# Calculation of Deadline Missing Probability in a QoS Capable Cluster Interconnect*

Eun Jung Kim Ki Hwan Yum Chita R. Das

Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
E-mail: {ejkim,yum,das}@cse.psu.edu

## Abstract

*The growing use of clusters in diverse applications, many of which have real-time constraints, requires Quality-of-Service (QoS) support from the underlying cluster interconnect. In this paper, we propose an analytical model that captures the characteristics of a QoS capable wormhole router, which is the basic building block of cluster networks. The model captures the behavior of integrated traffic in a cluster and computes the average deadline missing probability for real-time traffic. The cluster interconnect, considered here, is a hypercube network. Comparison of Deadline Missing Probability (DMP) using the proposed model with that of the simulation shows that our analytical model is accurate and useful.*

***Index Terms:*** *Analytical Model, Cluster Network, SAN, Quality-of-Service, Pipelined Router Architecture, VirtualClock, Wormhole Switching.*

## 1 Introduction

Quality-of-Service (QoS) provisioning in clusters has become a critical issue with the widespread use of clusters in diverse commercial applications. The traditional best-effort service model that has been used for scientific computing is not adequate to support many cluster applications with varying consumer expectations. For example, many real-time applications require timely delivery of messages. These applications mandates that the cluster system, and hence the cluster interconnect, should be able to handle user specified service demands instead of adopting the *same-service-to-all* model.

Most commercial routers (switches) such as SGI SPIDER [11], Cray T3D/E [20], Tandem Servernet-II [12], Intel Cavallino [4], IBM SP2 [22], and Myricom Myrinet [2] use wormhole switching to provide high performance. However, they have not been designed for QoS assurance except for the Servernet-II, which provides a link arbitration policy called ALU-biasing for implementing limited bandwidth and delay control. Hence, design and analysis of QoS capable routers and cluster networks has become a current research focus [17].

Recently a few router architectures with QoS provisioning have been proposed [17, 9, 3, 18, 10]. Most of these designs have used a hybrid approach with two different types of switching mechanisms within the same router — one for best-effort traffic and the other for real-time traffic. They have refrained from using wormhole switching because of potential unbounded delay for real-time traffic.

On the contrary, wormhole routers with some modifications have been considered for handling traffic priority [19, 5, 21, 24, 23]. The options vary from providing hardware support in the router for bandwidth assurance [19, 21, 24] to software solutions on existing routers [5]. In the hardware approach, the most logical solution is to assign separate virtual channels (VCs) to different traffic classes and use a rate-based scheduling mechanism such as Fair Queueing [7] or VirtualClock [25] to share the link

*This research was supported in part by NSF grants MIPS-9634197, CCR-9900701, CCR-0098149, and equipment grants from NSF and IBM.

bandwidth proportionately [19, 24]. Techniques such as preemption of lower priority traffic in favor of higher priority traffic have also been proposed [21]. Recently, we have proposed a QoS-aware pipelined router that supports features such as rate-based scheduling, preemption, and flit acceleration mechanism [23]. Software solution like the self-synchronizing scheduling [5] does not need any hardware modification, but the solution may not be scalable.

A limitation of all prior studies is that they use simulation to evaluate the performance of various design trade-offs. In addition, the evaluations are confined to a single router in many cases. Detailed flit-level simulation is quite expensive and prohibits full-blown analyses of various design trade-offs. On the other hand, an accurate analytical model can provide quick performance estimates and will be a valuable design tool. In this paper we present a mathematical model for analyzing QoS capable cluster networks. In [15], we had developed a mathematical model of a QoS-aware cluster network to compute the average network latency. While the average latency is an important performance metric for all types of traffic, it does not capture the behavior of real-time traffic in sufficient detail. For example, if an application messages requires low jitter tolerance, then jitter will be a main performance metric. In our case, since we consider time-constraint applications, delay bound is the primary objective function. Since wormhole-switched network cannot provide hard guarantees due to chained blocking, the system can provide soft guarantees in terms of deadline missing probability (DMP). The DMP of time-constrained applications was analyzed in [19, 14] via simulation. In this paper, we present an analytic approach to compute DMP of real-time applications.

We first develop the model for a single router and then extend it to a network. Here we use a hypercube-style cluster network primarily to keep the analysis tractable due to the symmetric nature of the network. However, our QoS-aware router model can be extended to any regular topology such as $k$-ary $n$-cubes and meshes as long as the topology and routing algorithm can be captured mathematically.

Like many commercial designs, we use a pipelined wormhole router architecture. The model considers an integrated workload consisting of $C$ different classes of traffic. $(C-1)$ classes represent real-time applications[1] with distinct service requirements. The last class is used for best-effort traffic applications. As proposed in our MediaWorm design [24], each class is statically assigned at least one VC, and the VCs are scheduled with a rate-based scheduling algorithm, VirtualClock [25], to regulate the bandwidth requirements. Average message latency for different traffic classes can be computed using this model.

The main contribution of this analytical model is that it provides an accurate estimation of the deadline missing probability of real-time traffic in QoS capable pipelined wormhole-switched networks. We validate the single router model (16-port) and the cluster network model (6-cubes) through extensive simulation. We use a mixed workload of three traffic classes ($C = 3$, two real-time and one best-effort) in this study. It is shown that the models are quite accurate in predicting DMP. Thus, it can be used as an efficient design tool to analyze network and application centric performance parameters.

The rest of the paper is organized as follows. In Section 2, the router architecture and the VirtualClock algorithm are discussed. In Section 3, we present the analytic models for DMP. The performance results are analyzed in Section 4, followed by the concluding remarks in Section 5.

