CONNECTION-BASED ADAPTIVE ROUTING USING DYNAMIC VIRTUAL CIRCUITS

YOSHIO F. TURNER and YUVAL TAMIR
Computer Science Department
University of California
Los Angeles, California 90095

ABSTRACT

Virtual circuits can reduce the routing overhead with irregular topologies and provide support for a mix of quality of service (QOS) requirements. Information about the state of the network and expected traffic patterns may be used during circuit establishment to utilize network resources more efficiently than is practical with packet routing. Most virtual circuit schemes are static — once the circuit is established, it remains unchanged until the connection is no longer needed. The *Dynamic Virtual Circuit* (DVC) mechanism allows existing circuits to be quickly torn down in order to free up resources needed for other circuits or in order to be re-established along routes that are better suited for current network conditions. We present a deadlock avoidance technique, based on unconstrained routing of DVCs combined with a deadlock-free virtual network. We describe key aspects of the implementation of our scheme and present performance evaluation results that explore its potential benefits.

1. INTRODUCTION

The routing scheme used in multicomputer interconnection networks should direct packets through the lowest latency paths from source to destination. It should take into account the topology of the network and *adapt* to the current workload and resource availability to route packets around congested or faulty areas [1, 2].

To maximize the performance of a network, it is important to minimize the addressing and control information that must be sent with each packet. Processing required to interpret and route each arriving packet must also be minimized. These goals can be achieved by using *connection-based* routing, in which resources are reserved in advance of communication at each node along the path from source to destination. To be effective, connection setup must be fast or infrequent relative to communication. Examples of connection-based routing differ in the amounts and types of resources reserved. At one extreme, *physical circuit switching* reserves the entire link bandwidth along the path from source to destination. That approach eliminates routing and the addressing and control information attached to each packet, but it also prevents sharing the reserved link bandwidth even when connections are idle. In contrast to physical circuit switching, *virtual circuit switching* [3, 4] reserves only an entry in a virtual channel table at each intermediate node. The entry specifies addressing and routing information for the virtual circuit. Once a *virtual circuit* is established on a path between a source and destination, packets can be sent along the path without the addressing and sequencing information needed in packet switched networks, and without time-consuming packet routing procedures. Compared to physical circuits, virtual circuits use network resources more efficiently since the physical resources (buffers, links, etc) are multiplexed between *active* connections.

With virtual circuits (as well as with physical circuit switching) it is possible to achieve good performance even in irregular topologies, where simple algorithmic routing is not possible. For such topologies, routing is based on tables, which are constructed when the system is initialized and may be changed later, as the load on the system changes [5]. Once a virtual circuit is established, forwarding of a packet along its path can be done very quickly — a single lookup in a small virtual channel table at each hop.

With most virtual circuit schemes, once the circuit is established, its route remains unchanged until the connection is no longer needed and the circuit is torn down. This prevents adaptation to changes in traffic patterns. Furthermore, it may prevent establishment of a new virtual circuit once all the required resources have been reserved by existing circuits. To resolve this problem, we have proposed the *Dynamic Virtual Circuits* (DVC) mechanism [6]. With DVCs, a virtual circuit can be torn down from an intermediate node, between the source and destination, and later re-established, possibly along a different route, while maintaining the virtual circuit semantics — reliable in-order delivery of packets.

In this paper we describe a practical implementation of DVCs. The key challenge when implementing DVCs is to simultaneously provide: fully adaptive routing, deadlock-freedom, and the low per-hop overhead of static virtual circuit switching. Our scheme places minimal restrictions on the use of buffer resources and does not constrain the routes of the virtual circuits, thus maximizing the network's ability to adapt to different traffic patterns. Deadlocks are avoided using a packet-routing deadlock-free virtual network. The deadlock avoidance scheme is simple to implement, uses few dedicated resources, and maintains the semantics of the virtual circuits. Section 2 describes some related work in this area. Section 3 describes our proposed technique. Section 4 presents simulation results that explore the potential performance benefits of DVCs. This is done by considering limit cases, where the more sophisticated (complex) routing possible with DVCs leads to significantly higher performance than can be achieved with conventional packet switched networks, which typically must use simple (e.g., algorithmic) routing.

