Components of Congestion Control

Ludmila Cherkasova, Al Davis[*], Robin Hodgson,
Vadim Kotov, Ian Robinson, Tomas Rokicki
Hewlett-Packard Laboratories
1501 Page Mill Road, Palo Alto, CA 94303
e-mail: {cherkasova,hodgson,kotov,irobinson,rokicki}@hpl.hp.com

Contents

1 Introduction

To design a high-speed, low latency, high capacity, reliable interconnection fabric for scalable parallel processing systems, one inevitably faces the problem of congestion control. Congestion management is a hot controversial topic. Contradictory beliefs supports some of the old myths and create new ones about congestion and mechanisms to control it [Jain92].

This paper presents three complementary components of the flow control solution adopted for the *Fed-X* fabric. Each of the three addresses performance problems caused by a particular characteristic of realistic network workloads.

The Fed-X fabric is a high-speed scalable interconnect for a multi-computer system [Davis92, Davis94]. The basic switch design used in the Fed-X interconnect uses two well-known techniques for reducing contention within the fabric: internal buffering and adaptive routing.

Internal buffering provides a measure of elasticity in response to contention [Novatsky95, Stunkel94]. It prevents contention at a port from immediately blocking the links used by incoming packets headed for that port—a situation that can cause early saturation in unbuffered designs. While this approach significantly reduces cogestion over an unbuffered approach, the buffers become resources in the switch fabric which any additional flow control schemes—or buffer allocation strategies—must allocate carefully. Adaptive routing also reduces congestion, especially in the presence of hot spots, but complicates flow control due to the increased number of paths, and thus buffers, through which packets may flow. The Fed-X fabric uses a deadlock-free adaptive routing algorithm to guarantee forward progress.

The three flow-control techniques introduced in this paper are backpressure flow control, alpha scheduling, and balanced injection. Our variation on backpressure flow control ensures that all of the buffers in a

[*]Department of Computer Science, University of Utah, Salt Lake City, UT 84112; e-mail: ald@endo.cs.utah.edu

node are not filled with one particular category of packet, such as those destined for the same node, so that buffers are available for other packets. Alpha scheduling exploits the distinction between message latency and packet latency to maximize the effective performance of the interconnect from the perspective of applications. Balanced injection throttles the injection of traffic into a node from outside the network based on local congestion as measured by local buffer usage. These three flow-control techniques, together, provide a significant increase in fabric performance, both in sustainable throughput and in message latency, under a wide variety of realistic traffic patterns.

Evaluations of interconnect performance generally focus on transport latencies for fixed-size packets as a function of traffic load. To an application, however, it is the latency of the variable-length messages, rather than the individual packets into which they get segmented, that is important. This shift in perspective—from packets to messages—has another effect for performance evaluation. The bursts of packets that result from message segmentation generate a different type of contention. Instead of occasional packets competing briefly for the same port in the interconnect, two bursts compete for some port over a longer period of time. In order to reflect realistic workloads and provide useful results, our simulation workloads included bursty, hot-spot traffic and our results are presented in terms of the average message latency delivered to applications.

2 The Fed-X Interconnect

The Fed-X interconnect is a direct network (each switch in the network attaches to a compute node, or PE). The switches are connected in a wrapped rectangular 2D mesh topology. Each includes a routing device with six half-duplex links providing connections to the adjacent switches. An additional port provides the connection to the local PE. Messages traveling through the interconnect are split into fixed-length packets. The first few words of a standard packet comprise the packet header which contains the source and destination addresses of the packet as well as a unique message and packet identifier. Each half-duplex link alternates directions between the switches on a per-packet basis. Each switch also includes a centralized buffer pool consisting of ten dual-ported, packet-sized buffers.

Routing logic decides which port or ports an arriving packet should be forwarded to. If the port is available, the packet transmission starts, even as it is still being received. This *virtual cut-through* technique minimizes per-hop latencies. If the port is not available the incoming packet accumulates in a switch buffer until the port becomes free. Since the buffer is large enough to absorb the whole packet the link used by the incoming packet can be freed up, even if congestion at the output port persists.

A bit within the packet header determines whether packets are routed deterministically (i.e. always following the same path), or adaptively (i.e., at each switch, taking any path that takes it closer to its destination). A cycle-free routing and buffer allocation strategy is used to avoid deadlock.

As a simple wrapped rectangular 2D mesh, the X-mesh can be readily scaled in either or both dimensions. For example one stackable switch hub or rack backplane could carry a 2x8 array of Fed-X switches. These units could be stacked to provide 32, 48, 64, or more ports. The performance results shown in the paper are for an 8x8 size interconnect. Our simulations indicate that the latency and throughput characteristics for alternative sizes are similar to those presented here.