## 2 A QoS-aware Router Architecture

Most routers now use a pipelined design to minimize the network cycle time. Accordingly, we use a pipelined, wormhole-switched router in this paper. Fig. 1 shows the pipelined router consisting of five stages. Stage 1 represents the functional units, which synchronize the incoming flits, demultiplex a flit so that it can go to the appropriate input virtual channel (VC) buffer to be subsequently decoded. If the flit is a header flit, routing decision and arbitration for the correct crossbar output are performed in the next two stages (stage 2 and stage 3). On the other hand, middle flits and the tail flit of a message directly move to stage 4. Flits get routed to the correct crossbar output port in stage 4. Finally, the last stage performs buffering for flits flowing out of the crossbar, multiplexes the physical channel bandwidth amongst multiple VCs, and transmits one flit at a time to the neighboring router or to the network interface of the node attached to this router.

In this $n$-port router architecture, we provide one VC for each of the $C$ traffic classes (thus $C$ input and $C$ output VCs). More VCs per class should improve the performance. Note that the crossbar used in our router is called

[1] Here a real-time application refers to any time-constrained application.

a *full crossbar* since it has $n \times C$ inputs and $n \times C$ outputs. The model can be modified for a multiplexed crossbar, where the VC multiplexing will be done before the crossbar stage.

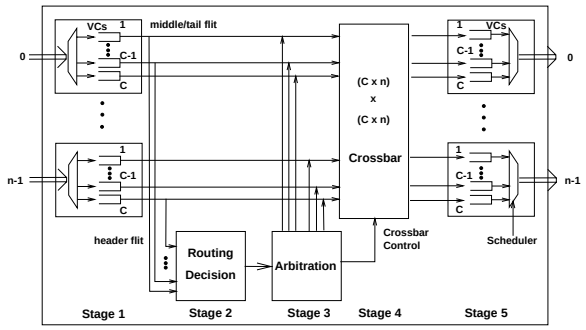



**Figure 1. The pipelined router architecture with a full crossbar.**

Unlike the lumped router models analyzed before [1, 16, 6, 13] that capture only the blocking delay caused by arbitration, a message entering the above pipelined router can experience delay at stages 1, 3 and 5 of the router. If the corresponding input buffer is full in stage 1, the message must wait outside the router until adequate space is available. In stage 3, the message again may be delayed because its destination crossbar output port could be busy. Crossbar output port arbitration is performed at a message level granularity. So the message has to wait until the output port is released by the message currently using it. Finally in stage 5, multiple VCs compete for the physical channel bandwidth. Traditionally, a Round Robin or FIFO scheduler is used to schedule the output channel in a time-division manner.

The above router design is modified to support QoS provisioning by simply incorporating a rate-based scheduling algorithm to share the physical channel bandwidth. Similar techniques have been proposed for the Internet router line cards. Our previous research [24, 23, 15] has shown that this architecture with the VirtualClock algorithm [25] can be effective in providing QoS for integrated traffic.

In the VirtualClock algorithm, there are two variables, called *auxVC* and *Vtick* for each connection. The values of these two variables are determined when a connection is set up. The *auxVC* indicates the virtual clock value of the connection, while the *Vtick* is the amount of time that should be incremented whenever a flit arrives at that connection. The *Vtick* value specifies the interarrival time of flits from the connection. Therefore, a smaller *Vtick* value implies higher bandwidth. Once these two values are set, the VirtualClock algorithm works as follows. For each connection $i$, when a flit arrives at the scheduler, the following computation is done.

$$auxVC_i \leftarrow \max(\text{real time}, auxVC_i),$$

$$auxVC_i \leftarrow auxVC_i + Vtick_i,$$

timestamp the flits with the $auxVC_i$.

The flits are queued and serviced in increasing timestamp order. For the best-effort traffic, the timestamp is set as $\infty$. So the best-effort flits are processed only if there are no other flits with lower timestamp values.

## 3 Deadline Missing Probability

As described in the previous section, the router model assumes a pipelined architecture with $P = 5$ stages. The model is derived for $C$ classes of traffic with different service requirements. Here we assume that there are $(C-1)$ real-time traffic classes and one class of best-effort traffic. Each class is assigned a dedicated VC. (This assumption can be relaxed to assign multiple VCs to a class.) In addition, the model is based on the following assumptions typically used in analytical models:

- The arrival pattern of each class $c$ follows the Poisson processes with an average arrival rate of $\lambda_c^g$.
- Message length is $M$ flits long.
- Message destination is uniformly distributed.
- The $Vtick_c$ value for real-time traffic belonging to class $c$ is given by $1/(\lambda_c^g M)$, and the *Vtick* value for best-effort traffic is set to $\infty$.
- The input and output buffers(VCs) in stages 1 and 5 can hold $b_s$ flits. Each class is assigned a dedicated injection/ejection queue outside the router, and these queues have infinite capacity.

For a given source and destination pair, the probability of missing the deadline is the probability that a message cannot be delivered within a specified time($D$). We only compute the deadline missing probability of real-time traffic which has time-constraints. Here we consider only deadline to traverse the network. Source queueing is not included in order to keep the discussion simple.

### 3.1 Single Router Model

We compute the DMP for a class $c$ traffic in a single router. The network latency of class $c$, $L_c$, is the time to traverse the router. The network latency($L_c$) of a message of class $c$ consists of two parts. The first part is the actual message transfer time, $T$. The second part is due to blocking caused by the wormhole switching scheme, and due to sharing of the physical channel bandwidth by multiple virtual channels at stage 5 of Fig. 1. The actual transmission time with $P$ pipeline stages in a single router is $(P-1+M)$ cycles for an $M$-flit message.