2. RELATED WORK

Proposals for connection-based routing have been motivated by the observation that many applications of practical interest exhibit spatial and temporal locality, in the sense that each node tends to communicate mostly with a slowly changing, small set of other nodes [4]. Hsu and Banerjee [7] proposed routing hardware called *cached circuits* that accommodates a small number of virtual circuits on each link to exploit locality. Dao et al [3] combined cached circuits with conventional wormhole routing for traffic not exhibiting locality. Our Dynamic Virtual Circuits (DVCs) [6] differ from these proposals by allowing virtual channel resources to be quickly reallocated through local operations at a switch, avoiding the long delays of schemes that require interactions with faraway nodes before releasing local resources. Resource reallocation avoids blocking circuit establishment, and it enables adaptive circuit rerouting.

Much of the research on adaptive routing for packet-switched networks has focused on the buffer deadlock problem. If the paths taken by packets are not constrained, the network may be susceptible to deadlocks, due to cyclic dependencies among buffer and link resources [8]. This problem can be overcome by detecting potential buffer dependency cycles when they occur and resolving them using dedicated resources (one packet buffer per switch) reserved especially for this purpose [9]. The scheme involves ‘‘rotating’’ blocked packets around the dependency cycle, utilizing the dedicated buffers, until one or more packets leave the cycle. Unfortunately, through simulation we have shown elsewhere [10] that detecting and rotating blocked cycles completes too slowly to prevent the arrival of new packets that nearly instantly replace packets removed by cycle rotation. Therefore, cycles of blocked packets may persist or grow to encompass large numbers of network buffers.

An alternative to the time-consuming deadlock resolution mechanism described above is to embed in the physical network a deadlock-free virtual network. The virtual network consists of a set of buffers and a scheme for routing packets between them. Deadlock-freedom is guaranteed by restricting the routing of packets using the dedicated buffers such that no cyclic buffer dependencies are possible. The deadlock-free virtual network must be always reachable by packets and once a packet enters this virtual network, it cannot leave it until it reaches its destination. Examples of this approach include the scheme of Dally and Aoki [1], and Disha Concurrent [2]. With some variations, all such schemes dedicate buffer resources to deadlock-free routing.

The approach of providing a deadlock-free escape path was generalized by Duato, who proved necessary and sufficient conditions for deadlock-freedom [11]. Stated informally, there must be a set of virtual channels that can be reached from any buffer in the network and acts as a deadlock-free escape path for the delivery of blocked packets. This basic approach is used for deadlock avoidance with our DVC mechanism.

3. DVC’S WITH DEADLOCK AVOIDANCE

Communication over virtual circuits requires that a circuit be established prior to or in conjunction with the transmission of the first packet from a source to a destination. Data packets on established circuits carry very few overhead bits for bookkeeping (routing and sequencing information). They are quickly routed and forwarded at each intermediate node. With DVCs, the circuits may be torn down starting from any intermediate switch and later re-established in order to handle additional packets [6]. This section shows how the mechanism for tearing down and re-establishing circuits can be combined with a deadlock avoidance scheme that provides packets with a deadlock-free escape path. Along the escape path, packets are routed individually, completely bypassing the virtual circuits mechanisms. At the destination, packets arriving along the original virtual circuit, re-established virtual circuits, and the escape path, must be sorted into proper order to maintain the semantics of FIFO packet delivery. Assuming that only a minority of packets will require the escape path, most of the advantages of virtual circuits can be maintained.

We consider a virtual cut-through network composed of point-to-point interconnected *n x n* switches. At each switch, one input and one output connect the switch to a host. The switch is input buffered, and a central *n x n* crossbar connects the inputs to the outputs.

A source node’s host initiates a DVC by creating and injecting a Circuit Establishment Packet (CEP). The CEP holds the DVC’s source and destination IDs, which are used to route the CEP adaptively. In general, CEP routing may be accomplished through the use of routing tables maintained at each switch, such as in the SGI Spider chip [5].

