SCALABLE STRATEGIES FOR ALLEVIATING THE HOL BLOCKING PRODUCED BY CONGESTION TREES IN LOSSLESS INTERCONNECTION NETWORKS

P. Nicolas Kokkalis, Njuguna Njoroge, Ernesto Staroswiecki

EE382C – Interconnection Networks
Department of Electrical Engineering
Stanford University

ABSTRACT

In this paper, we review congestion management strategies that attempt to alleviate or eliminate HOL blocking in lossless interconnection networks. We first look at DAMQ buffers, a method that alleviates HOL blocking within a single n x n switch. We then explore a strategy used in the ATLAS-I switch that manages to eliminate HOL blocking in multistage lossless interconnection networks. Finally, we investigate a new strategy, called RECN, which claims to be more scalable than ATLAS-I. We conclude with a comparison of these techniques.

Keywords: Congestion Trees, HOL Blocking, ATLAS-I, RECN, DAMQ, Buffering

1 Introduction

Congestion management is a very important issue in the design of lossless interconnection networks. In lossy networks, packets are dropped when congestion occurs, preventing the formation of congestion trees [8]. On the other hand, in lossless networks congestion trees may grow very quickly, exhausting network buffer resources and preventing packets from making progress. "A single hotspot may lead a large fraction of the network to total collapse unless some action is taken" [1]. Therefore, interconnection network designers are forced to take into account this issue very seriously.

The problem of managing congestion has been studied for more than 20 years [7] and the solutions proposed belong to one of three classes: proactive, reactive, and HOL blocking (head of line blocking) avoidance or reduction. The proactive strategies are based on a priori knowledge of resource requirements, and reserving them before the transmission. The reactive strategies are based on measuring congestion, notifying the sources, and reducing injection at those sources. In this study, we focus on HOL blocking avoidance/reduction, which we introduce in the following paragraph.

HOL blocking

The most important side effect of the existence of congestion in lossless interconnection networks is HOL blocking. HOL blocking is the situation where congestion leads to blocked packets that prevent the advance of other packets stored in the same queue. This happens even if those packets are not going to cross the congested area. It occurs when a packet in a queue could be forwarded but is blocked by the packet ahead of it that at this moment cannot be forwarded. It is important to realize that the cause for performance degradation in the presence of congestion is HOL blocking and not the congestion itself. Thus, if HOL blocking is eliminated then congestion trees are essentially harmless.

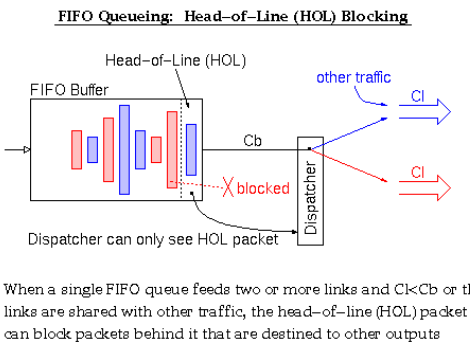

Figure 1 HOL blocking (figure from [5])

Given the fact that in any interconnection network buffer space is finite, the congestion itself may be caused by any non-uniform traffic pattern (i.e. any traffic pattern where any one destination receives more traffic than others). In cases of congestion, it is necessary to block traffic destined for the congested area until resources are no longer saturated. However, performance degradation and underutilization of resources result by the fact that other packets that

are not destined to the congested area, and therefore have resources available for them, are not forwarded since they are blocked behind packets that cannot move forward at this time.

In this paper, we focus on three techniques that attempt to alleviate or eliminate the effects of HOL blocking, namely: DAMQ buffers, ATLAS-I, and RECN. In Section 2, we describe related work. Section 3 elaborates on the paper of Tamir and Frazier that introduced DAMQ buffers, Section 4 analyzes the technique used by Katevenis, Serpanos, and Spyridakis in the design of ATLAS-I switch. Section 5 describes the recent strategy proposed by Duato, Johnson et al. called RECN. In Section 6, we conclude with a comparison of the 3 methods.

2 Background

In this section, we briefly survey various schemes that attempt to eliminate or reduce HOL blocking. The three papers we are reviewing refer to some of these techniques and/or improve upon them.

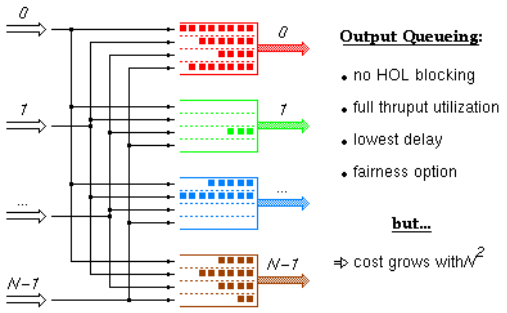

Figure 2 Output Queuing. The total number of memory ports is $O(N^2)$, distributed in N memories. The color of a flit indicates the output port of the fabric that this flit is destined to (Figure from [5]).

OQ Output Queuing. This architecture completely eliminates HOL blocking for single stage interconnection networks for both unicast and multicast traffic, while it offers ideal performance, minimum delays, and no internal blocking. In this organization, each output has a dedicated buffer for every input, whereas every input is directly connected with its corresponding buffer on every output. Its cost is unrealistically and unnecessarily high for large valency switches because its cost grows with the square of the number of links. Interconnection networks try to emulate this ideal architecture at lower cost. Figure 2 presents OQ graphically.

