Analytical Modeling of Routing Algorithms in Virtual Cut-Through Networks

Jennifer Rexford
Network Mathematics Research
Networking & Distributed Systems
AT&T Labs — Research
Florham Park, NJ 07932
jrex@research.att.com

Kang G. Shin
Real-Time Computing Laboratory
Electrical Engineering & Computer Science
University of Michigan
Ann Arbor, Michigan 48109
kgshin@eecs.umich.edu

Abstract

Contemporary multicomputer networks reduce communication latency by allowing an in-transit packet to "cut through" intermediate nodes to idle outgoing links, instead of buffering at each hop in its route. Under these cut-through switching schemes, routing algorithms influence performance by determining whether an incoming packet can locate an idle output link. This paper presents analytical models for a set of oblivious and adaptive routing algorithms, with different *selection functions* for ranking outgoing links, in torus networks with virtual cut-through switching. The analytical expressions can efficiently predict the behavior of large networks and help weigh the cost-performance trade-offs of different routing strategies. However, analytical performance models typically require simplifying assumptions about the underlying interconnection network and application traffic patterns, for the sake of tractability. We characterize these assumptions and their effects by evaluating the impact of packet history on communication performance, under different routing algorithms, network topologies, and application workloads. Simulation results show that cut-through switching introduces unique inter-node dependencies that affect the accuracy of analytical models, as well as actual performance under realistic network configurations. Based on these results, we propose new routing algorithms and task-allocation schemes that can improve network performance by capitalizing on the natural dependencies between adjacent nodes.

Keywords: Multicomputer, cut-through switching, routing, interconnection network, independence assumption

1 Introduction

Modern multicomputers can exploit the parallelism in a wide variety of applications by distributing computational tasks across a collection of processing nodes. Since these nodes cooperate by exchanging messages, application performance is extremely sensitive to the design of the underlying communication network [1–3]. Network policies are implemented in the router that connects an individual node to the interconnection fabric and manages traffic flowing through the node en route to other destination nodes; for example, Figure 1 shows a router in a 4×4 torus network. Network performance and implementation complexity hinge on the selection of an appropriate routing algorithm and switching scheme. This paper

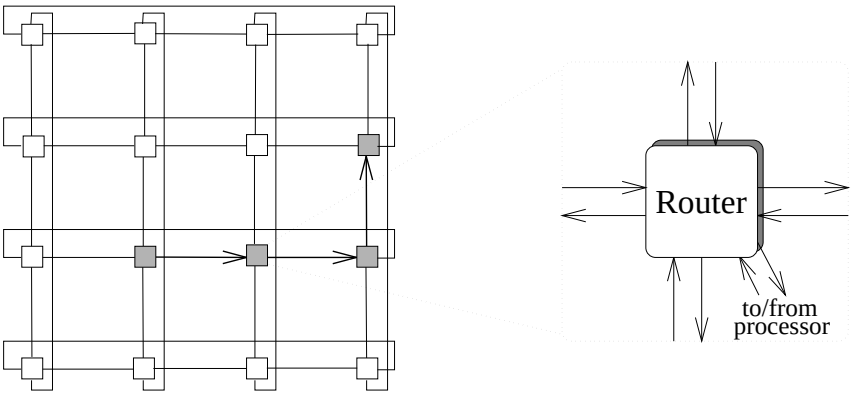


Figure 1: **Router in a Torus Network:** This figure shows a router in a 4×4 torus (wrapped square mesh) of processing nodes. To communicate with another node, a processor injects a packet into its router; then, the packet traverses one or more links before reaching the reception port of the router at the destination node.

examines the subtle, yet important, interplay between routing and switching, through a combination of analytical models and simulation experiments.

The switching scheme influences communication performance by determining which, and how much, link and buffer resources a packet consumes at each hop in its route. In traditional *packet switching*, an arriving packet must buffer completely before transmission to a subsequent node can begin. Hence, even in a lightly-loaded network, packet switching incurs delay proportional to the product of packet size and the length of the route. In contrast, *cut-through* switching schemes, such as wormhole [4] and virtual cut-through [5] switching, try to forward the incoming packet directly to an idle output link. If the packet encounters a busy outgoing link, virtual cut-through switching buffers the packet in the router, whereas wormhole switching stalls the packet in the network until its link becomes available. Consequently, wormhole networks typically require much less buffer space for storing blocked traffic, at the expense of fine-grain flow control and lower peak throughput [6–9]. When the network devotes more memory to each router, wormhole switching more closely resembles virtual cut-through switching in terms of both cost and performance, resulting in a spectrum of possible router designs.