Each physical link is logically subdivided into multiple *Routing Virtual Channels* (RVCs). Each packet has a header field that identifies the RVC used by the packet. The CEP acquires for the new DVC one RVC on each link it traverses on its path from source to destination (including the source and destination host interface links). At each switch, the mapping from input RVC to output RVC is recorded in an ‘‘Input Mapping Table’’ (IMT) at the input port. The IMT is indexed by input RVC identifier, and the entry records the following information about the DVC associated with the RVC: the output port, output RVC, source and destination IDs, and sequence number.

Note that we use the term ‘‘RVC’’ instead of the more familiar ‘‘virtual channel’’ to distinguish it from the same term commonly used to refer to flow-controlled buffers that prevent deadlock and increase performance. Those we call *Buffering Virtual Channels*, or ‘‘BVCs’’. In contrast, ‘‘RVCs’’ simply identify DVCs, and they do not require separate flow-controlled buffers.

Once DVC establishment commences via CEP injection, the source may inject data packets on the DVC (by using the same RVC on the host interface link as the CEP). The data packets are quickly routed at each hop via IMT lookup. The RVC identifier in the packet’s header indexes the IMT to retrieve the output port and output RVC. The RVC field of the packet’s header is replaced by the output RVC identifier, and the packet is enqueued for transmission to the next switch.

Once a DVC is no longer needed, the source injects a Circuit Destruction Packet (CDP) that traverses the DVC path and frees up the allocated RVCs. Also, throughout the lifetime of a DVC, any intermediate switch may tear the DVC from that point and later reroute it on demand, either to free up an output

RVC to allocate to a new circuit, or to adapt to traffic conditions by shifting traffic onto a lower latency path. A switch tears down a victim DVC by inserting a CDP into its path. A subsequent data packet arriving on the input RVC of a torn-down DVC triggers DVC re-establishment, in which the switch creates a new CEP using the information retained in the IMT. There are no restrictions on when to reroute DVCs or which new paths to take.

The general packet format is shown in Figure 1. It consists of a header followed by data phits. The first phit of the header identifies the RVC. An additional four bits of the RVC field indicate packet type and whether the packet is of maximum length. If not, a length field is present in the header. The remaining header fields shown in the figure are only occasionally present, as the following discussion will illustrate. Therefore, the minimum header (which should be the common case) is one that has only the RVC field.

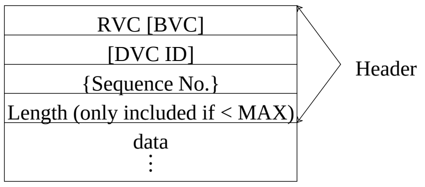

FIGURE 1: PACKET FORMAT. Fields in ‘‘[]’’ have the stated use only for diverted data packets. The sequence number field is used only for diverted packets and the next non-diverted packet.

To achieve low latency and high throughput with the largest possible variety of traffic patterns, adaptive routing is necessary. Eliminating restrictions on DVC paths avoids traffic congestion. However, without such restrictions, it is possible, though unlikely, that transient conditions cause deadlocks among the packet buffers. Also, since RVCs are also finite resources in contention, the circuit manipulation protocols triggered by CEPs and CDPs also pose the potential for deadlock. These catastrophic conditions must be prevented.

To avoid deadlock, we use three *Buffering Virtual Channels* (BVCs) per link. On each physical link I, the *primary BVC* is associated with the primary input buffer N_I for data packets and does not restrict packet routing. At any time, N_I may hold any number of data packets that arrived on mapped RVCs, but it may hold only a single data packet that arrived on an unmapped RVC. The *diversion BVC* is associated with the diversion buffer D_I and provides a routing-restricted, deadlock-free escape path for each blocked data packet in an otherwise deadlocked cycle [11]. Finally, the control BVC is exclusively for CEPs and CDPs. Separating control and data packets eliminates the dependence of control packets on the progress at the next switch of unmapped data packets, which in turn depend on the progress of control packets to obtain RVC mappings. Also, the control BVC is designed to prevent any inter-switch buffer dependencies involving control packets or their buffers, which considerably simplifies the hardware and operations necessary for deadlock avoidance.