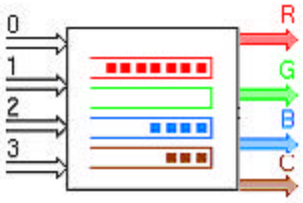

Figure 3 Shared Buffer. The total number of memory ports is O(N). The color of a flit indicates the output port of the fabric that this flit is destined to.

SB Shared Buffer. This architecture similarly to OQ completely eliminates HOL blocking for single stage interconnection networks for unicast traffic. A single central memory is used to store multiple queues arranged as one per output or one per flow. It appears to be more efficient than OQ, since it achieves higher memory utilization, but it requires a memory component with N input ports and N output ports (assuming that N is the number of network inputs and outputs). Some HOL blocking may arise on multicast traffic, due to limited memory bandwidth. Figure 3 presents a shared buffer switch.

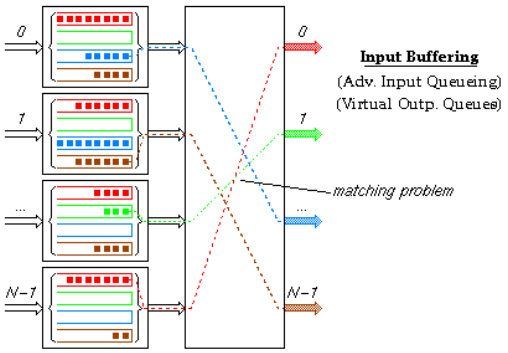

Figure 4 Virtual Output Queues. The color of a flit indicates the output port of the fabric that this cell is destined to (Figure taken from [REF])

VOQ Virtual Output Queues at the inputs. Also known as Input buffering or advanced input queuing. This architecture also applies in single stage networks. It avoids the head-of-line blocking problem of FIFO input queuing by providing multiple logical queues (one per output) in each input buffer.

Its disadvantage is that it introduces a matching problem in the switching topology that has to be solved once on every flit cycle: for each input buffer, choose one of the flit colors present in it, so that no two input buffers have the same color chosen, and so that the number of colors chosen is maximized. This approach is unrealistically expensive for high valency switches. Figure 4 gives an example of VOQ.

MLWC Multi-Lane Wormhole Credit Protocol [6]. Wormhole flow control is used in the design of many multiprocesssor, multistage interconnects. The multi-lane version of the wormhole protocol attempts to boost performance and delay HOL blocking by introducing a number of independent lanes (queues) per link. A new packet has to acquire a new lane before it can be transmitted, which is released one the transmission is over. Multiple packets destined to the same output are allowed to occupy different lanes per link at the same time. Such packet may originate from the same or different sources. Thus, all these packets will be blocked if one of them does, causing Lane Hogging. In other words, HOL blocking is somewhat delayed but it can occur. A disadvantage of this protocol is that packets from the same flow (or destined to the same output) cannot share the same lanes.

LECN Local Explicit Congestion Notification. This method alleviates HOL blocking locally in one stage of a multistage network topology. This technique improves on end-to-end schemes, which are not effective for handling transient congestion because end-to-end schemes respond to slowly and incur more overhead. The method works with congestion notifications between two stages. LECN also allows for a graceful implementation for end-to-end notifications for congestion that endures for longer periods.

3 DAMQs

The paper [3] addresses the design of buffers for n x n switches. The main problem it faces is Head-of Line blocking, and it tries to solve it or alleviate it by proposing a type of buffer called Dynamically Allocated Multiple-Queues buffer (DAMQ), which consists of n fixed-size input queues that are divided into n separate sub-queues, one for each output. The allocation of space for each output within each queue is done dynamically, and therefore the buffer space is used "better" than in other input buffers.

The paper explains the choice of input buffers (instead of output or centralized buffers) and the reasons to focus on n x n switches. It then introduces all the buffers models it will use to compare with the DAMQ. These buffers represent several combinations of multiple or single queues, and dynamically or statically allocated queues.

The DAMQ buffers are implemented as a single physical buffer, organized with several linked lists, one per output plus one for the free buffer space. These buffers require only a single read port and a single write port, but also include some overhead for control and pointer data.

In this work, the authors present simulation results for the DAMQ and the other models, for 2 x 2 and 4 x 4 switches of different types (discarding and blocking), with uniform and "hot spot" traffic patterns. With these results, they conclude that the DAMQ buffers are superior to the others studied, and that they work to alleviate congestion problems.

Paper Details

According to the authors, most large multiprocessor systems use multistage interconnection networks composed of small n x n switches, and therefore the improvement of such switches is of critical importance.

The main problem they try to address in this paper is Head-of-Line (HOL) blocking (although they do not refer to is as such anywhere in the paper). The authors describe the problem stating that "…buffers are finite and have a finite bandwidth. …[This can cause that] packets ready to be transmitted may be blocked behind packets waiting for their output port to free up…"

In order to alleviate this problem (although not solve it completely) they present an efficient buffering scheme for small n x n switches. They use this scheme to develop a Dynamically Allocated Multiple-Queues buffer (DAMQ).

To justify their design choices, the authors start by exploring the buffer design space. They claim that the optimal solution would be a shared central buffer, with all the input ports and all the output ports having access to it (although they also find that with complete sharing a "hogging" output may become a system-wide problem).

However, they find this solution infeasible due to the complexity of the needed buffer, and the multiple read and write ports needed. The next best solution would be to have dynamically allocated output buffers, since they have been proven to provide shorter mean queue length than input queues for equivalent systems. Yet, once more, they find that output queues are infeasible, and that the added complexity of multiple write ports would take up more space than the needed to add a larger input queue.

