, sources would increase their rates much faster than the rate at which they receive feedback which could be too aggressive a response. As an example, consider a single flow traversing a bottleneck link with capacity is $C = 25\,Kbps$ and $RTT = 500\,ms$. By the time, the source receives the first ACK, the source rate would be 125 Kbps which is five times larger than the bottleneck link rate. On the other hand, if T is much larger than the RTT of flows, source increase rates are conservative. Even though it is desirable to be conservative for stability purposes, being over-conservative is likely to cause inefficiencies on high BDP paths. In the topologies we consider, since the RTT is less than 50 ms, therefore, we set the value of T to 50 ms. We leave the dynamic adaptation of T according to the RTT for future work.

where Δ is the time elapsed since the last rate change before a source reduces its rate and $\beta = 0.875$ is the MD parameter. Note that when packet loss occurs, sources reduce their sending rates by a factor of two. The protocol uses the same algorithm for computing the retransmission timeout as used by TCP.

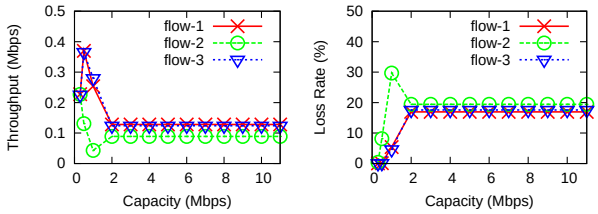
Figure 11 shows the throughput achieved by VRate flows when the capacity of each node along the path is set to 0.3 Mbps. Observe that all flows achieve the same throughput. This shows that as long as we ensure that the capacity parameter is set to no larger a value than the optimal rates, starvation will be prevented in the stack topology. Figure 12 shows the throughput of flows as a function of the capacity parameter. When $0.5 \leq C < 2$, the middle flow achieves very low throughput. Howver, for $C \geq 2$, the top and bottom flows achieve a throughput of around 130 Kbps whereas the middle flow achieves an average throughout of 89 Mbps. Note that for capacity parameter values larger than 2 Mbps, throughput behaviour doesn't change. This happens because the feedback doesn't change for such C values and therefore, the response is similar.

V. Topologies

We now analyze the performance of TCP, XCP, VCP, VRate, and UDP under other topologies in addition to the Stack topology.

A. Stack Topology

Figure 13 shows the throughput of flows under the stack topology for different transport protocols. For XCP, VCP and VRate the capacity estimate is set to the optimal value of 0.3 Mbps whereas in case of UDP, the CBR rate is set to 0.3 Mbps.



(a) Throughput (b) Loss Rate

Fig. 12. Throughput of FTP traffic over VRate as a function of capacity under the Stack Topology

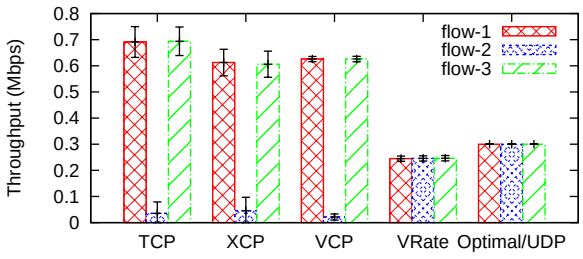


Fig. 13. Throughput of flows under the Stack topology

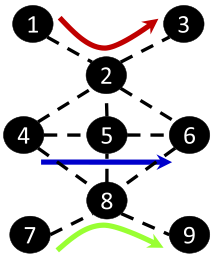


Fig. 14. Diamond topology

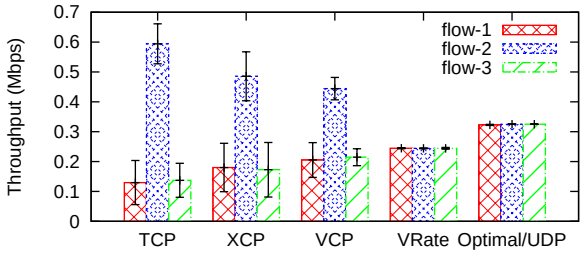


Fig. 15. Throughput of flows under the Diamond topology

B. Diamond Topology

Figure 14 shows the diamond topology. In this topology, when the offered load increases beyond a certain threshold, the top (at link 1→2) and the bottom (at link 7→8) flows experience congestion. Note that the difference between the diamond and the stack topology is the addition of extra links (4↔2, 4↔8, 2↔6, and 8↔6)[5] which introduces interference patterns that reduces the chances of a successful transmission from the top and the bottom flows. The optimal rates in this topology is 325 Kbps.

