Chapter 3

Routers With Tiny Buffers: Simulations

To validate the results of Chapter 2, we perform simulations using the ns-2 network simulator [6]. We have enhanced ns-2 to include an accurate CIOQ router model, and will study how much buffering this router needs at ingress and egress ports to achieve high utilization.

In this chapter, our metric for buffer sizing will be *link utilization*. This metric is operator-centric; if a congested link can keep operating at 100% utilization, then it makes efficient use of the operator's congested resource. This is not necessarily ideal for an individual end-user since the metric doesn't guarantee a short flow completion time (i.e., a quick download). However, if the buffer size is reduced, then the round-trip time will also be reduced, which could lead to higher per-flow throughput for TCP flows. The effect of tiny buffers on user-centric performance metrics will be discussed in Chapter 4.

3.1 Simulation setup

Figure 3.1 shows the topology of the simulated network. TCP flows are generated at separate source nodes (TCP servers), go through individual access links, and are multiplexed onto faster backbone links before reaching the input ports of the switch.

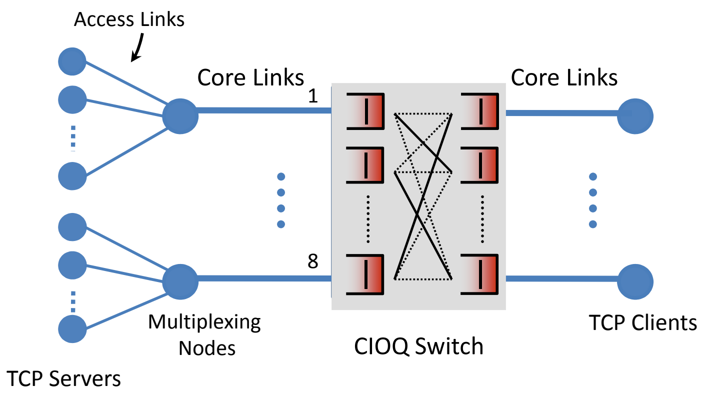


Figure 3.1: Simulated network topology.

Large buffers are used at the multiplexing routers to prevent drops at these nodes. All buffers in the network are drop-tail buffers.

The simulated switch is a CIOQ switch, which maintains virtual output queues (VOQ) at the input to eliminate head-of-line blocking. In each scheduling cycle, a scheduling algorithm configures the switch and matches input and output ports. Based on this configuration, either zero or one packet is removed from every input port, and is sent to the destination output port.

We define M to be the multiplexing factor, which is the ratio of the core link speed to the access link speed. Today, a typical user is connected to the network via a 10Mb/s DSL link, and backbone links often run at 40Gb/s; i.e., $M = 4,000$. In our simulations we conservatively pick M to be 100.

During an initialization phase, different flows start at different times and try to send an infinite amount of data. All data packets are 1000 bytes, and the control packets (SYN, SYN-ACK, FIN) are 40 bytes. The propagation delay between each server-client pair is uniformly picked from the interval 75-125ms (with an average of 100ms).

To measure link utilization, we first wait 30 seconds for the system to stabilize, and then take measurements for 60 seconds.

To study the effects of traffic characteristics and switch parameters in these simulations, we first choose a baseline setting with the following components:

1. The simulated switch is an 8×8 CIOQ switch with its linecards running at 2.5Gb/s.

2. The load is distributed uniformly among input and output ports of the router. In other words, all output ports are equally likely to be the destination port of a given flow.

3. The switch configuration is controlled by the Maximum Weight Matching (MWM) algorithm. MWM is known to deliver 100% utilization for admissible traffic distributions [34, 23], but the algorithm is too complex to be implemented in real routers.

4. We relax the over-provisioning assumption of Section 2.1.3, and offer 100% load to every output link of the router. In other words, we set the number of flows (N) sharing each output link and the TCP maximum window size ($W_{\max}$) such that the maximum aggregate rate of TCP sources sharing the link is equal to the link capacity:

$$\frac{N \times W_{max}}{RTT} = C. \tag{3.1}$$

   With an average RTT of 100ms and $W_{\max} = 64$KB, we need about 490 flows on each core link to fill the link.

5. The end hosts use TCP Reno with the Selective Acknowledgement (SACK) option disabled.

In Section 3.2, we present the results of simulating a router with the above baseline setting. Sections 3.3 and 3.4 examine how changing each of the components of the baseline setting affects link utilization.

