Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

MULTISTAGE NETWORKS WITH TRAFFIC WITH REAL-TIME CONSTRAINTS

Lance Kurisaki
Tomas Lang

December 1988
CSD-880101

Multistage Networks with Traffic with Real-Time Constraints

Lance Kurisaki and Tomas Lang

Computer Science Department
School of Engineering and Applied Science
University of California, Los Angeles
tel. (213) 825-6835
e-mail tomas@cs.ucla.edu

Abstract

Multistage interconnection networks (MIN) are used in multiprocessor systems to connect processors with other processors or with memory modules. It has been shown that the performance of these networks is satisfactory for moderate uniform traffic. Moreover, they can be used effectively with traffic producing nonuniform traffic spots (NUTS), if diverting switches are utilized instead of the more traditional blocking switches. Better throughput is obtained in all cases if the network is extended by adding intrastage links that provide alternate paths.

In this paper we consider the case in which the network carries the superposition of two types of traffic. One type is the high throughput data and instruction traffic, while the other consists of control packets (for singularity conditions, errors, etc.) and I/O packets. In many instances, this second type is of low throughput but with real-time constraints. In such a case it is necessary to assure low latency for these packets. We consider strategies to control the network to achieve the low latency for real-time traffic, especially in the case where the background traffic produces NUTS.

We evaluate the performance by simulations and conclude that diverting switches, which produce good performance for NUTS, are also suitable for assuring low latency for the real-time traffic, especially when using the displacing mode. Moreover, we show that the network with additional links, which provides the highest throughput in the presence of NUTS, is also suitable for low latency rtt traffic. We also conclude that even for uniform traffic it is not possible to assure low latency for the rtt when using blocking switches.

Key words: Multiprocessors, Multistage networks, Nonuniform traffic spots, Real-time traffic, Diverting switches.

Multistage Networks with Traffic with Real-Time Constraints

Lance Kurisaki and Tomas Lang

Computer Science Department
School of Engineering and Applied Science
University of California, Los Angeles

1. Introduction

Multistage interconnection networks (MIN) are used in multiprocessor systems to connect processors with other processors or with memory modules. These networks provide a compromise between networks of low latency and high cost, such as the crossbar, and networks of high latency and low cost, such as the shared bus. Moreover, MINs can be pipelined to provide a bandwidth comparable to that of the crossbar for suitable traffic patterns. In addition, the control of routing is simple. A large body of work has been done on the structure, operation, and performance of these networks; a comprehensive reference is [1]. These networks were initially introduced for use in array computers of the SIMD type; in this context the interconnection networks are sometimes called permutation networks. More recently, they are being proposed and used in multiprocessors of the MIMD type, especially of the shared-memory variety [2,3,4,5]. In this paper we are concerned with this second type of use.

In their basic form, these multistage networks provide a unique path between any source-destination pair. However, the paths for different pairs are not disjoint and, therefore, conflicts might occur when simultaneous communication is established between several source-destination pairs. The basic method used to handle this problem is to use a packet-switched type of operation and to buffer the packets in the switches. Blocking occurs whenever the buffers become full.

It has been shown that the performance of these networks is satisfactory for uniform traffic [6,7], that is, for traffic in which the destinations are generated by a random variable with uniform distribution. In [8] we have discussed the effect of nonuniform traffic spots (NUTS) and determined by simulations that the throughput is badly degraded when blocking switches are used. Moreover, we showed that the use of diverting switches improves the performance and that adding links to provide alternate paths produces the best throughput.

In this paper we extend our work on multistage networks with NUTS to the case in which the network carries the superposition of two types of traffic. One type is the high throughput data and instruction traffic, while the other consists of control packets (for singularity conditions, errors, etc.) and I/O packets. In many instances, this second type is of low throughput but with real-time constraints. In such a case it is necessary to assure low latency for these packets. In the sequel we will call the first type of traffic background traffic (bgt) and the second type real-time traffic (rtt). We consider here strategies to control the network to achieve the low latency for rtt, especially in the case where the bgt has nonuniform traffic spots (NUTS).

Three solutions to this situation come to mind. In the first, a separate network is used for the rtt, which would be designed to assure the low latency; the disadvantage of this is the cost of the additional network. A second solution is to share one network but have separate buffers in the switches for each traffic; moreover, in case of conflicts for the switch output ports, priority is given to the rtt. Still, this solution has the additional cost of the buffers and is more complex to control, especially when blocking switches are used because of the need for additional full signals. Finally, it is possible to share network and buffers and develop control strategies to assure low latency for the rtt. Here we explore this third alternative.

