An Analytical Model for Wormhole Routing with Finite Size Input Buffers

Po-Chi Hu[a] and Leonard Kleinrock[b]

[a]Lucent Technologies, Inc., 200 Schulz Drive, Red Bank, NJ 07701, USA

[b]Department of Computer Science, University of California at Los Angeles, Los Angeles, CA 90095-1596, USA*

Abstract

In this paper, we develop a queueing model for wormhole routing with finite size buffers. This model assumes the use of a deadlock-free routing scheme that guarantees no cycle of link dependency (defined in section 3). Several approximation methods for estimating the output link contention delay and buffer queueing delay are proposed. Comparing the analytical results to simulation, we show that the model is pessimistic with regard to network performance and that the difference in network throughput is less than 10 percent.

1 Introduction

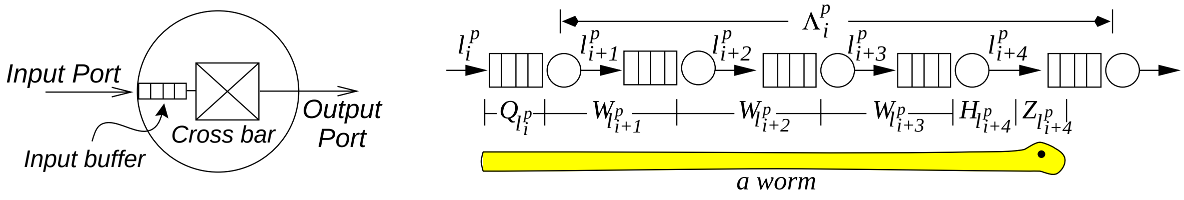
Wormhole routing is a simple, low-cost switching scheme often used for supercomputer interconnections. It has the merits of low latency, low cost, and simple implementation. In addition to its use for supercomputer interconnection, wormhole routing also has been applied to high-speed local area networks (LANs) [1, 2, 3] to support applications such as cluster computing that demand a very fast, high-data-rate communication media.

1.1 Wormhole Routing

Wormhole routing was developed from the earlier idea of *cut-through switching* [4], and was first introduced in [5]. A wormhole routing network is composed of several switches which have relatively small input buffers (see figure 1-a). As opposed to store-and-forward switching, a packet is forwarded to the next switch as soon as its header (or its routing information) is received (cut-through). If the outgoing link to the next switch is busy serving another packet, then the packet is blocked and resides in the network (see figure 1-b) until the outgoing link is available. In this case, called *blocking*, the switch must inform up-stream switches to stop transmission (i.e., it exercises *back-pressure flow control*) due to the limited size of buffers at each switch. A packet (which is also called a *worm*) may be buffered along a chain of switching nodes when blocked. Consequently,

*This work was supported by the Advanced Research Projects Agency, ARPA/CSTO, under Contract DABT63-93-C-0055 "The Distributed Supercomputer Supernet — A Multi Service Optical Intelligent Network".

deadlocks are possible unless a deadlock-free routing strategy is employed. A survey of wormhole routing can be found in [6].



(a) A wormhole routing switch (b) An illustration of notation and various delays

Figure 1: An illustration of wormhole routing.

1.2 Wormhole Routing Analysis

Many performance models for wormhole routing in a multi-processor environment have been proposed and presented in the literature [7, 8, 9, 10, 11]. However, they all assumed a negligible size of input buffers. This buffer size must increase in a LAN environment to accommodate transit data that cannot be stopped immediately due to the longer link propagation delay than in a multiprocessor interconnection application. As an example, a 640 Mbps Myrinet with a link length of 25 meters needs a buffer size of at least 54 bytes [2] per port to prevent data loss due to a buffer overflow or a transmission break due to the possibility of the buffer being empty before transmission is resumed. A LAN spanning hundreds of meters requires a buffer size larger than hundreds of bytes (a buffer size that could hold more than one packet). These buffers alleviate blocking problem. Thus, their effects must be captured in the model.

A finite size buffer complicates the analytical model in two ways. Firstly, the commonly used assumption that a worm reaches its destination before its tail leaves its source host, is no longer valid. It is now the case that a blocked worm may occupy only a fraction of the links along its path (not all of them). Secondly, a buffer may hold more than one, but not an infinite number, of worms. Buffering delay becomes difficult to estimate because the buffer size is finite (in terms of the amount of data).

To deal with the delay caused by blocking in the succeeding hops, knowledge of the dependency among all links is needed. To estimate the link blocking delay, the length of the link dependency chain must be resolved according to the worm size distribution. Approximations for determining the blocking chain length and the link blocking delay are presented in section 4. The finite size buffer is approximated through equivalent *M/G/1/K* queues with finite capacity. The structure of the equivalent queue and its solution is described in section 5. The entire modeling procedure is summarized in section 6. Section 7 shows comparison results with simulations. Section 8 concludes this paper.

2 Model Assumptions and Notation

The analysis work presented in this paper assumes the followings:

- a wormhole routing network using a deadlock-free routing that guarantees no cycle of link dependency. No cycle of link dependency is a sufficient, but not necessary, condition for deadlock free routing, as discussed in [12, 13].