The main parameters used in modeling the performance of the Fed-X interconnect were:

- Each port permits a byte of information to be transmitted in 1 time unit. Each standard packet is 160 bytes long and hence takes 160 time units to transmit.
- To receive a packet header and to compute the next available direction takes 12 time units.
- There are 10 buffers in each switch.

3 Experimental Workload Description

The Fed-X interconnect transfers variable-length messages by fragmentating them into a series of fixed-length packets. We used synthetic workloads to simulate the nature of this bursty traffic. In particular, we designed bimodal message length distributions consisting of short messages and long messages. Our basic workload has short messages of from one to five packets in length, with each length having equal probability, and long messages of twenty-five packets. Short messages contain 128 to 640 bytes of payload, typical for control messsages of various types, and long messages contain 3,200 bytes of payload, approximately the size of a disk or memory page.

We characterize our workloads by the percentage of long messages and short messages in the traffic. For instance, a typical workload might have 10% long messages and 90% short messages. The average message length for this workload is 5.2 packets. We assume a Poisson arrival process. Given a traffic density u between zero and one, we generate new messages with an average interarrival time of $5.2/u$.

Message latency is measured from the moment the message is sent by the application or operating system, as defined by the moment the message appears on the interconnect job list, to the moment all the packets of the message appear at the destination. Thus, this time includes both the queue wait time (in the source PE) and the interconnect transit time.

We also measure and report the packet latency, as measured from the moment when the source PE starts to inject the packet into the interconnect, until the moment when the packet is completely ejected to the destination PE. Packet latency does not include any queue wait time.

In order to validate the effectiveness of the various flow control strategies, special synthetic workloads with "hot spots" were designed. A few switches across the interconnect are declared to be the hot spots, and a specified subset of PEs aggressively saturate the hot spots with messages. The rest of the PEs send messages to each other.

The set of workloads that can be generated as described above reflect specific characteristics of real traffic patterns that illustrate the flow-control techniques we introduce in this paper. Our evaluation of Fed-X and testing of the flow-control techniques included many additional more complex workloads.

4 Backpressure Flow Control

Bursty traffic has particular consequences for contention within the interconnect. Instead of occasional packets competing briefly for the same port, two bursts compete for some port during a longer period of time. If this contention is not controlled, packets can build up in the preceding switches, increasing contention and eventually completely saturating the interconnect.

If the interconnect does not have any flow control mechanisms then long messages competing for output ports will quickly fill the buffer pools and deadlock the network—even under very light traffic loads. Fed-X uses a unique deadlock-free routing strategy and buffer allocation scheme to prevent this occurrence. While this strategy provides a basic flow-control mechanism that guarantees forward progress within the interconnect, it does not by itself guarantee good performance under a variety of traffic patterns.

The Fed-X link protocol is based on a negative acknowledgement protocol which can examine and refuse to accept an incoming packet based on a variety of specified conditions (a packet is not deleted from the source node until it is positively acknowledged). The basic condition is that a Fed-X switch will reject a packet if no buffers are available. A more restrictive backpressure flow-control mechanism can be introduced by extending this condition, rejecting a packet if too many packets that are similar in some way already exist in that node. Thus, this mechanism limits the resources that can be obtained by a particular message or type of message, leaving resources for other packets. We introduce and compare two different kinds of backpressure:

- *message-based backpressure*: a switch rejects a packet if it already contains a waiting packet from the same message.
- *destination-based backpressure*: a switch rejects a packet if it already contains a waiting packet with the same destination address.

The *message-based* backpressure stops the flow of packets from a particular message once its packets cause congestion on their way. We assume each message has a unique identifier that can be used to readily identify packets from the same message. This technique does not prevent several different messages sent to the same destination from entering the interconnect and competing for resources. The *destination-based*