We evaluate the performance by simulations. Since the objective is to explore alternative switch control strategies, we use a small set of traffic patterns that produce NUTS to compare the alternatives. We expect that similar results would be obtained for other patterns; however, for specific values of throughput and delay simulations should be done for the particular traffic.

We conclude that the diverting switches, which produce good performance for NUTS [8], are also suitable for assuring low latency for the rtt traffic, especially when using the displacing mode (see Section 3). Moreover, we show that the network with additional links, which provides the highest throughput in the presence of NUTS, is also suitable for low latency rtt traffic. We also conclude that even for uniform traffic it is not possible to assure low latency for the rtt when using blocking switches.

2. Multistage Network Structure and Operation

To make this paper relatively self-contained, we now repeat the brief description of the structure and operation of the multistage network, emphasising the assumptions we make [Lang87]. A more detailed discussion can be found in [Sieg85]. The type of multistage interconnection network we are considering has $N = 2^n$ inputs (sources) and outputs (destinations), both labeled from 0 to $N-1$. It consists of n stages of $N/2$ 2x2 switches, as shown in Figure 1. The outputs of stage-i switches are connected to the inputs of stage-$(i-1)$ switches, with the network inputs going to stage-$(n-1)$ switches and the network outputs coming from the stage-0 switches.

Several specific multistage networks have been proposed, differing in the interconnection pattern between stages. Since the characteristics, in terms of type of operation and performance, are similar for all these different topologies, we consider here the Omega network [9], which has been extensively studied [10] and is being used in several multiprocessor systems. In this network, the interconnection pattern between stages corresponds to the perfect shuffle connection, as shown in Figure 1.

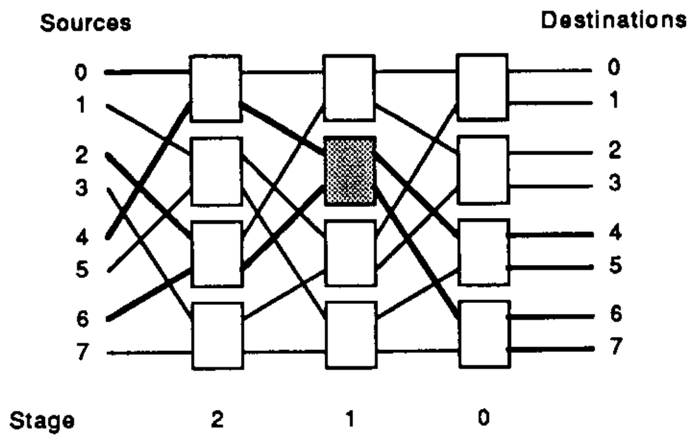


Figure 1. An 8×8 Omega network.

The routing of packets in the network is unique since there is a single path from a specific source to a specific destination. The control of routing is done using a destination tag that is associated with the message as part of each packet. At stage i, the routing depends only on the ith bit of the tag; if the bit has value 0(1), route to the upper(lower) output of the switch.

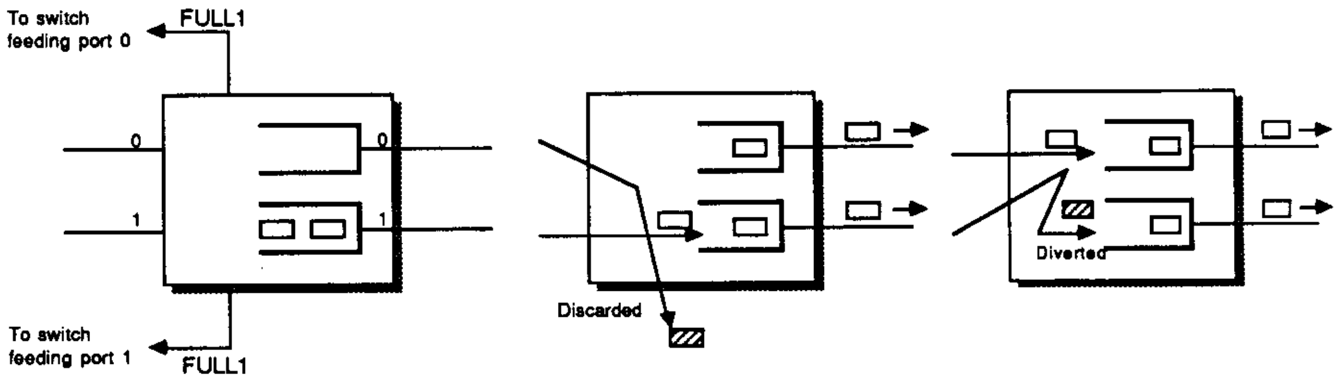
The operation of the network is synchronous and pipelined. In its basic form, each switch has one register per output and each cycle one packet is transferred from an output register in a switch of stage i to the corresponding output register of a switch of stage $i-1$. This implies that the packets are all of the same size. If the packet is large, the above mentioned cycle can be divided into several subcycles and a part of the packet be transferred per subcycle. However, we will not be concerned with this subdivision and will use the packet as the basic unit of transfer and the time of this transfer as the unit of time (one cycle).