- source routing. Routing is made by the source host and cannot be changed by switches (i.e., no deflection or adaptive routing).
- only one finite size buffer at each input port of a switch. Also, worms cannot share a link through interleaving (i.e., multiple virtual channels are not allowed).
- infinite size buffers at hosts.
- a Poisson worm arrival process and an arbitrary worm size distribution.

To facilitate this paper presentation, we measure packet length by *flits*, which is the amount of data that can be transmitted in one time unit. For example, the 640Mbps Myrinet [2] has one byte per flit lasting 12.5ns.

The followings define some notation used through this paper. The notation is also illustrated in figure 1-b.

$d_p =$ The length (number of hops) of path p.

$l_{ab} =$ The link that originates at node (a host or a switch) a and ends at node b.

$l^p_i =$ The ith link of path p; $1 \leq i \leq d_p$. If the ith link of path p originates at node a and ends at node b, then $l^p_i \equiv l_{ab}$.

$\eta_{l^p_i} =$ The propagation delay of link l^p_i.

$\mathcal{L}_p =$ The set of links which are traversed along path p.

$\mathcal{H}_a =$ The set of paths which originates at host a.

$\Delta =$ The buffer size, in terms of number of flits.

$\lambda_p =$ The arrival rate of worms that traverse along path p.

$\lambda_{l_{ab}} =$ The total worm arrival rate at l_{ab}.

$\gamma_a =$ The total worm arrival rate of worms at host a. $\gamma_a = \sum_{p:p \in \mathcal{H}_a} \lambda_p$.

$\ell =$ A random variable that denotes a worm size.

$L^*(s) =$ The Laplace-Stieltjes transform of the probability density function of ℓ.

$q_{l^p_i} =$ A random variable that denotes the delay of a worm head to reach the head of the input buffer for link l^p_i, after the worm has entered the buffer.

$Q^*_{l^p_j}(s) =$ The Laplace-Stieltjes transform of the probability density function of $q_{l^p_i}$.

$z_{l^p_i} =$ A random variable that denotes the delay of a worm head to reach the point where the accumulated buffer space is large enough to store the entire worm, after the worm head has entered the buffer for link l^p_i (see figure 1).

$Z^*_{l^p_j}(s) =$ The Laplace-Stieltjes transform of the probability density function of $z_{l^p_i}$.

$h_{l^p_i} =$ A random variable that denotes the contention delay for link l^p_i.

$H^*_{l^p_j}(s) =$ The Laplace-Stieltjes transform of the probability density function of $h_{l^p_i}$.

$\omega_{l^p_i} =$ A random variable that denotes the one-hop forwarding delay, excluding the link propagation delay, for the worm head to advance to the next hop (buffer head to buffer head) via link l^p_i.

$W^*_{l^p_j}(s) =$ The Laplace-Stieltjes transform of the probability density function of $\omega_{l^p_i}$.

$b_{l_i^p} =$ A random variable that denotes the link occupancy time of link l_i^p.

$B^*_{l_i^p}(s) =$ The Laplace-Stieltjes transform of the probability density function of $b_{l_i^p}$.

$B^*_{l_{ab}}(s) =$ The Laplace-Stieltjes transform of the probability density function of the link occupancy time at link l_{ab}.

$s_{l_i^p} =$ A random variable that denotes the service time of a worm via path p at the buffer for the ith link of path p.

$S^*_{l_i^p}(s) =$ The Laplace-Stieltjes transform of the probability density function of $s_{l_i^p}$.

$s_{l_{ab}} =$ A random variable that denotes the service time of a worm at the buffer for link l_{ab}.

$s_{l_{ab}}(\tau) =$ The probability density function of $s_{l_{ab}}$.

$S^*_{l_{ab}}(s) =$ The Laplace-Stieltjes transform of $s_{l_{ab}}(\tau)$.

$T_p =$ The average network delay for worms via path p.

3 Ordering Links

A wormhole routing network differs from a virtual cut-through network because of its link blocking feature. Blocking occurs due to the small size of the input buffers and results in increased link occupancy time. This occupancy time (defined as the time interval that a served worm holds this link) is not only a function of the worm size, but also a function of the blocking delay in the succeeding hops. As a consequence, it is important to find the dependency among links. The *link dependency* and the *cycle of link dependency* are defined as follows:

Definition 1 *We say that l_{ab} depends on l_{cd}, if $\exists p$, such that l_{cd} is a subsequent link of l_{ab} in path p. This dependency is represented as $l_{ab} \prec l_{cd}$. Moreover, if $l_{ab} \prec l_{cd}$, and $l_{cd} \prec l_{ef}$, then we say $l_{ab} \prec l_{ef}$, too (i.e., it is transitive).*

Note that it is possible that $l_{ab} \prec l_{ef}$ but l_{ab} is not a subsequent link of l_{ef} in any path, according to the transitive property.

Definition 2 *We say that there is a cycle of link dependency if $\exists l_{ab}, l_{cd}$ such that $l_{ab} \prec l_{cd}$ and $l_{cd} \prec l_{ab}$.*

Link dependency provides the relationship between link occupancy time and blocking time. In our earlier paper [11], we developed the relations between their distributions but relied on iterative methods to find the solution. Actually, a *computation order*, which indicates the sequence of links for blocking delay analysis can be derived if there is no cycle of link dependency, as illustrated in [14]. The method is simply the *topological sorting* [15]. For examples, if $l_{ab} \prec l_{cd} \prec l_{ef}$, we have a computation order, $l_{ef} \rightarrow l_{cd} \rightarrow l_{ab}$. Following the computation order, link occupancy time and blocking time can be evaluated link by link without iterations.