The primary buffer N_I may be a FIFO buffer, a more efficient Dynamically Allocated Multi-Queue (DAMQ) buffer [12], or any other type of buffer. Since DAMQ buffers avoid Head-Of-Line blocking and dynamically share the buffer space, we use them in our performance evaluation. Regardless of buffer type, when a packet arrives at an input port, it is first routed via IMT lookup before it is queued; with the DAMQ buffer, routing determines which logical queue the packet joins.

The diversion BVCs form a virtual network for diverted packets (the ‘‘diversion network’’). By restricting routing in the diversion network, we ensure its buffer dependency graph is acyclic. For example, in a mesh, Dimension-Order Routing (DOR) could be used. With rare diversion, it is sufficient for D_I to have capacity one.

A packet that has been blocked at the head of a primary buffer's queue becomes, after a timeout delay, a candidate for diversion to the diversion network. A single RVC can be dedicated to identify which packets use the diversion BVC. Any data packet arriving on that dedicated RVC is understood to be using the diversion BVC; otherwise, the packet uses the primary BVC. When a packet times out, its DVC information must be accessed to execute rerouting. When a packet is diverted, DVC information is attached to the packet header for identification. To access the correct IMT entry to get the DVC information, the primary buffer retains a record of the packet's input RVC until the packet is forwarded on the output RVC.

Diversion guarantees progress for each data packet that arrives on a mapped RVC. For unmapped data packets, ensuring progress requires that the data packet eventually acquires a mapping to an unallocated RVC via the deadlock-free creation, transmission, and processing of control packets. We guarantee progress by ensuring that no resource dependency cycles form in that process. The control BVC is associated with storage for each RVC; since that information is accessed infrequently, it may reside in slow RAM with negligible impact on performance. Allocating three entries for each RVC plus one entry for each input port is sufficient to guarantee that all arriving control packets can either be stored or else are unnecessary and can be dropped. This ensures that no inter-switch dependencies exist that involve the control BVC. Also, it can be shown that the process introduces no intra-switch dependency cycles. Therefore, both unmapped and mapped packets always make progress, and the whole network is deadlock free. The entire procedure is described and proved correct elsewhere [13].

Packet diversion helps to solve the packet deadlock problem but violates FIFO delivery, which is important for some applications. If packets were not diverted from their allocated paths, then FIFO delivery would be accomplished without stamping any packets with sequence numbers. However, here packets can occasionally be diverted, so minimal sequence information must be attached to some packets as follows.

Each input RVC's IMT entry records the sequence number of the data packet most recently transmitted on the output RVC to which the input RVC maps. When a packet is forwarded normally, the IMT is accessed, indexed by the packet's input RVC that is retained by the primary buffer. The sequence number is incremented, and the packet is forwarded.

But when a packet is diverted, its header is augmented by adding sequencing and routing information. The new header has two fields: 1) The source and destination, and 2) the sequence number from the IMT entry. The header of the next data packet transmitted normally from the same RVC is also augmented to include the sequence number, which is written to the IMT entry at each switch the packet subsequently visits until it is delivered or diverted.

The destination host interface uses the sequence numbers in packet headers to reconstruct FIFO ordering for each DVC. Packets are delivered from the host interface to the application in consecutive sequence number order. A packet arriving without a sequence number is assigned the sequence number one greater than that of the packet that arrived immediately previously on the same RVC. This achieves FIFO ordering while minimizing bandwidth used for sequence numbers, at the cost of resequencing hardware at the host interface and sequence number storage in the IMT.

4. PERFORMANCE EVALUATION

A primary advantage of DVCs is the potential for reducing overall network contention by establishing circuits or rerouting existing circuits onto low latency paths. In a realistic system, traffic patterns change dynamically, and circuits require time to adjust their paths to compensate. For a first-order evaluation of the performance potential for DVCs with adaptive routing, we abstract away from these transient complexities and consider simpler limit cases with stable traffic patterns and circuit placements.

