Support for Multiple Classes of Traffic in Multicomputer Routers *

Jennifer Rexford and Kang G. Shin

Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122

Abstract. Emerging parallel real-time and multimedia applications broaden the range of performance requirements imposed on the interconnection network. This communication typically consists of a mixture of different traffic classes, where *guaranteed* packets require bounds on latency or throughput while good average performance suffices for the *best-effort* traffic. This paper investigates how multicomputer routers can capitalize on low-latency routing and switching techniques for best-effort traffic while still supporting guaranteed communication. Through simulation experiments, we show that certain architectural features are best-suited to particular performance requirements. Based on these results, the paper proposes and evaluates a router architecture that tailors low-level routing, switching, and flow-control policies to the unique needs of best-effort and guaranteed traffic. Careful selection of these policies, coupled with fine-grain arbitration between the classes, allows the guaranteed and best-effort packets to share network bandwidth without sacrificing the performance of either class.

1 Introduction

Although multicomputer router design has traditionally emphasized providing low-latency communication, modern parallel applications require additional services from the interconnection network [1, 2]. Multimedia and real-time applications, such as scientific visualization and process control, necessitate control over delay variance and throughput, in addition to low average latency [3]. While *guaranteed* traffic necessitates deterministic or probabilistic bounds on throughput or end-to-end delay, *best-effort* service often suffices for the remaining traffic. For example, control or audio/video messages may mandate explicit performance guarantees, while data transfer may tolerate delay variability in exchange for improved average latency.

* The work reported in this paper was supported in part by the National Science Foundation under Grant MIP-9203895 and an Office of Naval Research graduate fellowship. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

Handling this mixture of disparate traffic classes affects the suitability of architectural features in multicomputer routers. While the router alone cannot satisfy application performance requirements, design decisions should not preclude the system from providing necessary guarantees. Servicing guaranteed traffic requires control over network access time and bandwidth allocation, so the router should bound the influence best-effort packets have on these parameters. The software, or even the hardware, can then utilize these bounds to satisfy quality-of-service requirements through packet scheduling and resource allocation for communicating tasks. Additionally, the design should not unduly penalize the performance of best-effort packets.

Modern parallel routers significantly reduce average latency by avoiding unnecessary packet delay at intermediate nodes; however, these low-latency techniques often impinge on control over packet scheduling. In particular, cut-through switching [4, 5] decentralizes bandwidth allocation and packet scheduling by allowing an incoming packet to proceed directly to the next node in its route if a suitable outgoing link is available. Other multicomputer router features, such as FIFO queueing and adaptive routing, further complicate the effort to provide predictable or guaranteed service.

Research in networking considers techniques for the effective mixing of multiple traffic classes in a communication fabric [6, 7]. However, the design trade-offs for parallel machines differ significantly from those in a heterogeneous, distributed environment. In parallel machines, router design trade-offs reflect the large network size and the tight coupling between nodes. Speed and area constraints motivate single-chip solutions, including designs that integrate the processing core and the communication subsystem [8–11]. Regular topologies facilitate efficient offset-based routing, avoiding the costs of implementing and maintaining a table-driven scheme at each node.

While these implementation constraints restrict some router design options, the tighter coupling between nodes enables multicomputer routers to consider more diverse routing and switching techniques for handling different traffic classes. The shorter, wider communication links in most parallel machines result in much lower packet transmission delays, compared to distributed systems. These low-latency channels broaden the spectrum of flow-control schemes that can be implemented efficiently. The fine-grain interaction between and within the nodes necessitates effective mapping of application tasks onto the interconnection network [12]. Effective router techniques for handling multiple traffic classes should provide useful software abstractions to parallel applications.

The remainder of this paper is organized as follows. Section 2 presents a simulation model for studying the impact of routing and switching on interconnection network performance. Using this model, Section 3 investigates how switching schemes affect the network's ability to service multiple traffic classes. Based on these results, Section 4 proposes and evaluates a router architecture that allows best-effort packets to capitalize on low-latency routing and switching techniques without compromising the performance of guaranteed traffic.

