Buffer Sizing in a Combined Input Output Queued (CIOQ) Switch

Neda Beheshti, Nick Mckeown
Stanford University

Abstract

In all internet routers buffers are needed to hold packets during times of congestion. In some recent work, the question of finding the minimum buffer size guaranteeing high throughput has been addressed [3] - [6]. The answer to this question is particularly important in building all-optical routers, where the optical technology allows buffering up to a few dozen packets [7]. While in practice most high speed routers use input queued switches, the previous works have all focused on the problem of buffer sizing in output queued switches.
In this work, we consider a combined input output queued switch in the core of a TCP network. Based on ns2 simulations and by adding a CIOQ switch to the simulation environment, we investigate how different parameters can affect the amount of buffering needed to have high link utilization.

1 Introduction

It has been known for a long time that having large buffers in an internet router would result in higher link utilization. In a network of TCP flows, a widely-used rule-of-thumb [2] suggests that the amount of buffering needed to have full utilization of bottleneck links is

$$B = \overline{RTT} \times C$$

where $\overline{RTT}$ is the average round-trip time of a TCP flow passing through the bottleneck link. In [3] it is shown that when a large number of flows are sharing the bottleneck link, this amount can be scaled down by a factor of $\sqrt{N}$, where N is the number of flows through the bottleneck link. The possibility of more reduction in buffer size is explored in [6]. The authors argue that if the TCP flows are not overly bursty, in an over-provisioned network $O(\log W)$ buffers are sufficient for high throughput. Here W is the maximum window size of each TCP flow.
The fabric and memory of an $N \times N$ output queued switch must run N times

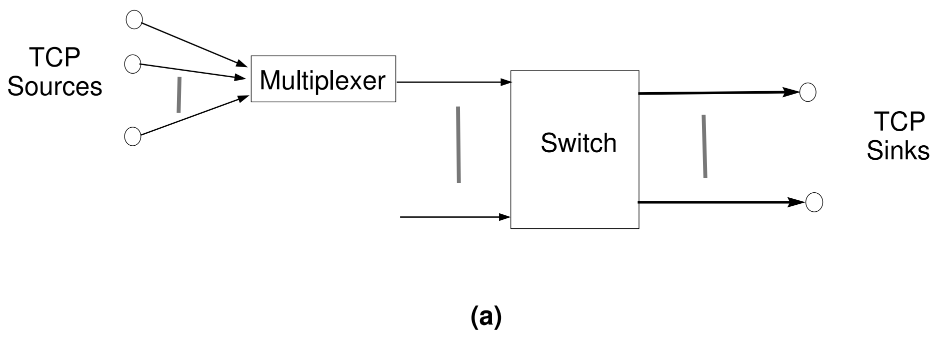
faster than the line rate. This makes output queueing impractical for switches with high line rates in general and for optical switches in particular.
In this paper we consider the problem of finding an appropriate buffer size in a combined input output queued (CIOQ) switch. A CIOQ switch with speed up of $1 \leq S \leq N$ runs S times faster than the input and output line rate. At every time slot, up to S cells can be removed from each input port and up to S cells can be transferred to each output port. By ns-2 simulations we show that with speedup of 2 and very small buffers at the input ports of the switch high link utilization is achievable. We also show how changing different parameters affects the throughput of the network.

2 CIOQ Switch in ns-2

Prior to this work, ns-2 was only capable of simulating networks with output queued switches, i.e., nodes in which all the arriving packets from different ports are immediately transferred to the output ports. In this work, an input queued switch module is added to ns-2 environment which is capable of modeling the contention between packets arriving at different input ports and destined for different output ports. In an IQ switch, arriving packets at each input port are queued in a separate buffer. A scheduling algorithm determines which packets to transfer from the input ports to the output ports under the 2 constraints that during each connection, 1) Either zero or one packet can be removed from an input, and 2) At most one packet can be delivered to an output.
Since the packets arriving to the switch may have variable length, we have the option of adding a block for segmenting arrival packets into fixed length cells. The cells are then sent to the internal switch and are transferred to the output ports under the scheduling policy. Each connection of the switch is long enough for one cell to be transferred to the output port. At the output side, the cells are reassembled back into packets before being sent across the output links.