We consider three traffic patterns: Transpose, which is heavily biased against static Dimension Order Routing (DOR) and is therefore a ‘‘best case’’ for DVCs; Bit-Reversal, which is not clearly biased; and finally Uniform, for which static routing performs well and is a ‘‘worst case’’ stable traffic pattern for DVCs. In all cases, we centrally precompute the paths used by DVCs in order to simulate conditions once circuits have settled into a steady-state, low contention configuration. We compare the performance of the resulting configuration against a packet switched network using DOR. The circuit path placement computation uses a simple heuristic shortest path algorithm that estimates link delay with a queueing model. Using a more optimal placement algorithm would only improve the simulated performance of DVCs relative to static DOR.

In all the following simulation results, packet size is 32 phits, and the switches have DAMQ primary input buffers and a diversion buffer of capacity one. Since we wish to evaluate the routing policy’s impact on throughput fairness as well as on aggregate performance, we prohibit local unfairness introduced at each switch through the crossbar arbitration policy. Specifically, the policy we simulate gives priority to the packet that has resided longest at the switch.

4.1. TRANSPOSE TRAFFIC

The transpose traffic pattern, which routes from the source at row i column j to the destination at row j column i, has poor performance with DOR. In a 2-D mesh, DOR transpose has low aggregate throughput and high unfairness, in the sense that some source-destination flows have high throughput while others experience much more congestion and therefore much lower throughput. Adaptive DVCs achieve higher throughput and much greater fairness for a given level of aggregate throughput.

Since each traffic flow must pass through a node along the diagonal, the overall throughput must be less than the bandwidth entering/leaving all diagonal nodes. In a mesh of size $\sqrt{n}\, x \sqrt{n}$, the diagonal bandwidth from either half of the network is $(\sqrt{n}-2)2+2$, and the throughput λ of the average source-destination flow is limited by the following:

$$(\sqrt{n}-2)\cdot 2 + 2 \geq \frac{n-\sqrt{n}}{2}\lambda \implies \lambda \leq \frac{4(\sqrt{n}-1)}{n-\sqrt{n}}.$$

Substituting $n=64$, we obtain $\lambda \leq 1/2$. If we normalize throughputs to the maximum value of λ, at saturation the maximum throughput obtained by DOR is 48% and the routed virtual circuits policy achieves 94%. This is reasonable since DOR uses only the horizontal links entering each node along the diagonal, limiting it to 50% of the incoming links available. With 1 phit of header per 32 phits of data, the effective throughput is reduced to 48%.

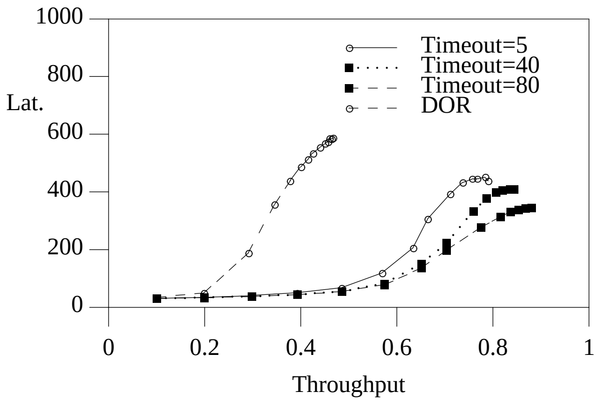

FIGURE 2: LATENCY VS. NORMALIZED THROUGHPUT. DAMQ buffer capacity = 64 phits. Transpose traffic pattern.

Figure 2 shows latency versus throughput for primary input DAMQ buffer capacity of 64 phits. With this traffic pattern, the only effect of increasing buffer size is to increase latency. Maximum throughput does not increase with buffer size, as it is limited by the bandwidth of individual saturated links along the mesh diagonal. However, the maximum throughput achieved with DVCs is significantly higher than that achieved with DOR.

Moreover, the results show that the routed virtual circuits perform best with long timeout. As timeout decreases, more packets are diverted and follow DOR routes to the destinations. Therefore, as the fraction of traffic diverted increases, the traffic and performance approach that of pure DOR.