In a lightly-loaded network, communication latency under cut-through switching is proportional to the sum of packet size and the length of the route, since a packet only incurs a small delay for processing the routing header at each node. As a result, most contemporary research and commercial multicomputer routers employ some form of cut-through switching. In addition, recent research considers the use of cut-through switching to improve communication performance in local-area networks and large, high-speed switches [9, 10]. Although wormhole switching is quite common, particularly in large-scale parallel machines [2, 3], several recent designs implement virtual cut-through switching [9, 11–16], or even both schemes [17, 18]. With the increasing prevalence of cut-through routers, there has been a re-

Selection Function	Ranking of x and y links
Dimension-ordered	Prefer x link over y link
Random	Select x and y links with equal probability
Diagonal	Prefer direction with most hops remaining

Table 1: **Selection Functions:** This table shows three selection functions for shortest-path routing in two-dimensional mesh and torus networks. Based on the header of the incoming packet, the router can determine which direction(s) lie along a minimal path to the destination node.

newed interest in analytical performance models for virtual cut-through networks [19–26]. However, for the most part, existing analytical work has not focused on the influence of different routing algorithms on the performance of virtual cut-through switching.

The routing algorithm determines which outgoing links a packet can consider at each node in its route. While *oblivious* routing generates a single outgoing link for an incoming packet, *adaptive* schemes can incorporate prevailing network conditions into the routing decision. By considering multiple outgoing links, adaptive algorithms can balance network load and increase a packet's chance of cutting through intermediate nodes, at the expense of out-of-order packet arrivals at the destination node [27] and an increase in router implementation complexity [28]. The cut-through probability also depends on the *selection function* [29] which determines which order the router considers the candidate outgoing links. This paper presents analytical models that compare the cut-through performance of a collection of oblivious and adaptive routing algorithms, with different selection functions, as shown in Table 1. The traditional dimension-ordered selection function favors the x-direction over the y-direction, as in the example in Figure 1, while the random selection function does not have preference for either direction. The diagonal selection function [30] tries to route packets in the direction that has the most hops remaining, in the hope of increasing the chance of having multiple routing choices at downstream nodes.

Cut-through switching schemes, coupled with effective routing algorithms, significantly reduce end-to-end packet delay. As networks grow larger, scalable performance requires packets to cut through as many nodes as possible. Fortunately, larger networks also provide more opportunities for adaptive routing algorithms, since packets travel longer distances with multiple possible paths to their destinations. Effective analytical models are necessary for predicting the behavior of large networks to help weigh the cost-performance trade-offs of various oblivious and adaptive routing algorithms. However, for tractability analytical models require certain simplifying assumptions about the underlying interconnection network and application traffic patterns. These assumptions can degrade the accuracy of the performance evaluation. In this paper, we focus specifically on the impact of the *independence assumption* [31] that permits analytical models to study each link in isolation.

The independence assumption has been investigated in the context of packet switching [31–34], but inter-node dependencies play a potentially larger role in cut-through networks, since performance is extremely sensitive to the likelihood that an incoming packet can locate an idle outgoing link. To study the independence assumption, and the phenomena it masks, we characterize the impact of packet history on cut-through performance through simulation experiments. Although these simulation results verify the basic trends in the analytical model, inter-node dependencies consistently cause the analytical model to overestimate or underestimate actual cut-through performance, depending on the network load. Further experiments demonstrate that the routing algorithms, network topologies, and application workloads common in modern multicomputers exacerbate these effects by limiting the mixing of traffic from different incoming links.

The following section provides an overview of the analytical model for evaluating a class of routing algorithms in $k \times k$ torus networks, under virtual cut-through switching and a uniform traffic load. Based on this model, Section 3 derives analytic expressions for the cut-through probability as a function of load and the distance a packet travels in the network. Section 4 verifies the analytical performance trends through simulation experiments on an idealized model of the network; evaluation of non-uniform traffic further highlights the benefits of adaptive routing algorithms that strive to increase the routing opportunities at subsequent nodes in a packet's journey. Based on discrepancies between the analytical and simulation results, Section 5 characterizes the effects of inter-node dependencies, as a function of the routing algorithm, network topology, and traffic pattern; by affecting the cut-through probability, these inter-node dependencies have a significant influence on communication performance in cut-through networks. After a discussion of related work in Section 6, we summarize the paper and present directions for future research in Section 7.

2 Router Model

To provide a framework for an end-to-end performance evaluation of routing algorithms and selection functions, this section first reviews a basic queuing model of virtual cut-through switching based on the work in [5]. Then, we extend this model to adaptive routing algorithms by expressing cut-through probability p_c in terms of the number of candidate outgoing links at each hop in a packet's route. By averaging across all pairs of nodes that are h hops apart, we define a recurrence that forms the basis of the combinatorial analysis of selection functions in Section 3. The resulting expressions for cut-through probability can be applied to a variety of analytic models of packet delay and queue lengths, including the queuing model in [5].

