5 Conclusions

This paper has derived an accurate model for predicting the performance of wormhole routed 2D-mesh networks. It has also presented empirical simulations and comparisons showing that the 2D-mesh network exhibits better performance than the torus.

Earlier comparisons among k-ary n-cubes [4] have favored low-dimensional networks under the important hardware cost measure of equal bisection width. The results of this paper show that an even more dramatic comparison can be obtained by stepping out of the general k-ary n-cube framework, in which the two-dimensional network is the torus, to consider the two-dimensional mesh.

One of the main directions for future research is to generalize beyond the uniform random message patterns considered here. It would be desirable to compare networks in terms of the performance of real parallel algorithms and to achieve a better understanding of the sensitivity of different algorithms to throughput and latency.

A Performance Model for the Wormhole Routed Torus

$$\lambda_{net} = \lambda_{node}\frac{k-1}{2}\frac{k^n}{k^n-1}$$

$$\lambda_{new} = \lambda_{node}\frac{k^{n-1}(k-1)}{k^n-1}$$

$$\lambda_{sw} = \lambda_{node}\frac{(k-1)^2}{k^n-1}$$

$$\lambda_j = \begin{cases} \lambda_{net} & \text{for } j \le 1 \\ \lambda_{node}\frac{k^{n-1}(k(k-1)-j(j-1))}{2(k^n-1)} & \text{for } j > 1 \end{cases}$$

$$flit_{xfer} = 1 + \frac{\lambda_{net}}{2}msgl$$

$$x_{base} = flit_{xfer}\cdot msgl$$

$$F_W(y,z) = \frac{yz^2\left(1+\left(\frac{z-x_{base}}{z}\right)^2\right)}{2(1-yz)}$$

$$x_{j_0} = \begin{cases} x_{base} & j_0 = 0;\text{ else} \\ \frac{k+j_0-3}{k+j_0-1}\left(x_{j_0-1} + W_{j_0-1}\frac{\lambda_{new}}{\lambda_{j_0-1}}\right) + \frac{2}{k+j_0-1}x_0 \end{cases}$$

$$W_{j_0} = F_W(\lambda_{j_0}, x_{j_0})$$

$$x_{j_0,j_1} = \begin{cases} \frac{x_{base}}{k} + \frac{k-1}{k}\left(x_{j_0} + W_{j_0}\left(1-\frac{\lambda_{sw}}{2\lambda_{j_0}}\right)\right) & j_1 = 0; \\ \frac{k+j_1-3}{k+j_1-1}\left(x_{j_0,j_1-1} + \frac{\lambda_{new}W_{j_0,j_1-1}}{\lambda_{j_1-1}}\right) + \frac{2x_{j_0,0}}{k+j_1-1} \end{cases}$$

$$W_{j_0,j_1} = F_W(\lambda_{j_1}, x_{j_0,j_1})$$

$$\begin{aligned} x^{inj}_{j_0,j_1} &= \left[x_{j_0} + \left(1-\frac{\lambda_{node}}{\lambda_{j_0}}\frac{1}{k+1}\right)W_{j_0}\right]\frac{1}{k+1} \\ &+ \left[x_{j_0,j_1} + \left(1-\frac{\lambda_{node}}{\lambda_{j_1}}\frac{k}{k+1}\right)W_{j_0,j_1}\right]\frac{k}{k+1}, \end{aligned}$$

$$W^{inj}_{j_0,j_1} = F_W(\lambda_{node}, x^{inj}_{j_0,j_1})$$

$$\overline{D} = \frac{k^2}{k+1}$$

$$\overline{L} = \frac{1}{k^2}\sum_{j_0=0}^{k-1}\sum_{j_1=0}^{k-1}\left(x^{inj}_{j_0,j_1} + W^{inj}_{j_0,j_1}\right) + (\overline{D}-1)flit_{xfer} + 2$$

References

[1] Seth Abraham and Krishnan Padmanabhan. Performance of multicomputer networks under pin-out constraints. *Journal of Parallel and Distributed Computing*, pages 237–248, December 1991.