In order to compute the second part of the network latency, let us define $B_c$ as the blocking length (in number of flits) seen by the header flit at the input, output, and arbitration stage in the router. $B_c$ captures the message blocking in a pipelined wormhole router. Then the effective length of the message becomes $(M + B_c)$ flits. Let $S_c$ be the average number of cycles required to transfer one flit of a class $c$ message. $S_c$ represents the effect of bandwidth sharing mechanism of the Virtual Clock algorithm. Thus, the network latency($L_c$) for $1 \leq c \leq C$ is

$$L_c = (B_c + M)S_c + P - 1. \quad (1)$$

While blocking happens among the same class of messages, the sharing depends on the traffic of other classes. Thus, these two random variables($B_c$ and $S_c$) are independent. We can combine them to a random parameter, $\beta_c = (B_c + M)S_c$. We know that $\beta_c = L_c - (P - 1)$ and $\beta_c \geq M$.

Let $P_{m,c}(D)$ be the probability of missing the deadline $D$. If we can find the c.d.f. of $L_c, P\{L_c \leq D\}$, then $P_{m,c}(D)$ is $1 - P\{L_c \leq D\}(= 1 - P\{\beta_c \leq D'\})$, where $D' = D - (P - 1)$.

For accurate estimation of $\beta_c$, first we consider the two random variables($B_c$ and $S_c$) separately and then combine them. To compute the blocking length $B_c$, note that blocking is possible at the input buffer stage, output buffer stage and arbitration stage. The worst case of blocking occurs when all these places are occupied by other messages. Thus the worst blocking length will be $(2\max(b_s, M) + M)$ where $b_s$ is the input/output buffer size and M is the message length. The $\max(b_s, M)$ term is used to capture the buffer length $b_s < M$, since a new message must wait until the service for the previous message is completed. Let us assume that we know the probability mass function $P_{m,c}(B)$ of $B_c$ ( $P_{m,c}(B) = P\{B_c = B\}$), which will be described later.

With a given blocking delay($B$), the effective message length will be $(M+B)$. When each flit of $(M+B)$ arrives at the head of the output VC, there are $2^{(C-2)}$ combinations of other real-time traffic that denote whether they occupy the corresponding output VCs or not. All these combinations will determine how to share the bandwidth. We number the combinations serially so that for each combination $k$ ($0 \leq k \leq 2^{(C-2)} - 1$) we can determine the number of cycles required to transfer a flit of class $c$ traffic, $S_c(k)$, and the probability of $k$th combination for traffic $c$, $P_c(k)$.

Let $X_c(i)$ be the number of flits, which needs $S_c(i)$ cycles at the output VC such that $\sum_{i=0}^{2^{(C-2)}-1} X_c(i) = B+M$, given that the blocking length is $B$. Then $\beta_c$, the actual delay for a blocking length $B$ can be denoted as

$$\beta_c = \sum_{i=0}^{2^{(C-2)}-1} X_c(i)S_c(i).$$

Let's define the c.d.f of $\beta_c$, $P\{\beta_c \leq D'\}$, as

$$\begin{aligned} P\{\beta_c \leq D'\} \quad = \quad & \\ \sum_{B=0}^{B^u} \sum_{X_0=0}^{X_0^u} \quad \cdots \quad & \sum_{X_{2^{(C-2)}-1}=0}^{X^u_{2^{(C-2)}-1}} P_{m,c}(B)P_{x,c}(X_0, 0|B) \\ \cdots \quad & P_{x,c}(X_{2^{(C-2)}-1}, 2^{(C-2)} - 1|B). \end{aligned} \quad (2)$$

There are $2^{(C-2)} + 1$ summation notations in Eq. 2. The first notation is for $B$ and the remaining $2^{(C-2)}$ notations correspond to the total number of combinations of the output VC status. In Eq. 2, $P_{x,c}(X, i|B)$ is the probability that $X_c(i) = X$ given the blocking length is $B$. $B^u$, the upper bound of $B$, is $2\max(b_s, M) + M$ which is the worst case of blocking, $X_0^u = \min(B + M, \frac{D'}{S_c(0)})$, and $X_i^u = \frac{D' - \sum_{j=0}^{i-1} S_c(j)X_j}{S_c(i)}$, for $1 \leq i \leq 2^{(C-2)} - 1$.

We need the solution of $P_{m,c}(B)$ and $P_{x,c}(X, i|B)$ to find the deadline missing probability. Since the exact estimation of the terms is extremely hard, we approximate these probabilities from the operational behavior of the router/network. If we have the blocking probability of class $c$ [2], $P_{b,c}$, then $P_{m,c}(0) = 1 - P_{b,c}$. Since the blocking length($B_c$) varies between 0 and $2\max(b_s, M) + M$, under

[2]The computation of $P_{b,c}$ and $P_c(k)$ is summarized in the Appendix. For full derivation, please refer to [15].

the uniform distribution assumption, $P_{m,c}(B)$ can be written as

$$P_{m,c}(B) \approx \begin{cases} 1 - P_{b,c}, & B = 0 \\ P_{b,c}/B^u, & 1 \le B \le B^u \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

where $B^u = 2\max(b_s, M) + M$.