Often, the performance of a distributed application is limited by the performance of its slowest member rather than by the aggregate throughput available to the application. As an example, an application whose nodes communicate via the transpose traffic pattern may occasionally need to synchronize to ensure that all sending nodes are in a known state. If some flow is particularly slow, all the other nodes will be throttled by

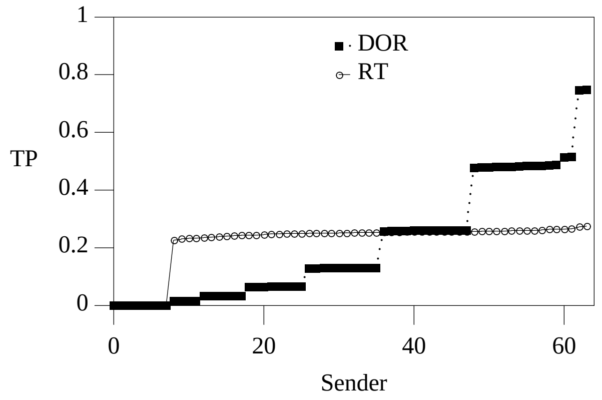

FIGURE 3: THROUGHPUT FAIRNESS. Throughput vs. Sender, sorted. Aggregate raw throughput = 0.233 for DOR, 0.242 for routed virtual circuits.

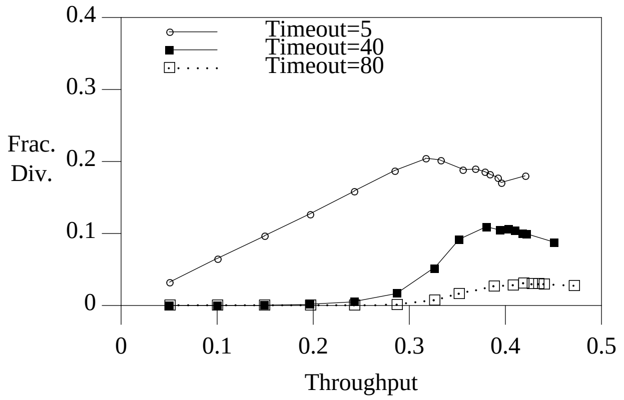

FIGURE 4: FRACTION OF TRAFFIC DIVERTED VERSUS AGGREGATE THROUGHPUT. Throughput is measured as useful phits received per cycle per receiver. DAMQ primary input buffer capacity = 64 phits.

the performance of that slow node upon synchronization. Therefore, it is interesting to examine the throughputs achieved by individual senders.

Figure 3 shows the raw throughput achieved by each sending node in the 8x8 mesh using the transpose traffic pattern. The throughputs from each sender are displayed, sorted so as to be monotonic (note that the first eight sources are along the diagonal and therefore do not generate packets). Throughputs for the routed virtual circuits policy and DOR are displayed as separate curves. An impartial evaluation of the fairness of the two policies is only possible when the two achieve the same aggregate throughput. For DOR, aggregate raw throughput is at its maximum, namely 0.233. The closest routed virtual circuit result has aggregate throughput 0.242. Since unfairness increases with aggregate throughput, the result in Figure 3 is biased slightly in favor of DOR, yet it can be seen that the routed virtual circuits achieve far greater uniformity of sender throughput than does DOR. DOR with transpose traffic causes some flows to experience large amounts of congestion with other flows, while others have no congestion whatsoever. As we increase applied load further, the routed virtual circuits policy also becomes very unfair, but only at higher levels of aggregate throughput than can be achieved with DOR.

To evaluate how much traffic is diverted from the virtual circuit path, figure 4 plots fraction of traffic diverted versus throughput with DAMQ primary buffer capacity 64. Again, packet length is 32 phits. The effects of varying the timeout interval are also shown. As timeout increases, the fraction diverted decreases significantly. For this particular traffic pattern and shortest path routing, no deadlocks are possible; hence, the best performance is achieved with no timeout at all. With different traffic patterns, cyclic dependencies may form temporarily, increasing the need for diversions.

4.2. BIT REVERSAL TRAFFIC

The bit reversal traffic pattern sends messages from each source $x_{n-1}x_{n-2}\cdots x_0 y_{n-1}y_{n-2}\cdots y_0$ to destination $y_0 y_1 \cdots y_{n-1} x_0 x_1 \cdots x_{n-1}$. We evaluate bit-reversal on an 8x8 mesh. Figure 5 shows latency versus throughput for primary input buffer capacity 64 phits. The reported throughput is normalized to the bisection bandwidth available with uniform traffic, and therefore it does not indicate an upper bound on throughput for the bit reversal traffic pattern.

