Effects of Interference on Wireless Mesh Networks: Pathologies and a Preliminary Solution

Yi Li   Lili Qiu   Yin Zhang   Ratul Mahajan
University of Texas, Austin   *Microsoft Research*

Zifei Zhong   Gaurav Deshpande   Eric Rozner
University of Texas, Austin

Abstract – *We highlight two fundamental problems that degrade the throughput of wireless mesh networks today. First, severe performance degradation can occur when sources send more traffic than what the network can support. The degradation can be sharp even in a simple setting of a single flow that traverses a network of two links. Second, current routing protocols fail to identify high throughput routing paths even when they exist. The underlying culprit in both cases is interference that is fundamental to wireless networks.*

As a first step towards a solution, we develop a novel approach to systematically account for and control interference in the network. Our approach uses (an approximation of) a formal model of interference to estimate the maximum rate at which flows can safely send traffic without overloading the network. Simulation and testbed experiments show that it can improve network throughput by as much as 50-100% in some configurations.

1. INTRODUCTION

Wireless mesh networks are an attractive communication paradigm because of their low cost and relative ease of deployment. These networks typically consist of many base stations (BSs), some of which are directly connected to the Internet. Users connect to one of the BSs, and the BSs form a multihop wireless network to route traffic between the Internet and the users.

Wireless meshes have witnessed significant research and deployment activity recently. Many researchers have focused on improving their throughput through better routing [1, 2, 22, 15]. At the same time, many cities across the world have already deployed or are planning to deploy these networks [21, 20].

But in spite of significant attention, the performance of these networks today leaves much to be desired [17, 19, 18]. Users of almost all existing deployments have been complaining about poor performance. In many cases, complaints occurred even when the users are close to the BSs. This suggests that the routing backhaul formed by the BSs might be a major contributor [17].

In this paper, we use testbed experiments and simulations to understand the performance shortcomings of the routing methodology in mesh networks today. In this methodology, BSs compute routing paths based on measurements of link quality such as average loss rate [1]. They are then free to send as much traffic along these paths as the MAC (medium access control) layer permits.

We uncover two fundamental problems with this routing methodology. First, not controlling how much nodes send can severely degrade network throughput when they send more than what the path can support. This occurs because, due to interference, any additional traffic reduces the capacity of bottleneck links. We show how the degradation can be sharp even in a simple setting of a single flow traversing two links. We also show that end-to-end congestion control (e.g., using TCP) is not sufficient by itself to prevent this behavior.

The second problem is that current protocols are unable to accurately estimate link and path quality for the purposes of path selection. The underlying issue is that quality is measured by sending probes, without consideration to interference. The probes measure quality under current routing patterns. However, due to interference, the quality can change arbitrarily with any change in the routing pattern. As such, these measurements have limited predictive value because they cannot tell whether re-routing existing flows would result in better network throughput or which path is best for a new flow.

Motivated by these observations, we seek to develop a fundamentally different approach to routing – one that systematically accounts for interference effects. Only then can we advise nodes on how much traffic they can safely send and identify high throughput routing paths. This is challenging because interference introduces complex interdependencies. For instance, how much a node can safely send depends on how much other nodes are sending and vice versa. As a result, most existing works that systematically consider interference effects fall in the analytical domain [8, 7, 10, 5]. They make several strong assumptions about topology, workload, or interference characteristics, and cannot be straightforwardly adapted for use in a practical system. The other extreme is current mesh routing protocols, which are practical but largely ignore interference or account for it in rudimentary ways.

This paper presents the first step towards our quest. We develop a practical algorithm that estimates the maximum rate at which each flow can safely send. It takes as input the topology of the network, the routing paths of flows, and their desired sending rate. It captures the network's interference dependencies as an approximate conflict graph [8] and uses an iterative process to estimate the (max-min fair) safe sending rate for each flow. Our algorithm can be implemented in a fully distributed manner. To our knowledge, it is the first algorithm that is both practical and systematically accounts for interference in wireless mesh networks.