Similarly we can get $P_{x,c}(X, k|B)$ with $P_c(k)$, which again for better readability[3], is deferred to the Appendix. Since $\overline{X_c(i)} = P_c(i) \cdot (B + M)$ for a given $B$, we assume that $X_c(i)$ varies between 0 and $(B+M)$ under the uniform distribution assumption. Hence,

$$P_{x,c}(X, k|B) \approx \begin{cases} 1 - P_c(k), & X = 0 \\ P_c(k)/(B + M), & 1 \le X \le (B + M) \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

### 3.2 Modeling of a Cluster Interconnect

The single router model can be extended to most of the regular networks as long as the topology and router algorithm can be captured analytically. Models for such topologies like hypercubes, meshes, and $k$-ary $n$-cubes have been developed to predict the performance of best-effort traffic [1, 16, 6, 8, 13]. Here, we consider integrated traffic in the network, and use a hypercube topology to demonstrate this idea. We use the deadlock-free e-cube routing algorithm for message transfer.

We compute the deadline missing probability for a class $c$ traffic that traverses $h$ hops in the network. The network latency of class $c$ is a discrete random variable, $L_c$. The network latency for a given *Path*, which is a set of $h$ channels traversed by a message due to e-cube routing, can be expressed as

$$L_c = \sum_{s \in \text{Path}} (B_{c,s}S_{c,s} + P) + ((B_{c,n} + M)S_{c,n} + P - 1), \quad (5)$$

where $B_{c,s}$ $(0 \le s \le n-1)$ is the blocking length seen by a header flit of class $c$ in channel $s$, and $S_{c,s}$ is the number of cycles for transferring a flit of class $c$ in channel $s$. The first term in Eq. 5 represents the time spent at each hop, and the last term denotes the time at the ejection channel. Note that this does not include the queueing delay outside the router. Since $B_{c,s}$ and $S_{c,s}$ are random variables, we can combine them to a random parameter, $\beta_{c,s}$. Thus, we write $\sum_{s \in \text{Path}_e} (\beta_{c,s}) = L_c - (P - 1 + Ph)$, where $\text{Path}_e$ includes the ejection channel ($\text{Path}_e = \text{Path} \cup \{n\}$). $\beta_{c,s}$ is given by $\beta_{c,s} = B_{c,s}S_{c,s}$ , $0 \le s \le n-1$ and $\beta_{c,n} = (B_{c,n}+M)S_{c,n}$. Let $D' = D-(P-1+Ph)$ and $P_{h,c,s}(D_s)$ be the p.m.f of $\beta_{c,s}(P_{h,c,s}(D_s) = P\{\beta_{c,s} = D_s\})$.

Let $P_{m,c}(D)$ be the probability of missing the deadline $D$ for a given $Path = \{s_1, s_2, \ldots s_h\}$. If we can find the c.d.f. of $L_c, P\{L_c \le D\}$, then $P_{m,c}(D)$ is $1 - P\{L_c \le D\}(= 1 - P\{\sum_{s \in \text{Path}_e} \beta_{c,s} \le D'\})$. Since delays of each hop($\beta_{c,s_1}, \beta_{c,s_2}$ ...) are independent each other, we can write

$$P\{\sum_{s \in \text{Path}_e} (\beta_{c,s}) \le D'\} = \sum_{D_{s_{h+1}}=M}^{D^u_{s_{h+1}}} \cdots \sum_{D_{s_1}=0}^{D^u_{s_1}} P_{h,c,s_1}(D_{s_1}) \cdots P_{h,c,s_{h+1}}(D_{s_{h+1}}). \quad (6)$$

The above equation has $(h+1)$ terms corresponding to $(h+1)$ hops a message travels(including ejection chance). The lower bound of each hop except the $(h+1)$th hop is zero. From $\sum_{s \in \text{Path}_e} (\beta_{c,s}) \le D'$, we can get the upper bounds as $D^u_{s_i} = D' - \sum_{j=i+1}^{h+1} D_{s_j}$ and $D^u_{s_{h+1}} = D'$.

We can compute the p.m.f($P_{h,c,s}(D_s)$) in Eq. 6 from the c.d.f($P\{\beta_{c,s} \le D_s\}$) by $P_{h,c,s}(D_s) = P\{\beta_{c,s} \le D_s\} - P\{\beta_{c,s} \le D_s - 1\}$. Like Eq. 2, we obtain $P\{\beta_{c,s} \le D_s\}$ as

$$P\{\beta_{c,s} \le D_s\} = \sum_{B=0}^{B^u} \sum_{X_0=0}^{X^u_0} \cdots \sum_{X_{2^{(C-2)}-1}=0}^{X^u_{2^{(C-2)}-1}} P_{m,c,s}(B) \; P_{x,c,s}(X_0, 0|B) \cdots P_{x,c,s}(X_{2^{(C-2)}-1}, 2^{(C-2)} - 1|B). \quad (7)$$

As explained in Eq. 2, there are $2^{(C-2)} + 1$ summation notations in Eq. 7. In Eq. 7, $P_{m,c,s}(B)$ is the p.m.f of $B_{c,s}$ and $P_{x,c,s}(X, i|B)$ is the probability that $X_{c,s}(i) = X$ for a given $B$ ($\sum_{i=0}^{2^{(C-2)}-1} X_{c,s}(i) = B$). And $\beta_{c,s} = \sum_{i=0}^{2^{(C-2)}-1} X_{c,s}(i)S_{c,s}(i)$. $B^u$, the upper bound of $B$, is $(2\max(b_s, M) + M)$, $X^u_0 = \min(B, \frac{D_s}{S_{c,s}(0)})$, and $X^u_i = \frac{D_s - \sum_{j=0}^{i-1} S_{c,s}(j)X_j}{S_{c,s}(i)}$. For the ejection channel $(s = n)$, the c.d.f is slightly different, and should include the message length $M$ with blocking length $B$. So $P_{x,c,s}(X, i|B), 0 \le i \le 2^{(C-2)}-1$, will be replaced by $P_{x,c,n}(X, i|B+M), 0 \le i \le 2^{(C-2)} - 1$. Also the upper bound of $X_0$ changes to $X^u_0 = \min(B + M, \frac{D_s}{S_{c,n}(0)})$.