They settle, then, with designing an input buffer, that tries to capture all the advantages and simplicity of a straightforward FIFO, yet avoids the HOL blocking problem.

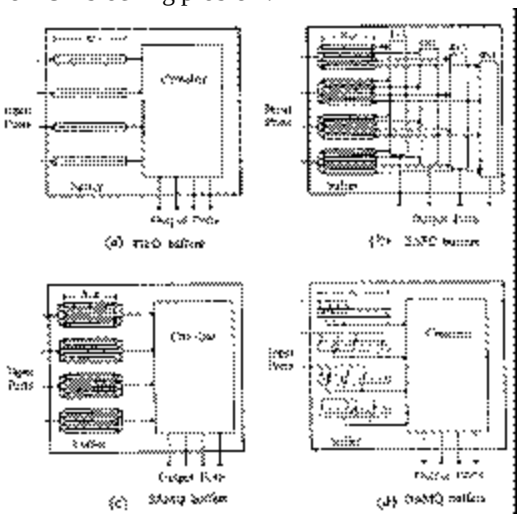

Figure 5 Alternative design of switches with input buffers (figure from [3])

The authors now explain the options they entertained for different input buffers, and that will be used over the paper as a basis for comparison for the DAMQ buffers. The first switch uses the simple input FIFO buffers. The second option uses statically allocated full control (SAFC) buffers, where each input buffer is divided into separate queues for each output, all of them the same size, the buffers have a separate output for each queue, and each output has it's own n x 1 crossbar to route from the input buffers. The next option involves statically allocated multiple queues (SAMQ) buffers. These buffers are also statically allocated as before, but now each buffer has a single output and we have only one n x n crossbar. Finally, we come to the DAMQ; where there is a single output per buffer, and the separate queues are allocated dynamically (see fig. 1). For evaluation purposes, they also model a centrally buffered dynamically allocated (CDBA) switch that is not practically realizable, but can work as a measuring patron.

The authors discuss at length the complexity of efficiently implementing the appropriate routing or pre-routing algorithms required for SAFC or SAMQ switches, and why the DAMQ option is superior for these considerations. In particular, they explain why SAFC or SAMQ switches, since their buffer space is divided into separate queues and cannot be modified, might lead to the blocking of packages while there is still buffer space, since the sought output queue might be full. In some cases, this could lead to even poorer performance than for FIFO switches, since these always use all available buffer space.

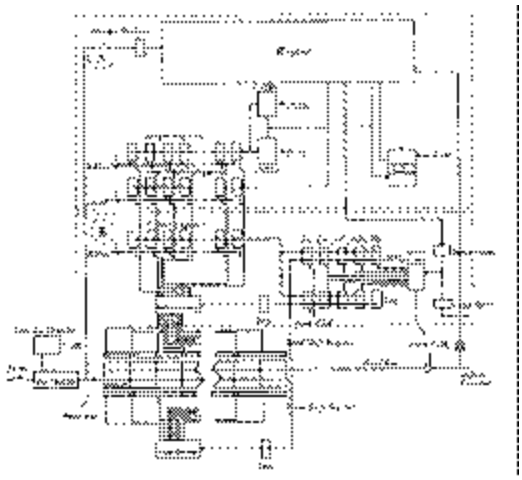

Figure 6 A DAMQ buffer layout (figure from [3])

In order to have multiple dynamically allocated queues in one DAMQ buffer, the queues are implemented as linked lists. The queues are a list of blocks, where each block also contains a pointer to the next block on the list. One aside of importance is the block size: the larger the size, the least flexibility to allocate them, but as we decrease the block size, the overhead needed for the pointers becomes increasingly important. For this paper, the authors consider variable packet sizes from 1 to 32 bytes, and decided to use a block size of 8 bytes.

In order to support this linked list configuration, they require a head and a tail pointer for each queue, and an additional one for the free space. The authors have provided a schematic layout of this switch (see fig. 2), and show that the area overhead needed for this DAMQ implementation (marked B) is not large compared with the entire buffer layout. With some interesting tricks, like having the head

pointer of an empty queue pointing to the first empty block, the authors manage to show that the switches support virtual cut-through, with a turn around time of about 4 cycles.

The largest section of the paper takes care of the evaluation of the DAMQ buffers. All the evaluation is done with simulation, since only the DAMQ switch was realized in hardware.

The first simulations they run were for 2 x 2 discarding switches (switches that discard a packet when it arrives to a full queue) using Markov models. They simulate the functioning of switches using FIFO, SAFC, SAMQ, DAMQ, and CBDA buffers. These simulations are done only on a single switch, and with fixed packet length and with uniform distribution of packet destinations. The results clearly show that the DAMQ buffer outperforms all others for all buffer sizes and input traffic rates (except one combination where the SAFC buffer performs better, this is with a 0.99 input traffic rate, where most buffers are full most of the time). Performance for all the discarding switches is measured as the fraction of packets that are discarded (the lowest the better).

The next simulations are run for a 64 x 64 Omega network constructed from three stages of 4 x 4 switches. Once again, all simulations are run with fixed packet length, and most of them with uniform traffic.

The results of the simulations with discarding switches show again that the DAMQ buffer performs significantly better that all the other ones.

The last set of simulations is for the same network as before, but now with blocking switches (switches that do not transmit until there is room in the downstream buffer). For these switches, performance is measured with a latency v. throughput curve. The results show that at lower throughputs all the switches achieve the very similar latencies. However, the DAMQ buffers can achieve significantly higher maximum throughput before latency become intractable. This performance advantage, however, decreases with increasing buffer size.