Figure 15 shows the throughput of flows under the diamond topology for different transport protocols. With TCP, the middle flow achieves about six times higher throughput. With XCP and VCP, the middle flow still achieves higher throughput, however, the top and the bottom flows achieve higher throughput than with TCP. In case of VRate, all flows achieve the same throughput, however, the throughout is about 50 Kbps less than optimal. The reason is that since the capacity parameter is set to 0.3 Mbps, the flow rates vary in [0.26,0.3] Mbps due to the AIMD nature of the protocol. This, however, is the case when there are no losses. When losses do occurs, sources back-off by a factor of 2. This losses are small, the average throughput is about 250 Kbps.

C. Half-Diamond Topology

In the half-diamond topology (see Figure 16), the middle flows experiences congestion. In this topology, nodes 4 and 6 are in the range of node 8 which isn't the case in the stack topology. Due to this, contention between the middle flow and the bottom flow is higher. Hence, the optimal rate for the top flow is 325 Kbps whereas for the middle and bottom flows, it is 315 Kbps. Figure 17 shows the throughput of flows under the half-diamond topology for different transport protocols. Observe that the top achieves the highest throughput across all protocols. With TCP, VCP and XCP, the middle flow achieves the lowest throughput. The difference in the throughput of flows is least with VCP among the three congestion control protocols. With VRate, the middle and the bottom flows achieve the same throughput with the top flow achieving a slightly higher throughput.

[5]where a↔b means that a and b are in range

D. Cross-Chain Topology

In the cross-chain topology (see Figure 18), flows 1 and 2 experience congestion at links 1↔2 and 1↔7, respectively. Note that unlike other topologies, flows in this topology do not have the same path length. Here, flow 2 traverses six hops whereas all other are one-hop flows. Observe that flows 1 and 2 experience the highest amount of contention whereas flow 3 the least, allowing it achieve a higher optimal rate. The optimal rate for flows 1, 2, 4 and 5, is 225 Kbps and for flow 3, it is 415 Kbps. Figure 19 shows the throughput of flows under the cross-chain topology for different transport protocols. Observe that with TCP, XCP and VCP, flows 1 are completely starved whereas all the bandwidth is grabbed by the other flows. With VRate, flows 1 and 2 achieve less throughput than other flows. In particular, flow 2 which has a path length of six hops, achieves almost half the throughput achieved by flows 4 and 5. This happens because it encounters the highest amount of contention. CBR traffic over UDP generated at the optimal rates, on the other hand, receives optimal throughputs.

VI. Discussion

Rate-based versus Window-based Designs: In window-based protocols such as TCP, the size of a window determines the maximum number of packets that can be transmitted in a given RTT. New packets are sent upon the receipt of