[2] Tilmann Bönniger, Rüdiger Esser, and Dietrich Krekel. CM-5E, KSR2, Paragon XP/S: A comparative description of massively parallel computers. *Parallel Computing*, 21:199–232, 1995.

[3] http://www.cray.com/products/systems/crayt3e.

[4] William J. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Trans. Computers*, 39(6):775–785, June 1990.

[5] William J. Dally and Charles L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Computers*, C-36(5):547–553, May 1987.

[6] Jeffrey T. Draper and Joydeep Ghosh. A comprehensive analytical model for wormhole routing in multicomputer systems. *Journal of Parallel and Distributed Computing*, 23:202–214, 1994.

[7] Jae H. Kim and Andrew A. Chien. Network performance under bimodal traffic loads. *Journal of Parallel and Distributed Computing*, 28:43–64, 1995.

[8] Jong Kim and Chita R. Das. Hypercube communication delay with wormhole routing. *IEEE Trans. Computers*, 43(7):806–814, July 1994.

$$x_{\langle j_0, j_1, ej\rangle} = s/f \quad (11)$$

$$x_{\langle j_0\rangle} = (s/f)\,\frac{1}{j_0} + \left[x_{\langle j_0-1\rangle} + \frac{1}{k-j_0+1}W_{\langle j_0-1\rangle}\right]\frac{j_0-1}{j_0} \quad (12)$$

$$\begin{aligned} x_{\langle j_0,j_1\rangle} &= (s/f)\,\frac{1}{j_1 k} + \left[x_{\langle j_0\rangle} + \frac{k(k-j_0)-(k-j_1)}{k(k-j_0)}W_{\langle j_0\rangle}\right]\frac{j_0}{j_1 k} \\ &+ \left[x_{\langle k-1-j_0\rangle} + \frac{kj_0+j_1}{k(j_0+1)}W_{\langle k-1-j_0\rangle}\right]\frac{k-1-j_0}{j_1 k} + \left[x_{\langle j_0,j_1-1\rangle} + \frac{1}{k-j_1+1}W_{\langle j_0,j_1-1\rangle}\right]\frac{j_1-1}{j_1} \quad (13)\end{aligned}$$

$$\begin{aligned} x_{\langle inj,j_0,j_1\rangle} &= \left[x_{\langle j_0\rangle} + \frac{k(k-j_0)-1}{k(k-j_0)}W_{\langle j_0\rangle}\right]\frac{j_0}{k^2-1} + \left[x_{\langle k-1-j_0\rangle} + \frac{k(1+j_0)-1}{k(1+j_0)}W_{\langle k-1-j_0\rangle}\right]\frac{k-1-j_0}{k^2-1} \\ &+ \left[x_{\langle j_0,j_1\rangle} + \frac{k-j_1-1}{k-j_1}W_{\langle j_0,j_1\rangle}\right]\frac{j_1 k}{k^2-1} + \left[x_{\langle j_0,k-1-j_1\rangle} + \frac{j_1}{1+j_1}W_{\langle j_0,k-1-j_1\rangle}\right]\frac{(k-1-j_1)k}{k^2-1} \quad (14)\end{aligned}$$

Figure 4: Service times for channels in the mesh. Here s and f are the message and flit size, respectively.

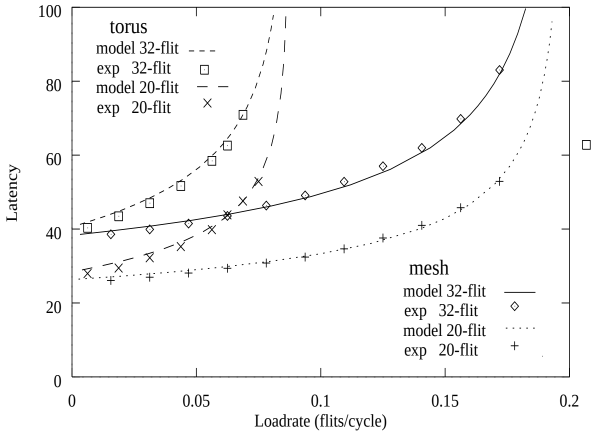

Figure 5: Comparison of model and simulation results for the mesh and torus with 64 nodes using message lengths of 20 and 32 flits.