This architecture uses virtual channels [13] to logically partition the inter-

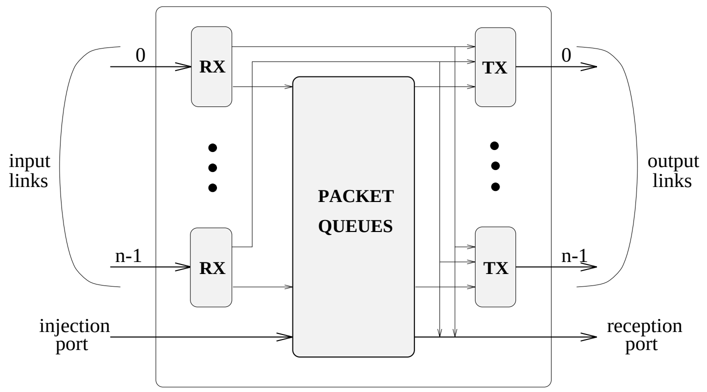

Fig. 1. Router model

connection network into multiple virtual networks that each employs different low-level policies for managing communication. Fine-grain, demand-slotted arbitration between the virtual networks regulates the intrusion of best-effort traffic on guaranteed packets, while allowing packets from either class to consume available link bandwidth. By tailoring the routing, switching, and flow-control policies for each virtual network, this architecture can accommodate the diverse performance requirements of parallel real-time and multimedia applications.

2 Evaluation Framework

The efficacy of a router architecture hinges on the policies for routing, switching, queueing, and resource arbitration. Supporting both guaranteed and best-effort traffic in a single design requires careful consideration of these low-level policies and how they influence the interaction between traffic classes. Evaluating design options for multicomputer routers requires the ability to vary low-level architectural parameters in a single unified framework. This section presents a flexible router model and simulation environment, which is used in Sections 3 and 4 to address effective architectural support for multiple traffic classes.

Packets enter the router from an injection port and n incoming links and depart the router through the reception port and n outgoing links, as shown in Figure 1. Each physical link multiplexes traffic for c virtual channels at the granularity of a flit cycle, while the injection and reception ports handle packets on behalf of the nc outgoing and incoming virtual channels, respectively. A crossbar connects the packet buffers and the incoming channels to the output links. Although each physical link services at most one virtual channel in each flit cycle, multiple virtual channels can be active at the injection and reception ports; this enables the model to represent router designs that have multiple physical or logical injection/reception ports [10, 14, 15].

Upon receiving the header flits of an incoming packet, the receiver (RX) decides whether to buffer, stall, or forward the packet, based on the routing and switching policies and prevailing network conditions. By treating outbound virtual channels as individually reservable resources, the model can invoke a variety of routing and switching schemes through flexible control over reservation policies. The routing algorithm selects candidate outgoing virtual channels, while the switching scheme determines whether or not an incoming packet waits to acquire a selected outgoing virtual channel or buffers instead. Once a packet reserves an outgoing virtual channel, it competes with other virtual channels for access to the physical link (TX) through an arbitration policy. The model includes several arbitration policies, including round-robin and priority-driven schemes.

The router model is evaluated in the pp-mess-sim (point-to-point message simulator) environment [16, 17]. Implemented in C++, pp-mess-sim is an object-oriented discrete-event simulation tool for evaluating multicomputer router architectures. Using a high-level specification language, the user can select the network topology, internal router policies, and the traffic patterns generated by each node. These communication patterns stem from a collection of independent traffic classes, each with its own performance metrics and packet characteristics. The simulator allows the derivation of packet length, interarrival times, and target nodes from a variety of stochastic processes.

To evaluate traffic mixing, the simulator associates each traffic class with a particular routing algorithm and switching scheme on a set of virtual channels. The tool includes an extensible set of routing-switching algorithms that interact with the router model through a well-defined set of instructions. This enables specification of routing-switching combinations separate from the router model. These algorithms can formally query the status of the router in order to execute state-dependent routing and switching decisions. The simulator supports wormhole, virtual cut-through, and packet switching, as well as hybrid schemes, each under a variety of routing algorithms.

The experiments in this paper evaluate an 8×8 torus (8-ary 2-cube) network carrying 16-flit packets using dimension-ordered routing; similar performance trends occur for other network and message sizes. Each traffic class independently generates packets at each node with exponentially-distributed inter-arrival times and uniform random selection of destination nodes. The simulator collects performance data only after receiving at least 200 packets from each traffic class on each source node to allow the network to reach steady state. Each traffic class then accumulates performance data for 2000 packets from each node, with each source continuing to generate packets until data collection completes throughout the network. For all results shown in the paper, the standard error of average latency is less than 10 cycles for the 95% confidence interval.