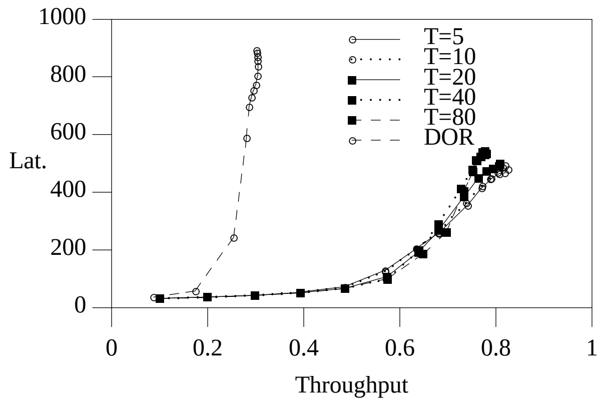

FIGURE 5: LATENCY VS. NORMALIZED THROUGHPUT. DAMQ buffer capacity = 64 phits. Bit reversal traffic pattern. T = Timeout.

The results demonstrate that routed virtual circuits significantly outperform DOR for this non-uniform traffic pattern. Moreover, the latency-throughput results are nearly independent of the diversion timeout value. This indicates that for routed virtual circuits with small timeout, the disadvantage of using the poorly performing DOR routing algorithm to route diverted traffic is completely offset by the advantage of having two BVCs per link. That is not the case for the transpose traffic pattern, because there the addition of BVCs does not alleviate the fundamental DOR transpose pattern bottleneck, which is the limited link bandwidth of the X-dimension links entering the mesh diagonal's switches.

Though not shown here, increasing the buffer capacity for the DOR case does not increase throughput. Also, the fraction of traffic diverted in the bit reversal case is similar to the transpose case: increasing timeout values dramatically decrease the fraction of traffic diverted with the bit reversal traffic pattern, and as load increases, diversion decreases since the DOR paths become congested, limiting injection into the diversion virtual network from the primary virtual network.

4.3. UNIFORM TRAFFIC

This section evaluates the performance of virtual circuits with diversion under uniform traffic. Under uniform traffic in the mesh, DOR should perform well, since as applied load increases, load across the horizontal and vertical bisections increases uniformly. In contrast, adaptive routing schemes tend to steer more traffic toward the center of the network, causing congestion.

Figure 6 shows latency versus throughput in uniform traffic with buffer capacity of 64 phits. In Figure 6, all the curves for routed flows correspond to a network where primary input buffer capacity is 32 phits, and the diversion buffer is also 32 phits, for a total of 64.

The results indicate that the performance of routed flows is close to that of DOR in uniform traffic. As timeout decreases, routed flows performance improves as it takes advantage of the 32 phit diversion buffers. Though not shown, with larger buffers (256 phits), routed flows and DOR achieve nearly identical performance even with large timeout.

One interesting feature of the routed flows results is reactivity, particularly with small timeouts. This can be explained by the observation that as applied load increases, the primary virtual network becomes congested, and packets become increasingly likely to enter the diversion virtual network, which has limited buffering and therefore limited throughput.

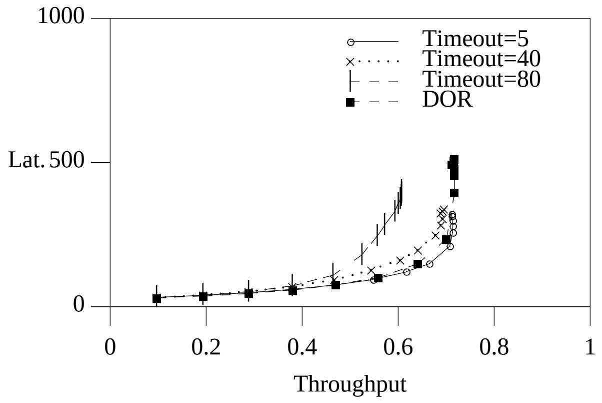

FIGURE 6: LATENCY VS. NORMALIZED THROUGHPUT. Total input buffer capacity = 64 phits. Uniform traffic pattern.