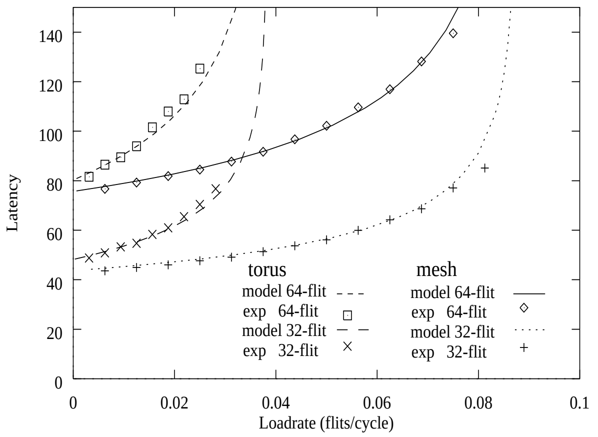

Figure 6: Comparison of model and simulation results for the mesh and torus with 256 nodes using message lengths of 32 and 64 flits.

a comparison of latencies between mesh and torus networks of 64 and 256 nodes. We see that the torus network tends to saturate at an earlier load rate (about one half that of the mesh). Also, at low load rates, the latency for the torus is slightly higher than that of the mesh under the same load condition and with the same message lengths.

In addition to showing better performance than the torus at any load rate, the mesh is easier to lay out. The networks have the same bisection width, and although the torus can be laid out with area and maximum wire length only a constant factor larger than for the mesh [4, Fig. 5], it cannot quite achieve the *same* area and wire length bounds as the mesh.

One may further note that under very low load rate, a very simple model for latency may be applied, namely number of flits plus average message distance. The key difference between the mesh and torus then is that the average message distance (excluding injection and ejection channels) is $2k/3$ for the mesh versus $k^2/(k+1)$ for the torus. The average distances may be computed by doubling the average distance in the x dimension, D_x, which may be found by considering a single row of the mesh or torus and utilizing the probability $k/(k^2-1)$ that a message from a node in a specified x-position goes to a node in some specific other x-position. We find

$$D_x = \frac{1}{k}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}|i-j|\frac{k}{k^2-1}$$

for the mesh, and

$$D_x = \frac{1}{k}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1} j\,\frac{k}{k^2-1}$$

for the torus.

arrival rate to a channel provided that the channel is stable (channel utilization factor < 1). The message rates for each channel is determined according to the number of source nodes that use the channel and the number of destination nodes that can be reached from the channel.

Extending the channel notation in a natural way to injection and ejection channels, we have:

$$\lambda_{\langle inj, j_0, j_1\rangle} = \lambda_{\langle j_0, j_1, ej\rangle} = \lambda_{node} \qquad (0 \leq j_0, j_1 < k) \ . \tag{8}$$

For the $k-1$ channel equivalence classes in the x-dimension, the message rates are

$$\lambda_{\langle j_0\rangle} = \frac{j_0(k-j_0)k}{k^2-1}\lambda_{node} \qquad (1 \leq j_0 < k) \ . \tag{9}$$

There are $k(k-1)$ channel equivalence classes in the y-dimension; the message rates are

$$\lambda_{\langle j_0, j_1\rangle} = \frac{j_1(k-j_1)k}{k^2-1}\lambda_{node} \quad \left\{ \begin{array}{l} 0 \leq j_0 < k \\ 1 \leq j_1 < k \end{array} \right. \ . \tag{10}$$

(Note that although the message rates for the channels in the y-dimension are independent of j_0, the service times are not, which is why there are different equivalence classes for different values of j_0.

3.2 Service Times

Service times are resolved in the order of ejection channels, x-dimension channels, y-dimension channels and injection channels. At the ejection channel, once the head flit of a message is received, the rest of the message will be received one flit at a time without any further blocking. Therefore the service time at the ejection channels is equal to the length (in number of flits) of the message as specified by Equation 11 in Figure 4.