Since each output register can receive only one message per cycle, there is a conflict when both packets entering a switch in a cycle have to be routed to the same output. One solution to this conflict is to have a buffer for each output and to store the additional packet in such a buffer. Of course, these buffers are finite so it is necessary to have an operation policy when the buffer is full. The basic scheme used is a blocking policy in which the predecessor switches do not send packets to a full buffer. To support this policy it is necessary to have signals from a switch to its predecessors indicating that the corresponding buffer(s) is full (Figure 2a).

To control contention it is also possible to use discarding and diverting switches, as shown in Figures 2b and 2c. In such cases, the packets are always sent to the successor switch but are discarded (and resent from the original source) or diverted to a wrong destination (and resent from the intermediate destination) when the buffers are full. It is important to note that in these cases, since packets always advance, there is at least one space in each output buffer so that at most one packet needs to be discarded or diverted per switch cycle, and no full signals are needed.

In [8] we showed that for traffic patterns that produce NUTS the diverting switches result in a larger throughput than either the blocking or the discarding switches.

To reduce further the contention it is possible to add intrastage links which produce alternate paths for the packets [11,12] (Figure 2d). Consequently, overflow packets can be routed through the additional link and still be able to reach the desired destination in a single pass through the network. This, of course, requires an augmented 3x3 switch with more complex control. We combine this alternate paths with diverting to obtain best throughput.



Figures 2a-c. Blocking, Discarding and Diverting Switches.

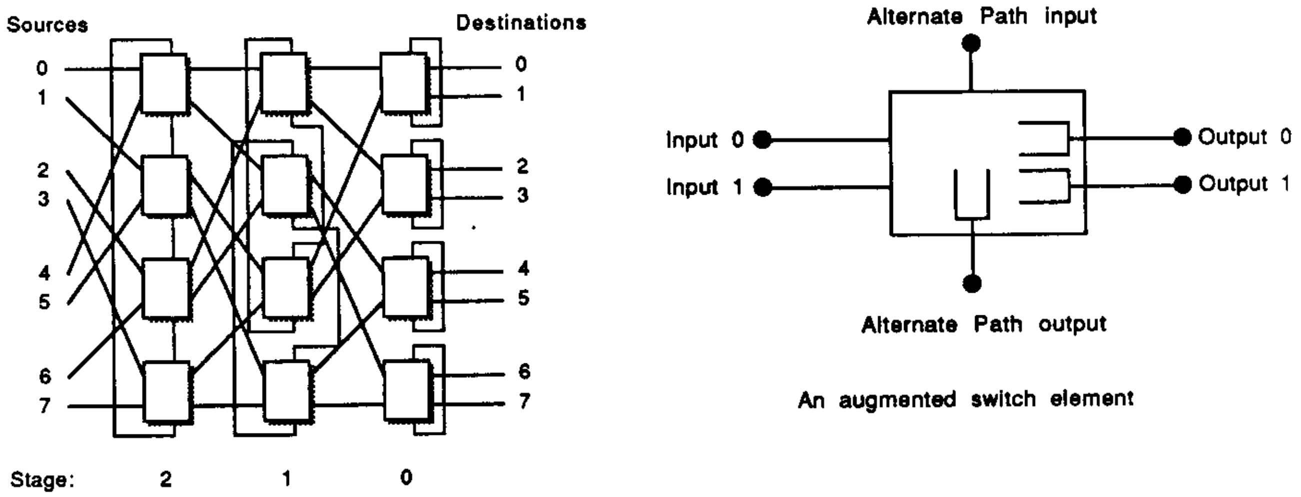


Figure 2d. Alternate Path Network and Switch Element.