We have implemented our algorithm in a testbed and in a simulator. Preliminary evaluation reveals that limiting the nodes to the rates computed by our protocol significantly increases the network throughput. The exact gain depends on various factors but can be as high as 50-100%.

Our current algorithm does not compute routing paths themselves. But given a routing path pattern as input, it can estimate total network throughput, as the sum of estimated flow rates. As part of ongoing work, we plan to identify high throughput routing paths by searching over the space of possible options. This search will be purely computational, i.e., the routing patterns do not have to be installed in the network, but only fed into our algorithm.

2. RELATED WORK

Even though interference is fundamental to wireless networks, the body of work on routing in mesh networks has been developed independently of the work on systematically characterizing interference. Early protocols [12, 9] implemented shortest path routing, mostly ignoring the impact of interference. The next generation of protocols, such as ETX [1] and ETT [2], route based on measured link quality, loss rate (ETX) and transmission time (ETT), in the hope that links that suffer from interference will have poor quality. Two recent proposals, MIC [22] and iAWARE [15], attempt to account for interference by modifying how they measure link quality. To make it less likely to select paths that contain nodes with more neighbors (and thus interference), MIC uses the product of ETX and the number of neighbors [22]. iAWARE scales ETX by a function of the signal strengths at the receiver from its neighbors [15].

All protocols above follow the least cost model of routing in which the quality (inversely, cost) of links is measured and the highest-quality path is chosen. No restrictions are placed on how much nodes can send on a path. The protocols differ only in how they measure link quality. The next section highlights the shortcomings of this methodology.

In contrast to the work on routing protocols, there is a rich body of analytical work that studies the impact of interference in wireless networks. Gupta and Kumar analyzed the asymptotic capacity of a wireless network under assumptions of homogeneity and randomness in the network topology and traffic demands [7]. Since then, other researchers have extended this work to other traffic patterns [10], mobility [6], and network coding [5].

Of particular relevance to our work is the conflict graph model [8]. This model enables the computation of the *exact* (as opposed to asymptotic) bounds on optimal network capacity for a given topologies and traffic demands, assuming that packet transmissions can be finely scheduled across links.

While the analytical models systematically account for interference, they do not prescribe routing paths or make unrealistic assumptions about traffic, topology, and scheduling. This makes it hard to directly employ them for the purposes of designing a routing protocol. One contribution of our work is showing that such models can in fact be leveraged in a practical setting.

A recent work developed a rate control algorithm (IFRC) for sensor networks in which all nodes send traffic towards a sink over a tree topology [14]. We share with it the idea of rate limiting sources to prevent overload. However our approach applies to general multihop wireless networks, while IFRC is specific to the topology and workload of its domain.

3. PATHOLOGIES

In this section, we use simulation and testbed experiments to show two problems with current routing protocols for wireless mesh networks.

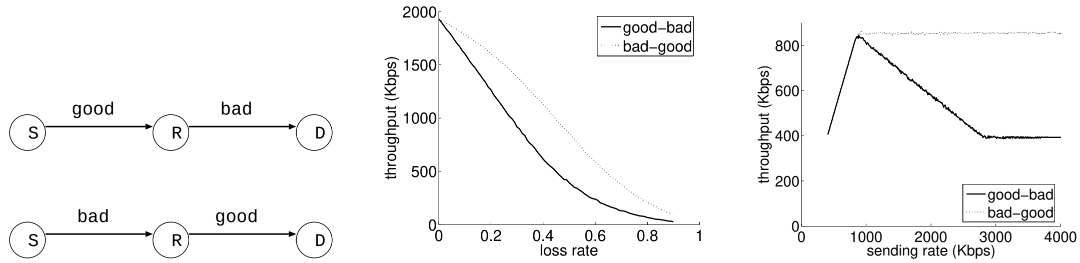
3.1 Lack of Rate Feedback

Current routing protocols provide no feedback as to how much traffic a node can send. In this section, we show that lack of rate feedback can lead to severe performance degradation.