3 Evaluation of Switching Schemes

In defining how packets flow through the network, the various switching schemes use different resources at nodes along a packet's route. This section evaluates the ability of wormhole, virtual cut-through, and packet switching to meet different

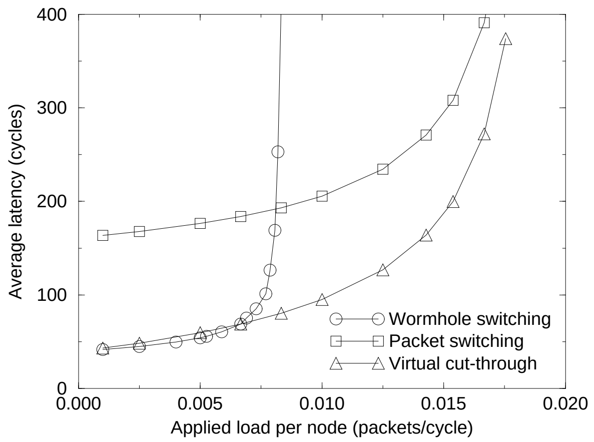

Fig. 2. Average packet latency

performance requirements in multicomputer routers. Each switching scheme is best-suited for certain traffic classes with particular characteristics and performance requirements [14, 18]. To effectively support multiple traffic classes, the router should bound both network access time and the service rate for guaranteed packets. These bounds provide necessary abstractions for the scheduling and mapping of communicating tasks. Best-effort packets, on the other hand, may forego these restrictions in exchange for lower latency and reduced buffer requirements.

3.1 Average Latency

Traditional *packet switching* requires an arriving packet to buffer completely before transmission to a subsequent node can begin. In contrast, cut-through switching schemes, such as *virtual cut-through* [4] and *wormhole* [5], try to forward an incoming packet directly to an idle output link. If the packet encounters a busy outgoing channel, virtual cut-through switching buffers the packet, while a blocked wormhole packet stalls pending access to the link. While first-generation multicomputers employed packet switching, most existing research and commercial routers utilize cut-through switching for lower latency and reduced buffer space requirements [19]. The usage of memory and link resources determines both average packet latency and the influence an in-transit packet can have on other network traffic.

Figure 2 shows the average end-to-end packet latency for the three switching schemes as a function of the packet injection rate. In the simulation experiments, virtual cut-through and packet switching utilize one virtual channel for each physical link and store buffered packets in output queues in the router. Wormhole packets employ deadlock-free routing on a pair of virtual channels [20] with demand-driven, round-robin arbitration amongst the virtual channels; each virtual channel can hold a single flit pending access to the output link. Even with this small amount of memory resources, wormhole switching performs well at low loads, slightly outperforming virtual cut-through switching. At high loads, virtual cut-through and packet switching performance gradually merge, as high network utilization decreases the likelihood that an in-transit packet encounters an idle output link.

By removing blocked packets from the network, virtual cut-through and packet switching consume network bandwidth proportional to the offered load. In contrast, a blocked wormhole packet stalls in the network, effectively dilating its length until its outgoing channel becomes available. As a result, wormhole networks typically utilize only a fraction of the available network bandwidth [13, 21], as seen by the early saturation of the wormhole plot in Figure 2. At higher loads, this effect enables packet switching to outperform wormhole switching, even though packet switching introduces buffering delay at each hop in a packet's route. While adding virtual channels can increase wormhole throughput [13], channel contention still creates dependencies amongst packets spanning multiple nodes.

The sensitivity of wormhole networks to slight changes in load, including short communication bursts [22], complicates the use of wormhole switching for guaranteed traffic. Still, wormhole switching is particularly well-suited to best-effort packets, due to its low latency and minimal buffer space requirements. While flow-control costs limit the utility of wormhole switching in distributed systems, parallel machines can dynamically transfer or stall wormhole flits without complicating buffer allocation for other traffic. Section 4 describes how, with effective flow-control and arbitration schemes, best-effort packets can employ wormhole switching without compromising the performance of the guaranteed traffic.

3.2 Predictability

While the router should provide low average latency for best-effort packets, guaranteed communication requires predictable network delay and throughput. Figure 3 shows the coefficient of variation for packet latency for the three switching schemes, where the coefficient of variation measures the ratio of the standard deviation to the mean [23]. Since latency characteristics vary depending on the distance between source-destination pairs, the graph shows results only for packets traveling a fixed distance in the network. While each source generates traffic with uniform random selection of destination nodes, data collection for Figure 3 includes only packets traveling exactly five hops.