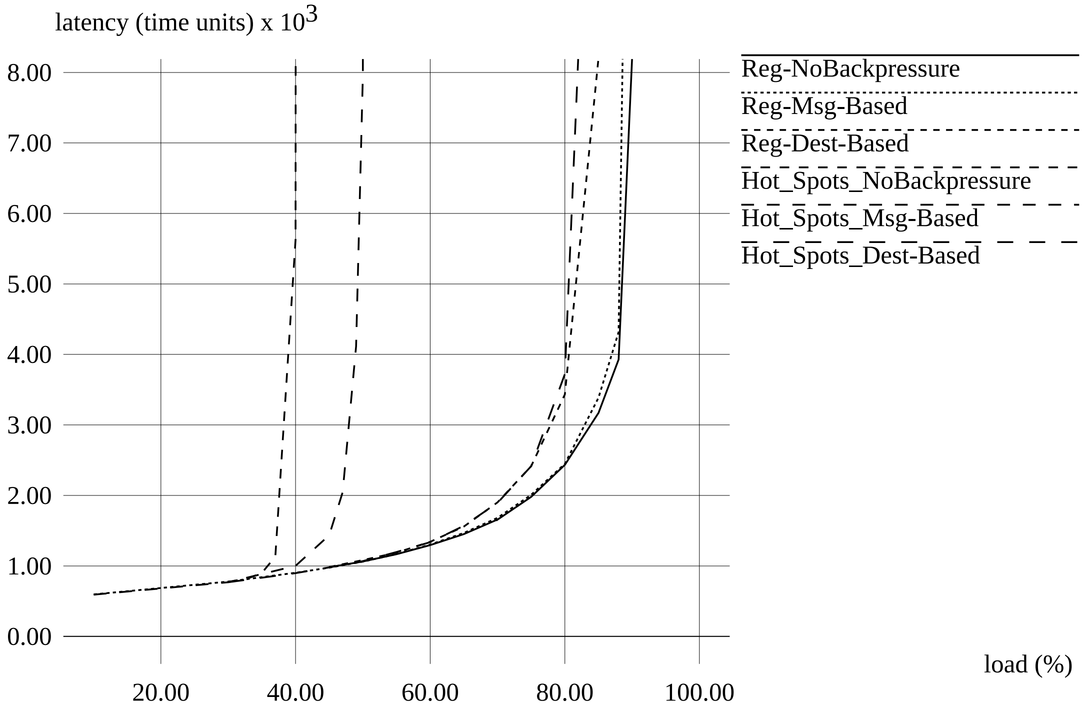

Figure 1: Message Latency for No-Backpressure, *Message-Based* and *Destination-Based* Backpressure (Workload *wk1-5*)

backpressure provides strictly stronger backpressure, because it does not allow more than one waiting packet with the same destination address in a switch.

For modeling we used bursty traffic with hot spots. In particular, 30% of the PEs will send messages to a few specific hot spots in the interconnect. The rest of the PEs send messages to each other; we refer to this latter category as independent nodes. This workload allows us to show the effect of traffic to the hot spots on traffic between the independent nodes. We measured message latency between independent nodes, both with and without hot-spot traffic.

We considered two different workloads which help to show the effect of the backpressure flow-control mechanism. The first workload, denoted *wk1-5*, contains only short messages. Figure 1 shows the average message latency for Fed-X without backpressure flow control as well as with the two different kinds of backpressure. (The results presented in this section also use balanced injection, but not alpha scheduling.)

First note that without hot-spot traffic, the performance difference between the three types of flow-control is small. With hot spots and no flow control, the attainable throughput of the interconnect among the independent nodes is less than half its normal performance. Using message-based backpressure increases the throughput significantly, but not nearly so much as destination-based backpressure. The throughput with destination-based backpressure is very nearly the ideal performance without hot spots. Thus, with destination-based backpressure, the traffic among the independent nodes is affected very little by the hot spot traffic.

The reason message-based backpressure did not work as well as destination-based backpressure is that with so many short messages, many different packets to the same destination but belonging to different messages can fill the buffers around the destination node, preventing other traffic from flowing. In the limit, with only single-packet messages, message-based backpressure does not perform any flow-control at all.

Thus, the strength of message-based backpressure is dependent on message length. This is supported by experiments on a workload consisting only of messages that are 30 packets long. Figure 2 shows the