The service times for x-dimension channels are determined as follows: Upon leaving channel $\langle j_0\rangle$, a message exits to the ejection channel available at that point, with probability $R_{\langle j_0\rangle|ej} = \frac{1}{j_0}$, or continues to the next channel in the same direction, with probability $R_{\langle j_0\rangle|\langle j_0-1\rangle} = \frac{j_0-1}{j_0}$. The service time $x_{\langle j_0\rangle}$, obtained using Equations 7 through 9 is given by Equation 12 in Figure 4.

The service times for the y dimension channels are determined as follows: Upon leaving channel $\langle j_0, j_1\rangle$, a message may exit to the ejection channel available at that point, with probability $R_{\langle j_0,j_1\rangle|ej} = \frac{1}{j_1 k}$, continue to the next channel in the same direction, with probability $R_{\langle j_0,j_1\rangle|\langle j_0,j_1-1\rangle} = \frac{j_1-1}{j_1}$, switch to a west-bound channel, with probability $R_{\langle j_0,j_1\rangle|\langle j_0\rangle} = \frac{j_0}{j_1 k}$, or switch to an east-bound channel, with probability $R_{\langle j_0,j_1\rangle|\langle j_0,E\rangle} = \frac{k-1-j_0}{j_1 k}$. The service time $x_{\langle j_0,j_1\rangle}$, obtained using Equations 7 through 10 is given by Equation 13 in Figure 4

The service times at the injection channels are determined as follows: Upon leaving an injection channel at node $\langle j_0, j_1\rangle$, a message may go north-bound, with probability $R_{inj|\langle j_0,j_1,N\rangle} = \frac{(k-1-j_1)k}{k^2-1}$, south-bound, with probability $R_{inj|\langle j_0,j_1\rangle} = \frac{j_1 k}{k^2-1}$, east-bound, with probability $R_{inj|\langle j_0,E\rangle} = \frac{k-1-j_0}{k^2-1}$, or west-bound with probability $R_{inj|\langle j_0\rangle} = \frac{j_0}{k^2-1}$ respectively. Therefore, the service time for an injection channel, obtained using Equations 7, 9, and 10 is given by Equation 14 in Figure 4.

The waiting times $W_{\langle inj,j_0,j_1\rangle}$ are computed using $x_{\langle inj,j_0,j_1\rangle}$ and Equation 4. The average latency for the entire network is then

$$\overline{L} = \frac{1}{k^2}\sum_{j_0=0}^{k-1}\sum_{j_1=0}^{k-1}(W_{\langle inj,j_0,j_1\rangle} + x_{\langle inj,j_0,j_1\rangle}) + \overline{D} - 1 \ . \tag{15}$$

For uniform random messages, $\overline{D} = \frac{2k}{3} + 2$.

4 Comparison of Mesh and Torus Models to Simulations and to Each Other

In this section we show the close correspondence of the analysis of Section 3 with simulations of the mesh, and we compare the mesh to the torus.

The lower right portions of Figures 5 and 6 show the comparison of latency results from the model and simulation for a 2D-mesh with 64 nodes and 256 nodes. The model agrees well with simulation results for all cases under study. The experimental values were obtained from simulation over 100,000 clock steps at each load rate, with data for the first 10,000 clocks discarded to get rid of start-up effects. At each load rate, the mean and standard deviation of the latency were computed; the network is stable at low load rates, but as the load rate approaches saturation, the latency deviations grow large. At each load rate, we computed a percentage error as the ratio of the standard deviation to the mean; we show empirical data points only for load rates at which the error is less than 5%.

For modeling of the torus network we use the results of Draper and Ghosh [6] with only slight modification as noted in footnote 1; see Appendix A. (One important difference that may be noted between the mesh and torus networks is that e-cube routing does not suffice to prevent deadlock in the torus, but adding in the use of virtual channels [5] does.)

Figures 5 and 6 show both the close correspondence of the torus model with experimental results and

But the M/G/1 model assumes independent arrivals at the inputs of a switch, all of which may block one another, which is not accurate for wormhole routing. In actuality, once an input link is occupied by a worm, there can be no more arrivals on that link until the first worm is fully serviced. Thus, once a worm arrives on a link, it only needs to wait for worms from other incoming links. Therefore, to use the M/G/1 waiting time result, we multiply by a blocking probability $P_{i|j}$:

$$w_j = P_{i|j} W_j \ , \tag{5}$$

where $P_{i|j}$ may be expressed as

$$P_{i|j} = 1 - \frac{\lambda_i^{\text{in}}}{\lambda_j} R_{i|j} \ . \tag{6}$$

following Draper and Ghosh [6]. We may view $P_{i|j}$ as the probability that a message deemed by the M/G/1 model to block an incoming message from input channel i is actually a possible blocking message, i.e., one that comes from some other input channel.