On this last simulation, the authors also simulated "hot spot" traffic. This traffic pattern is produced by making 5% of all the incoming traffic to be sent to a single destination. For this traffic pattern, they observe that all the switches saturate at about the same throughput, due to "tree saturation" (the saturation of all the buffers on the path to the hot spot), and no buffer gave an advantage.

In conclusion, the DAMQ buffer provides with an advantage in all the performance measures used by the authors (or at least no disadvantage), and it can be used for small n x n switches for multiprocessor or multicomputer systems.

Discussion

Strong points:

- The main strong point for this paper is the fact that this switch actually built. For this paper, the actual switch was laid out, and the schematic layout is presented. Also, the timing for this switch is calculated from SPICE simulations of several circuit sections.
- The implementation as linked lists is simple and understandable since it is based on a well-known data structure.
- The results truly show that for all cases studied the DAMQ buffer does outperform other buffers. The paper explains very well the principles of these results, justifying them.
- This buffer design should help minimize congestion (although not completely avoid it). This is shown by the results proving that this DAMQ buffer discards fewer packets than the others.

Weak points:

- The simulations ran are very limited. Some of the assumptions are too restrictive and need to be revised. For example, all simulations are run with fixed packet length, while the network that this switch is targeted for uses variable packet length. Also, most simulations use uniform traffic. It would be interesting to see other traffic patterns, for example a permutation traffic pattern. The "hot spot" traffic is defined without explanation as 5% of the uniform traffic destined to a single spot.
- It would be interesting to see how do DAMQ buffers and the other techniques perform with more than one hotspot or with more hotspot traffic.
- A further extension of DAMQ to avoid a single output to "hog" all of the buffer space could alleviate even further the "tree saturation" effect with "hot spot" traffic, and improve the saturation throughput for traffic patterns that produce congestion.
- Some of the architectural options (like centralized buffering or output buffering are dismissed without too much analysis, and hybrid techniques are not explored at all.
- The design is in many ways dated. Current VLSI trends, where silicon area is cheaper, yet

wiring resources are more expensive, might change several of the design decisions made.

4 ATLAS-I

According to Katevenis, Serpanos, and Spyridakis, the authors of [2], ATLAS-I is an actual chip designed at the Institute of Computer Science at the Foundation for Research and Technology in Greece. Its purpose is to be a building block for high-speed multistage interconnection networks, like the ones used on multiprocessors and Networks Of Workstations (NOWs). It uses Asynchronous Transfer Mode (ATM), which is an industry standard for local and wide area networks. The authors claim that if ATM is combined with credit based flow control (a form of backpressure) in the design of the switches, it can outperform multi-lane wormhole flow control, which they claim it is broadly used to interconnect high-speed multiprocessors.

The authors start their description by pointing that ATM's architecture can be considered to equivalent to wormhole's one. For instance ATM cells are analogous to wormhole flits and credit based ATM is analogous to multi-lane wormhole. They claim that their protocol and design outperforms wormhole, especially in the presence of hot-spot traffic and congestion. Also, they argue that their architecture completely solves the HOL blocking in a scalable and efficient manner, using many small internal buffers on each network stage, multiple lanes and backpressure to ensure that the small internal buffers do not overflow.

In brief, their strategy follows the architecture of VOQ attempting to emulate the solution of the matching problem of VOQ in a progressive and distributed manner, while it applies to multistage interconnects. Network traffic is divided to global flow groups and each stage keeps small internal buffers per flow group on each input. The maximum number of flow groups is statically determined by the chip characteristics. Each flow group does not cause any interference or HOL blocking to any other flow groups. Multiple flows can be joined to a single flow group. Internal and external credit based backpressure is used to keep the internal buffers from overflowing and the head flits as close to the outputs as possible, thus minimizing latency. Figure 7 shows a example network using ATLAS-I chips.

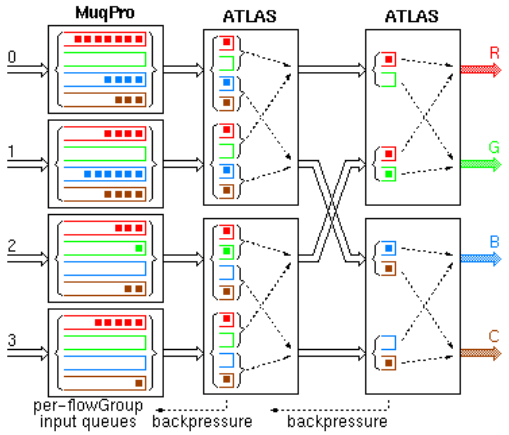

Figure 7 ATLAS-I Flow Control. The color of a flit indicates the flow group that the flit belongs to. Each MuqPro chip simply maintains separate logical queues for each flow group by the use of external memories. The only external memories in the organization are those attached to the MuqPro's, whose cost is the same as in input buffering

Overview of the ATLAS-I switch chip

Each chip is a 16x16 ATM switch with serial links of 622Mbits/sec each that can be configured as a smaller dimension switch with higher bandwidth per link, for instance as 4x4 of 2.5Gbit/sec. The chip operates as a crossbar with a 256-cell shared buffer, 54 logical output queues, including 3 multicast queues. At the same time it handles 3 different priority levels by allocating different buffers on each priority, while it deals with all the specifics needed by the ATM protocol.