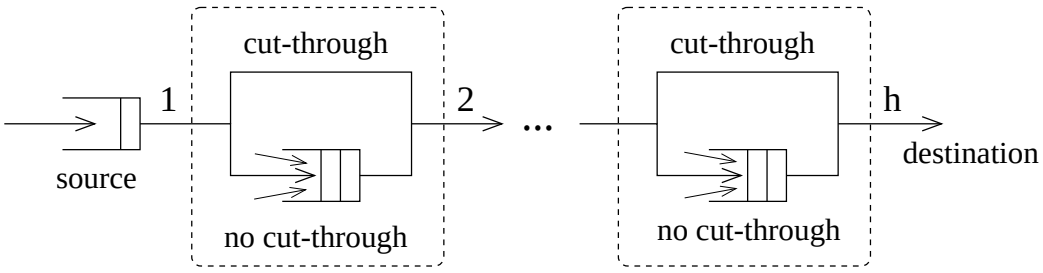


Figure 2: **Path in a Cut-Through Network:** This figure shows a conceptual model of an h-hop path in a cut-through network. Each outgoing link has a queue for blocked traffic from the multiple incoming links. At each hop in its route, a packet either cuts through to an idle output link or joins a service queue, depending on the cut-through probability p_c.

2.1 Network of Queues

A basic model of cut-through performance decomposes the interconnection network into a collection of independent links and queues, as shown in Figure 2. An arriving packet can cut through a node if its outgoing link is idle; otherwise, the packet buffers in a service queue. The various oblivious and adaptive routing algorithms differ in how they affect the cut-through probability p_c, which depends on the number of candidate output links a packet can consider at each node in its route. As in packet-switched networks, an M/M/1 queuing model can represent the service queue at each link, if queue size is not restricted, and packet lengths and interarrival times are exponentially distributed [35]. In this framework, each source node generates packets as a Poisson process with rate λ, which results in a stream of packets with exponentially distributed interarrival times with mean $1/\lambda$. Similarly, packet lengths are exponentially distributed with mean $\overline{\ell}$, as shown in Table 2.

With sufficient mixing of traffic from different sources, Kleinrock's independence assumption enables the analysis to decouple the packet length and interarrival distributions at each node [31]. This permits the analysis to model the network links as independent exponential servers operating at rate $1/\overline{\ell}$. In effect, then, a packet receives a new length at each node in its route. Under the independence assumption, a mixture of sources with Poisson arrivals produces a Poissonian output stream for the link [36]; this link, in turn, forms the traffic source for an input link to an adjacent node. With sufficient randomization of packet routes, the Poissonian traffic permits a product-form solution that separately analyzes each link on a packet's route, by Jackson's theorem [37]. In Section 4, we isolate the effects of the independence assumption in cut-through networks through a detailed comparison between the analytical model and the simulation results.

Since the network links are independent, a packet traveling h hops has $h-1$ independent opportunities to cut through intermediate nodes. Under oblivious routing, this results in a binomial distribution

Parameter	Setting
Switching scheme	Virtual cut-through switching
Buffer architecture	Infinite packet FIFO for each output link
Routing algorithm	Random, oblivious routing
Network topology	Homogeneous network ($k \times k$ torus)
Packet arrival	Poissonian packet generation (rate λ)
Packet destination	Uniform random traffic load
Packet length	Exponentially distributed (mean $\overline{\ell}$)

Table 2: **Analytical Model:** This table shows the idealized network and workload parameters for a tractable queuing model. Under these assumptions, the analysis can focus on a single link in isolation; in a torus network, a link is busy with probability $\rho = \lambda \overline{h}\,\overline{\ell}/4$, where packets travel an average of $\overline{h}$ hops to reach their destinations.

for the number of cut-throughs [5]; that is,

$$P[c \text{ cut-throughs}] = \left(\begin{array}{c} h-1 \\ c \end{array} \right) (1-\rho)^c \rho^{h-1-c} \qquad c = 0, 1, \ldots, h-1,$$

where $\rho < 1$ is the network utilization (i.e., the likelihood that a link is busy). Based on this distribution, analytical models can determine other higher-level metrics, such as average latency as well as the probability distributions for packet delay and queue lengths. These metrics can guide network designers in provisioning the link and buffer resources in emerging multicomputer routers.

For example, a packet traveling h hops has average delay [5]

$$\frac{h\overline{\ell}}{1-\rho} \; - \; p_c(h-1)\overline{\ell}.$$

The first term is the average delay in a packet-switched network, where the packet encounters h independent M/M/1 queues, each with mean service rate $1/\overline{\ell}$; in an M/M/1 queue, customers spend average time $\overline{\ell}/(1-\rho)$ in the system [35]. The second term captures the performance benefit of employing cut-through switching. When an arriving packet routes directly to an idle link, the router avoids buffering the packet, so the average packet reduces its latency by $\overline{\ell}$ whenever it cuts through one of the $h-1$ intermediate nodes in its route. To include header processing delay at each router, the $p_c(h-1)\overline{\ell}$ term reduces to $p_c(h-1)(\overline{\ell}-\tau)$, for a header of length τ. Although more complex selection functions may have slightly larger values of τ, the header processing time is typically small in comparison to $\overline{\ell}$ [5]. For simplicity, we ignore the constant τ in the remainder of the analysis.