From Eq. 3, $P_{m,c,s}(B)$ can be written as

$$P_{m,c,s}(B) \approx \begin{cases} 1 - P^s_{b,c}, & B = 0 \\ P^s_{b,c}/B^u, & 1 \leq B \leq B^u \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

where $P^s_{b,c}$ is the blocking probability of class $c$ in channel $s$.

Similarly from Eq. 4, we can get $P_{x,c,s}(X,k|B)$ with $P_{c,s}(k)$ as

$$P_{x,c,s}(X,k|B) \approx \begin{cases} 1 - P_{c,s}(k), & X = 0 \\ P_{c,s}(k)/B, & 1 \leq X \leq B \\ 0, & \text{otherwise.} \end{cases}$$

Note that all these equations can be derived from the single router model for a given number of hops($h$) and for a physical channel $s$ by setting the proper boundary values.

## 4 Performance Results

Using the equations derived in Section 3, we compute the DMPs in a single router and in a 6-cube. Some of the results are presented here for validating the model. We are unable to include results of other cube dimensions due to space limitation. We also implemented a corresponding simulation model as shown in Fig. 1 using CSIM. Note that we need a deadline parameter $D$ to estimate the DMP. In our pipelined router model, the minimum transfer time for a 32-flit message is 36 cycles ($= P + M - 1$). Hence, we set $D = 42$ cycles for the single router. Similarly for 2-hop messages in a 6-cube, the minimum transfer time is 46 cycles ($M + Ph + P - 1$). We set $D = 55$ or 60 cycles for 2-hop messages.

In Fig. 2, we plot the DMP results for two types of real-time traffic (R1 and R2) from the mathematical model (Math) and the simulation model (Sim). In a 6-cube, the DMPs of 2-hop and 5-hop messages are shown for different $D$ values. The graphs show that the single router results are more accurate compared to the 6-cube results. This is because we approximate the upper bound of blocking length in each hop to $(2 \max(b_s, M) + M)$ without accounting for the chained blocking. Since there is no chained blocking in a single router, the upper bound approximation is more accurate. Even with this approximation, the DMP results from the analytical model of a 6-cube match closely with the simulation results.

## 5 Concluding Remarks

This paper introduces an analytical approach for calculating the DMP of real-time traffic in a QoS-aware wormhole router and a hypercube-style cluster network, designed using such routers. For accurate calculation, the model captures the pipelined design, and analyzes the blocking delay at different stages of the pipe. In addition, the effect of VirtualClock scheduling algorithm is reflected in the model. Comparison with the simulation results indicates that the router as well as the hypercube models are quite accurate in predicting the DMP. Unlike the simulation model, the analytical model can be used as an efficient design tool in studying various design trade-offs. For example, the impact of message length ($M$), and other questions can be answered quickly using the model either for a single-cluster or for a multi-router cluster. Such performance estimates and quick design overviews are difficult to obtain via a simulation study.

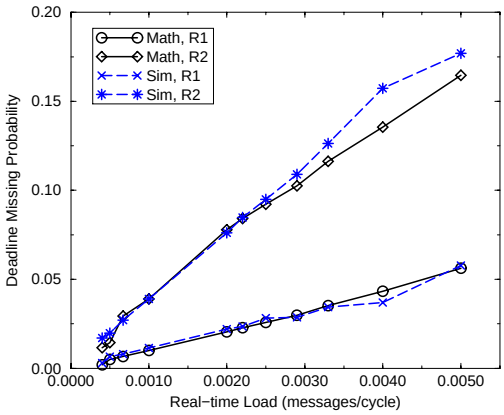
The model presented here can be improved in a variety of ways, and some of them are currently pursued in our group. First, the exponential arrival distribution for real-time traffic may not be quite practical to apply to media streams. We need to develop the model with a CBR/VBR source to capture inputs like media streams. Second, QoS comes with different connotations, and extension of the model to predict other performance parameters such as bandwidth assurance and jitter should be useful. Third, the model can be extended to other topologies. Finally, co-evaluation of the cluster network with a detailed network interface model should answer many questions regarding the QoS ability of the entire communication system.

## References

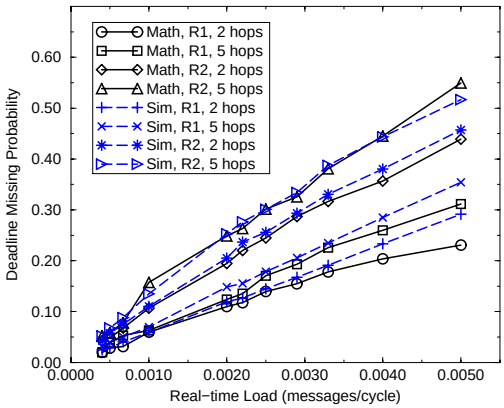

[1] V. S. Adve and M. K. Vernon. Performance Analysis of Mesh Interconnection Networks with Deterministic Routing. *IEEE Transactions on Parallel and Distributed Systems*, 5(3):225–246, March 1994.

[2] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W.-K. Su. Myrinet: A Gigabit-per-second Local Area Network. *IEEE Micro*, 15(1):29–36, February 1995.