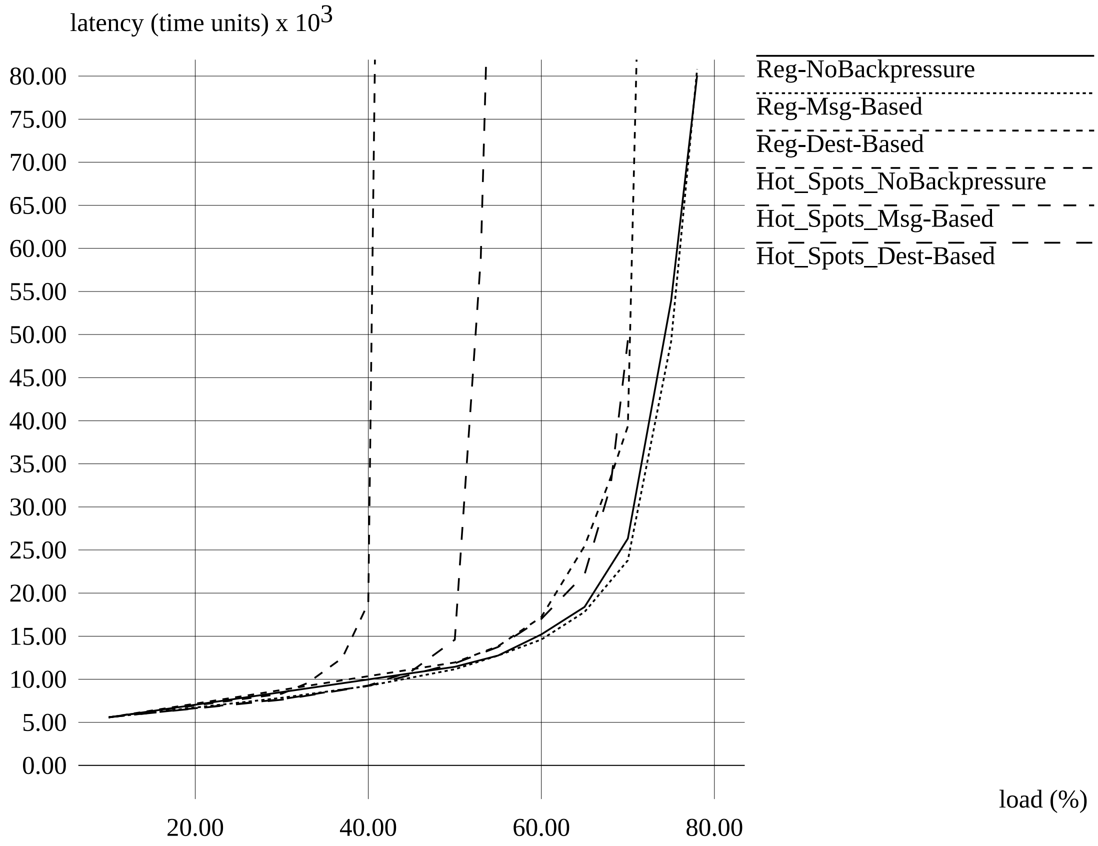

Figure 2: Message Latency for No-Backpressure, *Message-Based* and *Destination-Based* Backpressure (Workload *wk30*)

average message latency for Fed-X without backpressure as well as with message- and destination-based backpressure for both hot-spot and regular workloads. For this workload, message-based backpressure works much better than it did for the previous workload, but it is still significantly inferior to destination-based backpressure.

If the interconnect traffic does not have hot spots then the strong *destination-based* backpressure reduces the overall interconnect throughput by a few percent compared to the lighter *message-based* backpressure, or no backpressure at all. However, since realistic workloads exhibit hot-spot behavior, it is clear that destination-based backpressure is essential to guaranteeing good interconnect performance.

5 Alpha Scheduling

Evaluations of interconnect performance generally focus on transport latencies for fixed-size packets as a function of traffic load. To an application, however, it is the latency of the variable-length messages, rather than the individual packets into which they get segmented, that is important. This shift in perspective—from packets to messages—has important implications for performance evaluation. The naive FIFO injection strategy is not optimal in terms of message latency. Consider the case where a short message is added to the waiting queue immediately after a long message. In this case, the short message is delayed by the long message, and the average queue delay of both messages is high. If the short message were inserted before the long message, then only some packets from the long message would have a long waiting time; the average queue delay of both messages is shorter. In general, the optimal injection strategy with respect to optimizing queue waiting time is shortest message first. On the other hand, shortest message first can result in starvation of long messages.

Opportunistically inserting short messages early is also advantageous in that there is typically more of a premium attached to reducing their latency, even at the expense of latencies for longer (e.g. page-size) transfers.

We propose a scheduling strategy that ranges between FIFO and shortest-first, based on the value of a coefficient. The messages are stored in a priority queue. Three parameters control the ordering of messages in the queue:

- The node parameter c is a "clock" that starts at zero and increments for each packet injected into the interconnect from that node. It is easy to keep this value bounded without changing the scheduling solution.
- The message parameter l is the number of packets in the message that have not yet been sent. Initially this is just the length of the message. As each packet is sent out, the message priority is decremented by α to keep the head message priority up to date. Another strategy is to recalculate the head message priority before preempting it during the scan for insertion of a new message.
- The tuning parameter α controls the balance between fairness and latency minimization; it can range from 0 to ∞

Messages are inserted into the delivery queue with a priority of

$$c + \alpha l.$$

Messages with the lowest priorities get delivered first. A new message inserted into the queue with a priority lower than that of the sending message preempts the sending message.

If $\alpha = 0$, then this strategy is simply FIFO. If $\alpha = 1$ or some other finite positive value, then the strategy will not allow any single application to be delayed indefinitely by the other applications, no matter what their message stream looks like. Larger α provides better average latency; smaller α provides better fairness.

The backpressure flow control mechanism in Fed-X changes the interconnect behaviour in a particular way that allows a certain synergy with alpha scheduling. If a long message is being injected that passes through a congested region, backpressure will quickly stall the inject queue, rejecting further packets from that message. Injecting messages in a strict FIFO strategy will result in a very poor performance especially hurting latency of the short messages delayed by the stalled long messages in front of them [CKR94]. Injecting, at this point, a few short messages significantly decreases their waiting time, with little impact on the overall latency of the long message. Effectively, this automatic fragmentation of long messages by short ones acts to decrease traffic burstiness and randomize the traffic pattern by interleaving "chunks" of long message with short messages, decreasing the congestion caused by long messages.

