A Comparison of Input and Output Driven Routers

Melanie L. Fulgham and Lawrence Snyder

Department of Computer Science and Engineering, Box 352350
University of Washington, Seattle, WA 98195-2350 USA

Abstract. Communication in parallel computers requires a low latency router. Once a suitable routing algorithm is selected, an implementation must be designed. Issues such as whether the router should be input or output driven need to be considered. In this paper, we use simulations to compare input driven and output driven routing algorithms. Three algorithms, the Dally-Seitz oblivious router, the *-channels router, and the minimal triplex algorithm are evaluated. Each router is implemented as both an input and an output driven router. Experiments are run for each of the router implementations with seven different traffic patterns on both a 256-node two dimensional mesh and torus networks. The results show that in almost all cases, the output driven router matches or outperforms the input driven router. Furthermore, we find that randomization of output buffer selection in the input driven algorithm increases its performance and substantially reduces the performance discrepancy between the input and output driven algorithms. Although the findings apply to the routers considered, we believe the results generalize to other routers.

1 Introduction

Communication in parallel computers continues to be a very difficult problem. In this paper we consider communication in multicomputer networks, networks with point to point connections between processors. In this model, processors communicate by sending messages through the network. Messages are forwarded at each node to their destination by a hardware router that implements the routing algorithm, the rules of which specify the path a message takes to reach its destination. Initially the communication bottleneck was the large software overhead for sending and receiving messages. Nevertheless, research has reduced this overhead [20] and exposed the network interface. Designing low latency network interfaces has become a hot topic for research [2, 18]. Eventually network interfaces will no longer overshadow the routing time in the network. Then, both the routing algorithm and its implementation will have an impact on communication performance.

In this paper, we experimentally compare two methods of implementing a particular router, as an input driven or as an output driven algorithm. Although the two choices are conceptually similar, they result in noticeable performance differences when compared with each other. Three algorithms, the Dally-Seitz oblivious router, the *-channels router, and the minimal triplex router, are simulated on a 256-node two dimensional (2D)

This work was supported in part by the National Science Foundation Grant MIP-9213469 and by an ARPA Graduate Research Fellowship.

mesh and torus. Each algorithm is configured as both an input and an output driven router. The output driven versions of the algorithms are almost always superior when the networks are congested. This is the critical area of network performance, since communication is often bursty and does not always present easy, light loads to the network.

The paper is organized as follows. Section 2 defines input and output driven routers. The simulation methodology and the three routing algorithms are described in Section 3. Results follow in Section 4, related work in Section 5, and finally conclusions in Section 6.

2 Input versus Output Driven

Routers that make decisions based on local information can usually be classified as either input driven or output driven. The input driven algorithms make routing decisions from an occupied input buffer while the output driven algorithms select routes from an empty output buffer. Typically the input (output) driven algorithms service the input (output) buffers in round robin order. After routing, the pointer to the current buffer is advanced to the next dimension where a valid buffer resides. For input driven algorithms a valid (input) buffer is one that contains a message that is ready to be routed across the node. For output driven algorithms a valid (output) buffer is one that has room for a message to be routed to it.

An input driven router operates as follows. First it computes which output buffers the message in the current input buffer needs. Then it selects one of these output buffers, if available, and routes the message to that output buffer. Many of the routing algorithms in the literature fall into this category.

An output driven algorithm considers the current empty output buffer. The router tries to find a message in an input buffer that needs to be routed to the current output buffer. If messages are found, it selects one to be routed to the current output buffer. The Chaos router is a good example of an output driven router [16].

Many algorithms can be implemented as either input or output driven algorithms. For these algorithms, it would be advantageous to implement the router with the best performing routing mechanism. A later section shows that for the algorithms examined, the output driven algorithm almost always performed better than the input driven algorithm. Unfortunately some algorithms cannot easily be transformed into output driven algorithms, for example algorithms that take probabilities over the possible output choices.

