Experimental Evaluation of a Coarse-Grained Switch Scheduler

Charlie Wiseman
Washington University
+1-314-935-4586
wiseman@wustl.edu

Jon Turner
Washington University
+1-314-935-6132
jon.turner@wustl.edu

Ken Wong
Washington University
+1-314-935-7524
kenw@arl.wustl.edu

Brandon Heller
Washington University
+1-314-935-6160
bdh4@arl.wustl.edu

ABSTRACT

Modern high performance routers rely on sophisticated interconnection networks to meet ever increasing demands on capacity. Previous studies have used a combination of analysis and idealized simulations to show that *coarse-grained scheduling* of traffic flows can be effective in preventing interconnect congestion while ensuring high utilization. In this work, we study the performance of a coarse-grained scheduler in a real router with a scalable architecture similar to those found in high performance commercial systems. Our results are obtained by taking fine-grained measurements within the router that provide a detailed picture of the scheduler's behavior under a variety of conditions, giving a more complete and realistic understanding of the short time-scale dynamics than previous studies could provide.

Categories and Subject Descriptors. C.2.1 [**Computer-Communications Networks**]: Network Architecture and Design – *network communications, packet-switching networks.*

General Terms. algorithms, measurement, performance

Keywords. distributed scheduling, coarse-grained scheduling, high performance routers

1. INTRODUCTION

Modern high-end routers such as Cisco's CRS-1 [2] have hundreds or thousands of ports capable of supporting link speeds of 10 Gb/s or more. While it is easy to design an accompanying interconnect which works well under benign traffic conditions, it is also important that it operate well under more extreme conditions that can (and do) occur on a fairly routine basis. One way to meet performance needs under all conditions is to equip the system with a *scheduler* that controls the flow of data through the interconnect with the explicit goal of avoiding internal congestion, while moving traffic through the interconnect as quickly as possible. For large systems, the fine-grained scheduling method used in crossbars [1,3] becomes impractical, since it is difficult to make scheduling decisions for large numbers of ports in the short time available for scheduling. Instead of attempting to make scheduling decisions on a packet-by-packet basis, coarse-grained scheduling simply regulates the rates at which each input sends to each output over a scheduling interval that is much longer than the individual cell time, on the order of tens to hundreds of microseconds.

Copyright is held by the author/owners(s).
ANCS'07, December 3-4, 2007, Orlando, Florida, USA.
ACM 978-1-59593-945-6/07/0012.

2. CONTRIBUTIONS

In this work we implement and rigorously evaluate one distributed, coarse-grained scheduler in the extensible routers of the Open Network Laboratory [4]. In doing so, we have developed an infrastructure that allows this class of schedulers to be easily implemented and tested at Gb/s line rates. In particular, we have extended the ONL API to include functionality which provides ONL users the ability to place their own scheduler on the router, added measurement tools to the suite of standard router code that can run at up to a 100 μs granularity, and added flexible traffic generators which can produce Gb/s flows of any size packets with the additional ability to synchronize multiple generators to start flows to within 15 μs of one another.

Distributed Batch LOOFA (DBL) [5] is the coarse-grained scheduler we study here which is based on the Least Occupancy Output First Algorithm (LOOFA) [3] for fine-grained crossbar scheduling. The original version of DBL required some modifications to be used in practice. The most important change was the need for the algorithm to work asynchronously among each of the ports while still maintaining a congestion-free interconnect. For details of these enhancements, as well as further background, see [6].

3. EVALUATION

The ONL routers are built around a scalable switch core designed to support 1 Gb/s external links and provide a configurable internal bandwidth up to roughly twice the external bandwidth. Each of the 8 port processors is composed of an FPGA configured to handle all normal packet processing functions and an embedded general purpose processor for running user-customized code. In our case, we use this general purpose processor for both DBL as well as our measurement agents. Incoming packets are placed into the appropriate Virtual Output Queue (VOQ) at each input port before being transmitted over the switch. At each output port, packets are queued again to await transmission into the external link. DBL controls packet flow through the system by rate limiting the VOQs to ensure that no output is overwhelmed while still maintaining high overall throughput.

At the beginning of each experiment the measurement infrastructure synchronizes the time at each port so that all data obtained during the experiment are easily comparable. We have verified that this synchronization results in a spread of no more than 10 μs across all ports. In our experiments, the measurement code is run every 100 μs to record queue lengths (both input and output) and the VOQ rates assigned by DBL. DBL is configured to update the VOQ rates based on the queue lengths every 500 μs. This allows us to examine the overall behavior of the system at a finer time-scale than the scheduling period so that the effects of the scheduling decisions can be understood and characterized.