3 Experiment Settings

We consider a switch with 32 input and output ports in the core of a network with link capacity of 40 Gb/s. Each input port of the switch carries 500 different TCP/Reno flows multiplexed at a prior stage. Figure 1 shows the general topology of the simulated network. Each flow is generated at a separate source node which is connected to a multiplexer via an access link. The capacity of the access links varies in different simulations, and is assumed to be less than or equal to 40 Gb/s.
Output ports are divided into 4 groups of 8 channels. All the channels in a group conduct traffic to one destination. The generated flows are all long-lived TCP flows with an average RTT of 40ms. Different RTTs are modeled by adding random jitter to the propagation delay of the access links.



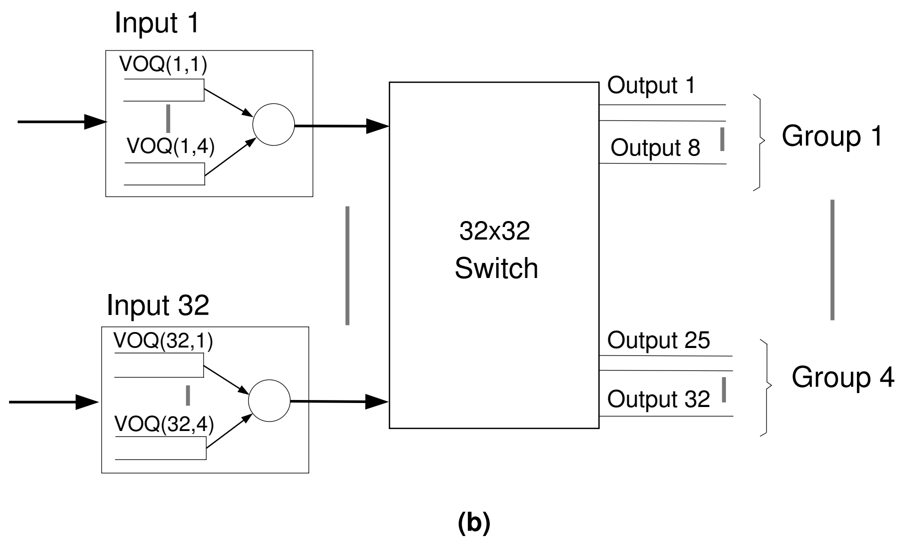


Figure 1: (a) General topology of the simulated network, (b) Input queued switch with virtual output queues. 8 channels carry traffic to each destination node.

Scheduling Algorithm- The scheduling policy that configures the switch connections is based on the GiSLIP algorithm proposed by Ramesh Rajaduray. At each connection of the switch, matching between input and output ports is determined by a simple request-grant-based policy. Once a connection is established between a pair, the pair remains connected until the packet is completely transferred to the output port (packet-mode scheduling).
If there are several choices at some step of the algorithm, the decision is made according to priority lists. There are 2 priority lists corresponding to each group of channels at the output (Channels that carry traffic to a common destination node). The *input-priority* list maintains the order in which the inputs should be selected in case of contention between different inputs. The *channel-priority* list keeps track of the selected channels at each connection. This makes the distribution of load uniform among different channels in a group.