Across all loads, packet switching incurs the least variability since packets deterministically buffer at intermediate nodes. Coupled with static routing, a packet-switched transfer utilizes deterministic memory and channel resources at fixed nodes and links along the route. This greatly simplifies the allocation and scheduling of resources throughout the interconnection network. In contrast, virtual cut-through imparts variable load on memory resources at intermediate

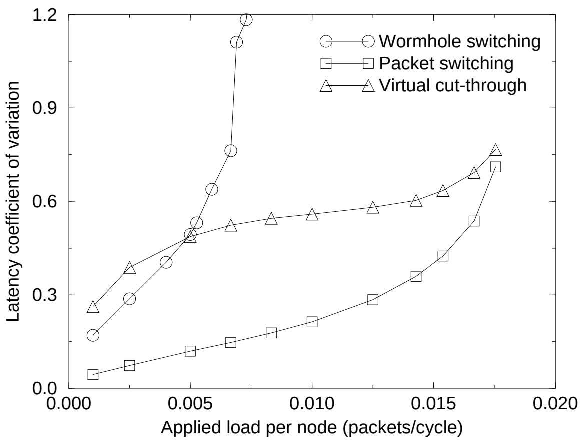

Fig. 3. Latency coefficient of variation (5-hop packets)

nodes by basing the buffering decision on the status of the output links. At high loads, virtual cut-through and packet switching merge, as in Figure 2, due to the decreasing likelihood of packet cut-throughs.

Wormhole switching, though conceptually similar to virtual cut-through, has quite different characteristics. Since a blocked wormhole packet never buffers, it imparts no memory demands on intermediate nodes, but instead consumes unpredictable amounts of channel bandwidth. In Figure 3, wormhole latency variation increases dramatically with rising load, even under a moderate injection rate below saturation throughput. Below the saturation load, wormhole switching results in a low average latency, as seen in Figure 2, but a portion of the traffic incurs larger delay due to pockets of channel contention and the small amount of buffer resources. In addition to a large coefficient of variation, wormhole traffic suffers a large standard deviation of packet latency, as shown in Figure 4.

Depending on the number of active virtual channels at each link, flits within a single wormhole packet may encounter different service rates. Demand-driven arbitration for access to the physical links, while important for low average latency, complicates the effort to export a predictable flit or packet service rate to a static or run-time scheduling algorithm. While adding virtual channels can reduce contention, additional virtual channels also increase the potential variability in the number of flits awaiting access to each physical link, further complicating the flit service rate.

3.3 Packet Scheduling

The router must have control over packet scheduling and bandwidth allocation to ensure that guaranteed packets meet their latency and bandwidth requirements.

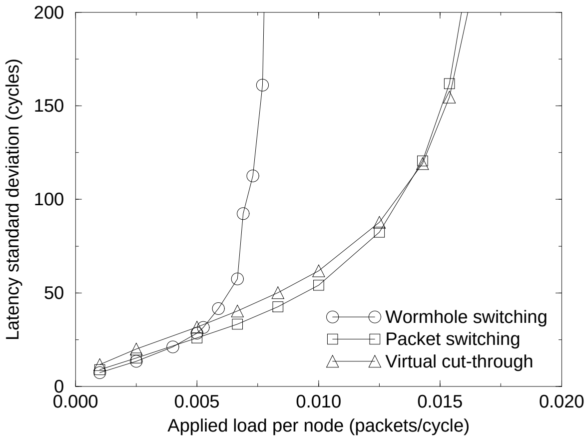

Fig. 4. Latency standard deviation (5-hop packets)

Virtual cut-through and packet switching generate physical queues in each node, facilitating priority-based scheduling amongst competing packets. In contrast, stalled wormhole packets form logical queues spanning multiple nodes. While these decentralized queues complicate packet scheduling, a wormhole router can influence resource allocation through the virtual channel reservation and arbitration policies. Priority assignment of virtual channels to incoming packets improves predictability; adaptive arbitration policies can further reduce variability by basing flit bandwidth allocation on packet deadlines or priority [13, 24–26].

While assigning priorities to virtual channels provides some control over packet scheduling, this ties priority resolution to the number of virtual channels. If packets at different priority levels share virtual channels, the application must account for blocking time when a lower priority packet holds resources needed by higher priority traffic. While adding more virtual channels can improve priority resolution, this also incurs increased latency overhead and implementation complexity for the router [27]. In addition, the router must enforce the multiple priority levels at its injection and reception ports to avoid unpredictable stalling at the network entry and exit points.