We performed experiments with a workload called *wk-mixture* with 10% long messages and 90% short messages. Figure 3 shows the effect of alpha message scheduling on the latency of short messages. Different lines on this graph illustrate the effect of scheduling with different values of α ranging from 0 to 8. When $\alpha = 0$, that is, when the strategy is simply FIFO, the latency of short messages is poor. Using $\alpha = 8$ improves the latency of short messages by up to a factor of five, and also improves the overall message latency by up to a factor of three.

Alpha scheduling does not affect the interconnect performance as dramatically for all workloads. For example, with 80% long messages and 20% short messages (in which only 2.9% of the packets belong to a short message) alpha scheduling has less of an impact on average message latency. However, we still see a significant improvement in the latency of short messages.

In general, alpha scheduling allows short messages to interrupt longer messages in such a way that overall queue waiting time is decreased. In the presence of the backpressure flow control mechanism, using alpha scheduling minimizes the latency of short messages without significantly affecting the latency of long messages (since they would be delayed by the backpressure mechanism anyway). Thus, backpressure flow control combined with alpha message scheduling yields an efficient way to control and optimize the traffic pattern.

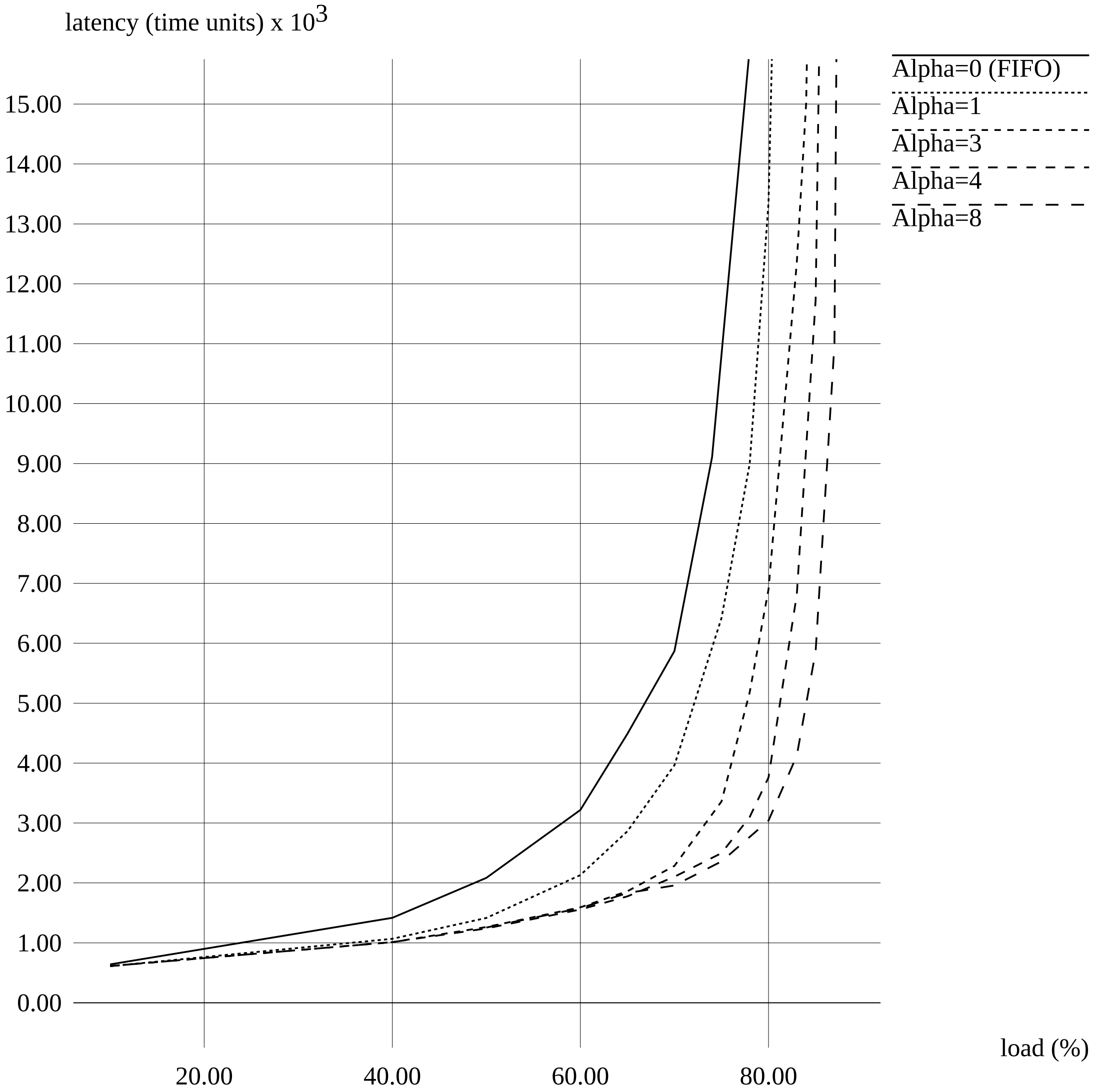

Figure 3: The Effect of Alpha Message Scheduling on the Latency of Short Messages (Workload *wk-mixture*)

