The Interaction between Virtual Channel Flow Control and Adaptive Routing in Wormhole Networks *

Swaminathan Ramany and Derek Eager
Department of Computational Science,
University of Saskatchewan,
Saskatoon, SK, Canada S7N 0W0
{swami, eager}@cs.usask.ca

Abstract

Multiprocessor interconnection networks based on low dimensional mesh or torus topologies and employing wormhole switching have become increasingly popular. Two concepts that have been proposed to improve the performance of such networks are Virtual Channel Flow Control (VCFC) and adaptive routing. Some previous studies have, however, found the latter technique to yield disappointing performance under uniform traffic patterns, while in this work we show that the former technique may result in degraded performance under certain non-uniform traffic patterns.

This paper studies the interaction between VCFC and adaptive routing. It is shown that each of these techniques may yield unsatisfactory performance only when used in isolation, and that through their appropriate combination uniformly improved performance can be achieved. The consistent performance improvement and the magnitude of the performance improvement in some cases offers compelling motivation for implementing a combination of VCFC and adaptive routing in future machines.

1 Introduction

The distributed memory MIMD multiprocessor, consisting of a collection of processor-memory pairs interconnected by a network, has become a popular parallel processing architecture. Interconnection networks appropriate for such systems can be broadly classified into two types: direct networks and indirect networks. In indirect networks the processing nodes are connected by intermediate switches to other nodes. In direct networks each node is connected directly to a few other neighboring nodes. The exact connection is defined by the topology of the network. Direct networks have become more popular in recent times since they can exploit locality in communication patterns, and are thus expected to more easily scale to support large systems. Owing to design and cost constraints low dimensional mesh and torus networks are favored by many researchers [1]. An example 2-dimensional mesh is shown in Figure 1; note that we assume here that each bidirectional link has associated with it two (uni-directional) physical channels.

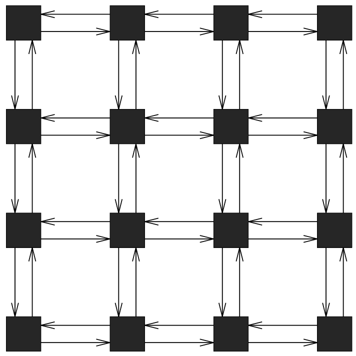

Figure 1: A 2-dimensional mesh of size 4×4

The most popular switching technology at present is wormhole switching [7] (often referred to as wormhole routing). In wormhole switching, a node in the network starts forwarding a message as soon as the first flit (unit of transmission on the physical link) is received and the required output channel is free, without waiting for the entire message to be received. Wormhole switching requires less buffer space and has lower communication latency compared to other switching techniques. In this paper networks employing wormhole switching are referred to as "wormhole networks". Two different concepts that have been proposed for performance enhancement in wormhole networks are "adaptive routing" and "Virtual Channel Flow Control"(VCFC).

Virtual channels [7] were initially proposed for deadlock avoidance in torus networks. A virtual channel is a "logical" channel with its own buffer, control and data path, that shares the same physical channel with other virtual channels. In the Virtual Channel Flow Control concept, the multiple virtual channels that are implemented on each physical channel are employed as multiple "lanes" with complete freedom in routing choice among them [6]. Owing to the resulting ability to allow multiple messages in the same physical channel, and thus allow messages to bypass blocked messages, higher throughputs can be achieved.

Most existing networks use deterministic routing. Deterministic routing allows just a single physical path from the source to destination. Thus there is a loss in flexibility which can lead to poor performance. Adaptive routing allows the density of network interconnection to be taken

*This research has been supported in part by grants from the Natural Sciences and Engineering Research Council of Canada.

Appeared in the Eighth ACM Interantional Conference on Supercomputing, Manchester, England, July 11-15, 1994

advantage of. Messages have several routing choices at each node and the path choices are made based on the local information available at the node. Adaptive routing can help in increasing the achievable network throughput [9, 10, 11]. Many new routing schemes have been proposed in recent research [3, 5, 8, 10, 12, 13, 16]. A detailed survey of various routing schemes in wormhole networks can be found in [14].

One disadvantage of adaptive routing in wormhole networks, however, is that it increases the likelihood of deadlocks. Thus, preventive measures have to be taken in the routing algorithm to avoid deadlocks. Deadlocks can often be avoided by partially restricting adaptivity. Generally, adaptive algorithms also use multiple virtual channels on each physical channel, for deadlock avoidance.

Some previous studies have shown that adaptive routing algorithms perform badly for uniform traffic patterns [10, 11]. Some researchers [4] also claim that adaptivity, full or partial, is not necessarily a benefit in wormhole networks and they show that certain adaptive algorithms can perform badly for both uniform and non-uniform traffic patterns. These results raise a certain amount of doubt as to whether adaptive algorithms would actually be implemented in future commercial machines. In this paper we also demonstrate that adding more virtual channels for VCFC alone can severely degrade performance for certain traffic patterns.

Thus adaptive routing by itself performs badly compared to deterministic routing under certain circumstances. Also, employing VCFC alone can degrade performance for certain traffic patterns. This paper investigates whether the problems of both these schemes can be mitigated by judiciously combining them. In this work we vary the amount of VCFC employed (by adding virtual channels) and study its effect on the performance of adaptive routing. A two dimensional mesh network is assumed for the purpose of this study.

This paper demonstrates that adaptive algorithms, when coupled with VCFC, can yield uniformly better performance for both uniform and nonuniform traffic with large performance improvements when the traffic is non-uniform. We also show that these benefits can be achieved at a reasonable cost in terms of number of virtual channels required per physical channel and algorithm complexity. We believe that adaptive routing is indeed beneficial, although its real benefit can be felt only if it is combined with VCFC.

The rest of this paper is organized as follows. In Section 2, we review required background. In Section 3, we describe the research methodology followed in this study. In Section 4, we analyze our simulation results and in Section 5 we present the conclusions and suggest some avenues for future work.

