Interaction between TCP and UDP flows in Wireless Multi-hop Ad hoc Networks

Erik Nordström and Christian Rohner
Uppsala University
chrohner|erikn@it.uu.se

Abstract

The interaction between TCP and UDP flows are examined experimentally in simple scenarios, involving between three and four nodes. Early results show that TCP's congestion control is not tuned to wireless multi-hop settings, resulting in significant jitter and loss in UDP flows.

1 Introduction

This paper examines the interaction between TCP and UDP flows in wireless multi-hop ad hoc networks. Through a set of simple experiments in a real world setting the effect of constant bit rate (CBR) UDP traffic on adaptive TCP and vice versa, is investigated. Previous work in the area mainly focus on TCP fairness between competing TCP flows. In case of un-fairness, it is caused by the fact that ad hoc nodes may, for a data flow, share the wireless transmission medium (e.g., IEEE 802.11 MAC), but the same nodes may not necessarily all be part of the data path. Therefore, contention is not just mainly for buffer space on routers, but also for the wireless transmission medium. However, there has been little work examining the behavior of TCP in the presence of UDP traffic that lacks congestion control, such as streaming music or voice.

We conduct experimental performance measurements using TCP and UDP traffic in simple scenarios ranging from three to four nodes. Previous work in this area use simulation as the preferred tool for studying protocol logic and large networks. Although valuable, it will not capture all the complexity that a truly wireless and mobile experimental test environment does.

Early results indicate that contention between flows are in both the wireless channel and for buffer space on routers, as expected. An observation is that not only TCP may suffer loss, but also that the UDP flows may suffer significant loss, giving raise to questions about TCP's ability to adapt to the available bandwidth in a fair way.

1.1 Problem Description

A link in a wireless network is an abstract notion defined by the connectivity between two nodes. Data sent by a node might be overheard by many other nodes or even be perceived as only interference. The presence of a link in a wireless network might be intermittent (e.g., due to mobility), links have varying quality (and hence bandwidth) and unlike wired networks, where congestion is in general caused by overfull queues, there may also be contention for the spatial reuse of the ether, causing congestion in the network.

The TCP feedback loop is responsible for adapting the sender data rate in response to, e.g., congestion. However, this end-to-end congestion control mechanism has reduced efficiency in wireless networks because transmission is inherently broadcast. Furthermore, there are different ranges for unicast radio transmission, broadcast radio transmission and interference. Even if the offered data rate is adapted to the bottleneck in a network path, transmissions might still contend and interfere with other. TCP has been designed to be fair between competing flows, i.e., TCP backs off in a way that an equilibrium is reached where all competing flows get an equal chunk of the available bandwidth at a bottleneck. CBR UDP flows are not rate adaptable and lacks congestion control. However, moderate rate flows, e.g., MP3 or voice streams could be considered marginal because they use relatively little bandwidth. In the presence of UDP flows, TCP should probe the data path by increasing its congestion window until loss occurs. In that case it should back off and not use more bandwidth than what is left after the UDP flow. The work in this paper examines the efficiency of the TCP rate adaptation in the presence of UDP flows.

The paper is structured as follows. In section 2

we discuss related work. We then turn to describing our experimental setup and the experiments conducted in section 3. Section 4 ends the paper, with a discussion and conclusions.

2 Related Work

Gupta et al. report in [2] on decreased TCP performance in the presence of interacting UDP flows. Their study is done in the ns-2 simulator on an artificial grid topology and with relatively high UDP data rate (800 Kb/s). Since UDP lacks rate adaptation and congestion control, a high data rate CBR flow will naturally congest the channel. Our work complement that work by performing real world measurements with UDP and TCP data flows in scenarios with increasing complexity. Moderate data rate UDP is used (typically streaming music) that we believe is more realistic.