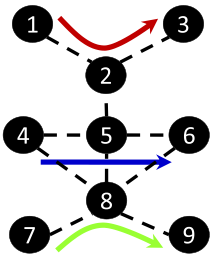


Fig. 16. Half-Diamond topology

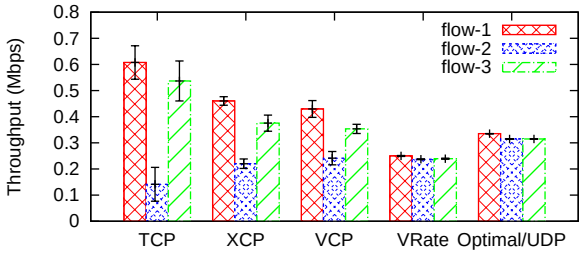


Fig. 17. Throughput of flows under the Half-Diamond topology

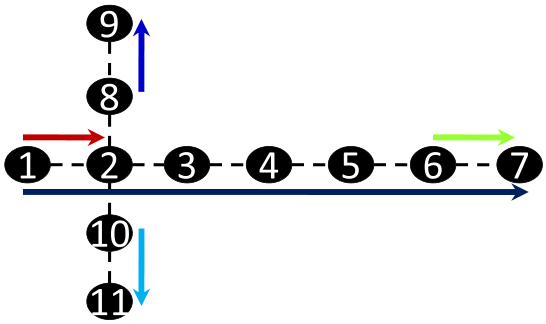


Fig. 18. Cross-Chain topology

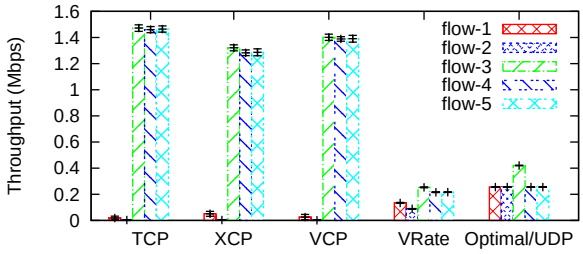


Fig. 19. Throughput of flows under the Cross-Chain topology

ACKs (a.k.a ACK Clocking). One concern with window-based protocols is that they tend to show bursty behavior due to ACK compression which can be quite pronounced in multi-hop wireless mesh networks. In the event of reverse-path congestion, ACKs experience queueing delays and their spacing can get reduced to a much smaller value than introduced by the bottleneck. This causes sources to send back-to-back packets as fast as it can, much higher than can be handled by the underlying network. Rate-based protocols maintain a dynamic rate and employ timers (rather than ACKs) for sending packets. At high sending rates, however, high precision timers are needed. Moderm operating systems now have available high precision timers that can be leveraged for this purpose. The key advantage of such protocols is their lack of burstiness.

VII. Related Work

[1] proposes an analytical model for determing per-flow throughput under aribitrary topoligies for understanding the root causes of starvation in CSMA-based multi-hop wireless networks. They show that the key reason for starvation is the lack of coordination in such protocols and not merely the number of competing nodes. They shows that losses can occur due to four reasons that correspond to different two-link topologies. They are as follows: losses due to collision between coordinated nodes (CO), losses due to Information Aysmmetry (IA), losses due to Near-Hidden terminals (NH) and losses due to Far-Hidden terminals (FH). In the experiments they conducted withy random topologies and one-hop flow demands, they witnessed that losses due to CO contributed to only a small fraction of the overall packet loss probability whereas losses IA was the dominant factor in most cases. In the topologies we consider, we have two-link sub-topologies where IA exists. Note, however, that they only used one-hop flows demands as opposed to multi-hop flow demands. [10] presented two congestion control protocols, WCP and WCPCap that outperformed TCP in the topologies we considered. WCP uses AIMD, RED with ECN support and congestion sharing to achieve better performance than TCP. They show that WCP doesn't achieve max-min rates. They hypothesize that it assigns rates that are inversely proportional to the number of congested neighborhoods due to the frequency of congestion feedback. They also proposed WCPCap, which estimates the capacity and uses explicit rate allocation. They, however, didn't test the communication overhead of capacity estimation. Another line of work uses back-pressure hop-by-hop congestion control. However, such protocols require per-destination queues [12]. Developing reliable capacity estimation algorithms is arguably the most important performance concern for protocols that rely on the capacity estimate for feedback computation. On this front, [2] proposed to use interference graphs given network topology and traffic demands and showed that the problem can be seen as a multi-commodity flow problem. However, they do not model the MAC and instead assume that packet transmissions can be finely scheduled across links. Such models tend to significantly over-estimate the performance of 802.11 networks. We saw in this work the performance is highly sensitive to capacity parameter over-estimation. [4] uses a theoretical model of 802.11 DCF and solves a centralized optimization problem for determining rate allocations to satisfy efficiency and/or fairness objectives. However, they do not discuss how to Capacity estimation can also be done via heutristics, which presents a tradeoff between accuracy and complexity.