3 Methodology

Three algorithms are simulated as both input and output driven routers: the Dally-Seitz oblivious router [8], the *-channels router [1], and the minimal triplex router [11] using a flit-level simulator of 256-node 2D torus and mesh networks. The number of virtual channels per channel per direction for the routers are 2 (1), 3 (2) and 3 (2), respectively, for the torus (mesh) algorithms. Each node has an injection and delivery buffer in addition to an input and an output buffer for each virtual channel in each dimension in each direction. The oblivious algorithm also has an extra set of virtual lanes [7], yielding a

total of 34 (18), 26 (18), and 26 (18) buffers per node for the torus (mesh) routers, respectively. Transmission of a flit across a channel from an output buffer to input buffer takes one cycle; the pipelined logic at each node takes 3 cycles for the oblivious, and 4 cycles for the adaptive routers [3]. Routing options are computed in parallel, although a router connects at most a single message from an input buffer to an output buffer per cycle. Each buffer holds exactly one 20-flit message, and virtual cut through flow control [14] is used to avoid store-and-forward penalties. Results are normalized to the maximum theoretically sustainable load, constrained by the network bisection bandwidth, of one message every 80 (160) cycles for the torus (mesh), respectively. Messages are introduced at each node at every cycle with a probability specified by the applied load. Traffic considered is random (all nodes equally likely), permutations including bit reversal, transpose, complement, and perfect shuffle, as well as hot spot traffic (ten random nodes four times more likely to be destinations). The hot spots are listed in Appendix A. The traffic patterns are found in the literature, and are generally thought to be difficult, useful, or both. See [10] and [4] for further details.

4 Results

The results are presented in two parts. The first set of experiments compare output driven routers to input driven routers that choose deterministically the first available output buffer from the set of needed output buffers. Any deterministic order suffices, in this case the routers search in dimension order. In the second set of experiments the input driven router selects an available output buffer at random from the set of needed output buffers. For this set-up, the input driven router is very similar to the output driven router which chooses messages from the input buffers at random. In both cases the output driven routers perform better than the input driven routers for all but a few cases. As expected, the results are more dramatic for the first case.

4.1 Fixed Order Output Buffer Selection

The first set of experiments consider input driven routers with a fixed order output buffer selection policy. Table 1 specifies the first normalized applied load, in increments of .05, at which messages arrive faster then they can be injected and delivered by the 256-node torus network. For all three algorithms and all traffic patterns except one, the point of saturation for the output driven router is at least as great as the saturation point of the corresponding input driven router. The advantage ranges from zero to seventy percent of the input driven saturation point.

We conjecture that the output driven algorithms are better at utilizing the channels between nodes because the empty output buffers are filled by dimension in round-robin order which naturally tries to keep all the physical channels busy. More specifically, when a message has been routed to an empty output buffer in a dimension, the router will consider the next dimension that has an empty output buffer. However the input driven algorithm routes a message to the first available output buffer it finds. This buffer may share the same physical channel as a recently routed message. With virtual cut-through routing the buffer chosen may even be the same buffer as the previously routed message.

Table 1. Minimum normalized applied load within .05 at which saturation is detected.

16x16 Torus Saturation, input driven is fixed order						
Traffic	oblivious		*-channels		min triplex	
	input	output	input	output	input	output
Random	0.70	0.80	0.85	0.95	0.80	0.85
Bit reversal	0.40	0.50	0.70	0.80	0.65	0.75
Complement	0.45	0.50	0.45	0.40	0.40	0.40
Perfect shuffle	0.40	0.45	0.50	0.50	0.40	0.45
Transpose	0.50	0.55	0.55	0.55	0.55	0.55
Hot Spot 1	0.60	0.65	0.70	0.90	0.65	0.85
Hot Spot 2	0.50	0.55	0.50	0.85	0.60	0.80

This cannot happen in an output driven router unless all the other output buffers are full, or no message in the input buffers needs any other output buffer. Thus, the benefit of output over input driven routers should be less pronounced in oblivious algorithms where a message only waits on one virtual channel, particularly those with fewer virtual lanes. The data supports this hypothesis for the torus and for the mesh which is presented later.