3.2 Simulation results – baseline setting

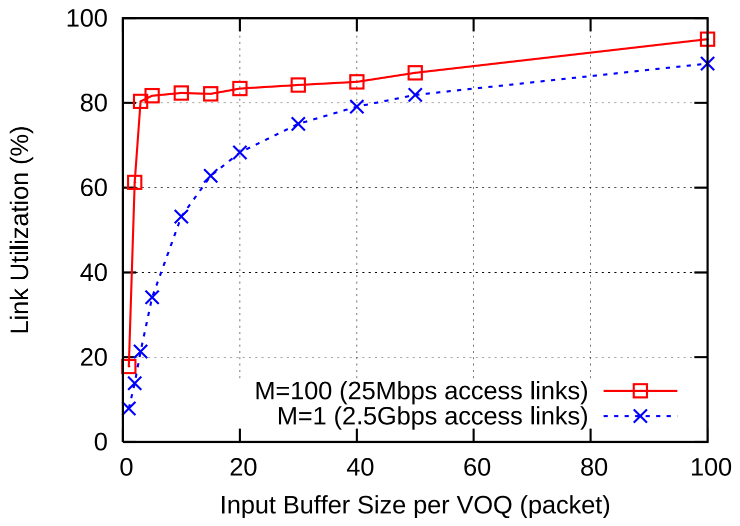
Figure 3.2 shows the average link utilization versus input and output buffer sizes in the baseline setting. To see the effect of the input and output buffer sizes independently, we first set the switch speedup to one, which makes the switch function as an input-queued switch. With speedup one, there is no queueing at egress ports, since the switch fabric runs no faster than the output links. Next, we set the switch speedup equal to the switch size (eight) to eliminate input queueing. With speedup eight, the switch functions as an output-queued switch and needs buffering only at the output side.

In both input-queued and output-queued scenarios, we run the simulations twice: first with $M = 1$, i.e., access links run at 2.5Gb/s, and then with $M = 100$, i.e., access links run at 25Mb/s.

Figure 3.2 shows the benefit of a larger M. Because the network naturally spaces out packets of each flow, much smaller buffer size is enough for high utilization. The plots show that when access links run at 25Mb/s (100 times slower than the core links), then buffering 3 packets in each VOQ and 15 packets at each output port suffices for 80% utilization. These numbers increase to 40 and more than 400 (not shown on this plot), respectively, when access links run as fast as core links.

With speedups between one and eight, we can combine the results shown in Figure 3.2: for each pair of input and output buffer sizes, utilization is not lower than the minimum utilization shown on these two graphs at the given input (top) and output (bottom) buffer sizes. This is because if speedup is greater than one, then packets are removed faster from the input queue, and the required buffer size goes down. If speedup is smaller than eight, then packets reach the output queue later, and hence the backlog is smaller.

Therefore, with any speedup, we can achieve more than 80% utilization with 3-packet VOQs and 15-packet output queues in the baseline setting. Remember that this result is with 100% load on the output links of the router. This suggests that the theoretical results of Section 2.1.3 are conservative in their over-provisioning assumptions.



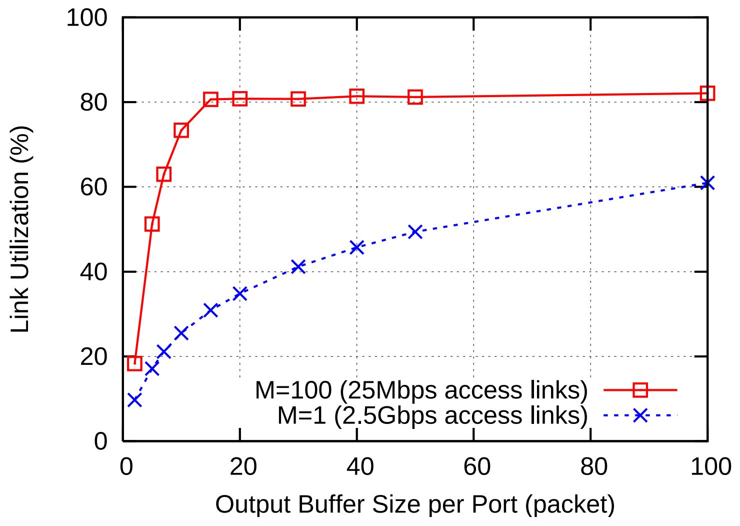


Figure 3.2: Link utilization vs. input and output buffer sizes. *Top*: speedup=1 and all the queueing takes place at the input; *bottom*: speedup=8 and all the queueing takes place at the output. With 25Mb/s access links, 3-packet VOQs, and 15-packet output buffers make the utilization above 80%.