2.2 Adaptive Routing and Selection Functions

Routing algorithms impact communication performance by affecting the cut-through probability p_c. Oblivious routing directs an incoming packet to a single outgoing link, selected randomly or statically by the selection function; hence, under the assumptions in Table 2, an arriving packet encounters an empty service queue with probability $p_c = 1 - \rho$. Adaptive routing algorithms are more difficult to analyze, since the cut-through probability depends on the *number* of links a packet can consider at each hop in its route. For example, cut-through probability in a torus network increases to $1 - \rho^2$ if *two* different links lie along minimal paths to the destination node, since the arriving packet can establish a cut-through unless *both* outgoing links are busy. If neither link is available, we assume that the packet enters a single service queue to await transmission, based on the first routing option. By focusing on a single service queue, our analysis is conservative in quantifying the potential performance improvements from adaptive routing.

For example, a blocked packet could conceivably join the shortest output queue, or even both queues along shortest-path routes; however, these alternatives improve performance at the expense of a significant increase in implementation complexity. Rather than focusing on how multiple routing choices could reduce the portion of delay that stems from queuing in the router, we characterize how adaptive routing and the choice of selection function affect the cut-through probability p_c. Similarly, we assume that the router has infinite packet buffers and, hence, do not model any particular deadlock-avoidance technique; the queuing model in Section 2.1 can be used to estimate the amount of memory necessary for a low probability of exhausting a finite packet buffer. This simple queuing model provides a basis for illustrating how the combination of adaptive routing and selection functions influence the cut-through probability and, in turn, the packet delay and buffer requirements. Finally, although we do not attempt to quantify the differences in the router clock frequency that may arise in implementing the more complex selection functions, the resulting analysis can be used to determine if the performance benefits of these selection functions outweigh the potential decrease in router speed.

In a two-dimensional torus network, the average cut-through probability depends on the likelihood P_2 that a packet has two routing options at an intermediate node. That is,

$$p_c = (1 - \rho^2)P_2 + (1 - \rho)(1 - P_2) = (1 - \rho)(1 + \rho P_2),$$

with link utilization $\rho = \lambda \overline{h}\,\overline{\ell}/4$, where packets travel an average of $\overline{h}$ hops in the network; the $\overline{h}\,\overline{\ell}$ term represents the average bandwidth consumed by a packet, while the denominator corresponds to the four links emanating from each node. Under adaptive routing, the probability P_2 depends on the selection function, as well as the relative position of the source and destination nodes. Although an empirical approach can tabulate P_2 for a specific topology and routing algorithm [26], analytical expressions

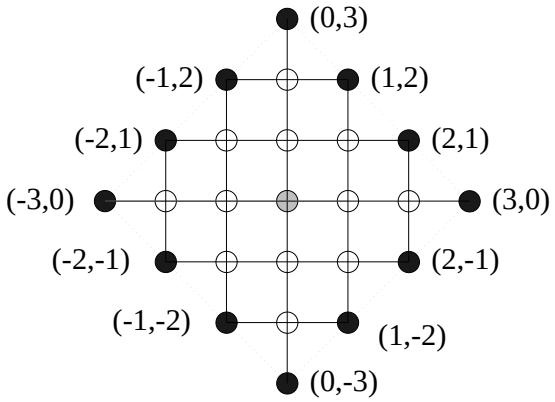


Figure 3: **Hop-Uniform Traffic:** This figure shows a source node $(0,0)$ that communicates with a ring of destination nodes that are $h = 3$ hops away. Under a homogeneous topology, such as a torus, hop-uniform traffic results in a uniform load on the network links.

facilitate more efficient prediction of communication performance, particularly for large networks. We derived closed-form analytical expressions for the random and dimension-ordered selection functions in Table 1; for the more complex diagonal selection function, we present an efficient recurrence for computing P_2.

2.3 Hop-Uniform Traffic Pattern

For a general model of homogeneous network traffic, we assume that a source node is equally likely to communicate with any destinations that are h hops away; this *hop-uniform* traffic pattern generates rings of destination nodes within a fixed distance of the source node, which can represent spheres of communication locality, as shown in Figure 3. For example, the node-uniform traffic pattern, which selects each node with equal probability, is a composition of several hop-uniform distributions. The hop-uniform traffic, coupled with the homogeneous network topology, permits the analysis to focus on a single source node $(0,0)$ and a "quadrant" of destination nodes $(0,h), (1,h-1), \ldots, (h-1,1)$, without loss of generality. Each of these destinations corresponds to a unique collection of possible shortest-path routes; the likelihood of selecting each path depends on the selection function and the average link utilization.