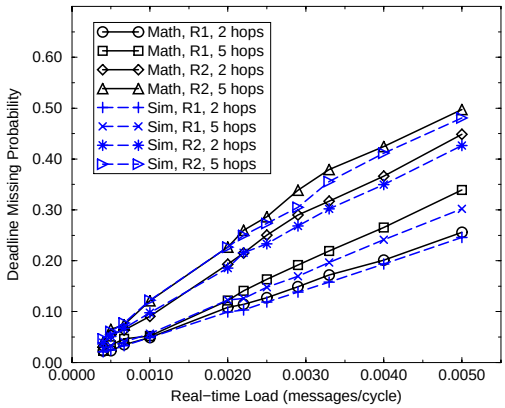
[3] M. B. Caminero, J. J. Quiles, J. Duato, D. S. Love, and S. Yalamanchili. Performance Evaluation of the Multimedia Router with MPEG-2 Video Traffic. In *Proceedings of the Third International Workshop on Communication, Architecture and Applications on Network Based Parallel Computing (CANPC'99)*, pages 62–76, January 1999.




(a) Single router with deadline 42 cycles




(b) Deadline : 55 cycles (2 hops), 70 cycles (5 hops)




(c) Deadline : 60 cycles (2 hops), 75 cycles (5 hops)

**Figure 2. DMP comparison of analytical model and simulation model in a single router and 6-cube with varying real-time load and fixed best-effort load (single router : 0.01 msgs/cycle, 6-cube : 0.002 msgs/cycle).**


[4] J. Carbonaro and F. Verhoorn. Cavallino: The Teraflops Router and NIC. In *Proc. Symp. High Performance Interconnects (Hot Interconnects 4)*, pages 157–160, August 1996.

[5] K. Connelly and A. A. Chien. FM-QoS: Real-Time Communication Using Self-Synchronizing Schedules. In *Proceedings of Supercomputing Conference*, November 1997.

[6] W. J. Dally. Performance Analysis of $k$-ary $n$-cube Interconnection Networks. *IEEE Transactions on Computers*, 39(6):775–785, June 1990.

[7] A. Demars and S. Shenker. Analysis and Simulation of a Fair Queueing Algorithm. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 1–12, 1989.

[8] J. T. Draper and J. Ghosh. A Comprehensive Analysis Model for Wormhole Routing in Multicomputer Systems. *Journal of Parallel and Distributed Computing*, 32:202–214, 1994.

[9] J. Duato, S. Yalamanchili, M. B. Caminero, D. Love, and F. J. Quiles. MMR: A High-Performance Multimedia Router-Architecture and Design-Tradeoffs. In *Proceedings of the IEEE International Symposium on High-Performance Computer Architecture*, pages 300–309, January 1999.

[10] H. Eberle and E. Oertli. Switcherland: A QoS Communication Architecture for Workstation Clusters. In *Proceedings of the International Symposium on Computer Architecture*, pages 98–108, June 1998.

[11] M. Galles. Scalable Pipelined Interconnect for Distributed Endpoint Routing : The SGI SPIDER Chip. In *Proceedings of Symposium on High Performance Interconnects (Hot Interconnects)*, pages 141–146, August 1996.

[12] D. Garcia and W. Watson. Servernet II. In *Proceedings of the 1997 Parallel Computing, Routing, and Communication Workshop (PCRCW'97)*, June 1997.

[13] P. T. Gaughan and S. Yalamanchili. A Performance Model of Pipelined $k$-ary $n$-cubes. *IEEE Transactions on Computers*, 44(8):1059–1063, Auguest 1995.

[14] B. Kim, J. Kim, S. Hong, and S. Lee. A Real-Time Communication Method for Wormhole Switching Networks. In *Proceedings of International Conference on Paralle Processing*, pages 527–534, August 1998.

[15] E. J. Kim, K. H. Yum, and C. R. Das. An Analytical Model for a QoS Capable Cluster Interconnect. To be presented at 11th GI/ITG Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB 2001), September 2001.

[16] J. Kim and C. R. Das. Hypercube Communication Delay with Wormhole Routing. *IEEE Transactions on Computers*, 43(7):806–814, July 1994.

[17] J. H. Kim. *Bandwidth and Latency Guarantees in Low-Cost, High-Performance Networks*. PhD thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champaign, 1997.

[18] J. H. Kim and A. A. Chien. Rotating Combined Queueing (RCQ): Bandwidth and Latency Gurantees in Low-Cost, High-Performance Networks. In *Proceedings of the International Symposium on Computer Architecture*, pages 226–236, May 1996.

[19] J.-P. Li and M. Mutka. Priority Based Real-Time Communication for Large Scale Wormhole Networks. In *Proceedings of International Parallel Processing Symposium*, pages 433–438, May 1994.

[20] S. L. Scott and G. M. Thorson. The Cray T3E Network: Adaptive Routing in a High Performance 3D Torus. In *Proceedings of Symposium on High Performance Interconnects (Hot Interconnects)*, pages 147–156, August 1996.

[21] H. Song, B. Kwon, and H. Yoon. Throttle and Preempt: A New Flow Control for Real-Time Communications in Wormhole Networks. In *Proceedings of International Conference on Paralle Processing*, pages 198–202, August 1997.

[22] C. B. Stunkel, D. G. Shea, B. Abali, M. G. Atkins, C. A. Bender, D. G. Grice, P. Hochschild, D. J. Joseph, B. J. Nathanson, R. A. Swetz, R. F. Stucke, M. Tsao, and P. R. Varker. The SP2 High-Performance Switch. *IBM Systems Journal*, 34(2):185–204, 1995.