By combining Equations 3, 5 and 6 we obtain the service time for messages incident to an incoming channel i:

$$x_i^{\text{in}} = \sum_j \left[x_j + (1 - \frac{\lambda_i^{\text{in}}}{\lambda_j} R_{i|j}) W_j \right] R_{i|j} \ . \tag{7}$$

Note here that $\overline{W}_j$ is the mean waiting time obtained from Equation 4, λ_i^{in} is the total message rate on incoming channel i and λ_j is the total message rate on outgoing channel j.

We now apply the framework of this section to the 2D-mesh in the next section.[1]

3 Analysis of 2D-Meshes

We now apply the model of Section 2 to the 2D-mesh network. In this network, nodes are connected by bidirectional links, without wrap-arounds at the edges of the mesh. Let the number of nodes be k^2, i.e., k is the side length in each dimension. Each node in the network represents a switch and a processor as in Figure 2.

An important issue for wormhole routing is deadlock avoidance. In a mesh network (and in k-ary n-cubes), deadlock may be prevented by using e-cube routing, i.e., messages are routed in one dimension and then the other (in the 2D case). In the analysis presented here, we assume routing in the y dimension first.

We label each node in the network using a pair of indices $\langle j_0, j_1 \rangle$, with j_0 and j_1 ($0 \le j_0, j_1 < k$) denoting the position of the node in the x and y-dimension, respectively. We can label each channel in the network using a node label together with a direction (N,S,E,W); for example, $\langle j_0, j_1, N \rangle$ denotes the channel going north from node $\langle j_0, j_1 \rangle$. Since each link is bidirectional, there are $2k(k-1)$ channels in each dimension.

We adopt the concept of channel equivalence class, defined by [6] as a set of channels that have identical statistical properties regarding message rate and service time. Because of the symmetry in a 2D-mesh network, an east-bound channel $\langle j_0, j_1, E \rangle$ has identical traffic behaviors as its corresponding west-bound channel $\langle k-1-j_0, j_1, W \rangle$, i.e., they belong to the same channel equivalence class. Similarly north-bound channel $\langle j_0, j_1, N \rangle$ and its corresponding south-bound channel $\langle j_0, k-1-j_1, S \rangle$ belong to the same class. Thus, we can reduce the number of equivalence classes to $k(k-1)$ in each dimension. We only have to determine the behavior of west and south-bound channels and then map the results to east and north-bound channels.

Furthermore, channels in the x-dimension with the same x-positions (i.e., $\langle j_0, j_1, E \rangle$, $j_1 = 0, 1, ..., k-1$) are equivalent, therefore the number of equivalence classes is reduced to $k-1$ for x-dimension channels and we can drop the y-index from the channel labels, i.e., we will use $\langle j_0, W \rangle$ to denote a west-bound channel from node $\langle j_0, j_1 \rangle$ with an arbitrary j_1 ($0 \le j_1 < k$). Since our analysis is focused on west and south-bound channels, for convenience we will drop the direction labels for them, i.e. we will use $\langle j_0 \rangle$ ($0 < j_0 < k$) to denote the $k-1$ channel equivalence classes in the x-dimension and $\langle j_0, j_1 \rangle$ ($0 \le j_0 < k, 0 < j_1 < k$) to denote the $k(k-1)$ channel equivalence classes in the y-dimension. We specify the direction labels explicitly when we refer to east and north-bound channels.

In this section, we first discuss message rates to each channel equivalence class. We then determine the service times for the $k-1$ channel equivalence classes in the x dimension, followed by the service times for the $k(k-1)$ channel equivalence classes in the y-dimension. Service times and waiting times at injection channels are then determined and average latency is determined using Equation 2.