5. CONCLUSION

Dynamic Virtual Circuits retain the traditional advantages of virtual circuit switching: low per-packet bandwidth overhead, FIFO delivery, and establishment on paths experiencing low contention. In addition, DVCs provide adaptive circuit rerouting and efficient circuit establishment even when RVCs are fully allocated.

Minimal routing restrictions and dedicated buffer resources avoid deadlock while allowing circuits to use fully-adaptive routing or rerouting. Avoiding inter-switch dependencies among control buffers and intra-switch dependency cycles eliminates deadlock caused by contention for RVCs. The low-cost implementation merges the normally empty control buffers with the off-chip tables necessary for maintaining circuit information.

The performance results show that with virtual circuits, global routing optimization is possible and provides performance superior to fixed routing. Furthermore, the use of deadlock-free escape paths is sufficiently infrequent to preserve the bandwidth efficiencies of the DVC mechanism.

Future work will evaluate the behavior of DVCs with shifting traffic patterns; in particular, there are many alternatives for choosing when and how to reroute existing circuits. Other opportunities for investigation include fault tolerance and multicast virtual circuits.

REFERENCES

1. W. J. Dally and H. Aoki, ‘‘Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels,’’ *IEEE Transactions on Parallel and Distributed Systems* **4**(4), pp. 466-475 (April 1993).
2. A. K. Venkatramani et al., ‘‘Generalized theory for deadlock-free adaptive wormhole routing and its application to Disha Concurrent,’’ *The 10th International Parallel Processing Symposium*, Honolulu, HI, pp. 815-21 (15-19 April 1996).
3. B. V. Dao et al., ‘‘Architectural support for reducing communication overhead in multiprocessor interconnection networks,’’ *Third International Symposium on High-Performance Computer Architecture*, San Antonio, TX, pp. 343-52 (1-5 Feb. 1997).
4. J.-M. Hsu and P. Banerjee, ‘‘Performance measurement and trace driven simulation of parallel CAD and numeric applications on a hypercube multicomputer,’’ *IEEE Transactions on Parallel and Distributed Systems* **3**(4), pp. 451-464 (July 1992).
5. M. Galles, ‘‘Spider: A High-Speed Network Interconnect,’’ *IEEE Micro* **17**(1), pp. 34-39 (January/February 1997).
6. Y. Tamir and Y. F. Turner, ‘‘High-Performance Adaptive Routing in Multicomputers Using Dynamic Virtual Circuits,’’ *6th Distributed Memory Computing Conference*, Portland, OR, pp. 404-411 (April 1991).
7. J.-M. Hsu and P. Banerjee, ‘‘Hardware Support for Message Routing in a Distributed Memory Multicomputer,’’ *1990 International Conference on Parallel Processing*, St. Charles, IL (August 1990).
8. W. J. Dally and C. L. Seitz, ‘‘Deadlock-Free Message Routing in Multiprocessor Interconnection Networks,’’ *IEEE Transactions on Computers* **C-36**(5), pp. 547-553 (May 1987).
9. J. M. Jaffe and M. Sidi, ‘‘Distributed Deadlock Resolution in Store-and-Forward Networks,’’ *Algorithmica* **4**(3), pp. 417-436 (1989).
10. Y. F. Turner and Y. Tamir, ‘‘Deadlock resolution in networks employing connection-based adaptive routing,’’ Computer Science Department Technical Report CSD-960032, University of California, Los Angeles, CA (August 1996).
11. J. Duato, ‘‘A necessary and sufficient condition for deadlock-free routing in cut-through and store-and-forward networks,’’ *IEEE Transactions on Parallel and Distributed Systems* **7**(8), pp. 841-54. (August 1996).
12. Y. Tamir and G. L. Frazier, ‘‘Dynamically-Allocated Multi-Queue Buffers for VLSI Communication Switches,’’ *IEEE Transactions on Computers* **41**(6), pp. 725-737 (June 1992).
13. Y. F. Turner, *High-Performance Adaptive Routing in Multicomputers Using Dynamic Virtual Circuits, Ph.D. Dissertation, in preparation*, 1998.