[23] K. H. Yum, E. J. Kim, and C. R. Das. QoS Provisioning in Clusters: An Investigation of Router and NIC Design. In *Proceedings of the International Symposium on Computer Architecture*, pages 120–129, June 2001.

[24] K. H. Yum, A. S. Vaidya, C. R. Das, and A. Sivasubramaniam. Investigating QoS Support for Traffic Mixes with the MediaWorm Router. In *Proceedings of the IEEE International Symposium on High-Performance Computer Architecture*, pages 97–106, January 2000.

[25] L. Zhang. VirtualClock: A New Traffic Control Algorithm for Packet-Switched Networks. *ACM Transactions on Computer Systems*, 9(2):101–124, May 1991.

## Appendix

**Computation of blocking probability($P_{b,c}$) in a single router**:

Here we describe the computation of blocking probability in a single router. Since the input/output buffer sizes are $b_s$, the blocking probability, $P_{b,c}$ for class $c$ ($1 \leq c \leq C$), can be expressed as

$$P_{b,c} = (\overline{L_c}\lambda'_c)^{1+2\frac{\max(b_s,M)}{M}} \tag{9}$$

where $\lambda'_c$ is the steady state message arrival rate of class $c$ traffic. ($\overline{L_c}\lambda'_c$) is the router utilization ($\rho_c$) for class $c$. Since $\lambda'_c$ and $\overline{L_c}$ are considered at the message-level granularity, the total buffer size($2b_s$ flits) of input and output queues becomes $2b_s/M$ when converted to message length. (Note that we are considering the worst case scenario here by using the entire buffer length $2b_s$.) Including the currently serviced flit/message, the total number of messages becomes $2b_s/M + 1$. Hence, the channel utilization (or blocking probability of class $c$) is given by Eq. 9.

The steady state arrival rate $\lambda'_c$ in Eq. 9 is given by

$$\lambda'_c = (1 - P_{b,c})\lambda^g_c. \tag{10}$$

The average network latency($\overline{L_c}$) of a message of class $c$ in Eq. 9 is given by

$$\overline{L_c} = P - 1 + (M + \overline{B_c})\overline{S_c}, \tag{11}$$

where $\overline{B_c} = P_{b,c}(\max(b_s, M) + M/2)$ and $\overline{S_c} = \sum_{k=0}^{2^{(C-2)}-1} S_c(k)P_c(k)$ . Note that due to the inter-dependencies between $P_{b,c}$ and $\lambda'_c$, the solution becomes iterative.

**Computation of blocking probability($P^s_{b,c}$) in a Cluster Interconnect**:

In order to get unknown term in Eq. 8, we need to compute the blocking probability($P^s_{b,c}$) in a Cluster Interconnect. From Eq. 9, the probability of blocking for class $c$ traffic in channel $s$ can be written as

$$P^s_{b,c} = (L_{c,s}\lambda_{c,s})^{1+2\frac{\max(b_s,M)}{M}}. \tag{12}$$

Note that $\lambda_{c,s}$ is the total rate(including transit rate and generation rate). To compute $\lambda_{c,s}$, we need to analysis the traffic rates in an $n$-cube router. Two types of messages arrive at a router using the input channels. One is called a terminating message, and the other is a transit message that passes through the router using one output channel. Let $\lambda^t_c$

be the total transit message rate of traffic $c$ at a router. The generation rate of traffic $c$ in the steady state is $\lambda'_c$. Therefore, the total message rate at the output of a router (over all the $n$ output channels) is $\lambda_c = \lambda^t_c + \lambda'_c$. Let $\lambda^{t,s}_c$ be the transit message arrival rate of traffic $c$ from other nodes at physical channel $s$ of a router. Similarly, $\lambda'_{c,s}$ represents traffic generated by the source node for a physical channel $s$ and virtual channel $c$. We give here the expressions derived in [16] for completeness.

The transient message arrival rate at physical channel $s$ and virtual channel $c$ of a router is given by

$$\lambda^{t,s}_c = \sum_{k=2}^{n} P_k \lambda'_c \left[ \sum_{j=m}^{M} \frac{{}_sC_{j-1} \cdot {}_{n-s-1}C_{k-j}}{{}_nC_k} \right]$$

where $0 \le s \le (n-1), m = \max(2, k-n+s+1)$, and $M = \min(s+1, k)$. The traffic generation rates have the following relations.

$$\lambda^t_c = \sum_{k=2}^{n} P_k (k-1)\lambda'_c, \quad \lambda_c = \lambda^t_c + \lambda'_c = \sum_{k=1}^{n} P_k k \lambda'_c$$

New messages for physical channel $s$ and VC $c$ are generated at a rate $\lambda^g_{c,s}$ by the local host, and is given by

$$\lambda^g_{c,s} = \sum_{k=1}^{n-s} P_k \lambda^g_c \cdot {}_{n-s-1}C_{k-1} / {}_nC_k.$$

The message rate for each virtual channel $c$ in an $n$-cube is the same regardless of its position, and is given as

$$\lambda_{c,s} = \lambda'_c \cdot \frac{\overline{h}}{n} = \frac{\lambda_c}{n}.$$

Similarly Eq. 10 is modified as

$$\lambda'_{c,s} = (1 - P^s_{b,c})\lambda^g_{c,s}. \tag{13}$$

In Eq. 12, $L_{c,s}$ is the latency of a class $c$ message when it uses in physical channel $s$ as the first path to traverse towards its destination. $L_{c,s}$ can be expressed as

$$\begin{aligned} L_{c,s} &= \{P - 1 + Ph_s\} \\ &+ \{(O_{c,n} + M) \cdot \overline{S_{c,n}}\} \\ &+ \{(I_{c,s} + B_{\text{middle}}(c,s)) \cdot \overline{S_{c,s}}\}. \end{aligned} \tag{14}$$