Several variations to the basic switch design are possible. The switch can have a single buffer pool to service both inputs/outputs, which leads to the best buffer utilization. However, this requires that two packets be accepted and sent from the queue per cycle and, if a FIFO policy is used, a packet in the front of the queue can block the sending of another packet. The other possibility is to have dedicated buffers, either servicing one input or one output. Input buffers have the advantage of simplifying the generation of the full signal and receive at most one packet per cycle. However, arbitration is necessary to determine which packets are sent to the output and a FIFO policy leads to the same blocking characteristics as for the single buffer. Output buffers have to be able to receive two packets per cycle and have a more complex generation of the full signal. Queue management policies can be FIFO or non-FIFO with some kind of priority scheme. More complex control algorithms are possible, leading to better utilization, but the cost and speed requirements of the switch limit the practicality of such complex algorithms. As technology improves, however, more options become available.

In this paper we do not evaluate the different buffering organizations and policies. The degradation produced by NUTS is inherent to the blocking operation of the network, which is present for any of the buffer organizations and policies. Moreover, the modifications we propose are applicable to all these organizations. Consequently, we perform our analysis using output buffers with FIFO policy.

As mentioned, the basic network is composed of 2x2 switches. However, generalizations are possible in which *kxk* switches are used. This has the advantage of reducing the number of stages to $\log_k N$, with the corresponding reduction in delay. In this paper, we only consider 2x2 switches, but the results should be equally applicable to the general case.

When processors send request packets to remote memory modules, traffic in the opposite direction is also generated. These return packets must traverse an analogous network to reach the processors. The analysis of this type of traffic is similar to the request traffic, and is not considered here.

3. Strategies for low-latency real-time traffic

We now present the strategies we propose and evaluate to obtain low latency for thr real-time traffic (rtt).

For discarding and diverting switches the basic strategy we use to reduce the latency of the rtt is to give priority to packets of this traffic type when one of the two packets coming to a buffer has to be discarded or diverted (as previously noted, both packets never have to be discarded/diverted because there is always at least one space in each buffer, left by the departing packets). Specifically, when one of the packets coming to a buffer is of the bgt type, this is the one discarded/diverted.

On top of this basic strategy we consider the following three alternatives, as illustrated in Figure 3. In the first, the rtt packets are placed at the back of the buffers, in the same way as the bgt packets (Figure 3a). In the second alternative, the rtt are placed in front of the buffer, while the bgt packets are placed at the back (Figure 3b). Finally, the third alternative always allows the placement of up to two rtt packets in the front of the buffer; if there is no space for both packets in the buffer, this requires the displacement of a packet from the end of the buffer (note that this displaced packet might be a bgt packet or a rtt packet). This displaced packet is discarded or diverted to another queue (this scheme does not apply to blocking switches) (Figure 3c).

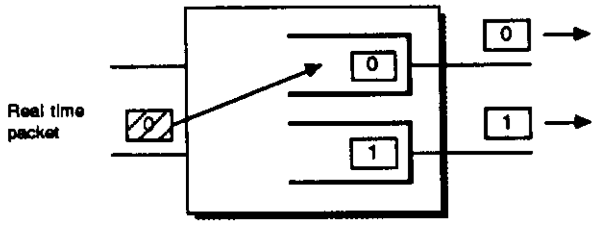


Figure 3a. RTT to the Back of the queue.

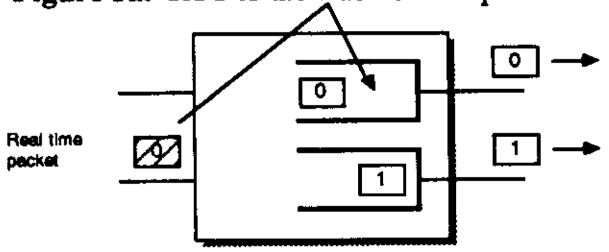


Figure 3b. RTT to the Front of the queue.