ATLAS-I credit-based flow control

ATLAS-I uses a credit-based (backpressure) flow control and therefore no cells are ever dropped because of overflow, since cells are only transmitted when buffer space is known to exist at the receiver: in other words, when the transmitter have a credit for such buffer space. Since single-lane backpressure could introduce HOL blocking the chip uses a multi-lane scheme. Moreover, the required buffer space is not statically allocated, as a naive approach would follow, but dynamically shared among connections.

In ATLAS-I credits operate in flow group granularity. A flow group is a set of connections over a common path through the network; their cells never overtake each other. ATLAS-I links

can each carry up to 4096 flow groups. The chip can merge multiple incoming flow groups into each outgoing flow group and therefore share the lanes and buffer space. In particular, every chip combines flows of the same destination and priority together. Figure 8 illustrates the protocol used by ATLAS-I.

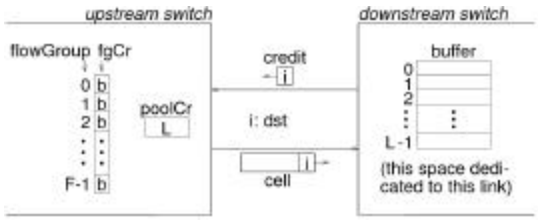

Figure 8 Credit protocol of ATLAS-I

The big picture is that in every input there is a shared buffer, which is essentially managed by the previous stage output. Each output maintains a credit pool (poolCr) to keep track of the available cell spaces in the input buffer of the next stage. Each output also maintains a credit pool per flow group (fgCr). An output needs to hold a credit fgCr(i) and poolCr>1 to submit a cell of flow group i, and consumes one credit of each kind on departure. Each input (downstream switch) forwards the appropriate credits every time a buffer space empties due to a cell departure.

The absolutely necessary buffer space to keep the link 100% working at full throughput, provided that there is no congestion, is equal to the channel bandwidth times the round-trip time (RTT) of data and credits. For the ATLAS-I it was calculated that the RTT is less than a cell time, so one cell buffer space appears to be enough to keep the links fully utilized when data are ready to be transmitted.

The effect of the ATLAS-I backpressure is to *push* most of the output queue cells *back* near the input ports. The *head* cells of the output queues, however, are still close to their respective output ports. The fabric operates like an input-buffered switch where the ATLAS chips implement the arbitration and scheduling function in a distributed, pipelined fashion.

Simulation Results

ATLAS-I performance was widely compared with multi-lane wormhole flow control protocol by the authors. Since ATM and Wormhole are different protocols the authors tried to factor out all of the differences and focus on the flow control algorithms. For instance they simulated same size and payload ATM cells with wormhole flits. Their simulations included both uniform traffic and hot spots. For the packet arrival time they mostly used a Poisson model. The reference topology was a butterfly (banyan) indirect network. According to their results ATLAS-I outperforms wormhole. The most important results acquired on a simulator of a 64x64 butterfly network are presented below.

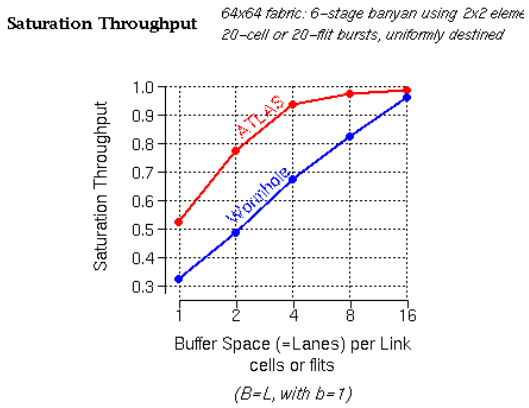

Figure 9 Buffer Space Comparison

As Figure 9 suggests in terms of buffer space in ATLAS-I, around 8 to 16 cells per incoming link suffices for the outgoing links to reach full utilization. They use the low priority traffic in this graph, which fills in the remaining capacity unused by higher priority traffic. The ATLAS protocol (red line) performs better than the multilane wormhole protocol.

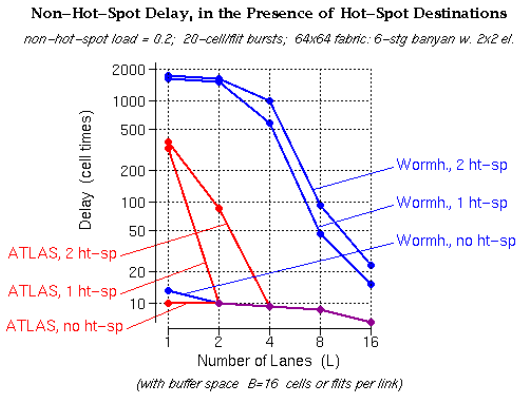

Figure 10 Latency

Figure 10 shows that with the ATLAS protocol, when the number of lanes is larger than the number of hot-spot output ports of the fabric (1 or 2 ports --upper two red curves), the delay to the non-hot-spot outputs remains unaffected by the presence of hot-spots (it is the same as when

there are no hot-spot outputs --bottom red curve). This is precisely the ideal desired behavior for hot-spot tolerance.

Discussion

In conclusion, ATLAS-I team claims that their design outperforms multi-lane wormhole credit protocol because it uses less buffers for similar performance and it has better tolerance to bursty and hotspot traffic. Their argument is mainly based on the fact that identically destined traffic is confined to one lane and that each lane can be shared by multiple packets.

In the following discussion, we present some of the strong and weak points, we think, of this paper.