Routing the complement permutation using the *-channels algorithm is the one case where the output driven saturation point did not equal or exceed the input driven saturation load. The complement is unusual since dimension order routing helps prevent conflicts in this traffic pattern, i.e. when two messages compete for the same output buffer. See [10] for an explanation. For the *-channels algorithm the input driven router has a slight advantage since it prefers the lowest dimension output buffer needed, while the output driven algorithm selects a random message that needs the current output buffer. However, the second set of experiments show this advantage disappears when the input driven algorithm picks from the needed output buffers at random.

Table 2 presents the saturation loads for the 256-node mesh and shows that for the mesh, output driven routers are superior to input driven routers. The advantage ranges from zero to twenty percent. Again, the only exception is the complement traffic for the *-channels and minimal triplex adaptive routers.

For the mesh, the oblivious algorithm was also simulated with a single virtual channel (vc) per channel, i.e. with no extra lanes. In this case, a message waiting in an input buffer needs exactly one output buffer. Therefore, the input and output driven algorithms do not make different routing decisions. Rather, the difference in performance is due to the ordering of messages and the router's ability to keep the channels busy. As hypothesized, the difference in saturation points between input and output driven routers is very modest, non-existent in four traffic patterns and within a normalized load of .05 for the remaining ones. For the two lane oblivious routers, the difference is more distinct. The output driven router has a higher saturation point than the input driven router in all but one of the traffic patterns, and the maximum difference in saturation points is .15.

Figures 1 and 2 in the Appendix show the expected throughput and latency versus the normalized applied load for each of the traffic patterns on the 256-node torus. Results for the second hot spot case are similar to the first hot spot case and have been

Table 2. Minimum normalized applied load within .05 at which saturation is detected.

16x16 Mesh Saturation, input driven is fixed order								
Traffic	oblivious (1 vc)		oblivious		*-channels		min triplex	
	input	output	input	output	input	output	input	output
Random	0.85	0.90	0.90	0.95	0.95	0.95	0.90	0.90
Bit reversal	0.50	0.50	0.50	0.50	0.80	0.80	0.70	0.75
Complement	0.45	0.50	0.45	0.50	0.45	0.35	0.45	0.35
Perfect shuffle	0.75	0.75	0.75	0.90	0.80	0.95	0.80	0.90
Transpose	0.50	0.50	0.50	0.55	0.85	0.85	0.85	0.85
Hot Spot 1	0.75	0.75	0.75	0.80	0.80	0.85	0.75	0.85
Hot Spot 2	0.70	0.70	0.70	0.75	0.75	0.85	0.75	0.85

omitted. Throughput represents messages delivered per cycle but is normalized to reflect bandwidth limitations of the network. Latency measures time in the network and does not include source queueing, since after saturation, the source queue length is not well defined.

Initially the input and output driven routers are indistinguishable. Nevertheless, output driven routers achieve a higher or equivalent throughput than the corresponding input driven router for all the traffic patterns and routers simulated. As with the saturation data, the complement traffic is the only exception. For the minimal triplex router, the peak throughput of the output driven router under the complement traffic does not quite reach that of the input driven router.

The expected latency curves have three phases. Initially input and output schemes have equivalent latencies. Any router and routing decision will do when the router is lightly loaded. When the applied load is in the neighborhood of saturation and the network is congested, the output driven routers exhibit a lower latency (and latency variance) than the input driven router. We believe this is because the output driven router is doing a better job at keeping the physical channels utilized. Finally after saturation, the latencies are less predictable, but often tend to converge. After saturation the network cannot keep up with the message arrivals; and again, any type of routing will do. The oblivious router is an exception since the sustained higher throughput of the output driven router results in higher latencies.

4.2 Random Output Buffer Selection

In order to validate our conjecture that the disadvantage of input driven routers is not from the deterministic search for available output buffers, the following changes were made to make the input driven routers as similar as possible to the output driven routers. Each input driven router was modified so that it selects a needed output buffer at random, instead of choosing the lowest dimension output buffer available. The output driven routers were unchanged and choose messages from the input buffers at random. Experiments showed the modified input driven algorithms to have better channel utilization resulting in improved performance[1].