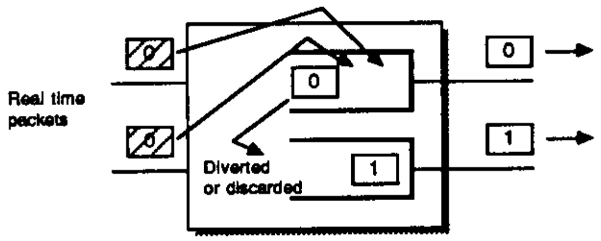


Figure 3c. Displacing.

4. Performance evaluation by simulation

To study the performance of the network using the alternative control policies presented in Section 3, we performed simulations. Fot this, we built a network simulator using as a basis SIMON, a general-purpose multiprocessor simulator developed at the University of Utah [13].

We evaluate the **steady-state** behavior of the system, that is, we assume that the traffic pattern under consideration remains for a period long enough to achieve this steady state.

The fundamental parameters for the network are its size and the size of the buffers. We have found that the relative performance of the network remains essentially the same for different values of these parameters. Consequently, we report our results for a network of size 64x64 and buffer of size 2; this buffer size is convenient because it produces a relatively small delay. We consider the worst case situation of maximum load on the network, that is, each input provides one packet per cycle (if the corresponding first-stage buffer is not full in the blocking case and if there are no packets to resend in the other cases). Finally, the total traffic is a mix of 95% of bgt and 5% of rtt.

As we did in [8], we choose traffic patterns to illustrate different types of behavior and show the effect in performance between the alternative ways of controlling contention. We are especially interested in cases in which the bgt traffic has NUTS because these are the difficult cases and where there is a difference between alternative control mechanisms. We assume that the rtt traffic is of low throughput so that it alone does not produce congestion in the network.

To allow for adequate analysis, we use a set of three traffic patterns that produce contention, along with uniform traffic for comparison; we expect that the behavior will be similar for other patterns.

The first type of traffic is produced when the even numbered sources send to the first half of the destinations, and odd sources send to the second half (**EFOS**). This pattern serves to illustrate a case in which each source accesses a subset of destinations.

The other two types of traffic are permutations. We use the **bit reversal** permutation since it is known to produce high contention in an Omega network. This permutation is an extreme case, and shows a lower bound on the improvement possible with the techniques used. As a more typical case, we also use a precomputed randomly generated **arbitrary** permutation.

To study both the effect of the bgt pattern and of the rtt pattern, we simulate two mixes: either the rtt is of the same pattern as the bgt or the rtt is uniform.

Since we are interested in providing a low latency for the rtt, we measure the distribution of delay for rtt packets. From this distribution we obtain as most significant measures the delay of the slowest rtt packet and the average latency of the slowest 10% of the rtt traffic. Moreover, since the network should carry the highest possible throughput of bgt, we also show this throughput in the tables.

5. Simulation Results

The simulation results are presented in the graphs of Figure 4 and in Tables 1 through 4.

Figure 4a shows the distribution of delays of rtt packets for one type of traffic (EFOS) and with diverting switches; the graphs for other types of traffic are very similar. We see that the policy with displacing is the best followed by rtt to front and finally rtt to back. The zoom view shows the 5% of slowest rtt packets.

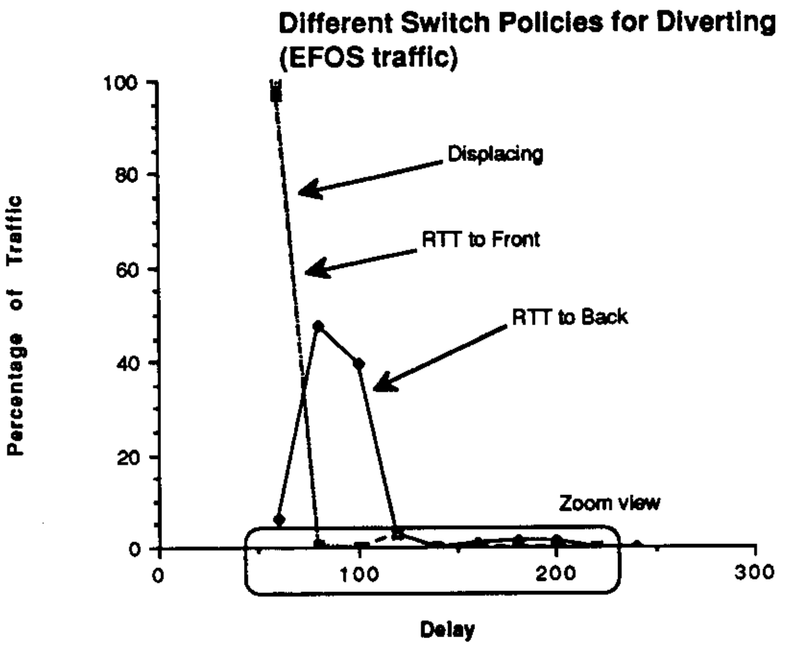
Figure 4b shows the effect of the bgt on the delay of the rtt traffic (for rtt to front policy). We see that with diverting switches there is practically no effect while the effect is significant when blocking switches are used. Moreover, we see that blocking switches have a much larger percentage of slow rtt packets.