3.3 When switch parameters change

In this section, we will see how the CIOQ switch parameters (scheduling algorithm, load distribution, switch size, and output port bandwidth) affect link utilization when tiny buffers are used. Network conditions and traffic characteristics are the same as in the baseline setting.

3.3.1 Switch scheduling algorithm and load distribution

In the baseline setting, we assumed that the switch was scheduled by the MWM algorithm, and that the load distribution was uniform. The MWM algorithm is very complex to implement and cannot be used in practice, but delivers full throughput for all admissible traffic.

Here, we relax these two assumptions and compare the results of the baseline setting to those obtained under the iSLIP scheduling algorithm [33] and non-uniform traffic.

The widely implemented iSLIP scheduling algorithm achieves 100% throughput for uniform traffic. This iterative round-robin-based algorithm is simple to implement in hardware, but the throughput is less than 100% in the presence of non-uniform bursty traffic.

Among various possible non-uniform distributions of load, we choose the diagonal load distribution. With a diagonal load, 2/3 of the total flows at a given input port i are destined for output port $i+1$, and the remaining 1/3 of the flows are destined for output $i+2$ (modular addition). The diagonal distribution is skewed in the sense that input i has packets only for outputs $i+1$ and $i+2$. This type of traffic is more difficult to schedule than uniformly distributed traffic, because arrivals favor the use of only two matchings out of all possible matchings. Under any scheduling algorithm, the diagonal load makes the router's buffers have the largest average backlog compared to any other distribution [45].

Figure 3.3 shows link utilization versus input buffer size per VOQ. With iSLIP and speedup 1 (top graph), there is no queueing at the output side of the switch. When the speedup is set to 1.2 (bottom graph), the switch fabric runs 1.2 times faster than

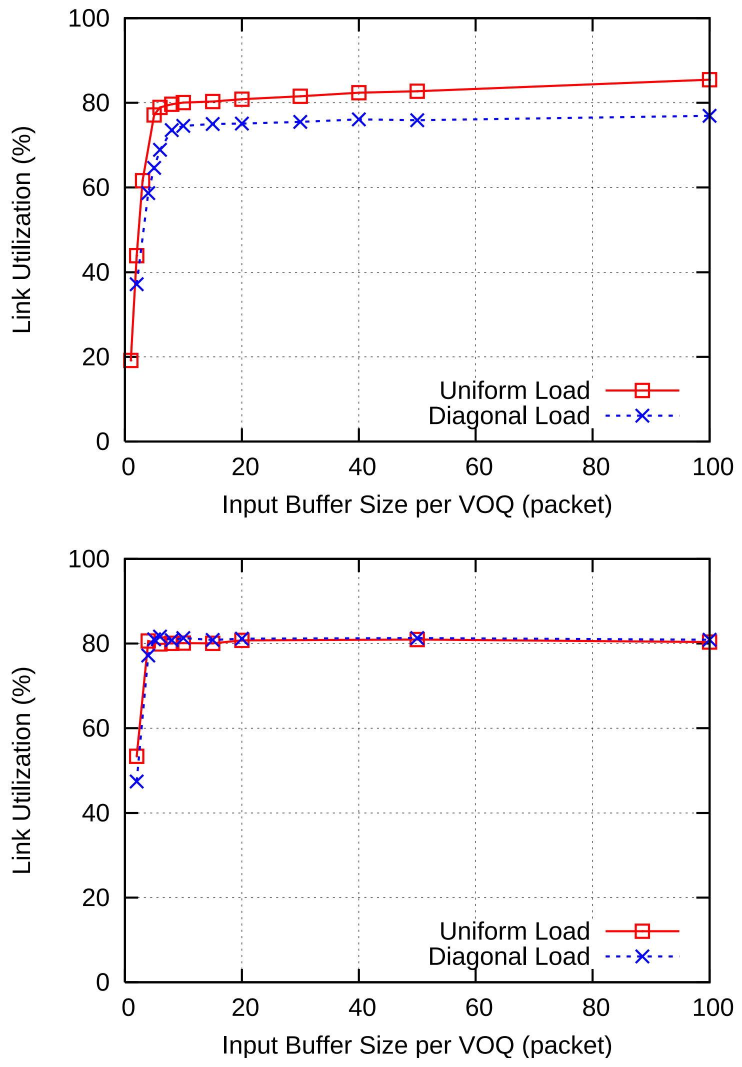


Figure 3.3: Link utilization vs. buffer size with iSLIP scheduling algorithm. *Top*: speedup=1; *bottom*: speedup=1.2.