[1] It is likely that round-robin selection could approximate this improvement.

Table 3 compares the saturation points of the input and the output driven routers for the mesh and torus. The change from fixed order output buffer selection to random does not change the routing decisions in the one lane oblivious input driven algorithm. The change between the two lane oblivious input driven algorithms is small; the two routers may select a different lane to route a message in the input buffer. Nevertheless, there is no noticeable change in saturation points in either the oblivious mesh or torus routers, most likely since the lanes share the same underlying physical channel.

Table 3. Minimum normalized applied load within .05 at which saturation is detected.

16x16 Network Saturation, input driven is random												
	Mesh						Torus					
Traffic	oblivious		*-channels		min triplex		oblivious		*-channels		min triplex	
	input	output	input	output	input	output	input	output	input	output	input	output
Random	0.90	0.95	0.95	0.95	0.90	0.90	0.70	0.80	0.95	0.95	0.85	0.85
Bit reversal	0.50	0.50	0.80	0.80	0.75	0.75	0.40	0.50	0.80	0.80	0.70	0.75
Complement	0.45	0.50	0.35	0.35	0.35	0.35	0.45	0.50	0.40	0.40	0.35	0.40
Perfect shuffle	0.75	0.90	0.95	0.95	0.90	0.90	0.40	0.45	0.50	0.50	0.45	0.45
Transpose	0.50	0.55	0.80	0.85	0.75	0.85	0.50	0.55	0.55	0.55	0.55	0.55
Hot Spot 1	0.75	0.80	0.90	0.85	0.85	0.85	0.60	0.65	0.85	0.90	0.80	0.85
Hot Spot 2	0.70	0.75	0.90	0.85	0.85	0.85	0.50	0.55	0.75	0.85	0.75	0.80

For the adaptive routers the difference between the saturation points of the input and output driven algorithms on the mesh and torus has nearly been eliminated for almost all the traffic patterns. The input driven routers with random output buffer selection are superior to those with fixed order output selection, since removing the bias of fixed order selection improves the utilization of the physical channels of a node. The only exception is the complement. The complement, as mentioned earlier, prefers dimension order routing, and hence saturates at a lower load than previously for both adaptive routers on the mesh and torus.

For the hot spot traffic, the input driven router with random selection has higher saturation point than the output driven router, even though the two routers achieve the same throughput. The input driven router is able to prevent congestion around the hot spots from spreading as quickly into the whole network compared to the output driven router. The reasons for this are unclear, but most likely related to the lack of wrap edges in the mesh since the torus does not exhibit this behavior.

Due to space limitations, the expected throughput and latency of the input and output driven algorithms on the torus are not presented, but can be found in [10]. The oblivious input driven algorithm does not exhibit significant change between the fixed order and random selection of output buffers, except for a decrease in latency for some of the traffic patterns including the complement, perfect shuffle, and transpose.

The input driven adaptive routers improve their maximum achieved throughput, and in some cases match the throughput of the corresponding output driven router, as with *-channels under bit reversal and random traffic and triplex under random traffic. For

most cases there is also a substantial decrease in the after saturation latency of the input driven algorithms for all the traffic patterns. In addition, the steep increase in latency is delayed and occurs at a higher applied load for random, bit reversal, perfect shuffle, and hot spot traffic.

5 Related Work

Most of the work in routing algorithms has been in developing deadlock-free algorithms. Numerous frameworks have been presented for developing deadlock-free algorithms with varying complexity, resource requirements, and switching techniques [8, 17, 9, 13, 6, 19]. Each of these factors influences the overall performance of the router. Nevertheless, only a few studies have been devoted to improving performance or comparing various implementations of a routing algorithm. Dally increased throughput of wormhole algorithms by adding virtual channels to separate the buffering resources from the transmission resources of the router [7]. Konstantinidou reduced overall message latency in bimodal length traffic by introducing segment routing [15]. Segment routing provides a separate buffer for large messages, allowing small messages to pass larger ones. Cherkasova and Rokicki replaced FIFO injection, the traditional method of introducing messages into the network, with alpha scheduling [5]. With variable length messages, alpha scheduling approximates the optimal average message latency of shortest first scheduling without introducing starvation. Finally, a study by Glass and Ni compared the performance of various policies for selecting input and output buffers for two different routing algorithms on the mesh [12].