Providing separate buffers for each priority level is effective for coarse-grain priority assignment, but this approach incurs significant cost for fine-grain resolution. With *packet* queues at each node, the router can effectively utilize fine-grain priorities, such as deadlines, to assign access to output links [7, 28]. Instead of providing separate logic and buffer space for each priority level, the router can include a single priority queue for each output link [29, 30]. By buffering packets at each node, packet switching enables the router to schedule traffic to provide latency or bandwidth guarantees [28]. For example, suppose a guaranteed packet enters an intermediate node well in advance of its deadline. The scheduler may wish to detain this packet, even if its outgoing link is available, to avoid unexpectedly overloading the subsequent node.

4 Controlled Traffic Mixing

Best-effort and guaranteed traffic have conflicting performance goals that complicate interconnection network design. The effective mixing of guaranteed and best-effort traffic hinges on controlling the interaction between these two classes. In particular, best-effort packets cannot consume arbitrary amounts of link or buffer resources while guaranteed packets await service.

4.1 Router Architecture

As seen in Section 3, wormhole and packet switching exercise complementary resources in the interconnection network, with wormhole switching reserving virtual channels and packet switching consuming buffers in the router. Hence, the combination of wormhole switching for best-effort traffic and packet switching for guaranteed communication enables effective partitioning of router resources. However, since the traffic classes share network bandwidth, the router must regulate access to the physical links to control the interaction between the two classes.

Assigning the best-effort and guaranteed packets to separate virtual networks can regulate this interaction between the traffic classes. The router divides each physical link into multiple virtual channels, where some virtual channels carry best-effort packets and the rest accept only guaranteed traffic. Virtual channels provide an effective mechanism for reducing the interaction between packets while still allowing traffic to share network bandwidth [8–10, 14, 31]. Exporting the virtual channel abstraction to the injection and reception ports further prevents intrusion between packets at the network entry and exit points [10, 14, 15].

By tailoring the routing, switching, and flow-control policies for each virtual network, multicomputer routers can support traffic classes with conflicting performance requirements. Packets on separate virtual networks interact only to compete for access to the physical links and ports. This bounds network access time for guaranteed packets, independent of the amount or length of best-effort packets. The communication software, or hardware, can then build on these underlying abstractions to provide various services, such as connection-oriented communication with latency or bandwidth guarantees. Fine-grain flow control on the wormhole virtual network enables best-effort flits to capitalize on slack link bandwidth left unclaimed by guaranteed packets.

4.2 Fair Arbitration

Figures 5, 6, and 7 evaluate the effect of increasing best-effort load on the performance of both best-effort and guaranteed traffic in this router architecture. In these experiments, the router interleaves three virtual channels on each link,

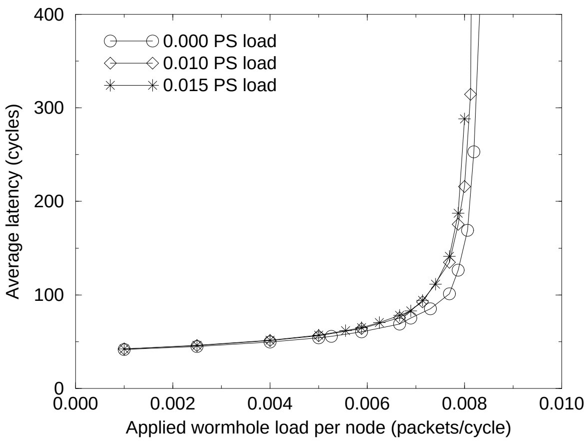

Fig. 5. Average wormhole latency

with two virtual channels allocated to best-effort packets for deadlock-free wormhole routing and one dedicated to guaranteed traffic using packet switching. Each curve shows the impact of changing best-effort load in the presence of a fixed rate of injection for guaranteed packets. The router employs round-robin arbitration amongst the active virtual channels contending for each link.

Figure 5 shows the average latency for the best-effort, wormhole packets, under three different injection rates for the packet-switched (PS) traffic. Note that the curve for zero packet-switched load corresponds to the wormhole latency data in Figure 2. As the amount of wormhole traffic increases, best-effort packets incur larger latency due to increased channel contention within the best-effort virtual network. Even with fairly heavy packet-switching load, the best-effort packets maintain low average latency until reaching the saturation throughput. The presence of packet-switched traffic does not significantly limit this achievable best-effort throughput, since the wormhole virtual network saturates due to virtual channel contention, not a shortage of network bandwidth.