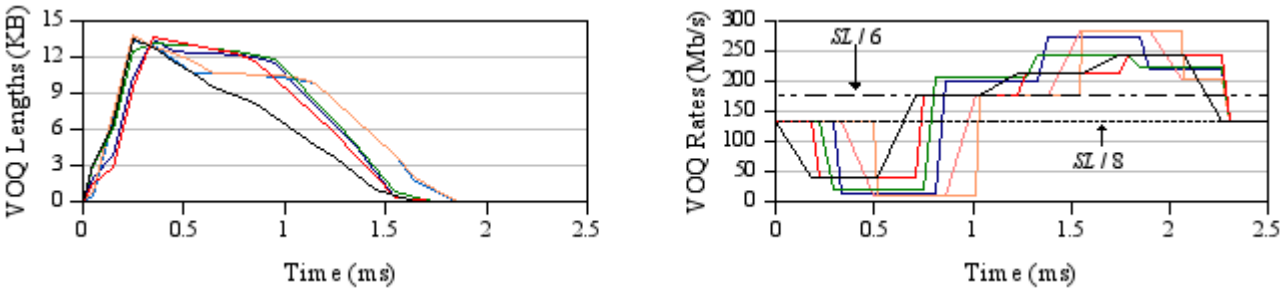

Figure 1. Response to a sudden burst of traffic.

Figure 1 shows the results of a 250 μs burst of traffic from six inputs all destined for one output. The packets are UDP packets with a 50 byte payload, sent to each port at a rate of L=900 Mb/s. The switch bandwidth is set to be S=1.2 times L, so SL=1080 Mb/s. Each curve in the figure represents the VOQ lengths (left chart) and VOQ rates (right chart) at one of the six inputs for the overloaded output. Ideally, the rates would all start and end at $SL/8$ Mb/s (the nominal rate when the system is idle) and rise to $SL/6$ Mb/s (each input equally sharing the capacity to the output) until the burst was finished. Instead, as the scheduler on each port runs for the first time since the burst began, the rates are set to near zero for one scheduling interval before rising to the expected rate. This is due to a mechanism in our implementation for safe asynchronous operation where each port delays using its own current queue lengths for one scheduling interval. The need for such a mechanism is detailed in [6], but is primarily due to the nature of the DBL algorithm which assigns VOQ rates based on input-side queue lengths in the absence of output-side constraints. This can lead to unexpected behavior when many new flows arrive close to one another. As can be seen in the figure, however, there is an unintended drawback. The delayed reaction has the opposite effect as the flows finish and the overloaded output is temporarily over-subscribed before the rates settle back to the nominal rate. Such behavior was only revealed once we moved to a real asynchronous implementation and only detectable due to our fine-grained measurement infrastructure.

The results for another experiment with bursty traffic are shown in figure 2. One output, the *subject*, receives a constant stream of packets at the line rate for the entire experiment. Only a single input receives traffic for the subject at any one moment and that input changes quickly over time. When an input is not receiving traffic for the subject, it sends randomly to any other output. This creates a challenging situation for the scheduler. There is significant input-side contention for the switch bandwidth, but the subject must be steadily supplied with packets in order to achieve maximum throughput. In this example, UDP packets with a 500 byte payload are used in conjunction with a line rate of 900 Mb/s. The curve in figure 2 shows the number of packets forwarded by the subject during each scheduling interval. The scheduler is unable to supply enough packets to the subject to maintain line rate for roughly half of the experiment.

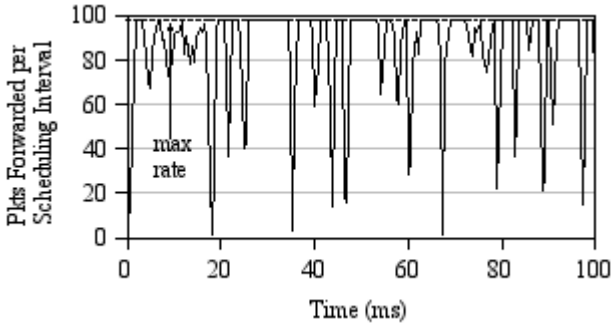

Figure 2. Response for a bursty traffic pattern.

4. SUMMARY

In addition to the many other experimental evaluations, [6] also contains simulation results for comparison as well as a discussion of the scalability of the scheduler. Based on our findings, we believe that distributed, coarse-grained schedulers represent an excellent choice for ensuring high performance in large, scalable routing systems even under extreme traffic patterns. Our measurement and traffic generation infrastructure which supports fine-grained time synchronization allows this class of schedulers to be implemented and tested with relative ease in the routers of the Open Network Laboratory.

5. REFERENCES

[1] Anderson, T., S. Owicki, J. Saxe and C. Thacker. "High Speed Switch Scheduling for Local Area Networks," *ACM Trans. on Computer Systems*, 11/93.

[2] Cisco Carrier Routing System. http://www.cisco.com/en/US/products/ps5763/index.html.

[3] Krishna, P., N. Patel, A. Charny and R. Simcoe. "On the Speedup Required for Work-conserving Crossbar Switches," *IEEE J. Selected Areas of Communications*, 6/99.

[4] Open Network Laboratory. http://www.onl.wustl.edu.

[5] Pappu, P., J. Turner and K. Wong. "Work-Conserving Distributed Schedulers for Terabit Routers," *Proceedings of SIGCOMM*, 9/04.

[6] Wiseman, C., J. Turner, K. Wong and B. Heller. "Experimental Evaluation of a Coarse-Grained Switch Scheduler", Washington University Computer Science and Engineering Technical Report, WUCSE-2007-51, 10/07.