We illustrate our point using the two simple topologies in Figure 1(a). Both have one reliable ("good") link and one lossy ("bad") link but the order of the two links is different. Using QualNet [13], we simulated the case of *S* sending 512-byte UDP packets to *D* as fast as possible. Unless otherwise specified, our evaluation uses 802.11a and 6Mbps MAC data rate throughout the paper.

Figure 1(b) shows that the throughput of the two topologies as a function of loss rate on the bad link are very different. At a loss rate of 0.5, the throughput of the good-bad topology is less than half of the bad-good topology.

The reason for this disparity is the following. For a successful reception in the good-bad topology, *S* needs to transmit a packet to *R* only once, but *R* has to transmit to *D* more than once. Since the 802.11 MAC allocates air time fairly among *S* and *R* under saturated demands, the incoming traffic at *R* is more than the outgoing traffic, and many packets sent by *S* are eventually dropped at *R*



(a) (b) (c)

Figure 1: The importance of rate feedback. (a) **Two topologies that differ in where the lossy link occurs.** (b) **Throughput as a function of loss rate when** S **sends as fast as possible.** (c) **Throughput as a function of the sending rate when the loss rate of the bad link is 0.50.**

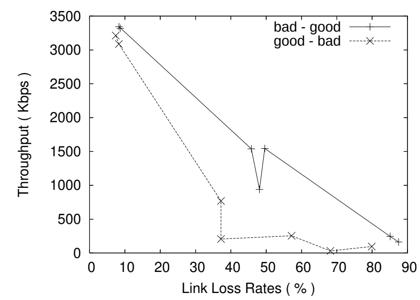
due to queue overflow. These wasted transmissions of S compete with R for air time and reduce the throughput of the good-bad topology. Such wastage does not exist in the bad-good topology because R can send all incoming traffic. To our knowledge, this sensitivity of wireless network throughput to bottleneck link location has not been reported previously.

Note that this problem cannot be solved by RTS/CTS because both transmitters can hear each other and there is no hidden terminal. Moreover, simply changing the MAC allocation policy will not fix the problem in the general case because the bottleneck can be multiple hops away from the source.

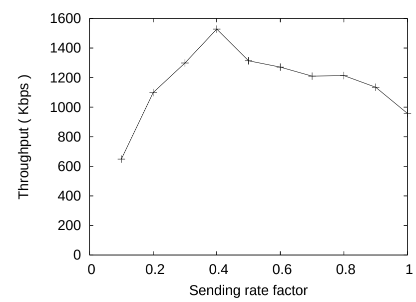
The wastage in the good-bad topology can lead to a very sudden decline in throughput as the sending rate is increased. Figure 1(c) plots the throughput of the two topologies as S increases its sending rate. The loss rate is configured to 0.5. In the good-bad topology, increasing the sending rate beyond a threshold sharply degrades throughput. This threshold represents the sending rate of S at which R receives enough air time to relay all received packets. Beyond it, R cannot keep up as it receives less air time and the medium is increasingly occupied by the transmissions from S that are eventually dropped. The throughput stabilizes when the air time utilization of R decreases to half.

The graph also shows that the two topologies have the same maximum capacity, but in the good-bad case, it can be achieved only if we limit S to the threshold sending rate. However, none of the current routing protocols give rate feedback. Moreover they cannot even distinguish between these two paths. The path quality as measured by current protocols will be the same for both topologies.

This sharp decline in throughput is reminiscent of congestion collapse in the Internet. But it is unique in that it can be caused by a single flow over a very simple topology. Known examples of congestion collapse in wired networks [3] involve more flows and complex topologies. A key difference is that the capacity of the bottleneck link in a wired network is not impacted by other links, but in wireless networks interference reduces bottleneck capacity when other links are active.