3.1 Message Rates

Assume messages arrives to each injection channel according to a Poisson process with rate λ_{node}. Also assume the mean departure rate is equal to the mean

[1] The specialization of our approach to k-ary n-cubes, and the 2D-torus in particular, differs from Draper and Ghosh only in that they essentially ignore injection and ejection channels. We treat injection and ejection channels like any other channel, except that we retain the simplification of omitting waiting time for ejection channels, which is relatively insignificant.

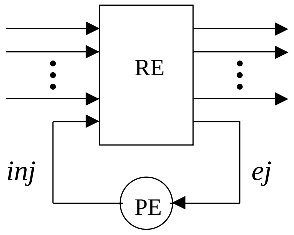

Figure 2: A node in the network has a routing element (RE) and a processing element (PE), which is attached to the RE through an injection channel and an ejection channel.

prised of a processing element (PE) connected by an injection channel and an ejection channel to a routing element (RE) that also has incoming and outgoing internode channels. A typical message path includes an injection channel, several inter-node channels and an ejection channel.

The latency for a message injected from a node in the network is determined by the waiting time and service time of the injection channel, with the service time depending on waiting and service times of successive channels in the path. When a message is generated at a PE attached to switch j, it will wait in an injection queue for a time denoted by $W_{inj,j}$. Once the head flit of the message is accepted by the network, the message will hold the injection channel for a service time denoted by $x_{inj,j}$. At the end of service time, i.e. when the tail of the message has left the injection channel, it will take another $D-1$ steps for the entire message to be received at the destination, where D is the number of channels in the path. This is because under the long worm assumption, when the tail of the message has left the injection channel, the head flit must have arrived at the destination where no further blocking is encountered. The latency L_j for the message injected at node j is then

$$L_j = W_{inj,j} + x_{inj,j} + D - 1 \ . \qquad (1)$$

Averaging over all nodes (and the probability distribution of message generation), the average latency $\overline{L}$ for the entire network is then

$$\begin{aligned} \overline{L} &= \frac{1}{N}\sum_j \overline{L}_j \\ &= \frac{1}{N}\sum_j \left(\overline{W}_{inj,j} + \overline{x}_{inj,j}\right) + \overline{D} - 1 \ , \end{aligned} \qquad (2)$$

where N is the number of nodes in the network and $\overline{D}$ is the average message distance.

The mean waiting time $\overline{W}_{inj,j}$ in the above equation can be approximated using a traditional M/G/1 queueing model once the service time $x_{inj,j}$ is known, since the queueing behavior at the injection queue is identical to that in store-and-forward routing. The source service time $x_{inj,j}$ however, is quite different from store-and-forward routing where service time is simply the length of the message. In wormhole routing, since a message is spread over many channels at a time step, the service time for a channel depends on the service times and the waiting times at the subsequent channels on the path to the destination. The service times for all the channels on a message's path need to be resolved in reverse order of the traversal order, i.e., from the ejection channel back towards the injection channel. For general k-ary n-cubes where e-cube routing is used to avoid deadlock, this implies the service times are resolved in the order of low dimensions followed by high dimensions.

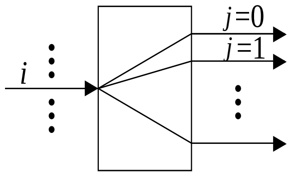

Figure 3: Possible subsequent channels for incoming channel i

The service time at an arbitrary channel on a message's path can be analyzed with the aid of Figure 3. Messages from an incoming channel i with arrival rate λ_i^{in} may be routed to outgoing channels denoted by j=0,1, etc. The service time for the incoming channel depends on the service times and waiting times at all possible outgoing channels. Denote the probability that a message from incoming channel i is routed to outgoing channel j by $R_{i|j}$, the service time for incoming channel i can then be expressed as

$$x_i^{\text{in}} = \sum_j (x_j + w_j) \cdot R_{i|j} \ , \qquad (3)$$

where x_j is the service time for the outgoing channel j and w_j is the waiting time for outgoing channel j. The above equation states that the service time at a channel depends on the service time of the subsequent channel and the waiting time for the subsequent channel.