4 Link Occupancy Time

To estimate the blocking delay at each switch, it is important to first analyze how the finite size buffer affects link status and worm transmission. When there is no buffer available at switches, the relation between the link occupancy time ($b_{l^p_i}$) and waiting time ($\omega_{l^p_i}$) has been well established in [11]. The Laplace-Stieltjes transform equation is:

$$B^*_{l^p_i}(s) = L^*(s) \prod_{j=i+1}^{d_p} W^*_{l^p_j}(s) \tag{1}$$

Introducing a finite size buffer on each input port reduces the number of links that a worm can spread over. In other words, the link occupancy time is only affected by a limited number of subsequent links, not all of them. Given a worm size ℓ, the number of effective subsequent links for a worm at the ith link of path p (l^p_i), $\Lambda^p_i(\ell)$, is derived by:

$$\Lambda^p_i(\ell) = \begin{cases} \left\lfloor \frac{\ell}{\Delta} \right\rfloor & \text{if } \frac{\ell}{\Delta} < d_p - i \\ d_p - i & \text{otherwise} \end{cases} \tag{2}$$

As shown in figures 1-b, blocking that occurs after the next $\Lambda^p_i(\ell)$ links does not affect the link occupancy time since the accumulated buffer space is large enough to hold the entire worm.

Now, we define random variables $x_{l^p_i}$ and $y_{l^p_i}$ as:

$$\begin{aligned} x_{l^p_i} &= \begin{cases} q_{l^p_i} + h_{l^p_{i+\Lambda^p_i(\ell)}} + z_{l^p_{i+\Lambda^p_i(\ell)}} + \sum_{j=i+1}^{i+\Lambda^p_i(\ell)-1} \omega_{l^p_j} & \text{if } \Lambda^p_i(\ell) \geq 1 \\ z_{l^p_i} & \text{otherwise} \end{cases} \\ y_{l^p_i} &= \ell + x_{l^p_i} = \begin{cases} \ell + q_{l^p_i} + h_{l^p_{i+\Lambda^p_i(\ell)}} + z_{l^p_{i+\Lambda^p_i(\ell)}} + \sum_{j=i+1}^{i+\Lambda^p_i(\ell)-1} \omega_{l^p_j} & \text{if } \Lambda^p_i(\ell) \geq 1 \\ \ell + z_{l^p_i} & \text{otherwise} \end{cases} \end{aligned}$$

The random variable $x_{l^p_i}$ represents the forwarding delay for the worm head to reach the position where a large enough buffer space has been accumulated to hold the entire worm (see figure 1-b).

Let $\Psi_{l^p_i}(k)$ denote the probability that $\Lambda^p_i(\ell) = k$. From equation (2), we have,

$$\Psi_{l^p_i}(k) = \begin{cases} \mathbf{Prob}\{\ell < (k+1)\Delta\} - \mathbf{Prob}\{\ell < k\Delta\} & \text{if } \frac{\ell}{\Delta} < d_p - i \\ 1 - \mathbf{Prob}\{\ell < k\Delta\} & \text{otherwise} \end{cases} \tag{3}$$

Then, the Laplace-Stieltjes transforms of the probability density functions of $x_{l^p_i}$ and $y_{l^p_i}$ are:

$$\mathcal{X}^*_{l^p_i}(s) = \Psi_{l^p_i}(0) Z^*_{l^p_i}(s) + \sum_{j=1}^{d_p-i} \left[\Psi_{l^p_i}(j) Q^*_{l^p_i}(s) H^*_{l^p_{i+j}}(s) Z^*_{l^p_{i+j}}(s) \prod_{k=i+1}^{i+j-1} W^*_{l^p_k}(s) \right] \tag{4}$$

$$\mathcal{Y}^*_{l^p_i}(s) = L^*(s) \mathcal{X}^*_{l^p_i}(s) \tag{5}$$

Observing figures 1-b and 2, the link occupancy time clearly has the range:

$$\max\left[\ell, x_{l^p_i}\right] \leq b_{l^p_i} \leq y_{l^p_i} \tag{6}$$

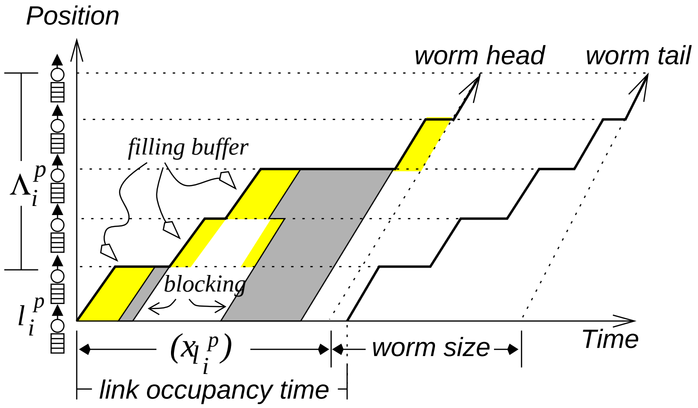


Figure 2: An illustration of the link occupancy time.