(a) Throughput vs. loss rates in the two topologies



(b) Throughput vs. sending rate in the good-bad topology

Figure 2: Testbed experiments confirm the importance of rate feedback.

Figure 2 confirms that the effect above is present in the more realistic testbed setting as well. We emulate different loss rates in the testbed by changing the distance between the machines and varying layers of foils around the wireless cards. Figure 2(b) shows that the two topologies perform differently when S sends as fast as possible. Figure 2(b) shows the sudden throughput decline in the good-bad topology when the bad link has roughly 50% loss. The x-axis in this graph denotes the fraction of the fastest possible sending rate (*e.g.*, sending rate factor $= 1$ indicates that the source sends packets back-to-back). The curve is not as smooth because the loss rate in the testbed cannot be precisely controlled. Overall, these results confirm the ill-effects of not providing rate feedback.

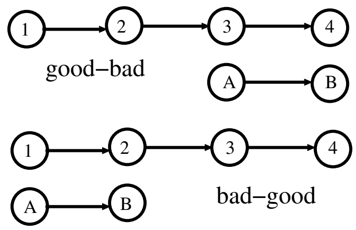


Figure 3: The topologies for the TCP case. The TCP flow goes from 1 to 4. The UDP flow goes from A to B.

TCP traffic We now show that similar problems occur with TCP as well because TCP's built-in rate control and congestion response is not sufficient to eliminate these problems. First, consider a star topology with 5 nodes, $A \cdots E$. Node C is in the middle and all links are reliable. There are two competing TCP flows *A-C-D* and *B-C-E*. We find performance degradation due to overload when the central node cannot relay all the traffic sent by its neighbors. As the maximum allowed rate of the TCP flow (controlled using receiver window) increases, the TCP throughput decreases. TCP is unable to appropriately set its rate to where it can maximize throughput, because (i) TCP's rate control is coarse grained – in the unit of packets per RTT and (ii) TCP's aggressive bandwidth probing makes the flows stabilize at a loss rate higher than the loss rate under maximum throughput [4].

Next consider the topologies in Figure 3 in which we introduce a TCP flow from Node 1 to 4. There is also a UDP flow with a sending rate of 1.3 Mbps from Node A to B. This "background" traffic creates a bottleneck at the link close to it, 3-4 or 1-2. We enable RTS/CTS so that there are no hidden terminals in either topology. These topologies are similar in essence to those in Figure 1 except that instead of loss the bottleneck is created by background traffic. Simulations reveal effects similar to those with UDP. TCP throughput is sensitive to bottleneck location, and the throughput of the TCP flow in the bad-good case is higher, by a factor of 5.6 (0.584 Mbps versus 0.104 Mbps). The background UDP flow obtains similar throughput in both cases.

3.2 Inaccurate Estimation of Path Quality

An important requirement for a routing protocol is to facilitate the selection of good routing paths. In this section, we show how current wireless routing protocols are ineffective at this function because their measures of link quality may not reflect actual quality. We first show problems with basic link quality measurements and then with path quality measurements. We use ETX [1] as the representative of current protocols. It characterizes link quality as the average number of transmissions required to get a packet across. Other protocols [2, 22, 15] also suffer from the pathologies described below, because they are based on ETX.