As seen in Figures 6 and 7, both the average latency and predictability of the guaranteed packets are largely unaffected by the best-effort traffic, due to fine-grain arbitration amongst the virtual channels. For both packet-switched loads, the mean and standard deviation of end-to-end latency closely match the corresponding values in Figures 2 and 4, even as the wormhole traffic exceeds its sustainable load. Channel contention on the best-effort virtual network does not impede the forward progress of guaranteed packets, since blocked wormhole packets temporarily stall in their own virtual network instead of depleting physical link or buffer resources. Demand-driven arbitration ensures that either class of traffic can improve throughput by capitalizing on the available link bandwidth.

While the separate virtual networks limit the interaction between the traf-

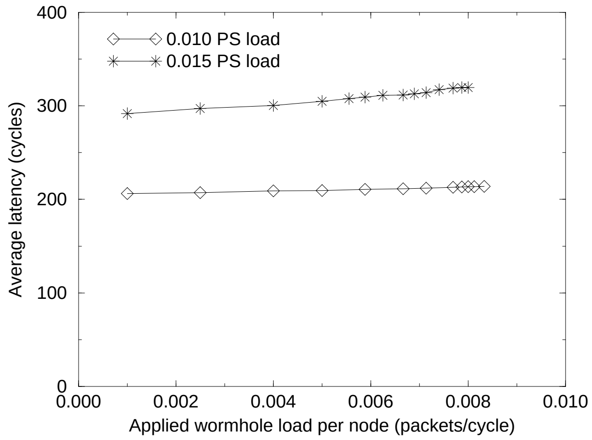

Fig. 6. Average packet switching latency

fic classes, the arbitration for access to the physical link still permits active best-effort virtual channels to increase delay for the guaranteed packets. This is manifested in Figures 6 and 7 by the slight increase in packet-switching latency and standard deviation in the presence of a heavier load of wormhole traffic. More significantly for the guaranteed traffic, fair arbitration amongst the virtual channels varies the service rate afforded both traffic classes, providing slower guaranteed service under increasing best-effort load.

4.3 Tighter Bounds for Guaranteed Traffic

The router can further minimize intrusion on guaranteed traffic by imposing priority arbitration between the virtual networks, where guaranteed packets always win arbitration over the best-effort packets. For a guaranteed packet, this effectively provides flit-level preemption of best-effort traffic across its entire path through the network. Unlike the results in Figures 6 and 7, assigning priority to guaranteed traffic removes any sensitivity to the best-effort load. Priority arbitration enables a guaranteed packet to travel at the same rate through each link in its journey, independent of the number of active best-effort virtual channels. This abstraction enables the scheduler to allocate resources based only on the worst-case requirements of the guaranteed traffic, while still enabling best-effort traffic to dynamically consume unused link bandwidth.

However, priority arbitration can exact a heavy toll on the best-effort packets, particularly at higher loads, as illustrated by Figure 8. This graph shows the average latency of best-effort wormhole packets in the presence of three different packet-switching loads under priority arbitration for the physical links. Unlike Figure 5, Figure 8 shows significant degradation of the performance of best-effort packets, since the strict priority-based scheme restricts their forward progress.

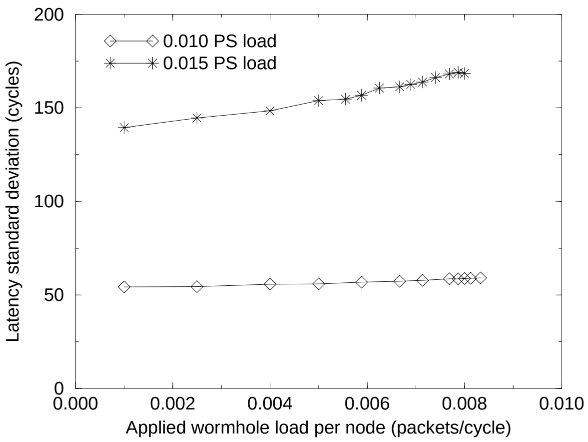

Fig. 7. Packet switching latency standard deviation

Even in the absence of livelock, lengthy blocking of wormhole flits increases contention delays in the best-effort virtual network.

Priority arbitration varies the service rate for the best-effort packets depending on the load of guaranteed traffic. To reduce contention, the best-effort virtual networks could employ adaptive routing to enable these packets to circumvent links and nodes serving a heavy load of guaranteed packets. Alternatively, the router could aid the forward progress of best-effort packets by ensuring predictable access to the physical link, even in the presence of guaranteed packets. The router can allow up to α best-effort flits to accompany the transmission of a guaranteed packet. Since the guaranteed traffic employs packet switching, a guaranteed packet holds the physical link for a bounded time proportional to its packet length ℓ. In effect, this dilates each guaranteed packet to a service time of at most $\ell + \alpha$ cycles, while dissipating contention in the best-effort virtual network. When no guaranteed packets await service, pending best-effort flits have free access to the outgoing link.