2 Routing in Wormhole Networks

A good routing algorithm should have freedom from deadlock, low latency, high throughput and ease of implementation in VLSI [11]. Deadlocks can be avoided by assigning each channel a unique number and ensuring that all paths include channels in strictly monotonic order [7]. This avoids circular waiting for resources and thus prevents deadlocks. Usually the numbering of the channels is based on the channel dimension. That is, the numbering is done in such a way that each message must be routed in one dimension until it arrives at the proper coordinate in that dimension before changing over to the next dimension. One such algorithm is the XY routing algorithm [18], and is applicable to 2D meshes. The XY routing algorithm sends messages in the X-dimension first and then in the Y-dimension.

Deterministic algorithms sacrifice adaptiveness so as to more easily ensure deadlock freedom. This can lead to a poor performance in terms of achievable throughput for many traffic patterns. To overcome this many new adaptive algorithms have been proposed in recent times. These adaptive algorithms typically avoid deadlocks through the use of extra virtual channels, and/or by partially restricting adaptivity. An algorithm is said to be minimal, if the set of valid paths allowed by the algorithm include only shortest paths from the source to the destination. A minimal adaptive algorithm is termed fully adaptive, if all shortest paths to the destination can be chosen. A minimal partially adaptive algorithm allows only a subset of the shortest paths to the destination. Some specific adaptive algorithms are discussed briefly in the reminder of this section.

Linder and Harden [13] extend the virtual channel concept to multiple virtual communication networks that allow a minimal fully adaptive algorithm for n-dimensional torus and mesh networks. The Linder and Harden algorithm for 2D meshes requires two virtual channels per physical channel in one of the two dimensions (either X or Y). Supposing that the extra channels are permitted in the Y dimension, then the algorithm routes all "east" bound messages using the +X physical channels (left to right) and one set of Y virtual channels and the "west" bound messages in the -X physical channels (right to left) and the other set of Y virtual channels. Jesshope, Miller and Yantchev [12] have proposed a similar algorithm for 2D meshes which partitions the network into 4 subnetworks. In this algorithm, messages traveling north-east, north-west, south-east and south-west are routed in separate virtual networks.

Dally and Aoki[9] propose two adaptive algorithms that are based on the concept of dimension reversals. Dally and Aoki define "dimension reversal" as a departure from the conventional dimension order routing. The number of virtual channels required per physical channel for the "static" dimension reversal algorithm is the number of dimension reversals required for the desired level of adaptivity minus one. In the case of their "dynamic" dimension reversal algorithm, the virtual channels are partitioned into two nonempty classes: adaptive and deterministic. Messages initially use only adaptive channels and can suffer any number of dimension reversals until they get blocked by a message which has suffered a lesser number of dimension reversals, after which the message has to be routed using the deterministic channels to the destination. The Dally and Aoki algorithms require a more complex router. Also, these algorithms require a header field in each message to record the number of dimension reversals suffered. The number of virtual channels assumed in their experiments (16 per physical channel) is also quite high.

Chien and Kim [5] propose a partially adaptive algorithm called the planar adaptive algorithm that requires only a constant number of virtual channels independent of network size and dimension. This algorithm allows messages to be routed on a series of two-dimensional adaptive planes. Within each adaptive plane, messages may use any channel leading to the destination. In the presence of faults, the planar adaptive routing can be extended to allow deadlock and livelock free non-minimal routing.

Berman et al. [3] propose an algorithm called *-Channels for torus networks which is fully-adaptive, deadlock and livelock free and requires a very moderate amount of resources in routing nodes. They divide the virtual channels into star and non-star channels. Messages move in the star channels for doing dimension order routing. The non-star channels are used to take transitions not allowed by dimension order routing. The *-Channels algorithm for 2 dimensional torus networks requires 3 virtual channels per bidirectional link (i.e., per pair of (uni-directional) physical channels) in the X dimension and 5 virtual channels per bidirectional link associated with the Y dimension. Similar algorithms have also been proposed recently by Su and Shin [17] and Duato [8].

The *-Channels algorithm can be easily modified to suit two dimensional meshes. In this context, just two virtual channels are required for each physical channel in the Y dimension and one virtual channel for each physical channel in the X dimension. Out of the two channels in the Y dimension, one channel is the star channel which is to be used only for dimension order routing, the other is the non-star channel which can be used for both dimension order routing and non-dimension order routing. If additional channels are added to the topology (for VCFC), they can be classified as non-star channels as the single star channel in the Y dimension is enough to guarantee deadlock freedom. [1] Thus on any physical channel in the X dimension, the number of virtual channels available for any message is the total number N of virtual channels, while on any physical channel in the Y dimension, at least $N-1$ virtual channels are available for any message, regardless of its routing. The number of virtual channels on each physical channel to choose among in the *-Channels algorithm compares favorably against that in the Linder and Harden algorithm. The Linder and Harden algorithm has N choices in the X dimension and only $N/2$ in the Y dimension due to the partitioning of the network into subnetworks, when VCFC is employed with N virtual channels per physical channel. (See Figures 2 and 3.)

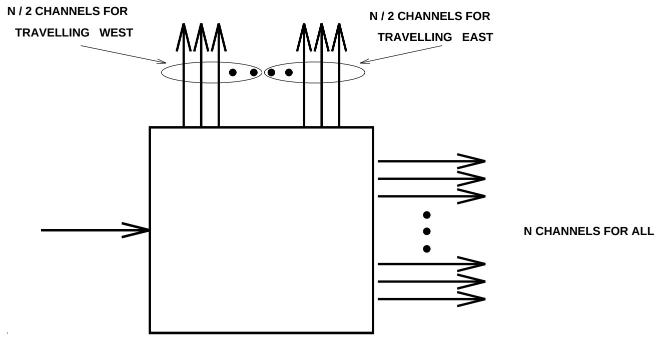

Figure 2: Channel choices available for the Linder and Harden Algorithm

All the above algorithms avoid deadlocks by using multiple virtual channels per physical channel, in some fashion, and restricting their usage. The turn model algorithm proposed by Glass and Ni [10] allows development of adaptive routing algorithms for any given topology (including topologies extended through the addition of virtual channels) without adding any further virtual channels. A turn is defined as a change in dimension. The basic concept of the turn model algorithm is to prohibit the least number of turns for every cycle that can result in a deadlock.