The tables show the maximum latency experienced by the real-time traffic (in cycles) and the average delay of the slowest 10% of the real-time traffic. Since the simulated network has 6 stages, the minimum delay is 6 cycles.

Table 1 shows the effect of the switch type when both types of traffic are uniform. We conclude that for very low rtt latency the policy with displacing should be used, while for somewhat larger delay the policy rtt to front is acceptable for discarding, diverting or alternate paths. Blocking switches are not suitable to assure a low latency for rtt. Also, the policy rtt to back produces a significant degradation. Notice also that the network with alternate paths carries a much higher total throughput.

A similar conclusion is obtained for the other traffic patterns (Tables 2 to 4). Note however that in these cases, in which the bgt produces NUTS, the total traffic carried by the net is significantly larger for diverting switches than for discarding or blocking switches (note c). In these cases also the switch with alternate paths (using diverting) has the largest throughput.

From the two tables in Table 2, we see that the pattern of the rtt is not very significant, since it is of very low rate. A small degradation occurs when the rtt has the same pattern as the bgt, because of the NUTS produced by the latter.



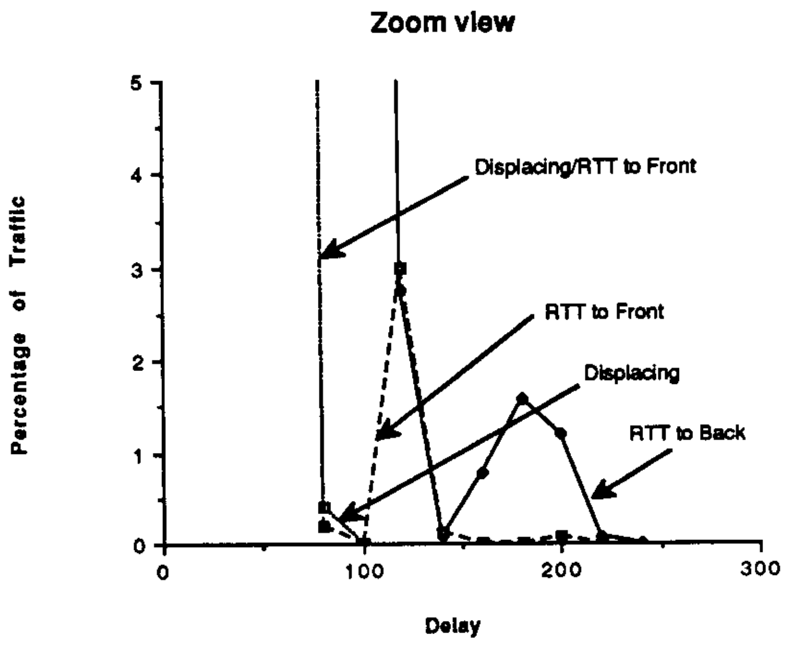


Figure 4a. Different Switch Policies for Diverting Switches.