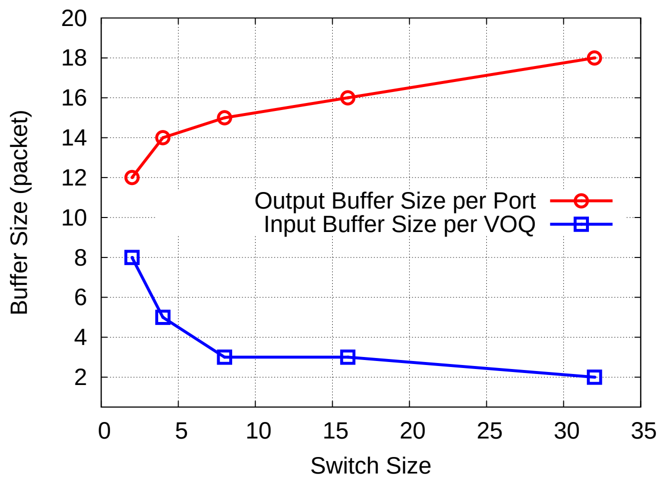


Figure 3.4: Minimum required buffer size for 80% utilization vs. switch size. The switch has a uniform load distribution, which results in more contention and short-term congestion when the number of ports increases.

the line rate, which may cause backlog in output buffers. In this case, we have set the output buffer size to only 20 packets per port. That is why we see some utilization loss when the speedup is increased from 1 to 1.2.

The results show that with speedup 1.2—for all combinations of scheduling algorithm and load distribution—setting the buffer size to 5 packets per VOQ and 20 packets per output port raises the utilization to more than 80%. Larger speedups make the impact of the scheduling algorithm even smaller because the switch behaves more and more like an output-queued switch.

3.3.2 Switch size

The output link utilization of a router depends on its size (number of ports). Increasing the number of ports creates more contention among the input ports, and adds to the short-term congestion (caused by the statistical arrival time of packets from different ingress ports) on the output links. Hence, with the same traffic, the utilization could be different for different switch sizes, depending on the distribution of the load.

Figure 3.4 shows the minimum required buffer size at input and output ports of a switch for 80% utilization on the output links. The simulation setting follows the baseline, except that the switch size is changed from 2 to 32. The number of flows on each link is kept constant for each switch size to maintain 100% offered load on the core links.

In this set of simulations, the switch has a uniform load distribution. If the traffic at a given output port comes from a fixed subset of the input ports, then we do not expect to see any changes when the switch size is varied. With diagonal load distribution, for example, where the traffic on each output link i comes only from ports $i-1$ and i, the required buffer size for 80% utilization remains constant as we change the number of ports.

We assume that the ingress ports maintain a separate VOQ for each output port. Therefore, despite the decrease in the VOQ size as the switch size grows, the total buffer size per ingress port (i.e., the size of the VOQ times the number of output ports) increases.

3.3.3 Output link bandwidth

The theoretical results of Section 2.1.3 shows that the required buffer size, for achieving high utilization, is independent of the absolute bandwidth of the bottleneck link. This is different from what the rule-of-thumb proposes. According to this rule, the buffer size needs to increase linearly as the bottleneck link bandwidth increases; hence, a 40Gb/s link needs 40 times as many buffers as a 1Gb/s link.

Figure 3.5 shows how link utilization stays unchanged when we increase the core link bandwidth, but keep $M = 100$ by increasing the access link bandwidth proportionally (the dotted curve). The buffer size is constant at 20 packets per port.

The solid curve in Figure 3.5 shows the bottleneck link utilization when the access bandwidth is fixed at 25Mb/s. Increasing the core bandwidth creates more spacing between packets, and reduces burst size; hence, the utilization improves.