Strong points:

- *Convincing and efficient.* It seems that the idea of small internal buffers combined with a well-calculated backpressure to keep the maximum throughput of links is a powerful one. The fact that the protocol was implemented as an actual chip is obviously an additional convincing factor. The presence of multiple lanes eliminates HOL blocking.
- *No lane hogging*. The fact that is technique does not allow data going to the same destination to occupy more than one lane in the intermediate switches is very good. Moreover, we believe it is the major advantage over wormhole protocol.

Weak points:

- *Deterministic Routing and indirect networks.* Everything was calculated assuming deterministic routing and indirect networks. This is expectable though since ATM is connection oriented, but it would be nice to see how this protocol would perform under adaptive routing, on direct networks, or non-ATM implementations.
- *Internal flow group interference*. The protocol merges flows going to the same destination into a single flow group. Although this is essential to efficiently manage buffer space and minimize the number of lanes, packets of the same flow group never overtake each other. This can introduce unfairness if different sources are injecting packets to the same source at different rates. For instance, sources that inject at lower rates are penalized from sources that inject at higher rates.
- *Per destination as opposed to per congested link*. In the cases where internal links are congested, many destinations appear to be congested. This approach allocates a separate buffer for each endpoint of the network. Where as in alternative schemes, a single buffer space is allocated.
- *Credit per cell.* Another noticeable point of this approach is that a credit is transmitted for every cell. A more efficient scheme would be to transmit congestion notifications, whenever congestion occurs.

5 RECN

The paper, "A New Scalable and Cost-Effective Congestion Management Strategy for Lossless Multistage Interconnection Networks"[ref], proposes "a congestion management strategy for lossless multistage interconnection networks that scales as network size and/or link bandwidth increase". In particular, Duato, Johnson et al. assert that their solution can eliminate HOL blocking due to the formation of congestion trees. Moreover, they suggest that by being able to eliminate HOL blocking, the cost and power constraints imposed on network systems are easier to meet because their techniques make more efficient use of existing links (i.e. they do not need to be overdimensioned).

Before delving into the details of their scheme, the paper considers other research that pertains to congestion management. The authors cite three main categories of congestion management solutions – proactive, reactive and HOL blocking minimization/elimination strategies. They discount the merits of first two techniques: the proactive method requires prior knowledge of the resource constraints in order to reserve resources. The authors are not convinced that this technique is effective because prior knowledge of the network state is not always available and resource reservation incurs substantial overhead.

On the other hand, reactive methods detect congestion and notify the sources. The authors are unimpressed with this reactive approach of congestion management because they assert that reactive techniques consume significant overhead with the transmission of the notifications. Moreover, such reactive schemes are too slow to respond, especially as the network diameter increases. Therefore, these reactive schemes scale poorly.

Finally, the authors argue that HOL blocking elimination techniques, which they claim are a subclass of reactive schemes, have been more effective in congestion management. The authors highlight the advantages and tradeoffs of some notable HOL blocking elimination schemes, like virtual output queues (VOQs), dynamically allocated multiqueues (DAMQs) and Credit Flow Controlled ATM. In particular, the authors point out that VOQs scale poorly because the number of storage resources increases quadratically with network size, DAMQs do not work with multistage networks and Credit Flow Controlled ATM schemes consume too much buffer storage.

After highlighting the limitations of prior schemes in congestion management, Duato, Johnson et al., describe their technique, which they refer to as regional explicitly congestion notification (RECN). They claim that RECN works by detecting and explicitly notifying congestion throughout the region that is affected by a congestion tree. The authors arrived at their scheme by observing, through simulations, that packets from non-congested traffic do not significantly interfere with each other when they are placed in the same queues. The ramifications of this observation is that the number of queues required to handle HOL blocking can be reduced substantially as compared to VOQs mechanisms. In fact, the authors noted that there is a significant amount of spatial and temporal locality in the flow of packets. Thus, packets from non-congested nodes can be placed in the same queues, which increases queue utilization. Another corollary of the observation is that congested traffic and non-congested traffic should be segregated into separate queues. From their observations, they developed RECN to use less buffer storage, which they argue scales well as the network scales up, both for link bandwidth and/or number of nodes (switches and/or terminal nodes). Furthermore, because the notifications are regional, RECN does not suffer the propagation delays experienced in the aforementioned reactive techniques. In this manner, RECN also scales well because it handles congestion regionally and it is not very dependant on the size of the network.

Prior to detailing how they implemented RECN, Duato, Johnson et al. constrained their network topology to multistage networks with switch-architectures based on internal multiplexed crossbar with input and output buffers. They chose to focus on this class of topologies and switch-architectures because they consider them representative of present commercial networks. Additionally, the authors restricted their discussion to deterministic routing, which they also argue is representative of commercial network routing schemes.

RECN has several stages of operation, the first being congestion detection, then congestion notification and queue allocation and finally, queue deallocation when the congestion subsides. For congestion detection, when an output port has become congested, by reaching a certain threshold of packets in its queue, it propagates a notification to the input ports the first time they route a packet to the congested output port. The output port tracks which inputs ports have been notified to avoid resending notifications, thus saving bandwidth. Additionally, when a notification is sent, it carries a token. Only a leaf input port of a congestion tree can be the owner of a token.

When the input port receives the notification, a *set aside queues* (SAQs) is locally allocated for the packets heading to the congested output. A *content addressable memory* (CAM) is also allocated to store path information associated with a SAQ. Consequently, all incoming packets requesting to be routed to the congested output port are stored in that SAQ, preventing the blocking of the other packets destined to a non-congested output port. When a SAQ reaches a certain threshold of packets being buffered, it sends notification upstream. This process propagates upstream, if necessary, up to the source nodes.