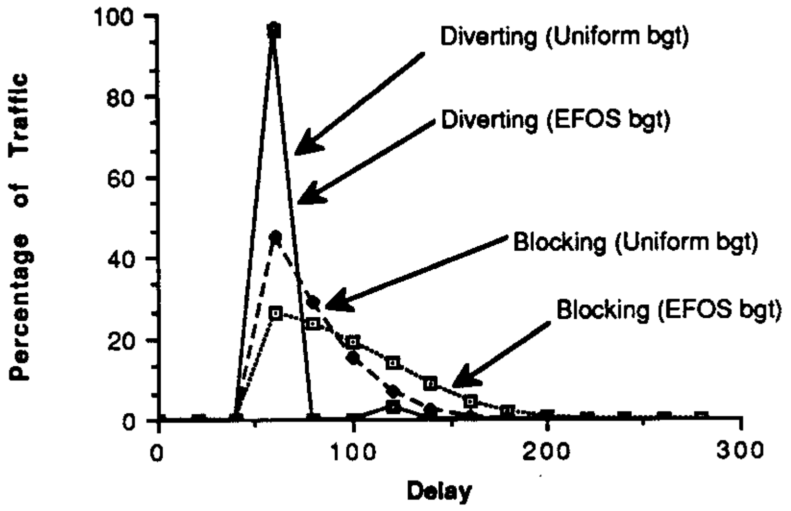


Figure 4b. Effect of Different Background Traffic (rtt to Front).

Table 1. Delay of RTT for Different Types of Switches.
(Uniform RTT/Uniform BGT)

RTT: Uniform(5%)/BGT: Uniform(95%)							
Switch Type	RTT to Back		RTT to Front		Displacing		Total Relative Throughput
	Max	Top 10%	Max	Top 10%	Max	Top 10%	
Blocking	32	16.9	22	12.9	*	*	.59
Discarding	18	10.2	12	6.3	8	6.1	.63
Diverting	20	11.7	14	7.4	8	6.0	.55
Alternate Paths	14	12.1	8	6.1	8	6.1	.81

Table 2. Delay of RTT for Different Types of Switches.
(EFOS BGT)

RTT: EFOS(5%)/BGT: EFOS(95%)							
Switch Type	RTT to Back		RTT to Front		Displacing		Total Relative Throughput
	Max	Top 10%	Max	Top 10%	Max	Top 10%	
Blocking	28	20.5	26	**16.3**	*	*	**.32**
Discarding	18	10.4	10	**6.4** (*b*)	8	**6.0**	**.35**
Diverting	22	13.6	20	**8.0**	8	**6.1** (*a*)	**.53** (*c*)
Alternate Paths	14	12.1	8	6.1	8	**6.3**	**.81**

RTT: Uniform(5%)/BGT: EFOS(95%)							
Switch Type	RTT to Back		RTT to Front		Displacing		Total Relative Throughput
	Max	Top 10%	Max	Top 10%	Max	Top 10%	
Blocking	26	18.1	22	14.3	*	*	.33
Discarding	16	10.0	10	6.1	6	6.0	.35
Diverting	22	12.0	14	7.1	8	6.1	.53
Alternate Paths	14	12.1	8	6.1	8	6.1	.81

Table 3. Delay of RTT for Different Types of Switches.
(Other BGT)

RTT: Uniform(5%)/BGT: Arbitrary(95%)							
Switch Type	RTT to Back		RTT to Front		Displacing		Total Relative Throughput
	Max	Top 10%	Max	Top 10%	Max	Top 10%	
Blocking	30	21.9	28	17.5	*	*	.46
Discarding	14	8.6	10	6.2	6	6.0	.45
Diverting	20	11.7	14	7.3	6	6.0	.53
Alternate Paths	14	12.0	8	6.1	8	6.1	.78

RTT: Uniform(5%)/BGT: Bit Reversal(95%)							
Switch Type	RTT to Back		RTT to Front		Displacing		Total Relative Throughput
	Max	Top 10%	Max	Top 10%	Max	Top 10%	
Blocking	>38	>38	>38	>34	*	*	.13
Discarding	14	8.0	8	6.0	6	6.0	.13
Diverting	22	11.4	14	7.2	8	6.0	.44
Alternate Paths	14	12.0	6	6.1	8	6.1	.74

Finally, from the operation of the network, we see that if the rtt corresponds to nonintersecting paths (Figure 5), the use of diverting switches with rtt to front produces the minimum delay, since never two rtt packets would conflict. For this type of rtt there would be no need to use the somewhat more complicated policy with displacing.