Routers in the Internet can improve the fairness in competition among the router's resources, for example, by using Random Early Detection (RED). Xu, Gerla, Qi and Shu study in [4], the TCP fairness among competing flows in wireless ad hoc networks. They look at how queuing policies can help increase TCP fairness and show that the RED scheme does not work, because flows do not only share individual queues on forwarding nodes, but also the wireless channel. They propose a Neighborhood RED (NRED) scheme that treats all individual queues of nodes within a neighborhood as a shared distributed queue. Thus, a forwarding node can drop packets to induce decreased send rate of one TCP flow for the benefit of another flow at different location in the same neighborhood.

3 Experiments

Through real world experiments we examine the contention between TCP and UDP data flows in three different scenarios with increasing complexity. The scenarios involve three to four nodes and start with a simple two-hop UDP flow in a stationary setting. A competing TCP flow is then added, first over one hop, later with mobility over up to three hops, sharing links with the UDP flow. Each scenario is run five times and average performance measures are computed.

The experiments are conducted indoors in a systematic way using the APE testbed [3]. APE provides a test environment that allows repeatability of experiments and provides support for extensive logging and analysis of experiment data. The APE testbed orchestrates the scenarios and provides verbose logging on the network interface and network layer. Standard laptops (IBM X31) with Lucent/ORiNOCO IEEE 802.11b network interfaces are used. Logged and time stamped data is time synchronized between nodes using periodic broadcasts of time information at a rate that has a negligible impact on the experiments. This time information is used to synchronize the experiment data collected from the different nodes.

The AODV-UU [1] implementation is used where dynamic routing is needed. AODV-UU was chosen because it is mature and has been interoperability tested.

3.1 Static Multi-hop UDP

The first experiment intends to measure the data rate limit and explore the interactions of links in a multi-hop path. For this purpose a scenario consisting of three nodes 0, 1, and 2 placed in line was constructed. Each node only has connectivity with its adjacent nodes. End node 2 sends a constant bit rate UDP flow to node 0, over the intermediate node 1. This setup constitutes our base line scenario and provides a calibration for the other tests. The setup is illustrated in Figure 1.

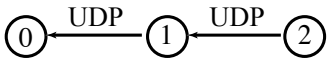

Figure 1: Static two-hop scenario with a constant bit rate UDP flow.

By using constant bit rate UDP, the influence of an adaptive transport protocol (e.g., TCP) can be avoided. For each test run, the offered data rate (rate of data from the application) is increased and we study the effect on the two links comprising the multi-hop path. Since the bandwidth at node 1's network interface is shared between the two links we expect the overall UDP throughput to decrease when the offered data rate at node 2 exceeds one half of the bandwidth at node 1.

Figure 2 shows the received data rate as a function of the transmitted data rate (rate of the data actually sent on the channel).

It can be observed that the transmitted data rate for this two hop path at around 3.5 Mbit/s achieves the best received data rate. Increasing the offered data rate beyond that has a negative impact on the overall throughput. Note that although we in-

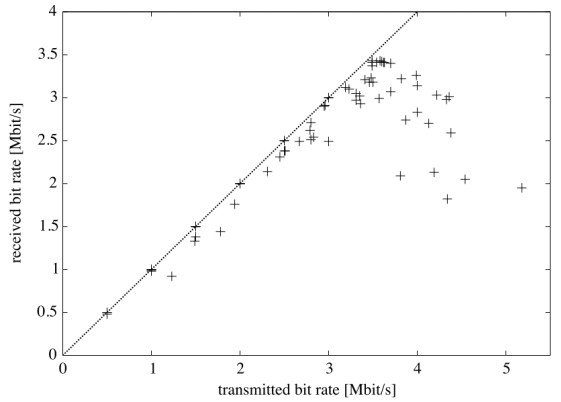

Figure 2: Received bit rate as a function of the transmitted bit rate in the two-hop scenario for transmitted rates between 0.5 Mbit/s and 5 Mbit/s (with 0.5 Mbit/s intervals in between). The line shows the optimal throughput when there is no packet loss.

creased the offered data rate up to 10 Mbit/s, the transmitted rate never exceeded 5.2 Mbit/s.