This permits forward progress for best-effort packets while still enforcing a tight bound on the intrusion on guaranteed traffic, without restricting packet size. Such a credit-based scheme preserves necessary delay abstractions for the scheduling of guaranteed traffic. For additional flexibility, a writeable register in each router would allow the system to set α when downloading tasks to the processing nodes. For example, the compiler could test the schedulability of the guaranteed communication under several candidate α values, selecting an α that does not disrupt the delay or bandwidth bounds for the guaranteed packets. This enables the compiler to determine the appropriate trade-off between the best-effort performance and the admission of guaranteed traffic for a given application.

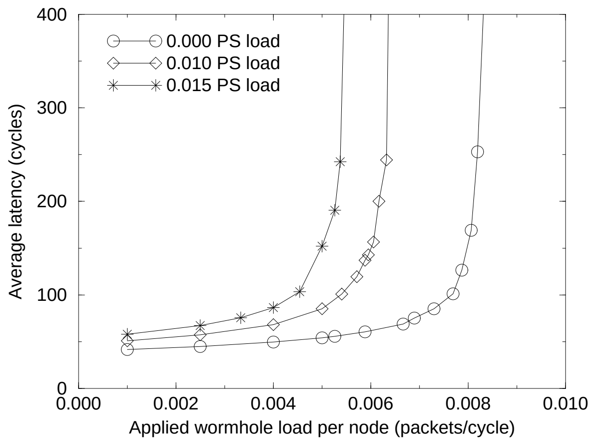

Fig. 8. Average wormhole latency with priority arbitration

5 Conclusions and Future Work

Parallel real-time and multimedia applications impose diverse communication requirements on multicomputer interconnection networks. The conflicting performance goals of best-effort and guaranteed traffic affect the suitability of routing, switching, and flow-control schemes. In this paper, we show that low-level control over routing and switching, coupled with fine-grain arbitration, enables multicomputer routers to effectively mix guaranteed and best-effort communication. This allows best-effort traffic to capitalize on flexible routing and switching schemes that improve average performance, without compromising the predictable, timely delivery of guaranteed packets.

Effective mixing of best-effort and guaranteed traffic requires a combination of low-level hardware support and higher-level protocols. This paper has addressed effective multicomputer router hardware for enabling the development of such higher-level protocols. Arbitration and flow-control schemes enable the router to export bounded network access delay, packet service time, and throughput for guaranteed traffic, even in the presence of best-effort flits. Hardware or software protocols can then build on these abstractions to allocate communication resources and schedule guaranteed packets [6, 7, 28].

Traditionally, real-time systems have employed packet switching, coupled with scheduling algorithms, for predictable performance. However, in tightly-coupled parallel machines, this approach unduly penalizes best-effort packets. As future work, we plan to compare the proposed router architecture to approaches that employ a single switching scheme, such as wormhole, virtual cut-through, or packet switching. Studying realistic communication patterns and scheduling algorithms should lend more insight into the cost-performance trade-offs in the proposed router model.