Since buffers tend to be fully utilized under severely blocking conditions, the left hand side of inequality (6) should be adopted to approximate $b_{l_i^p}$ when the forwarding delay, $x_{l_i^p}$, dominates. Also, the average link occupancy time should be monotonically increasing as the forwarding delay increases, and must be at least as large as the worm size. To satisfy all of the above, the following approximation is proposed for the link occupancy time distribution:

$$B^*_{l_i^p}(s) = \begin{cases} \left(\overline{\mathcal{Y}_{l_i^p}} + \overline{\mathcal{X}_{l_i^p}}\right)^{-1} \left[\left(\overline{\mathcal{Y}_{l_i^p}} - \overline{\mathcal{X}_{l_i^p}}\right) \mathcal{Y}^*_{l_i^p}(s) + 2\overline{\mathcal{X}_{l_i^p}} L^*(s)\right] & \text{if } \overline{L} > \overline{\mathcal{X}_{l_i^p}} \\ \left(\overline{\mathcal{Y}_{l_i^p}} + \overline{\mathcal{X}_{l_i^p}}\right)^{-1} \left[\left(\overline{\mathcal{Y}_{l_i^p}} - \overline{\mathcal{X}_{l_i^p}}\right) \mathcal{Y}^*_{l_i^p}(s) + 2\overline{\mathcal{X}_{l_i^p}} \mathcal{X}^*_{l_i^p}(s)\right] & \text{if } \overline{L} \le \overline{\mathcal{X}_{l_i^p}} \end{cases} \tag{7}$$

where $\overline{\mathcal{X}_{l_i^p}}$ is the first moment of $\mathcal{X}^*_{l_i^p}(s)$, and similarly for $\overline{\mathcal{Y}_{l_i^p}}$ and $\overline{L}$.

It can be shown that equation (7) has the limit values, $\lim_{\overline{\mathcal{X}_{l_i^p}} \to 0} B^*_{l_i^p}(s) = \mathcal{Y}^*_{l_i^p}(s)$ and $\lim_{\overline{\mathcal{X}_{l_i^p}} \to \infty} B^*_{l_i^p}(s) = \mathcal{X}^*_{l_i^p}(s)$, since $\overline{\mathcal{Y}_{l_i^p}} = \overline{\mathcal{X}_{l_i^p}} + \overline{L}$. Moreover, $\overline{B_{l_i^p}}$ derived by equation (7) is monotonically increasing with $\overline{\mathcal{X}_{l_i^p}}$, as proven in [14].

The remaining $W^*_{l_j^p}(s)$, $Z^*_{l_j^p}(s)$, $H^*_{l_j^p}(s)$, and $Q^*_{l_j^p}(s)$ quantities are discussed in section 5.

5 Modeling the Finite Size Buffer

Since buffer capacity is fixed in terms of the number of flits, the nature of the input buffer resembles a finite dam system. A worm flows in the buffer constantly when it is not full. However, the outgoing flow of the buffer may be interrupted due to worm blocking. The queueing model for a finite dam system developed in [16] cannot be applied directly in this case. Furthermore, the status of the buffer is tightly related to its upstream node, and vice versa. To analyze both independently could result in a poor model. For the sake of accuracy and simplicity, we use an alternative approach which treats both link contention and the input buffer as one single queue.

5.1 M/G/1/K Approximation

As shown in figure 3, the delay for a worm to seize its output link and reach the buffer head in the next hop (i.e., the one-hop forwarding delay, $\omega_{l_i^p}$) is exactly the waiting time of an *M/G/1* queue with finite capacity (denoted as *M/G/1/K*, for the case that capacity is K). With κ input ports, the *M/G/1/K* queue (see figure 3) has the capacity

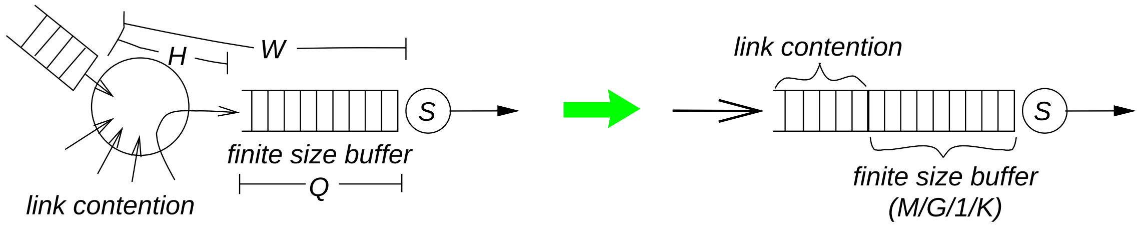


Figure 3: The forwarding delay is considered as a single queue with finite capacity. The queue includes the input buffer at the end of the link and contention for this link.