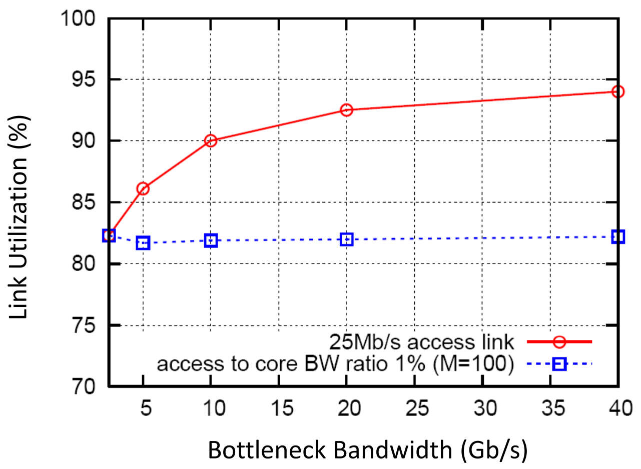


Figure 3.5: Link utilization vs. bottleneck link bandwidth. Utilization is determined by the core-to-access bandwidth ratio, not by the absolute core-link bandwidth. Buffer size at the bottleneck link is fixed at 20 packets.

3.4 When traffic characteristics change

This section considers two properties of the network traffic: the traffic load, and the TCP flavor implemented at the hosts. We will see how tweaking these parameters changes link utilization when tiny buffers are used in the simulated router.

3.4.1 Traffic load

The baseline setting assumes 100% offered load on the output links of the router. In other words, if all servers send traffic at their maximum rate ($W_{\max}/RTT$), then the aggregate traffic rate will be equal to the bottleneck bandwidth.

The offered load on the bottleneck link varies if either the number of flows sharing the link or the amount of the traffic that a single flow could generate changes. The effect on the link utilization in both cases is illustrated in Figure 3.6. The access bandwidth in both sets of simulations is fixed at 25Mb/s and the output buffer size is fixed at 20 packets (the simulated switch is an output-queued switch).

As the number of flows grows, the bandwidth share of each flow, the average

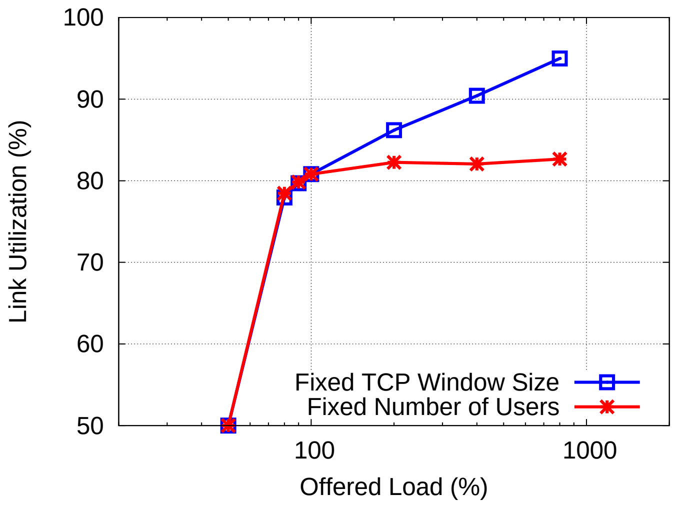


Figure 3.6: Link utilization vs. offered load. The offered load is changed by increasing the number of flows (with a fixed TCP window size) and increasing the maximum TCP window size (with a fixed number of flows).

window size, and the burst size a flow could possibly generate become smaller; hence, the link utilization improves.

Increasing the maximum window size (while keeping the number of flows constant) slightly improves utilization, from 81% to 83%. In this case, packet drops at the bottleneck link keep the average window size almost constant, regardless of how large the maximum allowed window size is.