The mean waiting times w_j may be approximated using the M/G/1 model, and the M/G/1 mean waiting time W_j may itself be approximated by incorporating a suggestion of Draper and Ghosh [6, pp. 206–207], yielding

$$\overline{W}_j = \frac{\lambda_j \overline{x}_j^2}{2(1-\lambda_j \overline{x}_j)} \cdot \left[1 + \frac{(\overline{x}_j - s/f)^2}{\overline{x}_j^2}\right] \ . \qquad (4)$$

where λ_j is the message rate on outgoing channel j, s denotes the size of the worm, and f denotes the flit size (so that s/f is the length of the worm in flits).

MODELING AND COMPARISON OF WORMHOLE ROUTED MESH AND TORUS NETWORKS

RONALD I. GREENBERG
Mathematical and Computer Sciences
Loyola University
6525 North Sheridan Road
Chicago, IL 60626
rig@math.luc.edu

LEE GUAN
Electrical Engineering
University of Maryland
College Park, MD 20742
leeguan@eng.umd.edu

Abstract

2D-mesh and torus networks have often been proposed as the interconnection pattern for parallel computers. In addition, wormhole routing has increasingly been advocated as a method of reducing latency. Most analysis of wormhole routed networks, however, has focused on the torus and the broader class of k-ary n-cubes to which it belongs. This paper presents a performance model for the wormhole routed mesh, and it compares the performance of the mesh and torus based on theoretical and empirical analyses.

keywords: interconnection network, wormhole routing, latency, throughput, 2D-mesh, torus

1 Introduction

2D-mesh and torus networks have often been proposed as the interconnection pattern for parallel computers. For example, the Intel Paragon [2] uses a 2D-mesh arrangement of processors. A recent example of a torus-based machine, albeit in three dimensions, is the Cray T3E [3].

In addition, wormhole routing [5] has increasingly been advocated as a method of reducing message routing latency. In this model, packets are composed of *flits* or *flow control digits*, and packets snake through the network one flit after another; only a constant number of flits may be stored in an intermediate node at any time. Most analysis of wormhole routed networks, however, has focused on the class of k-ary n-cubes networks (which includes the two-dimensional torus), e.g., [1, 4, 6, 7, 8].

Figure 1 illustrates the mesh and torus networks in a 4×4 size. The mesh links are bidirectional. Unidirectional links in the torus suffice for complete reachability and lead to a fairer comparison with the mesh than would bidirectional links, by giving the two networks the same bisection width.

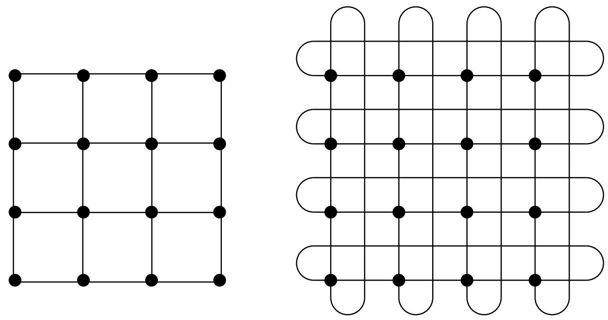

Figure 1: 4×4 mesh and torus networks.

This paper presents a performance model for the wormhole routed mesh in Section 3 after presenting a general framework for wormhole routing in Section 2. In Section 4, comparisons between the model and simulations are provided. In addition, the analytical and empirical results for the mesh are compared to corresponding results for the torus. Finally, Section 5 contains concluding remarks.

2 Wormhole Routing Performance Model

This section presents a general approach to analyze the performance of wormhole routed interconnection networks. We use average latency as the principal measure of network performance. The approach generalizes the analysis of Draper and Ghosh applied to k-ary n-cubes [6].

The model presented in this section is based on the following assumptions: (1) Arrivals at each source node are governed by a Poisson process, and destinations are uniformly random. (2) Worms have fixed length and are longer than the diameter of the network. (3) Contentions at incoming links to a node are resolved according to First-Come First-Served (FCFS) scheduling. (4) Messages arriving at destinations are immediately consumed at the rate of one flit per step, i.e., no blocking is encountered at destinations.

Figure 2 shows the switch model used in the analysis presented in this paper. A network node is com-