approximately $\kappa + \vartheta$, where ϑ is the number of worms that can be completely held in the portion of the finite size buffer. Unfortunately, the buffer size is determined as the number of flits, not the number of worms. For variable worm size cases, ϑ is not deterministic. To simplify the analysis of this finite size buffer, equivalent queues are used here instead. In general, an equivalent queue size specifies how many worms can be held in the buffer and is associated with a probability. Specifically, the buffer is approximated as a queue of capacity $\kappa + k$ with the probability $\vartheta(k)$ that,

$$\begin{aligned} \vartheta(k) &= \mathbf{Prob}\{\ell_1 + \cdots + \ell_k \leq \Delta < \ell_1 + \cdots + \ell_{k+1}\} \\ &= \mathbf{Prob}\{\ell_1 + \cdots + \ell_k \leq \Delta\} - \mathbf{Prob}\{\ell_1 + \cdots + \ell_{k+1} \leq \Delta\} \end{aligned} \tag{8}$$

Then, the one-hop forwarding delay is estimated as

$$W^*_{l_i^p}(s) = \sum_{j=0}^{\infty} \vartheta(j) \Upsilon^*_{l_i^p}(j + \kappa, s) \tag{9}$$

where $\Upsilon^*_{l_i^p}(j + \kappa, s)$ is the Laplace-Stieltjes transform of the probability density function of the waiting time for the equivalent queue that has capacity $j + \kappa$ (j is from the finite size buffer and κ is from the link contention). The range of the capacity is actually finite, since the worm size must be larger than a flit.

The finite capacity queue of figure 3 is modeled with the Poisson arrival assumption. This assumption is justified by the multiplexing of various inputs and demultiplexing of outputs [17]. The details of the procedure to solve the steady-state probability and waiting time distribution, $\Upsilon^*_{l_i^p}(j + \kappa, s)$, of an *M/G/1/K* queue is available in [18]. The solution is lengthy and hence not reproduced in this paper. Nevertheless, a few changes about the procedure should be noted here. First, the total worm arrival rate on link l_{ab} is derived as: $\lambda_{l_{ab}} = \sum_{p:l_{ab} \in \mathcal{L}_p} \lambda_p$. However, to apply the solutions for the *M/G/1/K* queue, $\lambda_{l_{ab}}$ needs to be normalized with the probability of encountering a full queue. That is,

$$\lambda'_{l_{ab}} = \frac{\lambda_{l_{ab}}}{1 - P_B} \tag{10}$$

where P_B is the probability of no waiting room left (blocking) in the *M/G/1/K* queueing system [18, Chapter 5, page 202], and it, as well as the steady-state probabilities, can be derived if the normalized traffic arrival rate, $\lambda'_{l_{ab}}$, is known. Therefore, an iterative method (e.g., *bi-section* [19]) needs to be applied to solve P_B and $\lambda'_{l_{ab}}$ first (see [14] for details).

Another change is about the integration [18, Chapter 5, equation (1.7)],

$$\int_0^\infty \frac{\left(\lambda'_{l_{ab}}\tau\right)^k}{k!} e^{-\lambda'_{l_{ab}}\tau} s_{l_{ab}}(\tau) d\tau$$

Though $s_{l_{ab}}(\tau)$ can be recovered by inverting its Laplace-Stieltjes transform, $S^*_{l_{ab}}(s)$, the inversion is not completely systematic. To ease this difficulty, a two-moment approximation can be exploited.

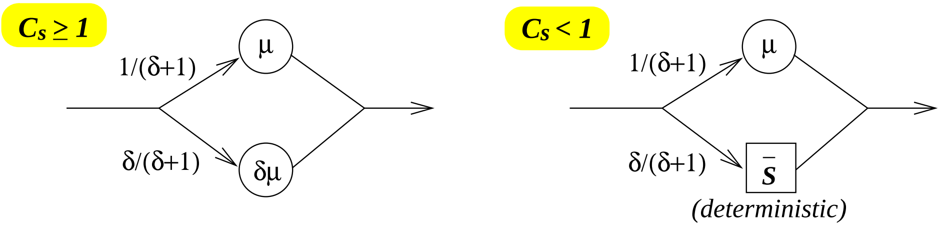


Figure 4: The two-stage approximation for a distribution function.

Moments of $s_{l_{ab}}$ are obtainable from $S^*_{l_{ab}}(s)$ by differentiation and setting $s = 0$. With the first two moments of $s_{l_{ab}}$, $\overline{S_{l_{ab}}}$ and $\overline{S^2_{l_{ab}}}$, the probability density function of $s_{l_{ab}}$ can be approximated as (figure 4):

$$s_{l_{ab}}(\tau) = \begin{cases} \frac{1}{\delta+1}\mu e^{-\mu\tau} + \frac{\delta}{\delta+1}\delta\mu e^{-\delta\mu\tau} & \text{if } C^2_{s_{l_{ab}}} \geq 1 \\ \frac{1}{\delta+1}\mu e^{-\mu\tau} + \frac{\delta}{\delta+1}u_0\left(\tau - \overline{S_{l_{ab}}}\right) & \text{if } C^2_{s_{l_{ab}}} < 1 \end{cases} \tag{11}$$

where $C_{s_{l_{ab}}} = \frac{\sqrt{\overline{S^2_{l_{ab}}} - \overline{S_{l_{ab}}}^2}}{\overline{S_{l_{ab}}}}$ is the coefficient of variation for $s_{l_{ab}}$, and $u_0(\tau)$ is the unit impulse function [20, Appendix I.3].