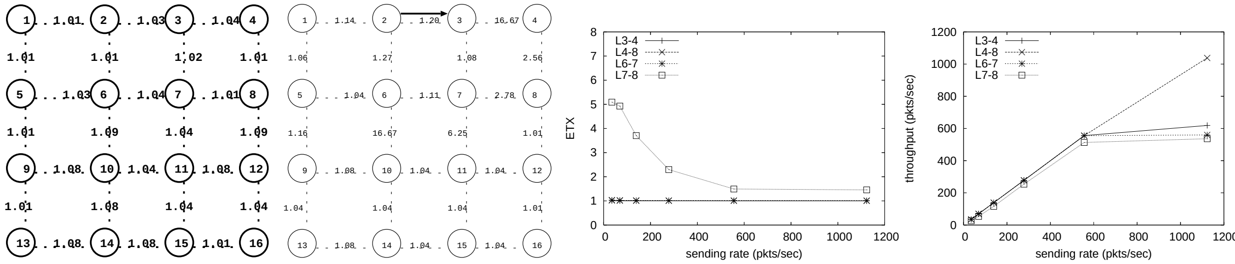
Link quality Measured ETX values may not reflect actual link quality that data traffic will experience when using that link. We demonstrate the problem with a concrete example. Figure 4(a) shows that the ETX values of all links in a 4x4 grid in absence of any traffic. Every link has good quality, with an ETX value close to 1. Figure 4(b) shows the snapshot of ETX values after a UDP flow is introduced from node 2 to 3. ETX values of links close to Link 2-3 increase because the probes on these links collide with data traffic on Link 2-3.

Do these ETX values reflect link quality that data traffic would experience? To answer this question, we focus on Links 3-4, 4-8, 6-7, and 7-8. We retain the flow on Link 2-3 and inject traffic with a varying rate on one of the other links. Figure 4(d) shows the throughput of the data traffic. Comparing with Figure 4(b), we observe that the measured ETX values are poor indicator of the actual performance experienced by data traffic. For example, the ETX value of Link 3-4 is 16.67, which is much higher than that of Link 6-7; however the throughput of the two links are similar across all sending rates. Similarly, even though Links 7-8 and 4-8 have higher ETX values than Link 6-7, they have similar or higher throughput than Link 6-7.

The discrepancy between measured ETX and actual traffic performance arises from two factors. First, the ETX metric is determined by packet loss rates at receivers, so it only captures receiver-side interference but fails to capture sender-side interference that stops nodes from transmitting. Link 6-7 has low ETX, because nodes 2 and 6 are close enough to avoid collision losses even though their transmissions interfere. To find high-throughput paths, the routing protocol must capture both receiver-side and sender-side interference.

Second, the characteristics of probing traffic and data traffic can be quite different in terms of, for instance, volume, packet sizes and generation pattern, which makes the two observe different loss rates. For example, Figure 4(c) shows that as Link 7-8 carries more traffic, its loss rate decreases due to decreasing competing background traffic. The loss rate of data traffic can be higher or lower than that of probe traffic depending on volume. Different packet sizes and generation pattern of DATA/ACK and probe packets also contribute to the discrepancy. For example, Link 3-4 has ETX value of 16.67 in Figure 4(b) because the probe traffic on Link 4-3 collides heavily with data traffic on Link 2-3 due to the hidden terminal problem. However, as shown in Figure 4(c), the data traffic on Link 3-4 has low ETX, because ACKs on Link 4-3 seldom collide with data traffic on Link 2-3. This collision rate is low because the ACKs have a smaller packet size and are generated immediately after the DATA packets on Link 3-4, during which time Node 2 often defers to them based on the NAV reservation in the DATA packets.

Path quality Even when the link quality itself is accurate, the relative path quality computed by ETX may not



(a) (b) (c) (d)

Figure 4: (a) Measured ETX values under no traffic. (b) Measured ETX values under one UDP flow from node 2 to 3. (c) Measured ETX of data traffic under two UDP flows (one on Link 2-3 and one on the link in the legends. (d) Throughput of a link when sending data traffic over it while keeping the same UDP flow from node 2 to 3 as competing traffic. Only directly connected nodes in the grid can carrier sense each other.

correctly rank paths according to their ability to carry traffic. In particular, ETX uses the sum of quality of all links along a path as the path quality, however the relationship between path and link quality does not follow summation due to self interference and interference with other ongoing traffic. Consider again the topology in Figure 4(a). Assume there are two flows, one from Node 5 to 8 and the other from 9 to 12. In our simulations, ETX picks the shortest paths 5-6-7-8 and 9-10-11-12, leading to a total throughput of 1.3 Mbps. But a better option is to use a slightly longer path for the top flow, 5-1-2-3-4-8, while retaining the same shortest path for the other flow. The total throughput then is 1.9 Mbps, which represents an improvement of 46%.