$h_s (= \sum_{k=0}^{n-s-1} (k+1) \cdot {}_{n-s-1}C_k / (2^{n-s-1}))$ in Eq.14 denotes the average number of hops a message travels starting with the physical channel $s$ as the first path. $I_{c,s} (= (P^s_{b,c} \cdot \max(b_s, M)/2))$ is the blocking length of a class $c$ real-time message at stage 1 of the first router that uses channel $s$ as the first route, and $O_{c,n} (= (P^n_{b,c} \cdot (\max(b_s, M)/2 + M/2)))$ is the blocking length at stages 3 and 5 in the ejection channel of the last router. Also, $B_{\text{middle}}(c,s)$ is the blocking length between the source and the destination (*i. e.* middle nodes) excluding the blocking length at stage 1 of the source and the blocking length at stages 3 and 5 of the destination. The computation of $B_{\text{middle}}(c,s)$ will be described later. $\overline{S_{c,s}} (= \sum_{k=0}^{2^{(C-2)}-1} S_{c,s}(k) P_{c,s}(k))$ is the average number of cycles required to transfer a flit of class $c$ message that uses channel $s$ for its first path, and $\overline{S_{c,n}} (= \sum_{k=0}^{2^{(C-2)}-1} S_{c,n}(k) P_{c,n}(k))$ is the average number of cycles per flit in the ejection channel.

**Computation of the probability of output VCs status, $P_c(k)$ :**

The probability of $k$th combination for class $c$, $P_c(k)$, can be determined using a Markov model. Let $S_1$ be a state such that the $c$th output buffer is empty and $S_2$ be the state such that the $c$th output buffer is not empty. The status of the rest $(C-2)$ buffers are all identical in the two states to make $S_1$ and $S_2$ adjacent. Let's assume the serial number of $S_2$ on class $c$ be $k$. (The detailed numbering function can be found in [15].) Now, the transition rate from state $S_1$ to $S_2$ is $\lambda'_c$, where $\lambda'_c$ is the traffic rate of the $c$th VC (Eq. 10), while the rate from $S_2$ to $S_1$ is $(1/L_c(k) - \lambda'_c)$, where $L_c(k) = P - 1 + (\overline{B}_c + M)S_c(k)$ from Eq. 11 and $S_c(k) = (\sum_{\forall j,j\text{th output VC is busy}} \frac{1}{Vtick_j}) / (\frac{1}{Vtick_c})$. The transition rate from $S_2$ is reduced by $\lambda'_c$ to account for the arrival of a message while channel $c$ is busy. From the Markov model, we get all the state probabilities, $\Pi_{S_i}$, Then,

$$P_c(k) = \frac{\Pi_{S_u}}{\sum_{\forall S_j \text{where } c\text{th VC is busy}} \Pi_{S_j}}, \tag{15}$$

where the serial number of $S_u$ on class $c$ is k.

The probability of output VCs status($P_{c,s}(k)$) in the network can be obtained similarly. Detailed computation of these probabilities can be found in [15].

**Computation of $B_{\text{middle}}(c,s)$ for Eq. 14**:

To compute $B_{\text{middle}}(c,s)$, we use the delay model from [16], except that we include the input and output queueing delay, while in [16] they capture only blocking delay. The average length of blocking in the middle nodes for a message which uses physical channel $s$ as the first path, $B_{\text{middle}}(c,s)$, is

$$B_{\text{middle}}(c,s) = (1 - \frac{P_1 \cdot \lambda'_c}{n \cdot \lambda'_{c,s}})$$

$$\times \quad \sum_{j=s+1}^{n-1} P_{b,c}^{j} \cdot (\max(M, b_s) + d_{c,j})$$

$$\times \quad \frac{\sum_{m=0}^{n-j-1} P_{m+2}(m+1)\frac{{}_{n-j-1}C_m}{{}_nC_{m+2}}}{\sum_{m=0}^{n-s-1} P_{m+2}\frac{{}_{n-s-1}C_{m+1}}{{}_nC_{m+2}}}$$

where $(1 - \frac{P_1 \cdot \lambda_c'}{n \cdot \lambda_{c,s}'})$ is the probability that a message does not terminate after using physical channel $s$ as the first path, and the last fractional expression represents the average number of hops the message travels when it takes a channel $j$ after using $s$.

The average length of messages involved in blocking for each channel is given as

$$d_{c,s} = \frac{1}{2}[\max(M, b_s) + M] + \frac{1}{2}(1 - P_{t,s})$$
$$\times \quad \sum_{j=s+1}^{n-1} P_{b,c}^{j} \cdot (d_{c,j} + \max(M, b_s)) \cdot H\{j|s\},$$

$$P_{t,s} = \sum_{k=0}^{s} P_{k+1} \cdot \frac{{}_sC_k}{{}_nC_{k+1}} \cdot \frac{1}{S(0)},$$

$$H\{j|s\} = \sum_{m=0}^{n-j-1} \sum_{k=0}^{s} P_{m+k+2} \cdot \frac{{}_{n-j-1}C_m \cdot {}_sC_k}{{}_nC_{m+k+2}} \cdot \frac{(m+1)}{S(1)},$$

where

$$S(j) = \sum_{m=j}^{n-s-1} {}_{n-s-1}C_m \cdot \sum_{k=0}^{s} P_{m+k+1} \cdot \frac{{}_sC_k}{{}_nC_{m+k+1}}.$$