After having determined the maximum data rate over two hops, the UDP data rate was fixed at 192 kbit/s to resemble the rate an MP3 data flow[1].

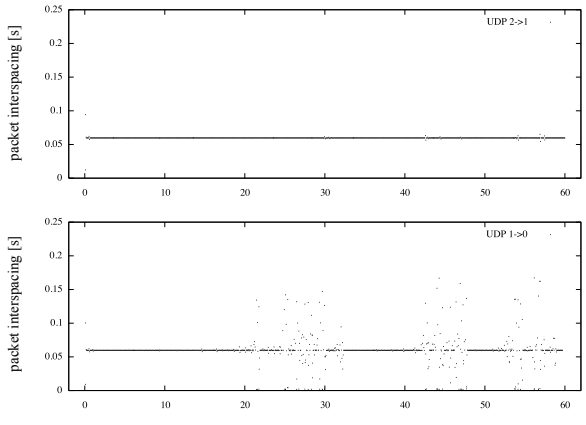

Figure 3: Example of UDP inter packet delivery time in the static multi-hop scenario.

Using this moderate data rate, the average UDP delivery ratio over five test runs was 99.2 %. Although close to 100 %, it could be verified that the majority of the packet losses occurred at the second hop link between node 1 and 0. Figure 3 shows the inter packet delivery time for the UDP flow between node 2 and 0 for a representative run. It can also be seen that the jitter is larger over this link, which is probably a combination of varying queuing delay at node 1 and retransmissions at the link layer. However, this jitter was not apparent in all test runs, but when it occurred, it was on the link between node 1 and 0.

[1]The actual data size for UDP packets (1470 bytes) does not match the frame size used in a real MP3 flow

3.2 Multi-hop UDP with TCP flow

In this scenario we increase the complexity by adding an extra node 3, which is mobile and sends a TCP flow to node 0. Node 3 starts at the far left position in Figure 4, outside the transmission range of node 1. Ten seconds into the scenario, node 3 starts moving toward node 0 and then when reaching that position (at time 38) it ends its movement.

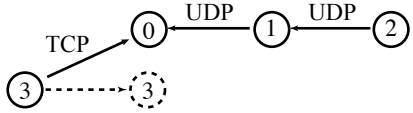

Figure 4: The TCP vs UDP scenario.

The purpose of this scenario is to see how TCP adapts its send rate (if at all) when moving to the position close to node 0, where it is potentially more affected by channel contention from both node 1 and node 2. We also want to see how the UDP flow from node 2 to node 0 is affected by the increased contention.

Over five test runs, the average UDP delivery ratio was 97.4 %, a slight decrease from the previous scenario. The average TCP throughput was 1.34 Mbit/s. At first this number seemed unexpectedly low. However, considering the overhead of TCP, feedback loop (packets in both directions and extra header size) and the contention with the UDP flow from both links (between nodes 2 and 1 and 1 and 0, respectively), the number 1.34 Mbit/s might not seem so unreasonable. Understanding the exact reasons for this particular throughput requires further investigation.

In Figure 5 we see the UDP inter packet delivery time on the link between node 2 and 1 and 1 and 0 respectively, along with the TCP time sequence graph for the TCP flow between node 3 and 0, for one of the experiments.

Although variations are apparent throughout all experiments, Figure 5 is representative for a typical test run. It can be seen that most of the packet loss on the UDP flow again occurs on the link between node 1 and 0. After node 3 has moved and is in direct contact with node 1, the jitter between node 2 and 1 is also more apparent, indicating that the

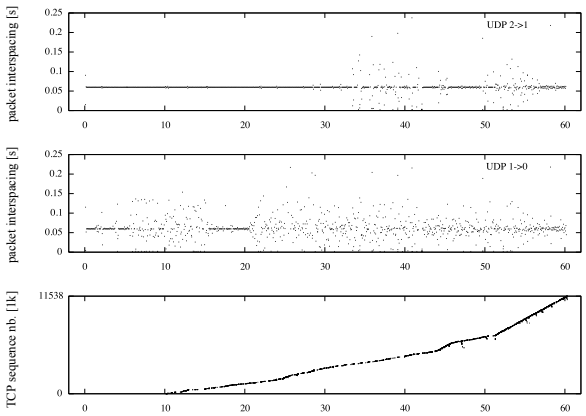

Figure 5: Example of UDP inter packet delivery time and TCP sequence graph.