Deallocation of the SAQs occurs in a similar manner to allocation. RECN uses tokens to tag the leaves of the congestion trees, so when the SAQ of a leaf becomes empty, it deallocates its SAQ and CAM, then sends a notification downstream. This process propagates downstream. The non-leaf ports need to wait for all tokens from its leaves (if it is part of multiple congestion trees) for it to completely deallocate.

Having described how RECN works, Duato, Johnson et al. evaluated it using an event driven simulator on three network configurations with various traffic patterns. The evaluation pitted RECN against VOQ at the network level (VOQnet), VOQ at the switch level (VOQsw), 4 queues at the ingress and egress ports (4Q) and 1 queue at the ingress and egress ports (1Q). The authors claim that VOQnet is used to demonstrate the maximum achievable performance for each test (i.e. complete elimination of HOL blocking), where 1Q demonstrates the worst achievable for

performance for each test. Also, in the simulations, RECN used a credit-based Xon/Xoff (stop/go) flow control for the 8 SAQs, each at the ingress and egress ports.

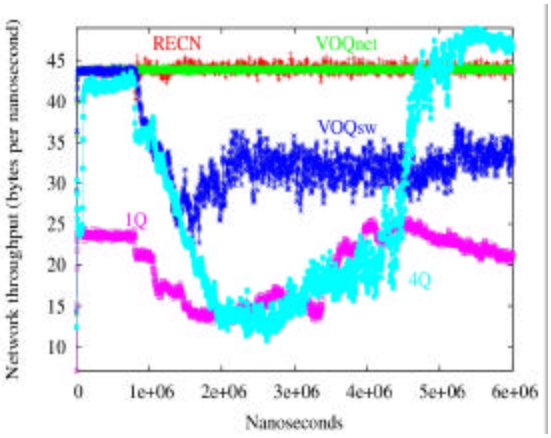

Figure 11 Overall Performance (figure from [1])

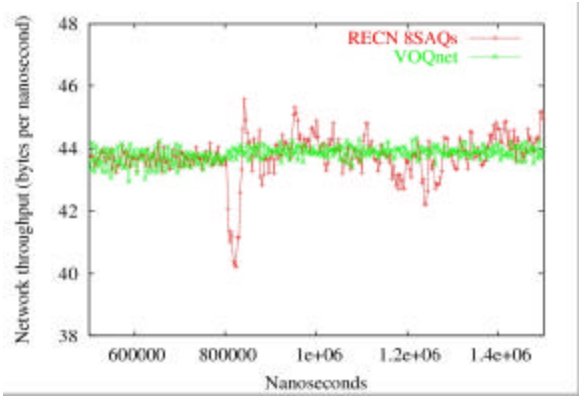

Figure 12 Start of Congestion (figure from [1])

Overall, the simulations results demonstrate that for all the traffic patterns and network sizes evaluated, RECN maintained performance like VOQnet, as Figure 11 represents the overall trends. In Figure 12, the authors zoomed into the period where the congestion started in order to compare VOQnet and RECN. As the graph illustrates, RECN drops to 90% of ideal total throughput for less than 50µs, which is considered a quick recovery by the authors.

Duato, Johnson et al. also evaluated how the usage of the SAQs varies with the traffic patterns and network configurations. They discovered that for most of the tests, the maximum of number of SAQs used per port did not exceed 8 (the number they allotted for each ingress and egress ports). In their analysis, the authors observed that the usage of SAQs is primarily correlated to the number of concurrent overlapping congestion trees, rather the size of the network. Thus, the paper suggests that if one has prior knowledge of how many congestion trees may overlap at the hottest spot in the network, she can use that as a lower bound for the number of SAQs per ingress and egress ports.

Based on the results, the paper asserts that RECN significantly outperforms systems without HOL blocking support and outperforms systems that implement VOQs at the switch level (VOQsw). Furthermore, the author claims that RECN achieves performance "identical" to ideal VOQ in fabric, except for a small initial period during the formation of the congestion tree, but at a small cost compared to VOQs (VOQnet in the graphs) implemented at the fabric level. Finally, the paper concludes that RECN also scales well.

Discussion

This method is based on principles similar to the one used in ATLAS-I and is inspired by LECN. It is based on the observation of the authors that uncongested flows can easily share the same buffers and queues without any considerable interference. Only congested flows need to be assigned to parallel queues (the SAQs), so that the flits of these congested flows that are blocked do not block flits of other flows. Even though this implementation necessitates some content addressable memories (for the SAQs), which are expensive, it appears to be more scalable than the ATLAS approach. Moreover, this approach allocates a single SAQ per congestion tree at each link regardless of packets final destination. This is beneficial when the root of the tree an internal link of the network since packets crossing this link may have different destinations. (ATLAS assigns a different queue per destination). An additional advantage of RECN over ATLAS-I is that backpressure information is communicated only the moment congestion arises and the moment it falls. On the other hand, ATLAS-I switches communicate credits for every cell departure.