Among the fully adaptive algorithms discussed in this section, the hardware requirements and the algorithm complexity for the Linder and Harden and *-Channels algorithms are comparatively low, for the case of a two dimensional mesh network. These algorithms are quite simple to implement and the router complexity is not likely to be very high, in this context. Also, these two algorithms require (at the minimum) only two virtual channels per physical channel in one dimension and one virtual channel per physical channel in the other.

We have chosen the Linder and Harden algorithm and the *-Channels algorithm as two representative adaptive algorithms for the performance study in this paper.

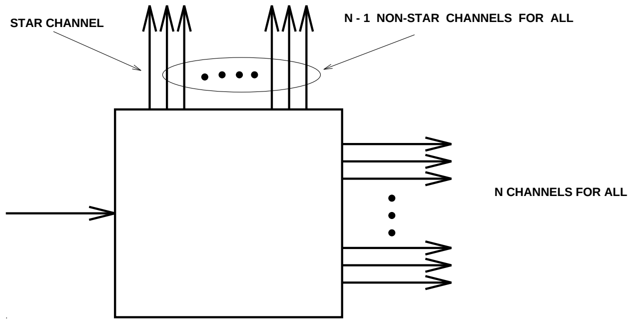

Figure 3: Channel choices available for the *-Channels Algorithm

3 Simulation Models

To evaluate the performance of the adaptive algorithms a network simulator was built. The simulation was performed at the flit level. The simulator has the flexibility to support various topologies, routing algorithms and traffic patterns.

Architectural Details

A two dimensional mesh network was assumed in the experiments whose results are reported here. Each pair of neighboring nodes in the mesh was assumed to be connected by a pair of (uni-directional) physical channels. The number of virtual channels per physical channel was varied during the experiments. Each router was assumed to be connected to the processor by means of a single consumption channel and a single injection channel, each with the same number of virtual channels as the other physical channels in the network. The flit transfer time between two nodes was set to be 1 clock cycle.

Routing and Transmission Policies

The two constituents of the routing policy are the 'input selection' policy and the 'output selection' policy [11]. The input selection policy is applied when the header flits of multiple messages contend for the same channel and the output selection policy is employed when the header flit of a message has to choose between multiple channels. We assume here that the output selection policy is applied before the input selection policy. For most of the simulations, the input selection policy was chosen as 'random select' and the output selection policy was set as 'minimum congestion' [9]. The minimum congestion policy picks that physical channel on which the message can be routed that has the most free virtual channels. In the case of a tie, the channel in the X dimension is chosen. The physical channel transmission policy, which is employed for multiplexing virtual channels over

[1] The proof that this modification does not introduce deadlocks is based on the fact that the *-Channels algorithm is proven to be deadlock free, if the channel dependency graph of the star channels is deadlock free [3]. Since we are enforcing only dimension-ordered, deterministic routing (with laning) among the star channels this condition is satisfied and hence the routing is deadlock free.

each physical channel, was set to be 'work conserving round robin' among the flits waiting to be transferred. We found that this policy has a better performance compared to other policies such as 'oldest flit first' and it is also straightforward to implement. The impact of some alternative policy choices is examined in section 4.4.

Workloads

A processor generates messages at a rate α whenever any virtual channel on its single injection channel is free. The size of each message was set to be 20 flits.

The different traffic patterns that we considered were 'uniform' traffic, 'matrix transpose' traffic and 'hot spot' traffic. With the uniform traffic pattern, a node sends each message to any other node with equal probability. Since the traffic in multiprocessor systems usually has some hot spots [15], a mild hot spot traffic pattern was also simulated. Six random nodes were fixed as hot spot nodes and each received three times the amount of traffic as did each of the other nodes. A type of matrix transpose traffic was selected as a representative of highly non-uniform traffic. In matrix-transpose traffic, the processor in column i and row j sends messages only to the processor in column j and row i. It was also assumed that the processor in column i and row i (i.e., on the diagonal) sends messages only to the processor in column $n-i+1$ and row $n-i+1$.

Performance Metrics

We were interested in two parameters, normalized network throughput, T, and the average message latency, l. The normalized network throughput was defined as

$$T = N * L/n^2$$

where N, L and n^2 are the number of messages delivered per unit time, worm length (message size in flits), and the number of network nodes, respectively. Since at most n^2/L messages can be delivered per unit time (given that each node can consume at most one flit in a clock cycle), $N*L/n^2$ gives the network throughput as a fraction of the maximum possible throughput.

Latency was measured by calculating the difference between the time when a message was created until the time the last flit of the message was accepted at the destination. Average latency was measured by averaging over all messages between all source-destination pairs.

4 Experimental Results

In our principal experiments that are reported here, the size of the two dimensional mesh was set to be 12 × 12. Experiments were also performed for other mesh sizes, but the results of these experiments did not differ qualitatively from the results for the 12 × 12 mesh. Simulation experiments were performed for the deterministic XY algorithm and the Linder and Harden and *-Channels adaptive routing algorithms, with different numbers of virtual channels per physical channel. First, the minimal configuration (for the adaptive algorithms) where there are two virtual channels for each physical channel in the Y dimension and one for each physical channel in the X dimension was simulated. Then, the topology was extended to have 2, 4, 6 and 10 virtual channels in both the X and Y dimensions. For all these cases we ran the simulations for all the three traffic patterns described in Section 3. When more virtual channels were added to the topology, we employed VCFC (i.e., allowed routing choice among the various "lanes" on a single physical channel) to the maximum extent possible. The amount of VCFC that can be employed depends to a large extent on the routing algorithm chosen. The idea behind adding lanes to the topology was to evaluate the impact of VCFC and how it would affect the performance of adaptive algorithms.

All average latency values were obtained with a confidence interval width of 3%, and a 95% confidence level. Results from the first 5000 clock cycles were discarded in each simulation, so as to remove some of the transient start-up effects.

4.1 Performance Under Uniform Traffic

When there are two virtual channels for each physical channel in the Y dimension and one virtual channel for each physical channel in the X dimension, the XY algorithm performs better than the adaptive algorithms at higher loads whereas the adaptive algorithms perform slightly better at lower loads. (See Figure 4.) This is due the fact that, at lower loads, adaptive algorithms provide more paths to the destination and can avoid unnecessary blocking which can occur in deterministic algorithms. At higher loads, when routing is adaptive, many messages take zig-zag routes within the network and block many other messages inside the network. For example, if a message traveling in the Y dimension makes a turn onto a channel in the X dimension, then it blocks all messages that want to travel in the X dimension on the same link. Thus, the greater the number of zig-zag turns, the greater the potential for blocking in the network. This effect has also been observed in previous studies [9, 10].