TCP flow is interfering with the UDP flow at that link.

An interesting observation can also be made at time 50, where there is an increase in the TCP throughput. This increase comes at a cost, inducing a higher amount of jitter at the link between node 2 and 1. This behavior is persistent in most of the experiments. It is not clear why TCP increases its throughput in this situation. One possible explanation could be the capture effect [5].

3.3 Roaming Node with Competing UDP and TCP flows

This scenario, called "Roaming node", is again an extension of the two previous scenarios. Complexity is increased by adding more mobility, multi-hop TCP, and dynamic routing using AODV-UU. The same UDP flow as in the previous scenarios is sent from node 2 to node 0. Node 3 now starts out alongside node 0 at position A, as illustrated in Figure 6. Node 2 starts its UDP flow to node 1 simultaneously as node 3 starts a TCP file transfer to node 0 and starts moving towards position D. After 62 seconds it will reach position D and then turn back and move towards node 0 again. During this time, the TCP flow to node 0 is sent over a path that increases from one hop to two and three hops, and reduced back to one hop as node 3 is on the way back. Note that in this scenario, TCP and UDP have competing data flows going over the same intermediate nodes.

From separate analysis we know that there is one hop connectivity between node 3 and node 0 up to about time 25, followed by two hop connectivity up to time 50, and then three hop connectivity until time 92 where the route is directly reduced to a one

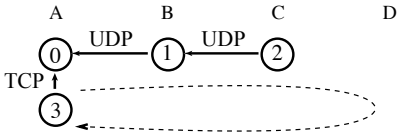

Figure 6: The Roaming Node scenario.

hop configuration until the end of the experiment.

The average UDP delivery ratio in this scenario decreased to 78.4 % and the average TCP throughput to 0.93 Mbit/s [2]. A summary of the different scenarios in UDP deliver ratio and TCP throughput is shown in Table 1. It can be seen that the general trend is that the increased complexity brings a decrease in the overall performance.

A decrease in the TCP throughput is expected as the data flow traverses multiple hops. Increasing the number of hops by a factor k, typically decreases the throughput of in the order of αk, where α is a constant close to 1. Factoring in the contention with the UDP flow, the observed TCP throughput appears reasonable.

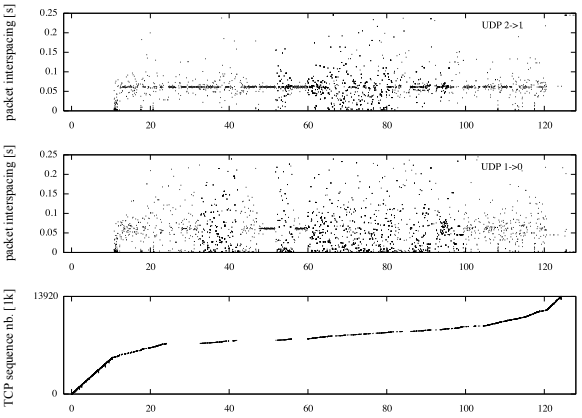

Figure 7: Example of UDP inter packet delivery time and TCP sequence graph in the Roaming node scenario.

In Figure 7 the UDP inter packet delivery time and TCP sequence number graph for the two different flows can be seen for one of the experiments. It can be observed that the TCP data flow stalls in particular on the change from a one hop to a two hop configuration (time 25..30), and on the change from a two hop to a three hop configuration (time 40..50). There are also stalls regularly during the three hop configuration (time 50..92). The

[2] In the Roaming node scenario we experienced an outlier in one of our five test runs. Those results were therefore discarded and the average calculated over the remaining four runs.