6 Balanced Injection

In this section we present the third and final facet of flow-control in Fed-X. This method is based on a local observation of the number of occupied buffers in a switch, yet provides good control over the tradeoff between latency and bandwidth over the entire interconnect.

The number of occupied buffers in a single switch implicitly reflects the level of congestion in the interconnect because it reflects, with a high probability, the level of port congestion at the switch. One can use this observation to control congestion by constraining a switch to only accept a new packet from the PE if the number of occupied buffers is below a threshold which we will define as *BufferLimit*. After a packet is accepted into the interconnect, further port and buffer reservations occur by the usual rule. Establishing, for example, $BufferLimit = 4$, a switch will accept a new packet from the PE if and only if it has less than 4 occupied buffers. To prevent starvation, this mechanism must be augmented with some sort of time-out or other progress guarantee.

The proposed method performs better if it is able to distinguish "destination" buffers (i.e. buffers which are used by packets that have arrived at their destination node and are only waiting to be ejected) and "transit" buffers used for the remainder of the packets (i.e. in-transit packets). By distinguishing the

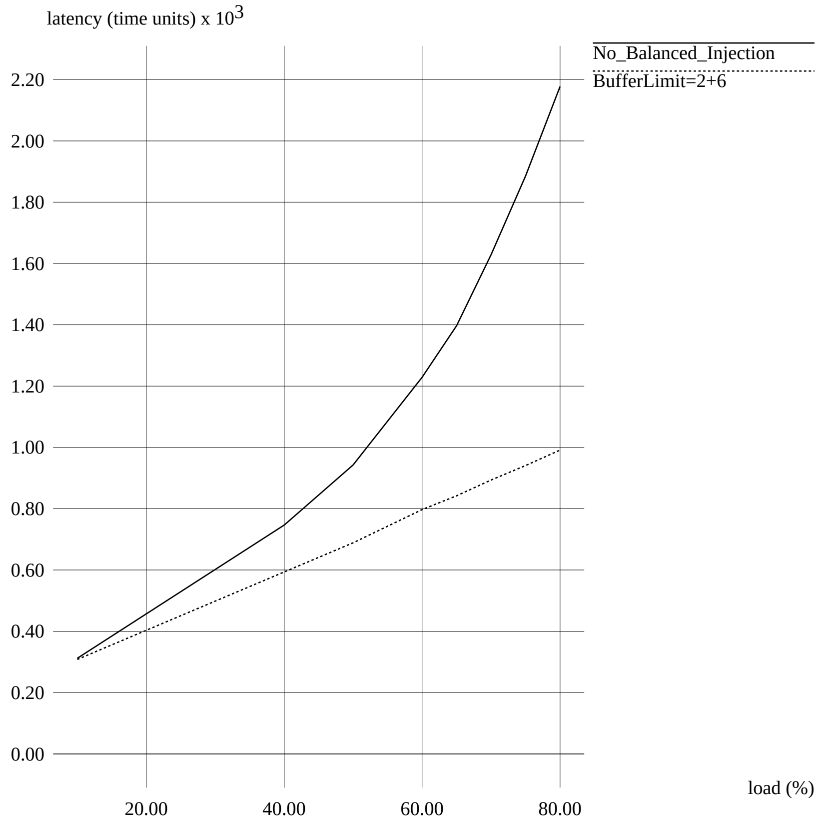

Figure 4: Packet Latency for Balanced Injection Interconnect vs Regular Case (Adaptive Routing, FIFO, Workload *wk-mixture*)

"destination" buffers, we can also refine the function of the PE port to help prevent saturation. Initially the PE port is designed to inject and eject packets in a fair manner. However, in the situation when there are several waiting packets to be ejected from the interconnect, it might be beneficial to give some temporary priority to the eject function.

We introduced two additional parameters: *BufferLimitDest* and *BufferLimitTrans*. If $BufferLimitDest =$ 2 and $BufferLimitTrans = 6$ then the switch accepts a new packet from the PE only if the number of buffers busy with destination packets is less than 2 and the number of buffers busy with transit packets is less than 6. These parameters are related to the architecture of the switch: if there are 2 or more destination packets then the priority will be given to the PE-port ejection direction over the injection direction. If there are 6 or more transit packets then the injection will be throttled because with high probability all the six adjacent ports are busy transferring packets.