When the number of virtual channels per physical channel is increased to two, six, and ten, we observe that the adaptive algorithms catch up with the XY algorithm in terms of performance, and even perform better than it. (See Figures 5, 6 and 7.) This improvement in performance occurs since the VCFC permitted by the additional virtual channels allow messages to bypass a blocked message. Thus VCFC alleviates the problem of blocking inside the network, which was exacerbated by adaptive algorithms. From Figures 4 and 5, we notice that there is not much improvement in the performance for the Linder and Harden algorithm when the number of virtual channels is increased from one to two for each physical channel in the X dimension. This is due to the fact that the channels in the Y dimension become the bottleneck as only one virtual channel of the two allocated to each physical channel in this dimension, is available to each message (effectively) due to the partitioning of the network into two subnetworks.

Since the zig-zag paths resulting from adaptive algorithms result in higher blocking inside the network for high loads, a drop in throughput can occur as the generation rate α increases, at very high loads and latencies. (See Figures 5 and 6.) Since adding virtual channels to the network avoids this blocking to a great extent, the drop in throughput also decreases as the number of channels increases.

In general the *-Channels algorithm performs better than the Linder and Harden algorithm because it offers more choices with respect to selection of virtual channels at every node, loads the network more evenly and maintains the uniformity of the traffic. We note, however, that the improvement in performance obtained by adding more lanes and employing VCFC is independent of the algorithm, as we observe good improvements for both adaptive algorithms. When used with VCFC, both adaptive algorithms yield performance similar to that of deterministic routing.

4.2 Performance Under Hot Spot Traffic

For the minimal topology, with two virtual channels per physical channel in the Y dimension and one per physical channel in the X dimension, there is not much difference between the observed performance under hot spot traffic, and that under uniform traffic. (Compare Figures 8 and 4.) This could be due to a bottleneck effect resulting from the lack of additional virtual channels in the X dimension. This effect may be stronger than the effect resulting from the traffic becoming mildly non-uniform.

When the number of virtual channels per physical channel is equal to two in each dimension, the performance of all three algorithms under the mild hot spot traffic pattern is slightly worse compared to that obtained with uniform traffic. (Compare Figures 5 and 9.) This relative drop in performance increases when the number of channels is made equal to four (not shown in the graphs included here). An explanation for this is that the effect of a lack of virtual channels is no longer a more limiting factor than the congestion caused by the hot spot nodes. The relative drop in performance under mild hot spot traffic (in comparison to that under uniform traffic) is substantially less for the adaptive algorithms as compared to for the deterministic XY algorithm. This is clearly visible when the number of virtual channels per physical channel is made equal to six. (See Figure 10.) This is because the adaptive algorithms are much better in spreading the traffic and thus avoiding congestion.

When the number of virtual channels per physical channel is made equal to ten (Figure 11) the *-Channels algorithm gives the same performance for the non-uniform traffic pattern as it gave for the uniform traffic pattern (max throughput of 0.3). This is because the number of virtual channels is sufficient for holding the non-uniform traffic without clogging all virtual channels and thus adaptive routing can be effectively employed to route traffic not destined for the hot spot nodes around links that would otherwise become congested. This is not possible with the XY routing algorithm (which employs VCFC but not adaptive routing).

From the above experiments, we note that, if traffic is mildly non-uniform and if the network does not have sufficient virtual channels, deterministic algorithms can do slightly better than adaptive algorithms. This effect disappears once more virtual channels are added to the network. Also, good adaptive algorithms, when used in combination with VCFC, can provide nearly the same performance for mildly non-uniform traffic as for uniform traffic. Thus, overall, the combination of VCFC and adaptive routing appears to be very beneficial.

4.3 Performance Under Matrix Transpose Traffic

Since the non-uniformity in the traffic is very high in the case of matrix transpose traffic, the (deterministic) XY algorithm performs badly compared to the adaptive algorithms even for the minimal topology with two virtual channels per physical channel in the Y dimension and one per physical channel in the X dimension. (See Figure 12.) From Figures 14 and 15, note that adaptive algorithms perform much better than the XY algorithm when the number of channels is increased.

Figures 16, 17 and 18 show the impact of adding virtual channels for all the 3 algorithms separately. From Figure 16, note that the performance of the XY algorithm actually decreases as the number of virtual channels is increased. Although VCFC allows messages to bypass blocked messages, the message multiplexing that it allows also may dramatically increase the latency of individual messages. Since matrix transpose traffic is highly non-uniform, the load (and hence multiplexing) on certain channels is very high. As the number of virtual channels is increased the load on these specific channels becomes much higher. Since the XY algorithm does not change this unevenness in load, it experiences increases in network latency. On the other hand the *-Channels algorithm spreads the traffic more evenly and thus reduces high multiplexing on any specific channel. Therefore, the negative impact of adding virtual channels is reduced. At the same time, the extra virtual channels help in avoiding blocking inside the network and thus improve the performance. (See Figure 17.) When the number of virtual channels is equal to ten there is a small drop in performance for the *-Channels algorithm. At this point, evidently, the negative impacts of multiplexing outweigh the positive effects of avoiding blocking.

When the number of virtual channels per physical channel is two in both X and Y dimensions there is a drop in performance for the Linder and Harden algorithm. (See Figures 13 and 18.) This is because the negative effect of multiplexing is quite high due to the extra virtual channel being added to each physical channel in the X dimension, and at the same time no positive effects exist as only a single virtual channel is available to each message in the Y dimension. But when the number of virtual channels is made equal to six, there is a slight improvement in performance. The performance of the Linder and Harden algorithm is not as good because it does not balance the load as uniformly as the *-Channels algorithm does and thus the negative effects of adding lanes in this traffic pattern are more pronounced.

In conclusion, we note that adding lanes (performing VCFC) in the context of a deterministic algorithm can severely degrade performance for certain traffic patterns. This degradation in performance can be avoided by employing adaptive routing algorithms. Since adaptive algorithms spread the traffic more evenly, they provide more robust flow control and less multiplexing which can improve performance by a great extent.