Scenario	UDP Delivery Ratio	TCP Throughput
Static multi-hop UDP	99.2 %	-
Multi-hop UDP with TCP flow	97.4 %	1.34 Mbit/s
Roaming node	78.4 %	0.93 Mbit/s

Table 1: Average UDP delivery ratio and TCP throughput for the different scenarios.

different configurations are visible in the (slightly) decreasing slope of the time sequence graph. The route switches on the way back are much smoother. This can be explained by AODV's HELLO messages working more proactively on the way back, discovering a more optimal (shorter) route before the old one is gone.

We further observe increased UDP jitter on the link between node 2 and 1 during three hop TCP connectivity (time 50..80). Increased UDP jitter on the link between node 1 and node 0 is observed whenever we have progressing TCP traffic, but the UDP flow immediately recovers during TCP stalls. The increased jitter is probably caused by the extra queuing delay on the intermediate nodes when the TCP flow also competes for buffer space.

An interesting observation can also be made in the beginning of the experiment. It seems that when the TCP and UDP flows start simultaneously and in combination with dynamic AODV routing, the multi-hop UDP path between node 2 and 0 is broken. We see two different explanations for this (or likely a combination). Either TCP captures the channel, causing significant loss on the first hop UDP link (consequently no traffic is seen on the second hop link either). A more probable explanation, though, is that TCP's aggressive start impacts AODV's HELLO neighbor sensing mechanism and broadcast route discovery. These probing packets are broadcasted on the link layer, hence without acknowledgments or retransmissions. If these packets are lost, no path will be established. Not until time 10, after some movement will TCP back off and allow the UDP flow between node 2 and node 0 to be established. The reason for this is probably that node 2 was previously a hidden terminal for node 3, causing massive collisions at node 1. Only after node 3 moves within contention range of node 2, will it back off.

4 Discussion and Conclusion

We have investigated the interaction between UDP and TCP flows through experimental tests in simple scenarios. From the results it can be concluded that due to wireless channel properties, data flows interfere with links that are not involved in the data path. This is in line with previous findings.

The results also indicate that both UDP and TCP influence and suffer from each other. Although we have moderate rate UDP flows, TCP's congestion control does not seem efficient enough to only have marginal impact on the other traffic in the network. When the two data flows do not share common links, we observe increased jitter in the UDP flow, but not significant packet loss. Instabilities in the form of short stalls are observed in TCP. Further analysis might show if TCP retransmissions are caused by lost packets or the high fluctuations in round trip time.

In the case where UDP and TCP share a common link, contention is significantly higher resulting in increased UDP packet loss and more significant TCP interruptions. This might in part also be explained by the instabilities in the multi-hop configurations.

Last, we conclude that dynamic routing, in particular when using broadcast neighbor sensing, adds another dimension of instability to ad hoc networking. Hidden terminals and channel capture effects cause otherwise stable routes to become unstable, simply because routing control messages are lost.

References

[1] The uppsala university ad hoc implementation portal. http://core.it.uu.se/adhoc.

[2] V. Gupta, S. V. Krishnamurthy, and M. Faloutsos. Improving the performance of tcp in the presence of interacting udp flows in ad hoc networks. Unpublished.

[3] E. Nordström, P. Gunningberg, and H. Lundgren. A testbed and methodology for experimental evaluation of wireless mobile ad hoc networks. In *Proceedings of First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities (Tridentcom)*, February 2005.

[4] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing TCP fairness in ad hoc wireless networks using neighborhood red. In *Proceedings of MobiCom'03, San Diego, USA*, September 2003.

[5] S. Xu and T. Saadawi. Does the ieee 802.11 mac protocol work well in multihop wireless ad hoc networks? *IEEE Communications Magazine*, 39(6), June 2001.