4. MODEL-BASED RATE COMPUTATION

The previous section demonstrated two problems with routing in mesh networks. Both stem from interference effects that are not accounted for by current protocols. We thus advocate an approach that systematically accounts for interference. As an important building block, we present an algorithm to estimate rates at which sources can safely send traffic. The challenge in developing this algorithm stems from the fact that interference induces complex interdependencies between safe sending rates of various nodes.

Our algorithm takes as input the network topology, the routing paths, and desired demand of each flow (*i.e.*, a unidirectional stream of traffic between two BSs), and outputs the maximum sending rate for each flow. This assumes that BSs can estimate future demand. Using recent history is one way to do so. Because BSs aggregate traffic from multiple users, their demand is likely to be relatively stable. We find this to be true in our own measurements of hotspot workloads.

We use the notion of a "conflict graph" [8] to systematically account for interference. In a conflict graph, vertices correspond to physical links and there is an edge between two vertices if the corresponding links interfere. Thus, given a clique in the conflict graph, only one of the links can be safely active in that clique.

```
ComputeRates(routes[], newRate[])
  flowSet = flows
  cliqueSet = set of cliques in the network
  while flowSet is not empty do
    find minimum α s.t. α × newRate[] saturates at least
    one clique in cliqueList
    saturatedFlows = flows traversing saturated cliques
    for all f in saturatedFlows do
      rate[f] = α × newRate[f]
    remove saturatedFlows from flowSet
    remove saturated cliques from cliqueList
    for all f in saturatedFlows do
      for all c in cliques traversed by f do
        update c's capacity by subtracting resources as-
        signed to f
```

Figure 5: Pseudo code for rate computation.

We first approximate the conflict graph of the wireless network. For this, we work with a simplified view of the network topology in which we consider a physical "link" to exist between two nodes if their bidirectional delivery rate is above a threshold (10% in our experiments). To capture interference experienced by a node, we generate a fixed number of cliques containing links that interfere with its incident links. For each node i, we first generate a candidate link set L_i which contains all links incident on the node and its neighbors. To generate a clique, we consider a random permutation of L_i. Starting from an empty clique, in order of the links in the permutation of L_i, we add the next link to the clique if it interferes with every link currently in the clique. A different permutation of L_i is used to generate another clique. Two links are considered as interfering if one of the following is true: i) they share an end point; ii) the two senders are neighbors; iii) the two senders are neighbors of the same receiver. If RTS/CTS is enabled, two links interfere also when the two receivers are neighbors.

We then compute the max-min fair sending rate for each flow based on its demand. These are determined such that no clique along the flow's path is utilized beyond its capacity. Figure 5 shows the pseudo code of this computation. We repeatedly use binary search to find the minimum multiplicative factor α such that routing α times the anticipated demand of each flow will saturate part of the network. Clique constraints help determine if a given traffic load results in network saturation under interference. Saturation occurs when at least one clique in the conflict graph reaches 100% utilization, i.e., the sum of the fraction of time that its links are active is 1. At this point, all flows that traverse the saturated clique are assigned a rate limit based on α. For the remaining flows, we remove the saturated flows and cliques, update the residual capacity by subtracting resources consumed by these flows, and find the new multiplicative factor α. This process iterates until the rates of all flows have been assigned.