At each decision time, an input port or an output channel is called *busy* if it is in the middle of transferring a packet. Otherwise, it is called *free.* Assuming that output i has k_i free channels, the algorithm goes through the following 4 steps:

- **Step 1.** Among all free inputs having a packet for Output i, this output gives grant to the first k_i inputs in its *input-priority* list.
- **Step 2.** If an input receives grants from more than one output, it accepts the one with the highest priority. The priority pointer of the input is then incremented by one(modulo N).
- **Step 3.** Each output updates its *input-priority* list by moving the inputs to which a connection is established at this decision cycle to the bottom of the list.
- **Step 4.** If output i receives accepts from $a_i \leq k_i$ inputs, it assigns the first a_i free channels from its *channel-priority* list to the accepting inputs. The selected channels are then moved to the bottom of the list.

At the completion of these 4 steps, all the selected input ports and output channels are flagged as busy until the packet transformation is complete.

4 Simulation Results

In this section we present the results of simulations using ns-2. The topology of the simulated network is as depicted in figure 1 and the general settings are as mentioned in the experiment settings section.
Due to memory and running time constraints of ns-2, the number of flows on each input port is bounded by 500 flows. That makes the total number of

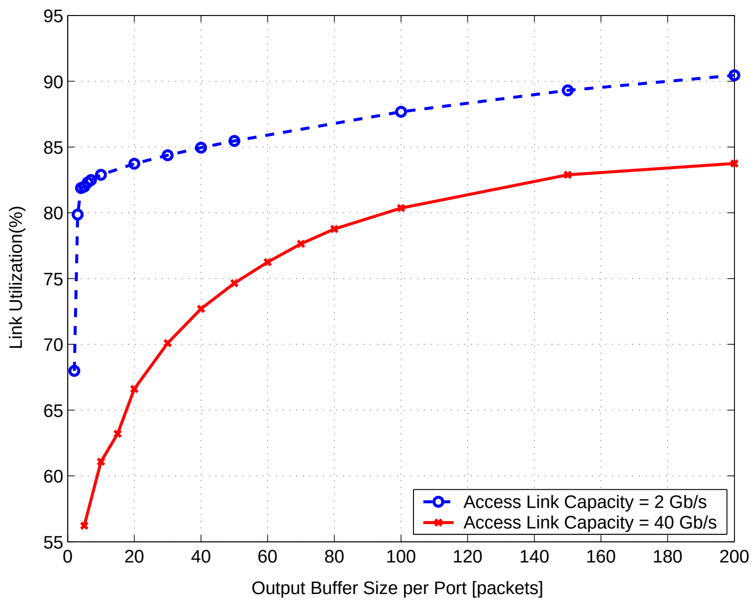


Figure 2: Average link utilization for different output buffer sizes; with limited access links the throughput increases significantly.

flows in the network equal to $16,000$. In these simulations we assume that all generated packets are destined for a common destination node. The switch is assumed to have speedup of 2 and only up to 2 packets can be buffered at each VOQ.

Figure 2 shows how link utilization varies as the size of output buffer increases considering 2 scenarios: 1) Access links have the same capacity as the core links (40 Gb/s) and 2) Capacity of each access link is limited to 5 percent of the core link capacity (2 Gb/s). In both cases the maximum offered load to the output links is equal to one, i.e., the maximum window size of each TCP flow W_{max} is chosen such that

$$\frac{NW_{max}}{RTT} = k * C$$

where N is the total number of flows, C is the capacity of core links (40 Gb/s) and k is the number of channels between the switch and each destination node (8). With a total number of $16,000$ flows passing through the switch, W_{max} should be equal to 100 packets to make the maximum offered load 100%.

The plot shows that having limited access links in the network, with buffering only up to 5 packets at each output port, link utilization of 80% is achievable. This number increases to 100 packets when all links are running at the same