To match the first two moments, we have,

$$\overline{S_{l_{ab}}} = \int_0^\infty \tau \left(\frac{1}{\delta+1}\mu e^{-\mu\tau} + \frac{\delta}{\delta+1}\delta\mu e^{-\delta\mu\tau}\right) d\tau = \frac{2}{(\delta+1)\mu} \tag{12}$$

$$\overline{S^2_{l_{ab}}} = \int_0^\infty \tau^2 \left(\frac{1}{\delta+1}\mu e^{-\mu\tau} + \frac{\delta}{\delta+1}\delta\mu e^{-\delta\mu\tau}\right) d\tau = \frac{2}{\delta\mu^2} \tag{13}$$

for the case, $C^2_{s_{l_{ab}}} \geq 1$.

After some manipulation of the above equations, we find: $\delta = C^2_{s_{l_{ab}}} \pm \sqrt{C^4_{s_{l_{ab}}} - 1}$ and $\mu = \frac{2}{(\delta+1)\overline{S_{l_{ab}}}}$. Similarly for the case $C^2_{s_{l_{ab}}} < 1$, we find: $\delta = \frac{1}{C^2_{s_{l_{ab}}}} - 1$ and $\mu = \frac{1}{\overline{S_{l_{ab}}}}$. With μ and δ, $s_{l_{ab}}(\tau)$ and its moments can be approximated. This two moment approximation is also applied for other distributions that need the probability density function explicitly.

Finally, $\Upsilon^*_{l^p_i}(K,s)$ is evaluated through [18, Chapter 5, equation (1.75)]:

$$\Upsilon^*_{l^p_i}(K,s) = \frac{\pi_0 s \left(1 - \left[\frac{\lambda'_{l^p_i} S^*_{l_{ab}}(s)}{\lambda'_{l^p_i} - s}\right]^K\right)}{s - \lambda'_{l^p_i} + \lambda'_{l^p_i} S^*_{l_{ab}}(s)} + \left[S^*_{l_{ab}}(s)\right]^{K-1} \sum_{j=0}^{K-1} \pi_j \left(\frac{\lambda'_{l^p_i}}{\lambda'_{l^p_i} - s}\right)^{K-j} \tag{14}$$

where π_j is the steady-state probability of j worms in the queue.

5.2 Buffering Delay and More

The buffering delay, $Q^*_{l^p_j}(s)$ is derived simply as $Q^*_{l^p_i}(s) = \frac{W^*_{l^p_i}(s)}{H^*_{l^p_i}(s)}$ because $\omega_{l^p_i} = q_{l^p_i} + h_{l^p_i}$, as shown in figure 3. The contention blocking, $H^*_{l^p_i}(s)$, can also be approximated as an *M/G/1/K* queue with $K = \kappa$, and the queue service time is exactly the link occupancy time, $B^*_{l_{ab}}(s)$, if $l^p_i \equiv l_{ab}$. However, $B^*_{l_{ab}}(s)$ is not available until $Q^*_{l^p_i}(s)$ is known, which requires knowledge of $H^*_{l^p_i}(s)$ as shown in the above. Consequently, $B^*_{l_{ab}}(s)$ must be properly approximated first in order to derive $H^*_{l^p_i}(s)$ and $Q^*_{l^p_i}(s)$. A simple approximation is proposed as the following:

$$B^*_{l_{ab}}(s) = \mathbf{Prob}\{\text{buffer full}\} S^*_{l_{ab}}(s) + (1 - \mathbf{Prob}\{\text{buffer full}\})\, L^*(s) \tag{15}$$

This approximation is based on the following observations:

1. When the buffer is full, it simply resembles a data pipe — one flit of data out of the buffer corresponds to one flit of data entering the buffer. Thus, $B^*_{l_{ab}}(s) = S^*_{l_{ab}}(s)$, in this case.
2. When there is space left in the buffer, a worm flows in the buffer without interruption. Thus, $B^*_{l_{ab}}(s) = L^*(s)$.

The buffer full probability can be closely estimated from the steady-state probability that is derived when we analyze $W^*_{l^p_i}(s)$, namely, the probability that more than ϑ worms are in the *M/G/1/K* queue used to approximate $W^*_{l^p_i}(s)$. ϑ is an equivalent queue size of the finite size buffer. Therefore, we have,

$$\mathbf{Prob}\{\text{buffer full}\} = \sum_{j=0}^{\infty} \vartheta(j) \left((1 - P^j_B) \sum_{k=j}^{j+\kappa-1} \pi^j_k + P^j_B \right) \tag{16}$$

where π^j_k, P^j_B denote the π_k, P_B of the *M/G/1/K* queue with $K = j + \kappa$.

Finally, $Z^*_{l^p_i}(s)$ is ignored, since it is small and implicitly included in $H^*_{l^p_i}(s)$ due to the fact that the equivalent queue used to approximate the finite size buffer does not count the buffer space that can only hold part of a worm. This delay is not recounted here. After $W^*_{l^p_i}(s)$, $H^*_{l^p_i}(s)$ and $Q^*_{l^p_i}(s)$ are derived, $B^*_{l^p_i}(s)$ is given by equation (7).