4.4 Impact of Other System and Policy Assumptions

In this section we will consider the effects of some alternative system assumptions on the results obtained so far. All these experiments were performed by using the *-Channels and XY algorithms only, with six virtual channels per physical channel in both X and Y dimensions.

Two variations of output selection policies were considered. In the first variation, the output selection policy chooses the dimension in which the message has to travel the farthest [2]. Due to the zig-zag nature of this output selection policy, the resulting performance was poorer compared to the minimum congestion policy used previously. The other variation was to choose the dimension which has the least load. [2] There was a slight improvement in performance, in some cases, by using this policy. But in general the trend of the results obtained were not altered when the output selection policy was changed.

Two different physical channel transmission policies were tried. One was the 'oldest flit first' policy, and the other

[2] This is different from the minimum congestion policy since here we base our choice on the number of active virtual channels (both star and non-star channels) in each dimension.

was the ‘switch when necessary’ policy. In the ‘oldest flit first’ policy, the flit belonging to the message that has spent the longest time in the system is transmitted first. The ‘switch when necessary’ policy transmits the same message as long as possible, before it encounters a gap in the message, [3] and then switches to the next message in a round robin fashion. The basic idea of this policy is to minimize the occurrence of gaps in messages. Figures 19 and 20 show the performance obtained using these policies for the *-Channels algorithm for two different traffic patterns. From these graphs we notice that at lower loads both the ‘oldest flit first’ and ‘switch when necessary’ policies perform better than the round robin policy. This could be due to the fact that these two policies tend to increase the possibility that successive flits transmitted over the channel are from the same message. This can reduce the multiplexing factor of the message and thus improve performance. At high loads, round robin does better, however. For the XY deterministic algorithm (which employs just VCFC), the behavior is quite similar. The relative performances of the XY algorithm and the *-Channels algorithm were not affected considerably.

Experiments were also performed in which the size of each message was either 35 or 5 flits, with equal probability. Note that the average size of a message is 20 flits, matching the size used in previous experiments. In this context the ‘round robin’ policy does better than ‘oldest flit first’ for both uniform and matrix transpose traffic. (See Figures 20 and 21.) This is because the ‘round robin’ policy does not make small messages (flit size = 5) wait behind larger ones (flit size = 35). For the XY deterministic algorithm, the behavior is quite similar. The relative performances of the XY algorithm and the *-Channels were not affected much by the change in the wormsize distribution.

In general, it appears that our results are not affected to any great extent by these modifications in the system and policy assumptions.

5 Conclusions

The main goal of this paper was to study the interaction between adaptive routing and Virtual Channel Flow Control. We varied the amount of VCFC in the context of two different adaptive algorithms and studied its effect on performance.

This study has shown that employing adaptive routing with a minimal channel topology (no extra virtual channels) can result in poor performance for uniform and some non-uniform traffic patterns. Also, using VCFC alone (using virtual channels to provide extra lanes, but not also to allow adaptivity in routing) can severely degrade performance for certain traffic patterns. But when VCFC is combined with adaptivity in routing, these effects are mitigated and good performance results. Thus we see a lot of interaction between VCFC and adaptivity. Adaptive algorithms perform well once VCFC is employed and VCFC gives good benefits with adaptive algorithms. Our main conclusion is therefore that both adaptive routing and VCFC are indeed beneficial and their real benefit can be felt when they are employed together.

This study also shows that only a limited number of virtual channels per physical channel are needed (at most six in our study) for good performance. Investing more channels beyond this returns only diminishing benefits and can even hurt performance due to effects of traffic multiplexing. Thus the choice of the number of virtual channels in the network should be made carefully, and relatively frugally. The algorithms that were implemented in this study do not appear to have high overheads and the router complexity for their implementations are also likely to be low. This leads us to the conjecture that adaptivity can be provided at a reasonable cost.

Finally, we notice that virtual channels have been used for various purposes: to design deadlock free routing algorithms, to provide better flow control, to increase adaptivity and to introduce fault tolerance. Some algorithms like the *-Channels algorithm, sacrifice fault tolerance for better performance. In the *-Channels algorithm, the star channels act as pathways to break deadlocks that could occur among non-star channels[3]. Therefore, failure of certain star channels can lead to deadlock situations due to the absence of these release mechanisms. More work needs to be done in designing high performance deadlock free routing algorithms that are also fault-tolerant.

[3] Gaps tend to occur between successive flits in a message due to blocking and multiplexing in the network.

References

[1] A. Agarwal, “Limits of Interconnection Network Performance”, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 2, No. 4, Oct. 1991, pp. 398-412.

[2] H.G. Badr and S. Podar, “An Optimal Shortest-Path Routing Policy for Network Computers with Regular Mesh-Connected Topologies”, *IEEE Transactions on Computers*, Vol. 38, No. 10, Oct. 1989, pp. 1362-1371.

[3] P.E. Berman, L. Gravano, G.D. Pifarre and J.L.C. Sanz, “Adaptive Deadlock- and Livelock-Free Routing with All Minimal Paths in Torus Networks”, *Proceedings of the 4th Annual ACM Symposium on Parallel Algorithms and Architectures*, June 1992.

[4] R.V. Boppana and S. Chalasani, “A Comparison of Adaptive Wormhole Routing Algorithms”, *Proceedings of the 20th Annual Symposium on Computer Architecture*, May 1993, pp. 351-360.

[5] A.A. Chien and J.H. Kim, “Planar-Adaptive Routing: Low-cost Adaptive Networks for Multiprocessors”, *Proceedings of the 19th Annual Symposium on Computer Architecture*, May 1992, pp. 278-287.

[6] W.J. Dally, “Virtual-Channel Flow Control”, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 2, March 1992, pp. 194-205.

[7] W.J. Dally and C.L. Seitz, “Deadlock-Free Message Routing in Multiprocessor Interconnection Networks”, *IEEE Transactions on Computers*, Vol. 36, No. 5, May 1987, pp. 547-553.

[8] J. Duato, “A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks”, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 12, Dec. 1993, pp. 1320-1331.