Our algorithm is amenable to a fully distributed implementation. All nodes share with each other the link delivery rates and flow demands. Then all nodes separately run the algorithm to compute their sending rates. Given the same inputs, they arrive at consistent answers. This is similar to link-state protocols such as OSPF. When a node is a source of a flow, it then directly limits its flow's rate according to the derived rate limit.

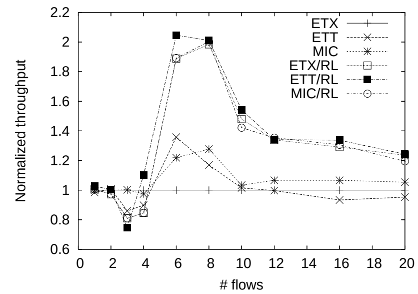
By necessity, our approach makes a few simplifying assumptions. Our view of the conflict graph is approximate. Given the computational difficulty of finding all cliques, we only consider a fixed number of cliques for each node. We treat interference between two links to be a binary property, and our determination of a clique's saturation level assumes perfect scheduling in which interfering nodes do not transmit simultaneously. In the next section, we show that in spite of these simplifications our approach significantly improves network throughput.

5. EVALUATION

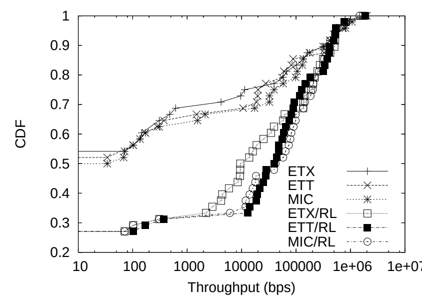
We evaluate our approach by comparing throughput under our rate limiting algorithm with the cases where flows are not rate limited. We conduct both simulation and testbed experiments.

To show that the benefit of our algorithm persists across a range of routing path selection methods, we use three different methods: *i*) ETX selects paths that minimize the total number of transmissions from source to destination [1]; *ii*) ETT selects paths that minimize the total transmission time [2]; and *iii*) MIC selects paths with minimum sum of the product of link ETX and the number of neighbors of the two end points [22].

Figure 6 shows the throughput with and without rate limiting for the three methods. These results are obtained using 25-node random topologies and a varying number of UDP flows with infinite demands between randomly chosen node pairs (*i.e.*, the sources always have data to



(a) Normalized throughput



(b) Normalized throughput distribution under 16 flows

Figure 6: Rate limiting improves normalized total throughput and fairness.

send). We obtained qualitatively similar results for several other types of topologies and workloads, but we omit those from this paper.

Figure 6(a) plots throughput normalized by that of ETX. It shows that under medium to high workloads our algorithm significantly boosts network throughput for each routing method, by as much as 100% in some cases. The throughput deteriorates slightly under lower load, possibly because the rate limits are aggressive in that regime. We are working towards improving performance in this regime as well.

The benefit of rate-limiting under high load appears relatively less than that under medium load because we aim for fairness when assigning flow rates. This reduces starvation, which has been observed in mesh networks [16], but at the cost of total network throughput. To show this, Figure 6(b) plots the distribution of throughput obtained by various flows under high load (16 flows). We see that in the absence of rate-limiting, roughly half of the flows are completely starved. With rate limiting, fewer flows are starved and the overall distribution is more fair. We consider this trade-off between total throughput and fairness to be a good one. However, if so desired, our approach can be modified to ignore fairness and maximize total throughput instead.

We now study the benefit of our algorithm in the more realistic testbed setting. Our testbed consists of 21 nodes spread across a floor of an office building. Each node is equipped with a NetGear WAG511 NIC and runs Win-

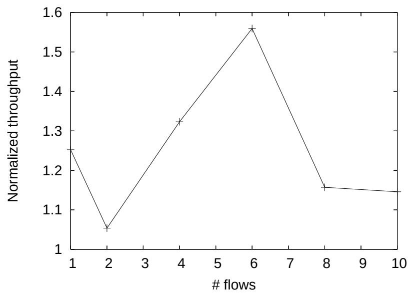


Figure 7: Normalized throughput with our rate limiting method. ETX is used to select routing paths.

dows XP. We use Mesh Connectivity Layer (MCL) [11] from MSR to implement our algorithm. MCL already implements ETX.