6 Conclusions

We have experimentally compared the performance of input and output driven algorithms on the mesh and torus. Although the two are conceptually similar, for almost all the cases examined, the performance of the output driven algorithms is equivalent or superior to that of the input driven algorithms. The difference is diminished when randomization is added to the output buffer selection of the input driven algorithm. Although the findings presented only apply to the routers considered, we believe that the results can be generalized to routers where the designer is indifferent to which approach to use. Future work may compare input and output driven routers using longer messages or a non-minimal router.

A Appendix

Assuming the nodes are labeled in row major order from 0 to 255, the hot spot nodes for case 1 are 158, 186, 216, 236, 121, 86, 6, 152, 201, and 123. For case 2 they are 51, 92, 254, 140, 51, 70, 201, 155, 124, and 245.

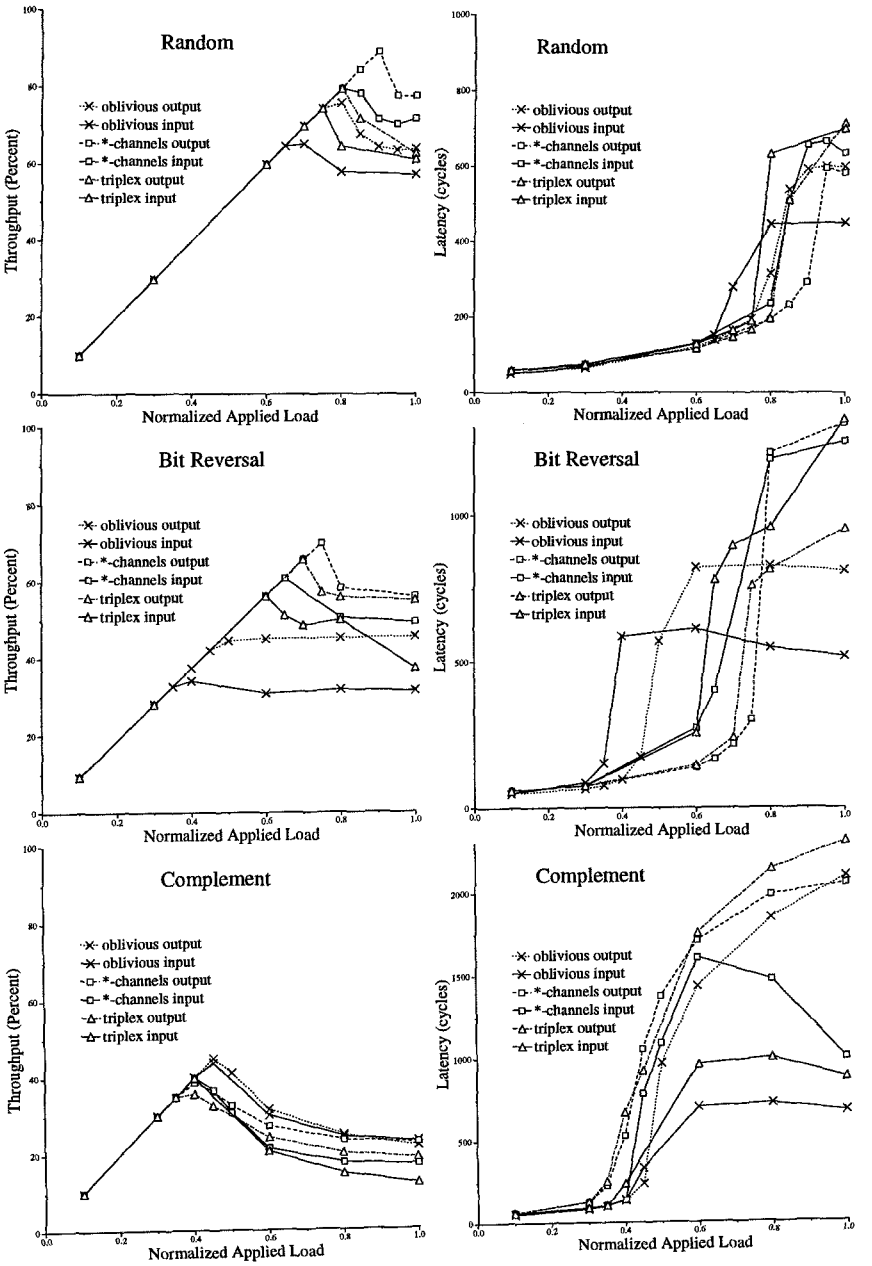

Fig. 1. Throughput and latency on a 256-node 2D torus with fixed output buffer selection.