[9] W.J. Dally and H. Aoki, “Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels”, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 4, April 1993, pp. 466-475.

[10] C.J. Glass and L.M. Ni, “The Turn Model of Adaptive Routing”, *Proceedings of the 19th Annual Symposium on Computer Architecture*, May 1992, pp. 278-287.

[11] C.J. Glass and L.M. Ni, “Adaptive Routing in Mesh-Connected Networks”, Technical Report MSU-CPS-ACS-45, Michigan State University, Oct. 1991.

[12] C.R. Jesshope, P.R. Miller and J.T. Yantchev, “High Performance Communications in Processor Networks”, *Proceedings of the 16th Annual Symposium on Computer Architecture*, May 1989, pp. 150-157.

[13] D.H. Linder and J.C. Harden, “An Adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes”, *IEEE Transactions on Computers*, Vol. 40, No. 1, Jan. 1991, pp. 2-12.

[14] L.M. Ni and P.K. McKinley, “A Survey of Wormhole Routing Techniques in Direct Networks”, *IEEE Computer*, Vol. 26, No. 2, Feb. 1993, pp. 62-76.

[15] G.F. Pfister and V.A. Norton, “Hot Spot Contention and Combining in Multistage Interconnection Networks”, *IEEE Transactions on Computers*, Vol. 34, No. 10, Oct. 1985, pp. 943-948.

[16] L. Schwiebert and D.N. Jayasimha, “Optimal Fully Adaptive Wormhole Routing for Meshes”, *Proceedings of Supercomputing 1993*, Nov. 1993, pp. 782-791.

[17] C. Su and K.G. Shin, “Adaptive Deadlock-Free Routing in Multicomputers Using Only One Extra Virtual Channel”, *Proceedings of the 1993 International Conference on Parallel Processing*, Aug. 1993, Vol. I, pp. 227-231.

[18] H. Sullivan and T.R. Bashkow, “A Large Scale, Homogeneous, Fully Distributed Parallel Machine”, *Proceedings of the 4th Annual Symposium on Computer Architecture*, Mar. 1977, pp. 105-124.

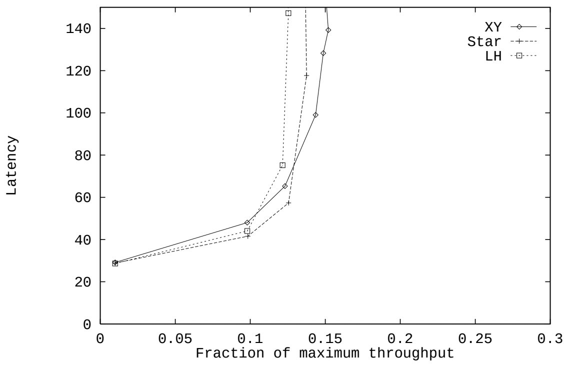

Figure 4: Comparison of routing algorithms for uniform traffic in a 12 X 12 mesh with 2 virtual channels in Y dimension and 1 in X dimension

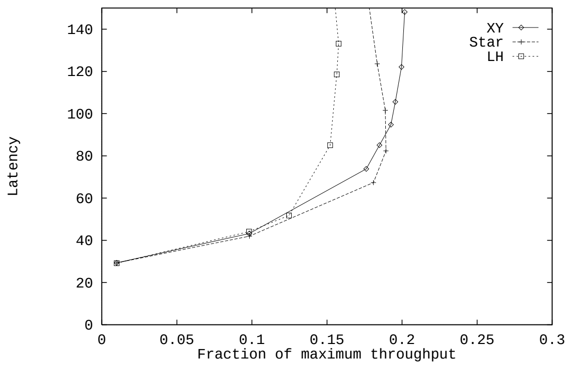

Figure 5: Comparison of routing algorithms for uniform traffic in a 12 X 12 mesh with 2 virtual channels in X and Y dimension

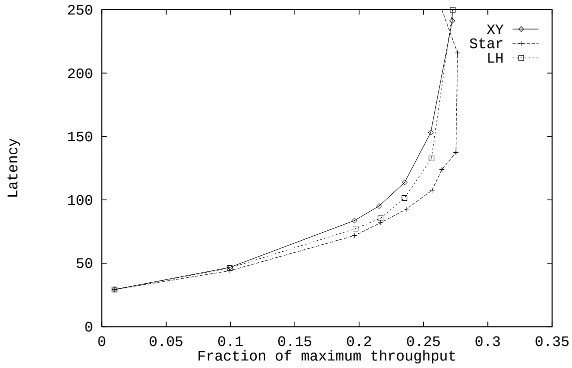

Figure 6: Comparison of routing algorithms for uniform traffic in a 12 X 12 mesh with 6 virtual channels in X and Y dimension

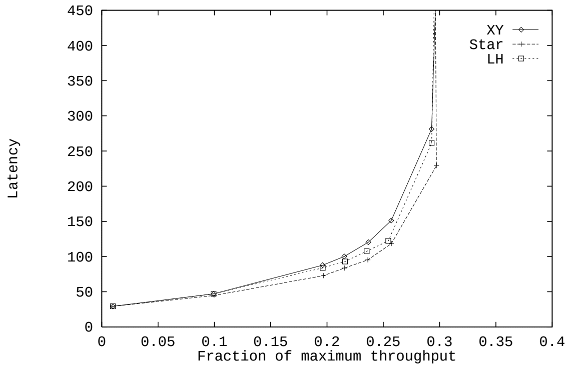

Figure 7: Comparison of routing algorithms for uniform traffic in a 12 X 12 mesh with 10 virtual channels in X and Y dimension

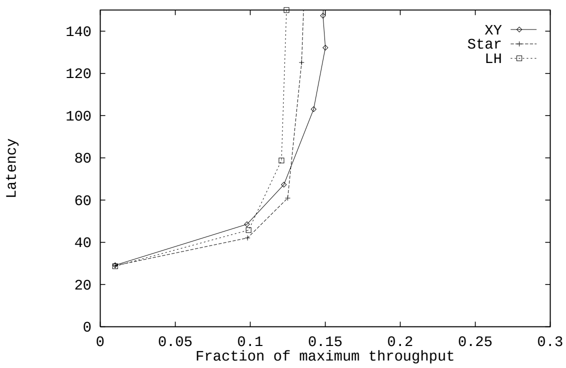

Figure 8: Comparison of routing algorithms for mild hot spot traffic in a 12 X 12 mesh with 2 virtual channels in Y dimension and 1 in X dimension