Strong points:
In summary, the paper presents RECN, a compelling solution for congestion management to thwart HOL blocking. In particular, the following bolstered the quality of the paper:
– The authors effectively demonstrate that power and cost constraints of network systems are positively influenced by effectively managing congestion. In particular, the paper

pointed how effective HOL blocking techniques can efficiently use the resources of the network (link bandwidth, need for fewer switches, etc.). In opening the paper with the issue of power and cost constraints, the authors present an interesting case for motivating their research.
- The paper constrains the analysis of RECN to deterministically-routed, multistage networks because the authors claim these types of network systems are representative of most commercial systems, which we agree is a reasonable assumption to make. Therefore, by constraining their analysis, RECN is easier to explain and to implement.
- Finally, from a theoretical standpoint, the savings in buffer storage and performance makes RECN standout to other schemes like VOQs.

Weak points:

While the authors presented some strong arguments for RECN, there are aspects of the paper with which we disagree or require further investigation/refinement. In particular, the following retracted from the quality of paper:
- While the Duato, Johnson et al. made sound qualifications of the important role that HOL blocking elimination plays in reducing power and cost constraints, they did not quantify their claims. Thus, it is questionable how much influence HOL blocking elimination can exert on power and cost limitations. This would be an interesting extension of their research since the paper makes strong claims.
- RECN performs well for the various network configurations and traffic patterns chosen in the paper's simulator. However, it seems that these traffic patterns were selected such that the maximum number of overlapping congestion trees is no more than 8, the number of SAQs per port. While the authors concede that the maximum number of overlapping congestion trees in a traffic pattern places a lower-bound on the number of SAQs, they do not discuss how easy/difficult it is to determine this figure à priori you set-up your network system. In other words, how does a network system manager practically determine that her storage-area-network typically has a certain number of overlapping congestion trees? This concern is worrisome because of the case when the network manager uses too few SAQ—once you run out of SAQs, the network looks like it has one queue ingress and egress, which is essentially the worst case scenario (as illustrated by 1Q in the graphs).
- The authors did not discuss the design complexity of RECN. Furthermore, there are unanswered questions such as how the complexity changes as you vary the number of SAQs, size of the buffers, etc. These tradeoff considerations point to our earlier concern of how effective HOL blocking elimination is for power and cost reduction. If RECN has a modest impact on these factors, but is overly complicated to implement, then it may behoove us to use alternative (like VOQs) HOL blocking elimination techniques.
- It is worth noting that the reference about VOQ was probably wrong. Additionally, they claim that DAMQ buffers can only be used in direct networks with little substantiation, where as the authors of [3] claim that DAMQ buffers can be implemented in both direct and indirect topologies.

6 Conclusion

It was interesting to review the three papers and analyze how they chose to address the problems that are caused by head of line (HOL) blocking. In their respective solutions, the papers differed on the degree to which they claimed to solve the problem and the level of granularity they employed. For instance, both ATLAS-I and RECN authors assert that their solutions eliminate HOL blocking at the onset of congestion while the authors of the DAMQ paper suggest their solution alleviates HOL blocking because it efficiently utilizes the buffer storage space. Consequently, the DAMQ buffer solution works on the granularity of a switch such that a DAMQ-based switch can be used in a network containing non-DAMQ switches. In contrast, both RECN and ATLAS-I necessitate a network-level compliance amongst the switches so that their respective flow control schemes are adhered. In addition, RECN, which is newer than ATLAS, adopts some techniques from ATLAS' implementation and improves upon them. Consequently, RECN seems to be more efficient in some aspects, particularly, with the notification schemes it employs. For example, for RECN backpressure information is only communicated the moment congestion arises and the moment it falls, which with ATLAS-I, the switches communicate credits for every cell departure.

Another distinguishing difference between the three papers was that ATLAS-I and DAMQ switches have been implemented in silicon, which lends more credence to the feasibility of their schemes. While simulations are commonly used in academic research, having working hardware is a strong selling point of one's research.

Finally, the common thread that links these three papers is that the authors identify that HOL blocking is an important issue in modern interconnection networks, especially as the power consumption and costs continue to increase.

References

[1] J. Duato, I. Johnson, J. Flich, F. Naven, P. García, and T. Nachiondo, "A New Scalable and Cost-Effective Congestion Management Strategy for Lossless Multistage Interconnection Networks", *in Proc. 11th Int. Symp. on High-Performance Computer Architecture*, pp 108-119., Feb. 2005.

[2] M. Katevenis, D. Serpanos, E. Spyridakis, "Credit-Flow-Controlled ATM for MP Interconnection: the ATLAS I Single-Chip ATM Switch", *in Proc. 4th Int. Symp. on High-Performance Computer Architecture*, pp. 47–56, Feb. 1998.

[3] Y. Tamir and G. L. Frazier, "Dynamically-Allocated Multi-Queue Buffers for VLSI Communication Switches", *IEEE Trans. on Computers*, vol. 41, no. 6, June 1992.

[4] V. Krishnan and D. Mayhew, "A Localized Congestion Control Mechanism for PCI Express Advanced Switching Fabrics", *in Proc. 12th IEEE Symp. on Hot Interconnects*, Aug. 2004.

[5] M. Katevenis, "CS534 Packet Switch Architecture", class notes. http://archvlsi.ics.forth.gr/~kateveni/534/

[6] W. J. Dally, "Virtual-channel flow control," *IEEE Trans. on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, March 1992.

[7] G. Pfister and A. Norton, "Hot Spot Contention and Combining in Multistage Interconnect Networks", *IEEE Trans. on Computers*, vol. C-34, pp. 943–948, Oct. 1985.

[8] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End To End Congestion Avoidance on a Global Internet", *IEEE Journal on Selected Areas in Communication*, vol.13, no. 8, pp. 1465–1480, Oct. 1995.