To determine P_2, consider a packet that travels from node (x,y) to node $(0,0)$, where $x, y \geq 0$ and $x + y = h$. Any nodes (i,j) with $0 \leq i \leq x$ and $0 \leq j \leq y$ may lie on a shortest-path route. Nodes (i,j) with $i > 0$ and $j > 0$ have *two* outgoing links on minimal paths to the destination node, as shown in Figure 4. These *internal* nodes have greater routing flexibility than the remaining *border* nodes [30]. The next section analyzes various adaptive routing algorithms by deriving

$$S^n_{(x,y)} = P[\text{packet visits } n \text{ internal nodes}], \quad n = 0, 1, \ldots, h-1$$

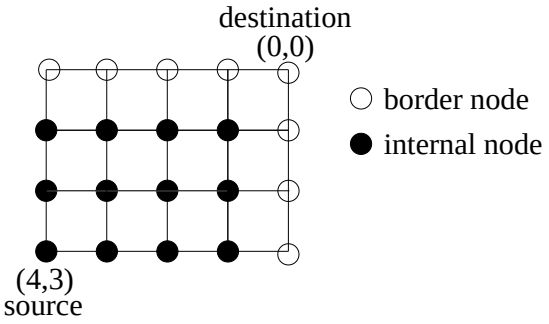


Figure 4: **Internal and Border Nodes:** This figure shows all of the nodes that lie along minimal paths from node $(4, 3)$ to node $(0, 0)$. At an *internal* node, an incoming packet can select from two candidate outgoing links; however, once the packet has only one routing option after reaching a border node.

for a packet traveling x hops in the x-direction and y-hops in the y-direction, where $h = x + y$. After defining recurrences for $S^n_{(x,y)}$, the analysis computes P_2 by averaging across a quadrant of destinations that are h hops away from the source node. Deriving P_2 as a function of h enables us to derive expressions for p_c. These expressions can, in turn, be applied to a variety of different network models, including the basic queuing analysis in Figure 2.

3 Analysis for Cut-Through Probability

The analysis evaluates three adaptive, minimal routing algorithms in a torus network to study the influence of selection functions on communication performance, as shown in Table 1. In contrast to the random and dimension-ordered selection functions, the diagonal adaptive algorithm actively strives to improve a packet's chance of having multiple routing options at intermediate nodes. This can significantly improve the average cut-through probability p_c, particularly in large networks. To analyze cut-through performance, we derive a recurrence for $S^n_{(x,y)}$, the likelihood that a packet traveling from node (x, y) to node $(0, 0)$ encounters exactly n internal nodes, for each of the adaptive routing algorithms.

3.1 Random and Dimension-Ordered Adaptive Routing

When either $x = 0$ or $y = 0$, the packet has already reached a border node and, hence, can only consider one direction for the remainder of the route, resulting in

$$S^n_{(x,y)} = \begin{cases} 1 & n = 0 \\ 0 & n > 0 \end{cases}$$

for each of the routing algorithms. If both x and y are non-zero, the packet travels in the x-direction with probability α, where α depends on the selection function; if a packet cannot establish a cut-through

on either link, we assume that the packet joins the service queue for its first-choice direction. Under the random selection function, $\alpha = 1/2$ since neither direction has preference over the other. In contrast, the dimension-ordered selection function routes a packet in the x-direction, unless the x link is busy and the y link is idle, resulting in $\alpha = 1 - \rho(1-\rho)$.

When both x and y are non-zero, the source (x, y) is an internal node; any other internal nodes also appear in the routes from nodes $(x-1, y)$ and $(x, y-1)$. Thus, for $x, y > 0$,

$$S^n_{(x,y)} = \begin{cases} 0 & \text{if } n = 0 \\ \alpha S^{n-1}_{(x-1,y)} + (1-\alpha) S^{n-1}_{(x,y-1)} & \text{otherwise.} \end{cases}$$

As discussed in Section 2.3, the mean value of $S^n_{(x,y)}$ can be computed by averaging across a single quadrant of destination nodes $(0, h), (1, h-1), \ldots, (h-1, 1)$, where $h = x + y$. For $n > 0$, summing the values of $S^n_{(x,y)}$ across the quadrant yields

$$\begin{aligned} S^n_h = \sum_{\substack{x=0 \\ y=h-x}}^{h-1} S^n_{(x,y)} &= \alpha \sum_{x=1}^{h-1} S^{n-1}_{(x-1,y)} + (1-\alpha) \sum_{x=1}^{h-1} S^{n-1}_{(x,y-1)} \\ &= \alpha \sum_{\substack{x=0 \\ y=(h-1)-x}}^{(h-1)-1} S^{n-1}_{(x,y)} + (1-\alpha) \sum_{\substack{x=1 \\ y=(h-1)-x}}^{h-1} S^{n-1}_{(x,y)}. \end{aligned}$$

To simplify the recurrence, we first determine S^1_h. If $n = 1$, both summations evaluate to 1 because $S^0_{(0,y)} = S^0_{(x,0)} = 1$, while all other terms are zero, resulting in $S^1_h = 1$. Based on this result, we can compute $S^n_{x,y}$ for $n > 1$. In this case, $S^{n-1}_{(0,y)} = S^{n-1}_{(x,0)} = 0$, so both summations range over $x = 1, \ldots, (h-1) - 1$. Thus, for $n > 1$,

$$S^n_h = \alpha S^{n-1}_{h-1} + (1-\alpha) S^{n-1}_{h-1} = S^{n-1}_{h-1}\,.$$

Hence, $S^n_h = S^{n-1}_{h-1} = \ldots = S^0_{h-n}$. Since $S^0_h = 1$ for all $h > 0$, this implies $S^n_h = 1$, for $0 \le n < h$. Thus, for h-hop packets, routes with $0, 1, \ldots, h-1$ internal nodes are encountered with equal probability $1/h$.