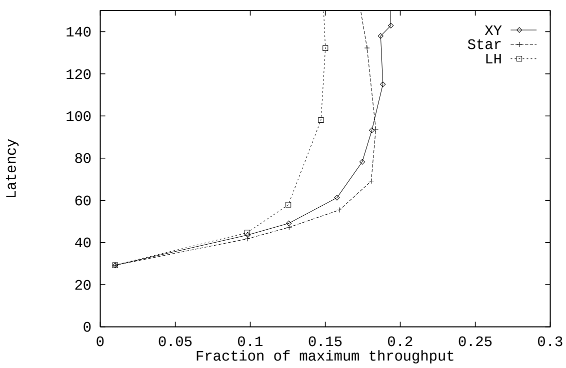

Figure 9: Comparison of routing algorithms for mild hot spot traffic in a 12 X 12 mesh with 2 virtual channels in X and Y dimension

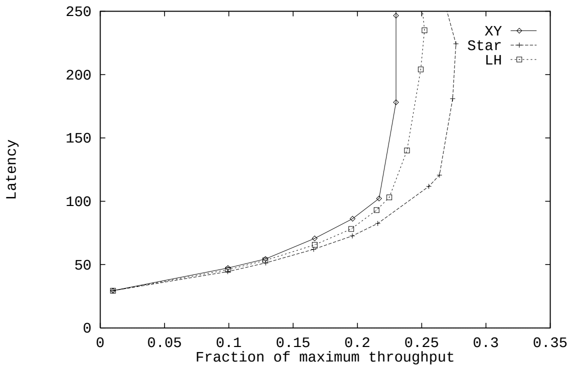

Figure 10: Comparison of routing algorithms for mild hot spot traffic in a 12 X 12 mesh with 6 virtual channels in X and Y dimension

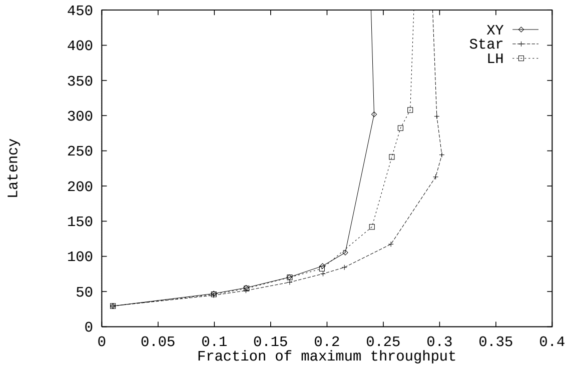

Figure 11: Comparison of routing algorithms for mild hot spot traffic in a 12 X 12 mesh with 10 virtual channels in X and Y dimension

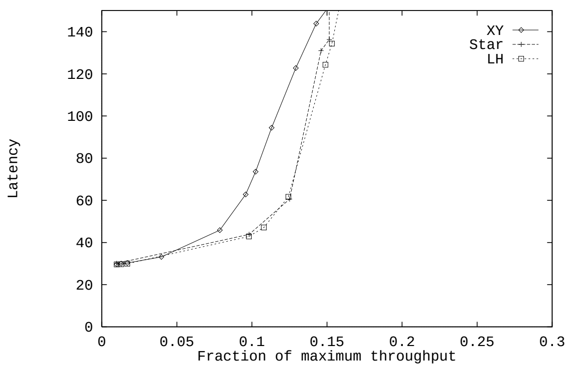

Figure 12: Comparison of routing algorithms for matrix transpose traffic in a 12 X 12 mesh with 2 virtual channels in Y dimension and 1 in X dimension

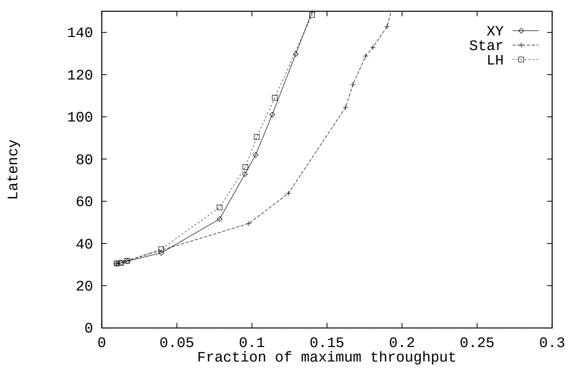

Figure 13: Comparison of routing algorithms for matrix transpose traffic in a 12 X 12 mesh with 2 virtual channels in X and Y dimension

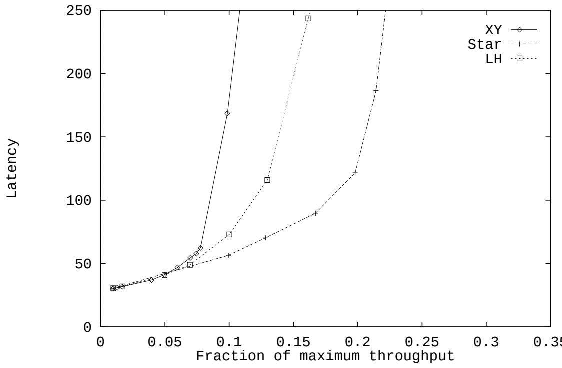

Figure 14: Comparison of routing algorithms for matrix transpose traffic in a 12 X 12 mesh with 6 virtual channels in X and Y dimension

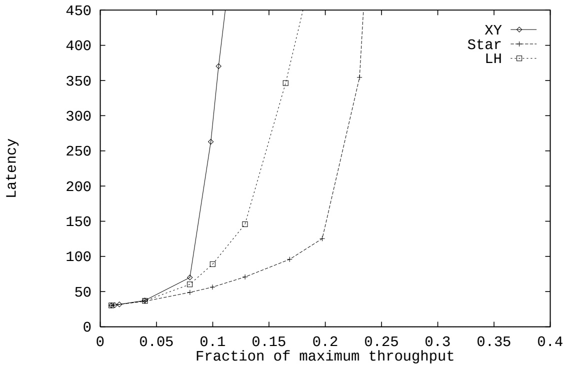

Figure 15: Comparison of routing algorithms for matrix transpose traffic in a 12 X 12 mesh with 10 virtual channels in X and Y dimension