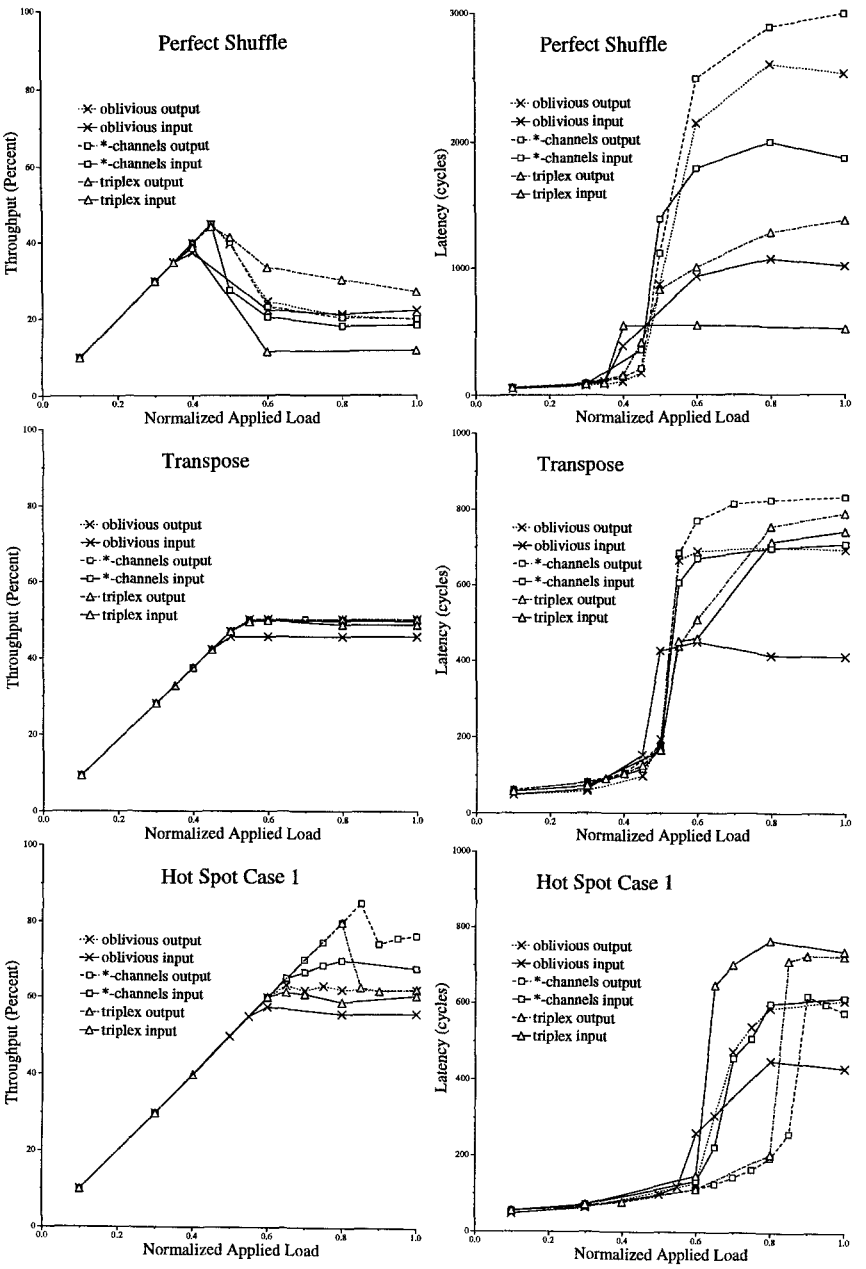

Fig. 2. Throughput and latency on a 256-node 2D torus with fixed output buffer selection.

References

1. P. Berman, L. Gravano, G. Pifarré, and J. Sanz. Adaptive deadlock- and livelock-free routing with all minimal paths in torus networks. In *Proc. of SPAA*, 1992.
2. M. Blumrich, K. Li, R. Alpert, C. Dubnicki, and E. Felten. Virtual memory mapped network interface for the SHRIMP multicomputer. In *Proc. of ISCA*, pages 142–153, 1994.
3. K. Bolding. *Chaotic Routing: Design and Implementation of an Adaptive Multicomputer Network Router.* PhD thesis, University of Washington, Seattle, July 1993.
4. K. Bolding, M. Fulgham, and L. Snyder. The case for chaotic adaptive routing. *IEEE Transactions on Computers*, to appear.
5. L. Cherkasova and T. Rokicki. Alpha message scheduling for packet-switched interconnects. Technical Report TR HPL-94-72, Hewlett-Packard Labs, 1994.
6. A. Chien and J. Kim. Planar-adaptive routing: Low-cost adaptive networks for multiprocessors. In *Proc. of the Intl. Sym. on Computer Architecture*, pages 268–277, 1992.
7. W. Dally. Virtual–channel flow control. In *Proc. of the Intl. Sym. on Comp. Arch.*, May 1990.
8. W. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, C-36(5):547–553, May 1987.
9. J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Transactions on Parallel and Distributed Systems*, 4(4):466–475, Apr. 1993.
10. M. Fulgham and L. Snyder. A comparison of input and output driven routers. Technical Report UW-CSE-96-06-01, Univ. of Washington, Seattle, 1996.