Figure 7 shows the relative (to ETX) throughput of rate limiting for different numbers of flows between randomly chosen node pairs. It confirms the benefits that we demonstrated in the simulation setting. The throughput with rate limiting is higher than without it, especially under medium load levels. At high loads we trade off total throughput for increased fairness among flows.

6. SUMMARY

We described two fundamental shortcomings of the state-of-the-art in routing methodology for wireless mesh networks. First, sources are not given any feedback on how much traffic they can safely send. As a result, severe performance degradation may occur when they send more than what the network can safely carry. The second shortcoming is the inability of the protocols to reliably identify high throughput paths in the face of wireless interference.

These observations led us to begin developing a novel approach that systematically accounts for interference. Our approach is centered around a formal model of interference that is used to systematically account for and limit interference. Given flow demands and routing paths, it lets us estimate the amount of traffic that sources can safely send. Preliminary evaluation suggests that our approach can significantly improve the throughput and fairness of wireless mesh networks.

Acknowledgements

This research is sponsored in part by National Science Foundation grants CNS-0434515, CNS-0546755, CNS-0627020, CNS-0546720, and CNS-0615104.

7. REFERENCES

[1] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *ACM MOBICOM*, Sept. 2003.

[2] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of ACM MOBICOM*, Sept. - Oct. 2004.

[3] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Transactions on Networking*, 7(4), Aug. 1999.

[4] Z. Fu, H. Luo, P. Zerfos, S. Lu, L. Zhang, and M. Gerla. The impact of of multihop wireless channel on tcp performance. In *IEEE Trans. on Mobile Computing*, Mar. 2005.

[5] M. Gastpar and M. Vetterli. On the capacity of wireless networks: the relay case. In *Proc. of IEEE INFOCOM*, Jun. 2002.

[6] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. In *Proc. of IEEE INFOCOM*, Apr. 2001.

[7] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), Mar. 2000.

[8] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *MOBICOM*, Sept. 2003.

[9] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multihop wireless ad hoc networks. In *Ad Hoc Networking*, 2001.

[10] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *In Proc. of ACM MOBICOM*, Jul. 2001.

[11] Mesh connectivity layer. `http://research.microsoft.com/mesh/\#software`.

[12] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, Feb. 1999.

[13] The Qualnet simulator from Scalable Networks Inc. `http://www.scalable-networks.com/`.

[14] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis. Interference-aware fair rate control in wireless sensor networks. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[15] A. P. Subramanian, M. M. Buddhikot, and S. Miller. Interference aware routing in multi-radio wireless mesh networks. In *Proc. of IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Sept. 2006.

[16] X. Wang and K. Kar. Throughput modeling and fairness issues in CSMA/CA based ad-hoc networks. In *Proc. of IEEE INFOCOM*, Mar. 2005.

[17] Questions for Tropos: Does Google's mountain view network fold under pressure? `http://www.muniwireless.com/article/articleview/5395`.

[18] Wi-Fi city sees startup woes. `http://www.wired.com/techbiz/media/news/2006/04/70720`.

[19] Another muni WiFi network gets an early thumbs down. `http://www.techdirt.com/blog/wireless/articles/20061226/145441.shtml`.

[20] City-wide Wi-Fi rolls out in UK. `http://news.bbc.co.uk/2/hi/technology/4578114.stm`.

[21] Cities unleash free Wi-Fi. `http://www.wired.com/gadgets/wireless/news/2005/10/68999`.

[22] Y. Yang, J. Wang, and R. Kravets. Designing routing metrics for mesh networks. In *IEEE Wireless Mesh Networks (WiMesh)*, Sept. 2005.