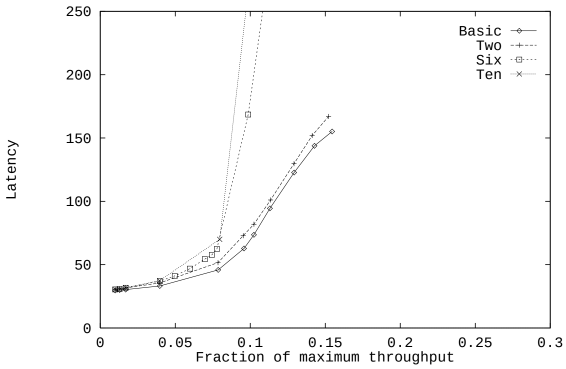

Figure 16: Impact of adding virtual channels to the topology for matrix transpose traffic and employing the XY algorithm

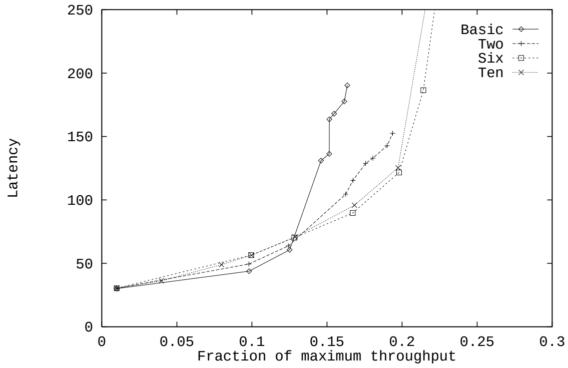

Figure 17: Impact of adding virtual channels to the topology for matrix transpose traffic and employing the *-Channels algorithm

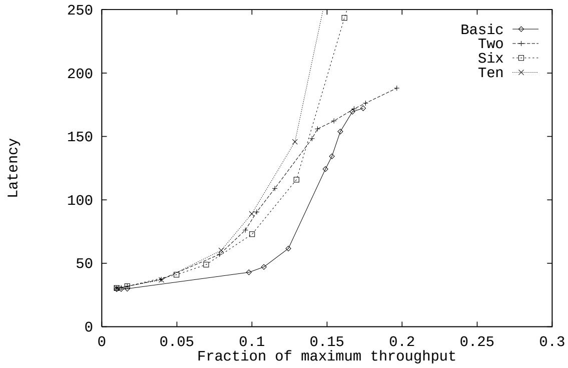

Figure 18: Impact of adding virtual channels to the topology for matrix transpose traffic and employing the Linder and Harden algorithm

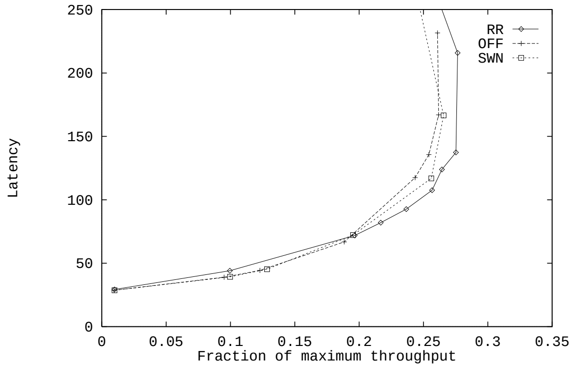

Figure 19: Impact of various physical transmission control policies on the performance of the *-Channels algorithm for uniform traffic

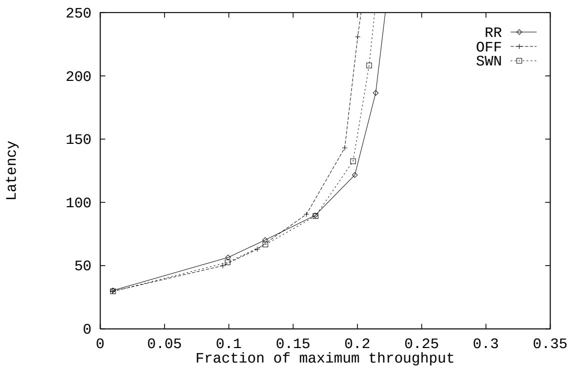

Figure 20: Impact of various physical transmission control policies on the performance of the *-Channels algorithm for matrix transpose traffic

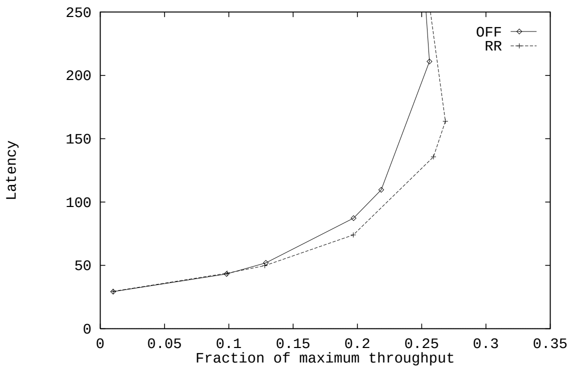

Figure 21: Performance under a bimodal wormsize distribution, using the *-Channels algorithm, for uniform traffic

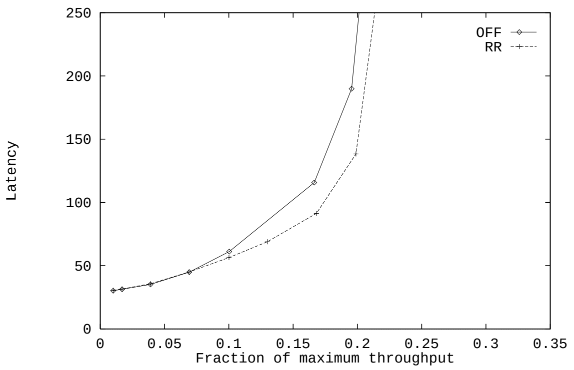

Figure 22: Performance under a bimodal wormsize distribution, using the *-Channels algorithm, for matrix transpose traffic