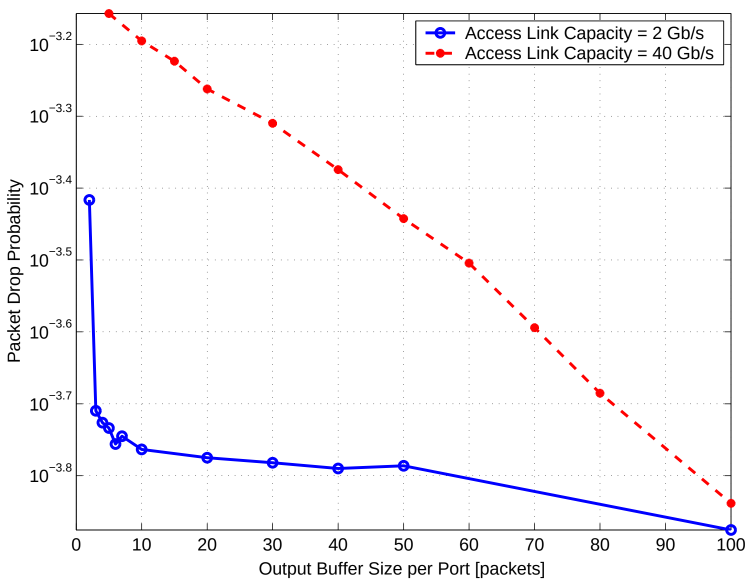


Figure 3: Packet drop rate for different output buffer sizes.

capacity. In both cases there is no need for buffering more than 2 packets at the input VOQs. The Packet drop rate versus the output buffer size in both cases in depicted in figure 3.
Next we consider a switch with over-loaded output links. In this case, if each flow sends packets at its maximum possible rate W_{max}/RTT, the load would be 1.5 times what can be carried over output links. Since the flows can get more bursty, we expect a little degradation in the throughput. Figure 4 compares the link utilization of an over-loaded network with that of a network with 100% maximum offered load.
The amount of buffering needed in a router depends on how bursty the TCP traffic is. As argued in [6], when the access links have a limited capacity compared to the core links, the router finds the arrival traffic less bursty and this reduces the size of buffers. With a fixed buffer size, dependency of the link utilization on the access link capacity is shown in figure 5. When access links are running at 1 Gb/s, the utilization increases from 72% to 88%, with output buffers of size 40 packets.

5 Conclusions

We considered the buffer sizing problem in a combined input output queued switch in a network of TCP flows. Based on ns-2 simulations, we showed that in a 32x32 switch with speedup of 2, there is no need for buffering more than

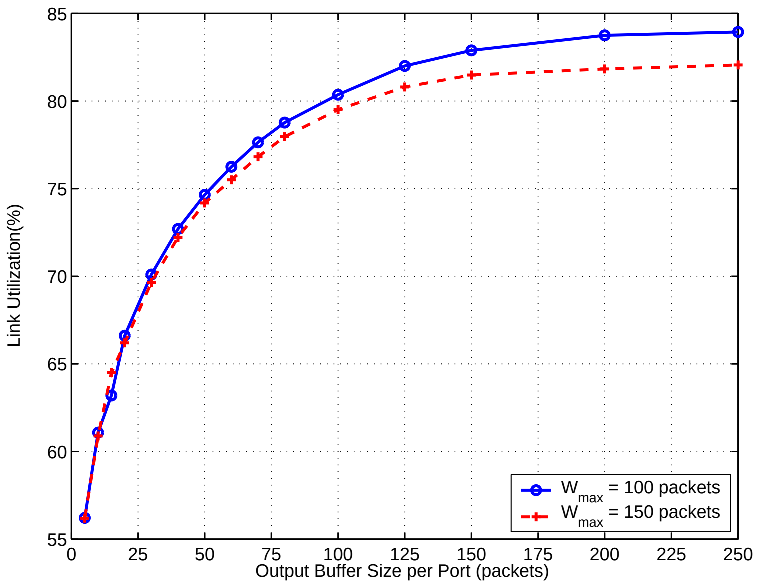


Figure 4: Link utilization vs. the buffer size. In the over-loaded network the throughput degradation increases at larger buffer sizes.