3.4.2 High-speed TCPs

Figure 3.7 compares the link utilization achieved by two high-speed TCPs, TCP BIC [50](default in Linux kernels 2.6.8 through 2.6.18) and TCP CUBIC [42] (default in Linux Kernel since version 2.6.19), with that achieved by TCP Reno [1]. The plot

[1]The main difference between these high-speed TCPs and TCP Reno is in their window growth functions. When TCP BIC gets a packet loss event, it reduces its window by a multiplicative factor. Then it performs a binary search to find the right window size (rather than additively increasing the window size as in Reno). CUBIC is an enhanced version of BIC; it simplifies the BIC window

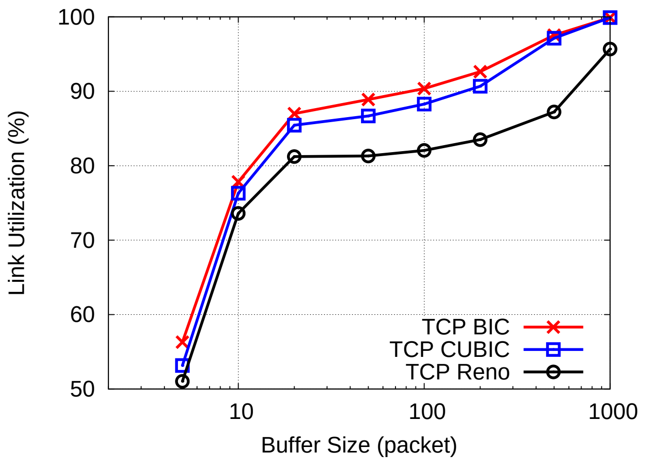


Figure 3.7: Comparison of utilization achieved by TCP Reno, BIC, and CUBIC.

shows that for all buffer sizes, these newer variants of TCP consistently outperform TCP Reno.

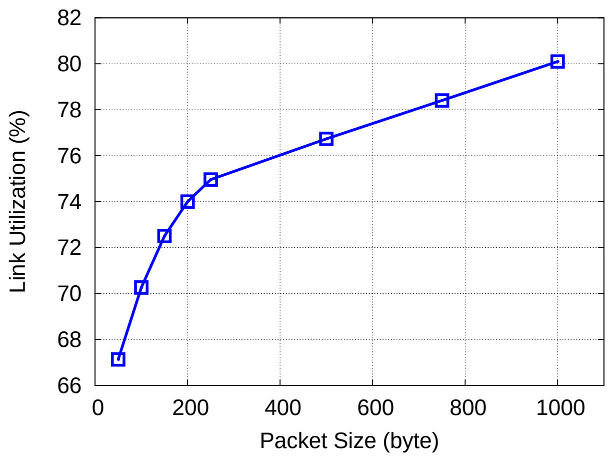
High-speed flavors of TCP are designed to improve the performance of long-lived large-bandwidth flows. In finding an individual flow's share of bandwidth, they are more aggressive than the traditional TCP Reno (and hence more responsive to the available bandwidth), but have also a higher packet drop rate compared to TCP Reno.

3.4.3 Packet size

So far, we have assumed that the size of data packets is 1000 bytes, and the unit of our buffer size measurements has been the *number of packets*. However, the buffer size required to achieve a certain link utilization is not independent of the packet size.

Figure 3.8 shows the bottleneck link utilization in a set of simulations with packet sizes 50-1000B. In these simulations, independent of the packet size, the offered load

control and improves its RTT-fairness. The window growth function of CUBIC is governed by a cubic function in terms of the elapsed time since the last loss event.



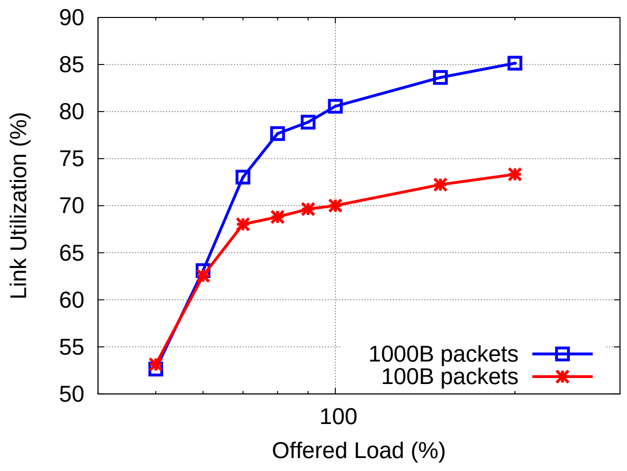


Figure 3.8: Link utilization vs. packet size. *Top*: on a congested link, smaller packets result in a lower utilization; *bottom*: at loads below 60%, same utilization is achieved with 100B and 1000B packets.