Now, the service time distribution for a worm through path p at the head of its ith hop input buffer is derived as $s_{l^p_i} = h_{l^p_{i+1}} + b_{l^p_{i+1}}$, which gives us:

$$S^*_{l^p_i}(s) = H^*_{l^p_{i+1}}(s) B^*_{l^p_{i+1}}(s) \tag{17}$$

Considering worms from different paths, the service time distribution for a finite size input buffer is:

$$S^*_{l_{ab}}(s) = \sum_{p: l_{ab} \in \mathcal{L}_p} \frac{\lambda_p}{\lambda_{l_{ab}}} S^*_{l^p_{\xi_p(l_{ab})}}(s) \tag{18}$$

where $\xi_p(l_{ab})$ is a function which returns i if link l_{ab} is the ith link of path p.

Once the service time and mean forwarding delay at each hop is derived, the network delay is obtained as (see figures 1-b and 2):

$$T_p = \overline{v_a} + \sum_{i=2}^{d_p} \overline{w_{l_i^p}} + \overline{L} + \sum_{i=1}^{d_p} \eta_{l_i^p} \tag{19}$$

if path p originates at host a. $\overline{v_a}$ is the mean of the queueing delay at host a. Note that the buffering delay at the first link is not counted, since it is part of the host queueing delay, which is directly derived from the *M/G/1* queue solution [20], $\overline{v_a} = \frac{\gamma_a \overline{S^2_{l_{ab}}}}{2(1-\gamma_a \overline{S_{l_{ab}}})}$.

6 Model Summary

Here, we summarize the full modeling process.

1. Read in the network topology.
2. Read in all paths and their worm arrival rates, λ_p.
3. Compute the worm arrival rate at each single link (e.g., $\lambda_{l_{ab}}$).
4. With the given worm size distribution, compute $\Psi_{l_i^p}(j)$ and $\vartheta(j)$, $\forall j$ and l_i^p.
5. Use *topological sorting* (see [14]) to construct the link computation order.
6. For $k = 1$ to the highest order, compute (in the following order) $S^*(s)$, $W^*(s)$, $H^*(s)$, $Q^*(s)$, and $B^*(s)$ for all links belonging to order k. The distribution of all of the above may actually be characterized by their first two moments.
7. Compute T_p for all paths p.

The entire procedure can be computerized except for step 4. Step 4 involves integration and other operations that require manual effort. However, once they are completed for a given worm size distribution, the rest can be done automatically for any network configuration. Note that the Laplace-Stieltjes transform for each probability density function does not need to be solved explicitly. They are only used for the convenience of presentation. Only the first two moments of each distribution are required.

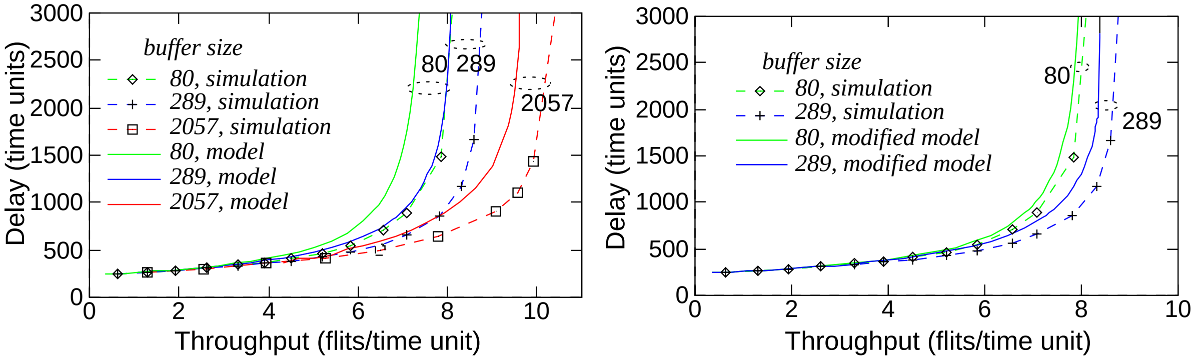
7 Comparison of Results

Using a 3×3 torus (totally, 9 switches and 36 hosts) with up/down deadlock-free routing [1] and symmetric traffic load (see [14] for details), the performance results estimated by both the model and simulation are shown in figure 5-a. The results are derived with the assumptions of exponential worm size distribution and Poisson worm arrivals. Figure 5-a indicates a ten percent difference between the network throughput estimated by the model and the simulation, in both small buffer size and large buffer size cases. Also, the analytical results are always pessimistic, compared to the simulation.

Some of the approximations could be modified or improved to compensate for the pessimism of the model. First, the finite size buffer may be better approximated by an equivalent queue with higher capacity. The current approximation:

$$\vartheta(j) = \textbf{Prob}\,\{\ell_1 + \cdots + \ell_k \leq \Delta < \ell_1 + \cdots + \ell_{k+1}\}$$

ignores the buffer space that can not hold a full worm. The buffer size is underestimated and consequently the analysis overestimates the network delay. Furthermore, the service



(a) The original model

(b) The modified model

Figure 5: Results of the finite size buffer models (worm size = 200 flits, propagation delay = 10 time units).

time of the equivalent queue depends on the queue capacity (in terms of number of worms). A larger capacity clearly implies a smaller average size of worms in the queue, due to the fact that the buffer size is fixed in terms of number of flits. As a result, the service rate must be higher in a high capacity case than in a low capacity one. Without consideration of the above dependency, the worm forwarding delay is overestimated.