References

1. D. Cohen, G. G. Finn, R. Felderman, and A. DeSchon, "The use of message-based multicomputer components to construct gigabit networks," *Computer Communication Review*, vol. 23, no. 3, pp. 32–44, July 1993.
2. R. Cypher, A. Ho, S. Konstantinidou, and P. Messina, "Architectural requirements of parallel scientific applications with explicit communication," in *Proc. Int'l Symposium on Computer Architecture*, pp. 2–13, May 1993.
3. D. Ferrari, "Client requirements for real-time communication services," *IEEE Communications Magazine*, pp. 65–72, November 1990.
4. P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique," *Computer Networks*, vol. 3, no. 4, pp. 267–286, September 1979.
5. W. J. Dally and C. L. Seitz, "The torus routing chip," *Journal of Distributed Computing*, vol. 1, no. 3, pp. 187–196, 1986.
6. J. J. Bae and T. Suda, "Survey of traffic control schemes and protocols in ATM networks," *Proceedings of the IEEE*, vol. 79, no. 2, pp. 170–189, February 1991.
7. C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time communication in packet-switched networks," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 122–139, January 1994.
8. W. J. Dally and P. Song, "Design of a self-timed VLSI multicomputer communication controller," in *IEEE Int'l Conf. on Computer Design: VLSI in Computers*, pp. 230–234, 1987.
9. W. J. Dally, J. A. S. Fiske, J. S. Keen, R. A. Lethin, M. D. Noakes, P. R. Nuth, R. E. Davison, and G. A. Fyler, "The Message-Driven Processor: A multicomputer processing node with efficient mechanisms," *IEEE Micro*, pp. 23–39, April 1992.
10. C. Peterson, J. Sutton, and P. Wiley, "iWarp: A 100-MOPS LIW microprocessor for multicomputers," *IEEE Micro*, pp. 26–29,81–87, June 1991.
11. D. Talia, "Message-routing systems for transputer-based multicomputers," *IEEE Micro*, pp. 62–72, June 1993.
12. M. G. Norman and P. Thanisch, "Models of machines and computation for mapping in multicomputers," *ACM Computing Surveys*, vol. 25, no. 3, pp. 263–302, September 1993.
13. W. Dally, "Virtual-channel flow control," *IEEE Trans. Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, March 1992.
14. J. Dolter, S. Daniel, A. Mehra, J. Rexford, W. Feng, and K. G. Shin, "SPIDER: Flexible and efficient communication support for point-to-point distributed systems," Technical Report CSE-TR-180-93, University of Michigan, October 1993. To appear in *Proc. Int. Conf. on Distributed Computing Systems*, June 1994.
15. J. H. Kim and A. A. Chien, "Evaluation of wormhole routed networks under hybrid traffic loads," in *Proc. Hawaii Int'l Conf. on System Sciences*, pp. 276–285, January 1993.
16. J. Dolter, *A Programmable Routing Controller Supporting Multi-mode Routing and Switching in Distributed Real-Time Systems*, PhD thesis, University of Michigan, September 1993.
17. W. Feng, J. Rexford, A. Mehra, S. Daniel, J. Dolter, and K. Shin, "Architectural support for managing communication in point-to-point distributed systems," Technical Report CSE-TR-197-94, University of Michigan, March 1994.

18. J. Rexford, J. Dolter, and K. G. Shin, "Hardware support for controlled interaction of guaranteed and best-effort communication," in *Proc. Workshop on Parallel and Distributed Real-Time Systems*, April 1994.
19. X. Zhang, "System effects of interprocessor communication latency in multicomputers," *IEEE Micro*, pp. 12–15,52–55, April 1991.
20. W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Computers*, vol. C-36, no. 5, pp. 547–553, May 1987.
21. J. Ngai and C. Seitz, "A framework for adaptive routing in multicomputer networks," in *Symposium on Parallel Algorithms and Architectures*, pp. 1–9, June 1989.
22. W. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 4, pp. 466–475, April 1993.
23. R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, Inc., 1991.
24. B. Tsai and K. G. Shin, "Sequencing of concurrent communication traffic in a mesh multicomputer with virtual channels," to appear in *Proc. Int'l Conf. on Parallel Processing*, August 1994.
25. J.-P. Li and M. W. Mutka, "Priority based real-time communication for large scale wormhole networks," in *Proc. International Parallel Processing Symposium*, pp. 433–438, April 1994.
26. J.-P. Li and M. W. Mutka, "Real-time virtual channel flow control," in *Phoenix Conference on Computers and Communication*, April 1994.
27. A. A. Chien, "A cost and speed model for k-ary n-cube wormhole routers," in *Proc. Hot Interconnects*, August 1993.
28. D. D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," in *Proc. Int. Conf. on Distributed Computer Systems*, pp. 300–307, May 1991.
29. H. J. Chao and N. Uzun, "A VLSI sequencer chip for ATM traffic shaper and queue manager," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 11, pp. 1634–1643, November 1992.
30. K. Toda, K. Nishida, E. Takahashi, N. Michell, and Y. Yamaguchi, "Implementation of a priority forwarding router chip for real-time interconnection networks," in *Proc. Workshop on Parallel and Distributed Real-Time Systems*, April 1994.
31. S. Konstantinidou, "Segment router: A novel router design for parallel computers," to appear in *Proc. Symposium on Parallel Algorithms and Architectures*, June 1994.
32. M. W. Mutka, "Using rate monotonic scheduling technology for real-time communications in a wormhole network," in *Proc. Workshop on Parallel and Distributed Real-Time Systems*, April 1994.

This article was processed using the LaTeX macro package with LLNCS style