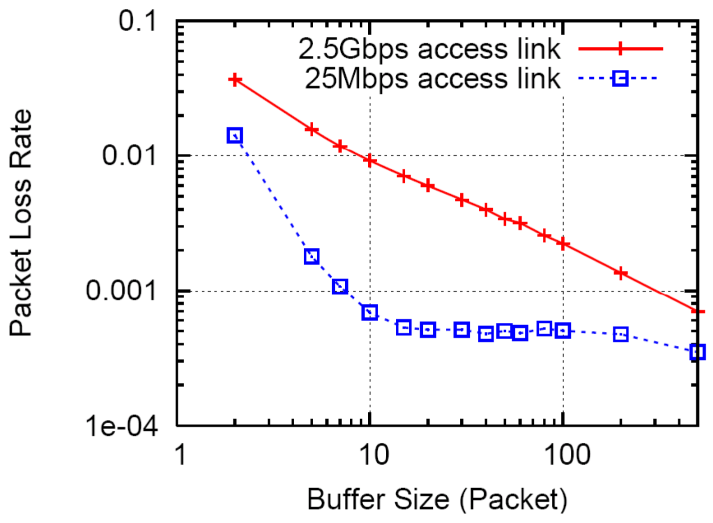


Figure 3.9: Packet drop rate vs. buffer size. The offered load on the bottleneck link is 100%.

is 100%, the buffer size at the bottleneck link is 15 packets, and the maximum TCP window size is 64KB. The window size in number of packets increases as the packet size becomes smaller.

With smaller packet sizes, recovery from packet drops becomes slower. In the congestion avoidance mode, TCP increases the congestion window by one segment every round-trip time; therefore, smaller packets make the ramp-up periods longer and the utilization loss larger. As the load on the bottleneck link (and consequently the drop rate at the buffer) reduces, the difference in utilization becomes smaller (Figure 3.8). At loads below 60%, link utilization is the same with both 100B and 1000B packets.

3.5 Packet drop rate

When a network link is congested, there have to be some drops at the link to notify the TCP senders of the congestion, and make them reduce their transmission rates.

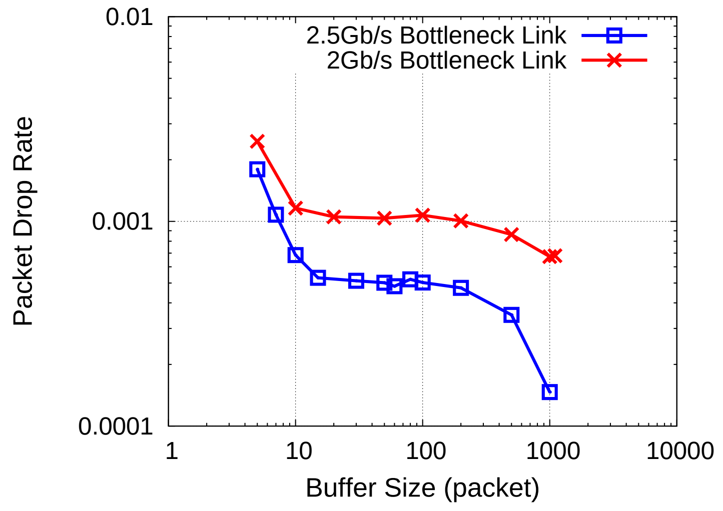


Figure 3.10: Comparison of packet drop rate when tiny buffers are used versus when bottleneck bandwidth is reduced. Under the same load, the 2.5Gb/s link with a tiny 10-packet buffer results in fewer drops than a 20% slower link with much larger buffers.

Figure 3.9 shows the average drop rate at the output ports of the simulated router in the baseline setting.

The offered load in the above simulation is 100%. Repeating the simulation with an offered load below 70% does not show any packet drops at the router, even with 15-packet buffers at output ports. This suggests that under typical conditions, when the core links are not congested, using tiny buffers does not affect the drop rate.

As noted previously, losing about 20% of the link utilization is equivalent to running the backbone link 20% slower. The plot in Figure 3.10 compares the packet drop rate of a 2.5Gb/s link with that of a 2Gb/s link (with 20% reduced bandwidth) under the same offered load. We can see that the drop rate of the faster link at 15-packet buffer is smaller than the drop rate of the slower link at $\frac{RTT \times C}{\sqrt{N}} \approx 1100$-packet buffer. In other words, the fast link with tiny buffers drops fewer packets than the link with 20% reduced bandwidth, but with much larger buffers.