11. M. Fulgham and L. Snyder. Triplex router: a versatile torus routing algorithm. Technical Report UW-CSE-96-01-11, Univ. of Washington, Seattle, 1996.
12. C. Glass and L. Ni. Adaptive routing in mesh-connected networks. In *Proc. of the Intl. Conf. on Distributed Computing Systems*, pages 12–19, 1992.
13. C. J. Glass and L. M. Ni. The turn model for adaptive routing. *JACM*, 41(5):874–902, 1994.
14. P. Kermani and L. Kleinrock. Virtual cut-through: A new computer communication switching technique. *Computer Networks*, 3:267–286, 1979.
15. S. Konstantinidou. The segment router: a novel router design for parallel computers. In *Proc. of the Sym. of Parallel Algorithms and Architectures*, pages 364–373, 1994.
16. S. Konstantinidou and L. Snyder. The chaos router. *IEEE Transactions on Computers*, 43(12):1386–97, Dec. 1994.
17. D. H. Linder and J. C. Harden. An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes. *IEEE Transactions on Computers*, C-40(1):2–12, Jan. 1991.
18. N. McKenzie, K. Bolding, C. Ebeling, and L. Snyder. CRANIUM: An interface for message passing on adaptive packet routing networks. In *Lecture Notes in Computer Science*, volume 853, pages 266–80, 1994.
19. L. Schwiebert and D. Jayasimha. A universal proof technique for deadlock-free routing in interconnection networks. In *Proc. of the Sym. on Par. Alg. and Arch.*, pages 175–184, 1995.
20. T. von Eicken D.E. Culler, S. Goldstein, and K. Schauser. Active messages: a mechanism for integrated communication and computation. In *Proc. of ISCA*, pages 256–266, May 1992.