Based on this expression for S^n_h, we can now determine the probability P_2 that a packet has two routing choices. However, the definition of $S^n_{(x,y)}$ allows the source (x, y) to count as one of the packet's internal nodes, even though a packet cannot actually cut through the first link in its route. To accurately compute cut-through probability, we must remove the source whenever a route has one or more internal nodes, resulting in

$$P[N = n \text{ internal } \textit{intermediate} \text{ nodes}] = \begin{cases} \frac{2}{h} & \text{if } n = 0 \\ \frac{1}{h} & \text{if } n = 1, 2, \ldots, h-2 \\ 0 & \text{if } n = h-1\,. \end{cases}$$

Note that each packet has at least one intermediate node on the *border*, since the packet has only one routing option for its final hop. The probability P_2 is the proportion of the $h-1$ intermediate nodes that are *internal* nodes, resulting in

$$P_2 = \frac{\sum_{n=0}^{h-1} n \cdot P[N=n]}{h-1} = \frac{1}{2} - \frac{1}{h} \qquad \text{for } h \geq 2.$$

Using the expression for p_c from Section 2.2, the cut-through probability simplifies to

$$p_c = (1-\rho)\left(1+\rho\left(\frac{1}{2}-\frac{1}{h}\right)\right),$$

which approaches $p_c = (1-\rho)(1+\rho/2)$ for large values of h. Considering a second outgoing link, when possible, increases the cut-through probability by nearly a factor of $\rho/2$ over the oblivious routing algorithms. It is interesting to note that P_2, and hence p_c, does not depend on α. This implies that choosing the first-choice link randomly (instead of using a static strategy, such as dimension-order routing) does not improve the likelihood of encountering internal nodes; on average, both approaches lead the packet to a border node in the same number of hops.

3.2 Diagonal Adaptive Routing

As seen in the previous subsection, capitalizing on multiple shortest paths improves the likelihood of cut-throughs. Oblivious routing ignores this opportunity, while the random and dimension-ordered adaptive schemes capitalize on it. To further improve communication performance, diagonal routing actively creates such opportunities by favoring the x-direction when $x > y$ and the y-direction when $y > x$ [11, 30]; when $x = y$, we assume that the algorithm breaks ties in favor of the x-direction[1]. To determine the recurrence for $S^n_{(x,y)}$, we define α as the likelihood that a packet travels in the preferred direction. The packet travels in the alternate direction only when the preferred link is busy *and* the alternative link is idle, so $1-\alpha = \rho(1-\rho)$; since $\rho \in [0,1]$, $\alpha \geq 3/4$, ensuring that packets travel in the preferred directions at least three-fourths of the time.

As in Section 3.1, if $x=0$ or $y=0$, $S^0_{(x,y)}=1$ and $S^{n\neq 0}_{(x,y)}=0$, since the packet has already reached a border node. When $x, y > 0$, the recurrence depends on the relative values of x and y, resulting in

$$S^n_{(x,y)} = \begin{cases} 0 & \text{if } n = 0 \\ \alpha S^{n-1}_{(x-1,y)} + (1-\alpha) S^{n-1}_{(x,y-1)} & x \geq y, n > 0 \\ \alpha S^{n-1}_{(x,y-1)} + (1-\alpha) S^{n-1}_{(x-1,y)} & y > x, n > 0\,. \end{cases}$$

[1]This assumption simplifies the presentation of the derivation; it can be shown that any tie-breaking policy results in the same analytical expression for cut-through probability.

First, we determine S_h^0 to initialize the recurrence. When $n = 0$, only $(x, y) = (0, h)$ contributes to the expression, resulting in $S_h^0 = S_{(0,h)}^0 = 1$. When $n > 0$, the route the expression for S_h^n depends on whether h is even or odd. For *even* values of h,

$$S_h^n = \sum_{\substack{x=1 \\ y=h-x}}^{h-1} S_{(x,y)}^n = \sum_{\substack{x=1 \\ y=h-x}}^{\frac{h}{2}-1} \left\{ \alpha S_{(x,y-1)}^{n-1} + (1-\alpha) S_{(x-1,y)}^{n-1} \right\} + \sum_{\substack{x=h/2 \\ y=h-x}}^{h-1} \left\{ \alpha S_{(x-1,y)}^{n-1} + (1-\alpha) S_{(x,y-1)}^{n-1} \right\} .$$

This simplifies to

$$S_h^n = \alpha (S_{h-1}^{n-1} + S_{(\frac{h}{2}-1,\frac{h}{2})}^{n-1}) + (1-\alpha)(S_{h-1}^{n-1} - S_{(\frac{h}{2}-1,\frac{h}{2})}^{n-1}) .$$