VIII. Future Work

We plan to extend this work by considering HTTP, SMTP and other common applications with realistic traffic workloads to assess the performance of these congestion control protocols over multi-hop wireless mesh networks. Another possible direction for extending this work is the incorporation of mobility. The addition of mobility, however, introduces mobility related losses due to link breakages which must be carefully considered in the design of protocols. In this work, we only considered static routing, it would be interesting to look into the impact of MANET routing protocols and ETX, ETT etc route selection strategies. There is also room for modeling the performance of WCP and WCPCap-like protocols. In the experiments we conducted, we made the interference range equal to the transmission range to create interesting topologies. However, generally, the intereference range is larger than the transmission range and 802.11 nodes back-off by EIFS rather than the DIFS interval when they can't decode the overheard packet correctly. EIFS is much larger than DIFS and it has been shown that this can cause unfairness in wireless networks. It would be interesting to analyze the impact of this change on the performance of transport protocols in multi-hop wireless mesh networks. Further, it would useful to determine the set of possible topologies that can exist under this assumption.

IX. Conclusion

In this paper, we analyzed the performance of TCP, XCP, VCP and UDP using different application sources in a number of multi-hop wireless mesh network topologies. We showed that with FTP traffic sources, TCP, XCP and VCP result in significant unfairness in the topologies we consider, often leading to starvation. In case of TCP, it happens because of lack of coordination among flows causing congestion whereas with XCP and VCP starvation exists even when the capacity estimate is set to the optimal value. The happens due to the fact that they are unable to reduce their rates below a certain value due to the window-based nature of the protocols. We proposed VRate, a rate-based congestion control protocol that uses the two ECN bits to guide sending rates. VRate when configured with the correct capacity estimate is able to achieve near optimal rates. When using CBR traffic sources, we observe that when the load is below a certain threshold no starvation occurs.

References

[1] Garetto, M., Salonidis, T., and Knightly, E. Modeling Per-flow Throughput and Capturing Starvation in CSMA Multi-hop Wireless Networks. In *IEEE INFOCOM* (2006).

[2] Jain, K., Padhye, J., Padmanabhan, V. N., and Qiu, L. Impact of interference on multi-hop wireless network performance. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking* (New York, NY, USA, 2003), ACM, pp. 66–80.

[3] Katabi, D., Handley, M., and Rohrs, C. Internet Congestion Control for High Bandwidth-Delay Product Networks. In *ACM SIGCOMM* (Aug 2002).

[4] Li, Y., Qiu, L., Zhang, Y., Mahajan, R., and Rozner, E. Predictable performance optimization for wireless networks. *SIGCOMM Comput. Commun. Rev. 38*, 4 (2008), 413–426.

[5] Self-organizing neighborhood wireless mesh networks. http://research.microsoft.com/mesh/.

[6] ns-2 Network Simulator. http://www.isi.edu/nsnam/ns/.

[7] Qazi, I. A., Andrew, L. L. H., and Znati, T. Congestion Control Using Efficient Explicit Feedback. Tech. rep., Department of Computer Science, University of Pittsburgh, August 2008.

[8] Qazi, I. A., Andrew, L. L. H., and Znati, T. Two bits are enough. In *ACM SIGCOMM* (Seattle, WA, 17-22 Aug 2008).

[9] Qazi, I. A., and Znati, T. On the Design of Load Factor based Congestion Control Protocols for Next-Generation Networks. In *IEEE INFOCOM* (Apr 2008).

[10] Rangwala, S., Jindal, A., Jang, K.-Y., Psounis, K., and Govindan, R. Understanding Congestion Control in Multi-hop Wireless Mesh Networks. In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)* (San Francisco, September 2008).

[11] Roofnet. http://pdos.csail.mit.edu/roofnet/.

[12] Tassiulas, L., and Ephremides, A. Stability properties of constrained queueing systems and scheduling policies for maximum throughout in multihop radio networks. In *IEEE Transactions on Automatic Control* (1992).

[13] Xia, Y., Subramanian, L., Stoica, I., and Kalyanaraman, S. One more bit is enough. *SIGCOMM Comput. Commun. Rev. 35*, 4 (2005), 37–48.