In figure 5-b, we show the analytical results of a modified model (see below) with regard to the above discussion. The probability of the number of worms that can be held in the finite size buffer, $\vartheta(j)$, is reformalized by enlarging the buffer size to $\Delta + \frac{\overline{L}}{2}$. The $\frac{\overline{L}}{2}$ portion counts the buffer space that cannot hold a full worm. Also, the moments of the buffer service time distribution are adjusted with the queue capacity, which gives a new average worm size. Namely, an equivalent queue with capacity $j + \kappa$ (j from the finite size buffer, and κ from the link contention) has a new mean buffer service time: $\left[new\ \overline{S_{l_{ab}}}\right] = \frac{\Delta+\kappa\overline{L}}{(\kappa+j)\overline{L}}\overline{S_{l_{ab}}}$ and similarly, $\left[new\ \overline{S^2_{l_{ab}}}\right] = \left(\frac{\Delta+\kappa\overline{L}}{(\kappa+j)\overline{L}}\right)^2\overline{S^2_{l_{ab}}}$. The predicted network performance in figure 5-b is closer to the simulation. However, the model is still pessimistic.

8 Summary

In this paper, a finite size buffer model for wormhole routing is developed. It is shown that this analysis is not trivial and needs many approximations. To further improve these approximations require intensive study of several sophisticated queueing models. However, the full modeling procedure presented in this paper is systematic and could be implemented as a useful tool.

References

[1] M.D. Schroeder, A.D. Birrell, M. Burrows, H. Murray, et al. "Autonet: A High-speed, Self-configuring Local Area Network Using Point-to-point Links". *IEEE Journal on Selected Areas in Communications*, 9(8):1318–1335, October 1991.

[2] C. Seitz, D. Cohen, and R. Felderman. "Myrinet—A Gigabit-per-second Local-Area Network". *IEEE Micro*, 15(1):29–36, February 1995.

[3] et. al L. Kleinrock. "The Supercomputer Supernet: A Scalable Distributed Terabit Network". *Journal of High Speed Networks: special issue on Optical Networks*, 4(4):407–24, 1995.

[4] P. Kermani and L. Kleinrock. "Virtual cut-through: A New Computer Communication Switching Technique". *Computer Networks*, 3:267–289, 1979.

[5] C. Seitz *et al.* "The Hypercube Communications Chip". Technical report, Dep. Computer Science, California Inst., March 1985. Display File 5128:DF:85.

[6] L. M. Ni and P. K. McKinley. "A Survey of Wormhole Routing Techniques in Direct Networks". *Computer*, pages 62–76, February 1993.

[7] W. J. Dally. "Performance Analysis of K-ary n-cube Interconnection Networks". *IEEE Trans. on Computers*, 39(6), June 1990.

[8] W-J. Guan, W. K. Tsai, and D. Blough. "An Analytical Model for Wormhole Routing in Multicomputer Interconnection Networks". In *Proceedings of Seventh International Parallel Processing Symposium*, pages 650–654, April 1993.

[9] J. Kim and C. R. Das. "Hypercube Communication Delay with Wormhole Routing". *IEEE Trans. on Computers*, 43(7), July 1994.

[10] J. T. Draper and J. Ghosh. "A Comprehensive Analytical Model for Wormhole Routing in Multicomputer Systems". *Journal of Parallel and Distributed Computing*, 23:202–214, November 1994.

[11] Po-Chi Hu and L. Kleinrock. "A Queueing Model for Wormhole Routing with Timeout". In *Proceedings of the 4th International Conference on Computer Communications and Networks*, pages 584–593, Las Vegas, NV, U.S., September 1995.

[12] J. Duato. "A Necessary and Sufficient Condition for Deadlock-free Adaptive Routing in Wormhole Networks". *IEEE Transactions on Parallel and Distributed Systems*, 6(10):1055–67, October 1995.

[13] L. Schwiebert and D.N. Jayasimha. "A Necessary and Sufficient Condition for Deadlock-free Wormhole Routing". *Journal of Parallel and Distributed Computing*, 32(1):103–117, January 1996.

[14] Po-Chi Hu. *High-Speed Local Area Networks Using Wormhole Routing: Modeling and Extensions*. PhD thesis, University of California, Los Angeles, June 1996.

[15] Ralph P. Grimaldi. *Discrete and Combinatorial Mathematics : An Applied Introduction*. Addison-Wesley, Reading, Mass., 2nd edition, 1989.

[16] Jacob Willem Cohen. *The Single Server Queue*. North-Holland Pub. Co., revised edition, 1982.

[17] L. Kleinrock. *Communication Nets: Stochastic Message Flow and Delay*. MgGraw-Hill, New York, 1964. Reprinted by Dover Publications, 1972.

[18] Hideaki Takagi. *Queueing Analysis : A Foundation of Performance Evaluation*, volume 2. North-Holland, New York, NY, U.S.A., 1993.

[19] William H. Press *et al. Numerical Recipes: The Art of Scientific computing*. Cambridge University Press, New York, 1986.

[20] L. Kleinrock. *Queueing Systems, Vol. I: Theory*. Wiley Interscience, New York, 1975.