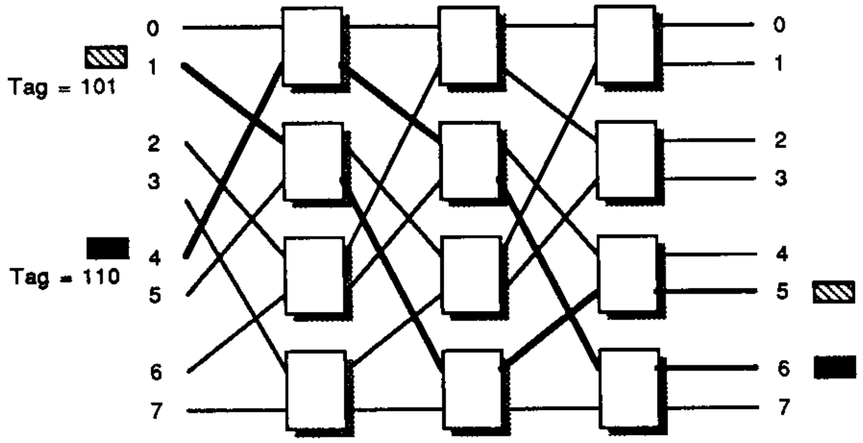


Figure 5. Two non-intersecting paths.

6. Conclusions

We have presented several strategies to assure low latency for low rate real-time traffic in the presence of high rate background traffic, even when there exist NUTS produced by the latter. For the basic network with single path between any source-destination pair, the best strategy is to use diverting switches with displacing policy. The use of diverting switches provides a large throughput for background traffic producing NUTS, while this type of switches with displacing assures the low latency for rtt. Discarding switches are not good because they have a significant degradation with NUTS and blocking switches degrade both types of traffic.

Extension of the network obtained by adding intrastage links enhance even further the throughput of the bgt while having still a low latency for the rtt. The disadvantage of this type of extension is that it requires 3x3 switches which are somewhat more complex than the basic 2x2 type.

Acknowledgement

This research was sponsored in part by the State of California MICRO program and Hughes Aircraft Co.

7. References.

[1] H. J. Siegel, "Interconnection Networks for Large-Scale Parallel Processing, Theory and Case Studies", Lexington Books, 1985.

[2] K. Hwang and F. Briggs, "Computer Architecture and Parallel Processing", McGraw Hill, 1984.

[3] A. Gottlieb, et al. "The NYU Ultracomputer--Designing an MIMD Shared Memory Parallel Computer", IEEE Transactions on Computers, Vol. C-32, No. 2, pp. 175-189.

[4] G. F. Pfister, W. C. Brantley, D. A. George, S. L. Harvey, W. J. Kleinfelder, K. P. McAuliffe, E. A. Melton, V. A. Norton, J. Weiss, "The IBM Research Parallel Processor Prototype (RP3): Introduction and Architecture", Proceedings of the 1985 International Conference in Parallel Processing, August 1985, pp.764-771.

[5] R. H. Thomas, "Behavior of the Butterfly Parallel Processor in the Presence of Memory Hot Spots", IEEE Parallel Processing, 1986.

[6] D. Dias and R. Jump, "Packet Switching Interconnection Networks for Modular Systems", Computer, December 1981, pp. 43-54.

[7] C.P. Krustal and M. Snir, "The Performance of Multistage Interconnection Networks for Multiprocessors", IEEE Transactions on Computers, Vol. 32, No. 12, December 1983, pp. 1091-1098.

[8] T. Lang and L. Kurisaki, "Nonuniform Traffic Spots (NUTS) In Multistage Interconnection Networks", UCLA Technical Report CSD-880001, January 1988.

[9] D.H. Lawrie, "Access and Alignment of Data in an Array Processor", IEEE Transactions on Computers, Vol. C-24, No. 12, Dec. 1975, pp. 1145-1155.

[10] P. Chen, D. Lawrie, and P. Yew, "Interconnection Networks Using Shuffles", Computer, December 1981, pp. 55-64.

[11] V. P. Kumar and S. M. Reddy, "Design and Analysis of Fault-Tolerant Multistage Interconnection Networks With Low Link Complexity", 12th Annual Symposium on Computer Architecture, pp. 376-386 (June 1985).

[12] N. Tzeng, P. Yew, and C. Zhu, "A Fault-Tolerant Scheme for Multistage Interconnection Networks", 12th Annual Symposium on Computer Architecture, pp. 368-375 (June 1985).

[13] R.M. Fujimoto, "The CSIMON Interface", Computer Science Department, University of Utah, 1986.