Balanced injection is complementary to backpressure. The backpressure mechanism restricts resource utilization of a particular type of packet, so that other types may acquire resources. Balanced injection attempts to keep the overall resource utilization in the interconnect low enough to minimize overall congestion and allow high throughput.

For this experiment, we used the adaptive routing strategy and message-based backpressure flow control, using alpha message scheduling with parameter $\alpha = 8$ and with a workload *wk-mixture*. We compared the interconnect performance and behaviour under the prior assumptions against an interconnect with our modified *BufferLimit* strategy, controlled by parameters set at $BufferLimitDest = 2$ and $BufferLimitTrans = 6$. We will refer to this latter interconnect as a "balanced injection" interconnect.

Figure 4 shows the average packet latency for a balanced injection interconnect against that for the regular interconnect. The packet latency for a balanced injection interconnect is much less, especially under heavy traffic conditions. The explanation of this phenomena is quite clear: we restrict the injection

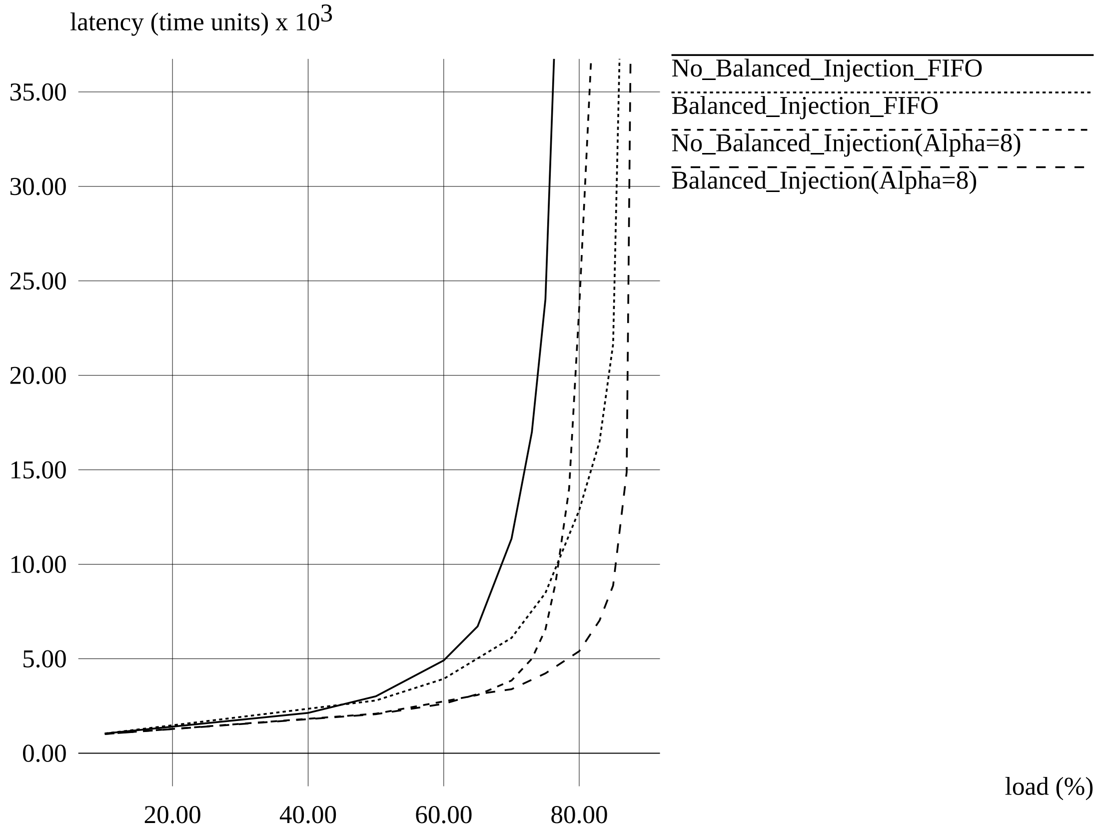

Figure 5: Message Latency for Balanced Injection Interconnect vs Regular Case, Alpha Scheduling ($\alpha = 8$) vs FIFO, Workload *wk-mixture*)

of new packets into the interconnect when we observe local congestion in a switch. We therefore maintain fewer packets inside the interconnect and, as a consequence, transfer the packets through interconnect faster.