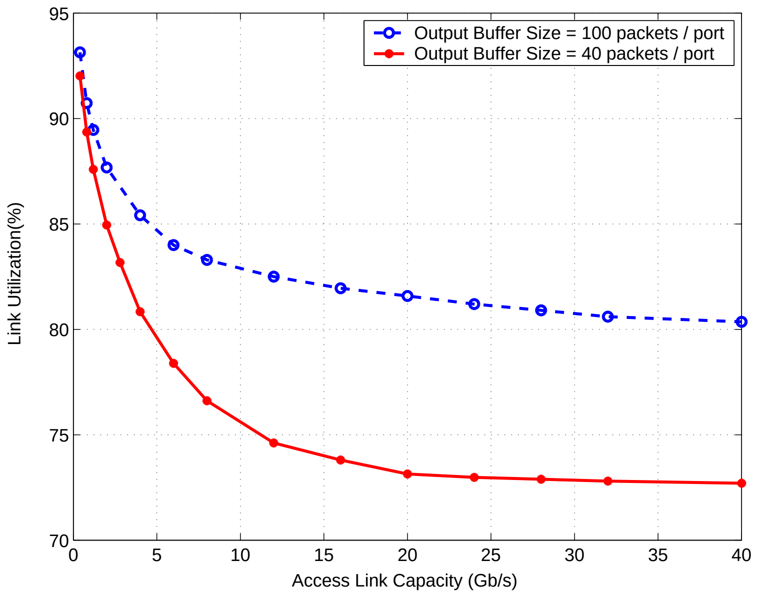


Figure 5: Link utilization for different access link capacities.

2 packets at the input VOQs when the number of flows on each input port increases up to 500.
The simulation results show that the required buffer size at the output of the switch is a decreasing function of the capacity of access links. The slower the access links are, the less buffer is needed at the output ports. But even when the access links run as fast as the core links, by sacrificing a fraction of the link capacity, the output buffers can be made very small. This would be a reasonable trade-off in an all-optical network where link capacity is plentiful.
At 40 Gb/s with high capacity access links, the proposed buffer size is 30-50 packets per output port for achieving 70-75 % link utilization. The results show that doubling the buffer size to 100 packets would increase the utilization by about 5%. The throughput can go well above 75% in a network with slow access links where each flow is less bursty. When the access link capacity is 2 Gb/s for example, buffer size of 50 packets results in a throughput of about 85%. In practice, the capacity of access links can be several orders of magnitude smaller than that of the core links.
In all the simulation results shown in this paper, it was assumed that there are 500 flows on each input port of the switch. The number of flows was bounded above because of the memory constraints of the ns-2 simulator, but the results give a lower bound on the throughput of a network with tens of thousands of flows.

6 Acknowledgement

The authors would like to thank Ramesh Rajaduray for helpful discussions and for proposing the GiSLIP algorithm.

References

[1] The network simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[2] C. Villamizar and C. song. High performance TCP in ANSNET. *ACM Computer Communications Review,* 24(5):45-60, 1994.

[3] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing Router Buffers. *SIGCOMM '04*, pages 281-292, New York, NY, USA, 2004. ACM Press.

[4] Damon Wischik and Nick Mckeown. Part I: Buffer sizes for core routers. *ACM SIGCOMM Computer Communication Review.* Volume 35, Number 2, July 2005

[5] Gaurav Raina, Don Towsley, and Damon Wischik. Part II: Control theory for buffer sizing. *ACM SIGCOMM Computer Communications Review.* Volume 35, Number 2, July 2005.

[6] Mihaela Enachescu, Yashar Ganjali, Ashish Goel, Nick McKeown, and Tim Roughgarden. Part III: Routers with very small buffers. *ACM SIGCOMM Computer Communications Review.* Volume 35, Number 2, July 2005.

[7] H. Park, E. F. Burmeister, S. Bjorlin, and J. E. Bowers. 40 gb/s optical buffer design and simulations. *Numerical Simulations of Optoelectronic Devices (NUSOD)*, 2004.