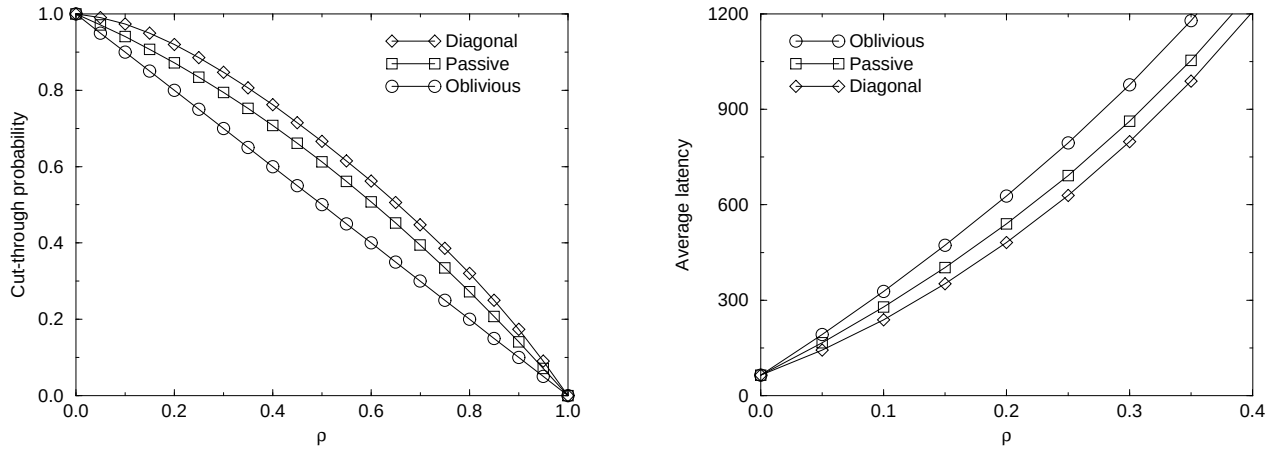
A similar expression holds for *odd* values of h, resulting in

$$S_h^n = \begin{cases} S_{h-1}^{n-1} + (2\alpha - 1) S_{(\frac{h}{2}-1,\frac{h}{2})}^{n-1} & \text{for even } h \\ S_{h-1}^{n-1} + (2\alpha - 1) S_{(\frac{h-1}{2},\frac{h-1}{2})}^{n-1} & \text{for odd } h . \end{cases}$$

Note that when $\alpha = 1/2$ the recurrence reduces to the expression for random and dimension-ordered adaptive routing in Section 3.1. Since diagonal routing has $\alpha \geq 3/4$, the algorithm increases S_h^n for larger n, since routes with a larger number of internal nodes become more likely. Although this analysis does not result in a closed-form expression for the cut-through probability, the recurrence for S_h^n provides a very efficient way to generate P_2 and, subsequently, p_c.

3.3 Performance Comparison

The analytical expressions for p_c enable performance comparisons between the oblivious and adaptive routing algorithms. Figure 5 plots the cut-through probability and the average packet latency for the three algorithms with $\overline{\ell} = 64$ and $h = 20$ under changing load ρ; in Figure 5, the "passive" curve corresponds to the random and dimension-ordered adaptive routing algorithms. At low loads, packets almost always cut through intermediate nodes, so p_c is nearly unity. In this situation, transmission delay is the main component of packet latency, so all three algorithms have an average latency of just over 64 cycles (the packet length, plus processing delay for the one-byte header at each hop in the route); to construct the latency plot, we apply the analysis for p_c to the expression for average delay for packets with $h - 1$ intermediate nodes, as described in Section 2.1. As load increases, the adaptive schemes outperform the oblivious algorithm by increasing p_c. In Figure 5(a), diagonal routing operating at 50% network load can achieve the same cut-through probability that oblivious routing does at 30% load.



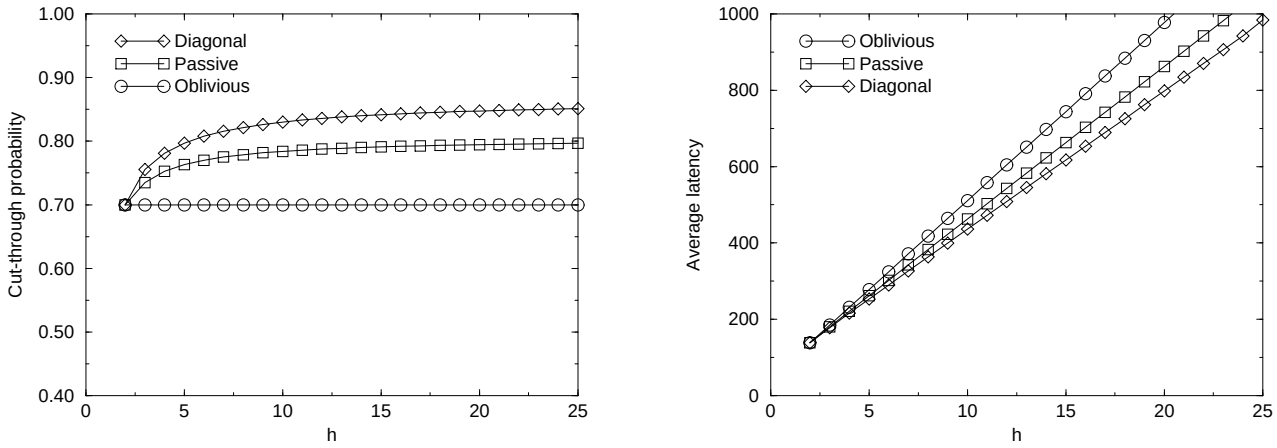
(a) Cut-through probability (b) Average latency