Figure 5 compares the average message latency for a balanced injection interconnect with the regular interconnect, both with and without alpha scheduling. One result presented by this graph is surprising: throttling the packet injection did not increase the message latency, despite increasing overall waiting time. Without alpha scheduling, we also observed a 10% increase in throughput using balanced injection over the regular interconnect: instead of handling a maximum of 78% utilization, as the regular interconnect does, the balanced injection interconnect paces the injection to successfully handle an 85% overall utilization. In general, with or without alpha scheduling, the primary impact of balanced injection was to increase the attainable throughput of the interconnect, along with a slight improvement in latency.

This same figure also allows us to evaluate the interaction between alpha scheduling and balanced injection. In general, alpha scheduling significantly improves the interconnect message latency, with an additional slight improvement in attainable throughput. Together, balanced injection and alpha scheduling combine to provide significant improvements in both message latency and attainable throughput.

This performance improvement was consistent across a wide variety of workloads. These results show that the balanced injection strategy is another efficient flow control mechanism that regulates the interconnect under bursty traffic conditions.

7 Conclusion

This paper presented three original and complementary ideas on flow control mechanisms for packet switched interconnects:

- backpressure flow control (both message-based and destination-based);
- alpha message scheduling;
- balanced injection.

Each of these schemes is a different piece of the congestion control puzzle providing only a partial solution to a whole problem of congestion management.

Balanced injection throttles injection based on local measure of the occupied buffers in a node, which reflects the amount of congestion around a particular switch. Over a wide variety of different workloads, and both with and without our other congestion control techniques, balanced injection increases the sustainable throughput of the interconnect by about 10%.

Balanced injection does not solve congestion caused by hot spots, where packets for a particular destination are injected by many different nodes. In this case, the congestion occurs at the destination node and is not quickly reflected by buffer utilization at the injection node. Instead, buffers around the destination fill with packets for that destination, preventing other traffic from getting through. Backpressure ensures that contention for a destination port only consumes a limited number of buffers in each node. This provides a reasonable balance between allowing few packets from a long message to be buffered, and allowing all buffers to be occupied by packets for a particular destination. Our results show that backpressure allows traffic independent of the hot spots to flow virtually unhindered by hot spot traffic.

The final piece of the congestion control puzzle is alpha scheduling. The primary effect of alpha scheduling is to reorder packet injection to optimize message latency, but it also has an important second order effect of interleaving chunks of long messages with shorter messages, decreasing traffic burstiness and randomizing the traffic pattern. Our results show that alpha scheduling significantly decreases average message latency. It complements backpressure by interleaving short messages unaffected by backpressure with long messages that have a higher incidence of blocking due to backpressure, allowing short message latency to be significantly reduced with virtually no effect on long message latency.

The combination of these three mechanisms provides a synergistic solution for congestion control allowing the Fed-X fabric to maintain high throughput in the presence of realistic workloads.

8 References

[Cherkasova94] L. Cherkasova, A. Davis, V. Kotov, I. Robinson, T. Rokicki: How Much Adaptivity is Required for Bursty Traffic? In Proceedings of Seventh International Conference on Parallel and Distributed Computing, Las-Vegas, October, 1994, ISCA, Raleigh, NC, pp 208-213.

[CKR94] L. Cherkasova, V. Kotov , T. Rokicki. On The Effects of Message Scheduling for Packet Switching Interconnect Fabric. HP Laboratories Report No. HPL-94-70, August, 1994.

[Chien93] Chien, Andrew A.: A Cost and Speed Model for k-ary n-cube Wormwhole Routers. In *Proceedings of Hot Interconnects'93, A Symposium on High Performance Interconnects*, 1993.

[Dally89] Dally, W. J. et al.: The J-Machine: A Fine-Grain Concurrent Computer. In *Proceedings of the IFIP Conference*, North-Holland, pp. 1147–1153, 1989.

[Davis92] A. Davis. Mayfly: A General-Purpose, Scalable, Parallel Processing Architecture. J. LISP and Symbolic Computation, vol.5, No.1/2, May 1992.

[Davis94] A. Davis, R. Hodgson, I. Robinson, L. Cherkasova, V. Kotov, T. Rokicki. R2: A Damped Adaptive Router Design. In Proceedings of First International Workshop on Parallel Computer Routing and Communication. *Lecture Notes in Computer Science*, Springer-Verlag, Vol. 853, 1994, pp. 295-309.

[Jain92] Jain, R. Myths About Congestion Management in High-speed Networks. Internetworking: Research and Experience, Vol. 3, pp. 101–113, 1992.

[Novatsky95] Novatsky, Andreas G. et al.: S-Connect: from Networks of Workstations to Supercomputer Performance, Proc. 22nd Int'l Symp. on Comp Arch, June 1995.

[Seitz8] Seitz, C.: The Cosmic Cube. J.*Communications of the ACM*, Vol.28, No.1, pp. 22-33, 1984.

[Stunkel94] Stunkel, Craig B. et al.: The SP-1 High Performance Switch, Scalable High Performance Computing Conference, May 1994.