Figure 5: **Routing Performance as a Function of Load:** These graphs illustrate the performance of the three types routing algorithms as a function of load ρ. Packets have mean length $\overline{\ell} = 64$ bytes and travel $h = 20$ hops to reach their destinations. Passive (random and dimension-ordered) adaptive routing outperforms oblivious routing, but diagonal routing excels over the other algorithms by actively increasing the likelihood that packets can select from multiple outgoing links.

To evaluate how well the algorithms scale to larger networks, Figure 6 compares the three routing algorithms as h increases, for a fixed value of ρ. For example, for oblivious routing at 30% load, the latency expression $h\overline{\ell}/(1-\rho) - p_c(h-1)\overline{\ell}$ reduces to $46.6h+44.8$, resulting in an average end-to-end delay of just over 500 time units for 10-hop routes. Even at this moderate load, oblivious routing introduces queuing delay at 30% of the links, while avoiding the time to buffer the packet at the remaining 70% of links. Since cutting through intermediate nodes significantly shortens packet delay, even small increases in p_c can translate into significant reductions in end-to-end latency. With larger values of h, packets have more opportunities to consider multiple outgoing links, widening the performance gap between adaptive and oblivious routing. Diagonal routing performs especially well for large values of h, since more source-destination pairs require packets to travel in both the x and y direction; this results in a significant drop in average latency, as shown in Figure 6(b). The benefits of adaptive routing can be even more dramatic under non-uniform traffic patterns, as shown in the next section.

4 Simulation Results

The analytical expressions for p_c and average latency estimate the routing performance, but some low-level design decisions and inter-node dependencies are difficult to capture mathematically. Experiments with a discrete-event simulation model provide deeper insight into the interaction between routing and cut-through switching. To verify the analytical model and assess the effectiveness of the independence



(a) Cut-through probability (b) Average latency

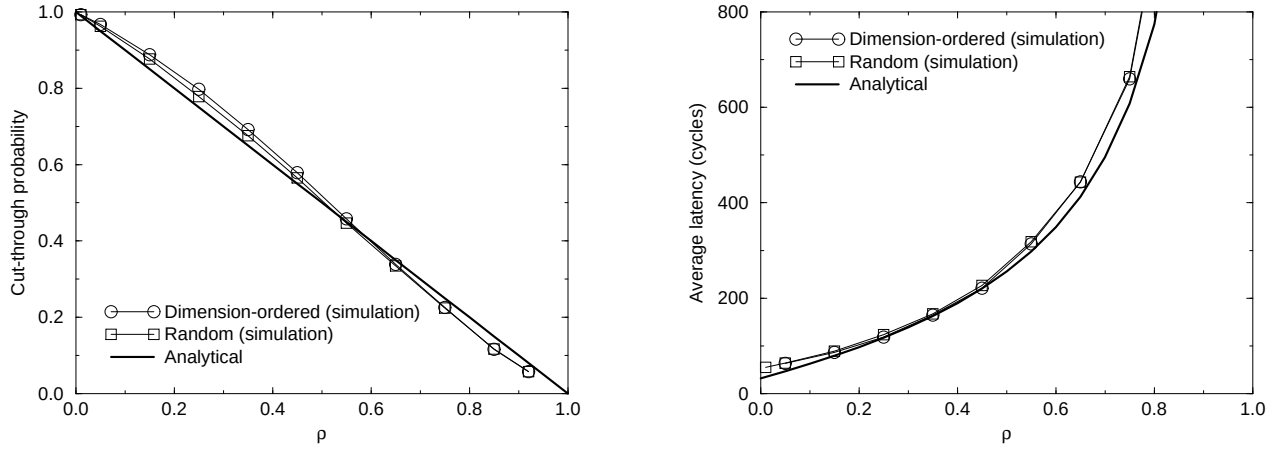
Figure 6: **Routing Performance as a Function of Hop-Count:** These graphs illustrate the performance of the three types routing algorithms as a function of h. Packets have mean length $\overline{\ell} = 64$ bytes and link utilization is $\rho = 0.3$. Diagonal adaptive routing outperforms the random and dimension-ordered adaptive routing, which outperforms oblivious routing particularly for larger values of h.

assumption, the initial simulation experiments evaluate idealized network and workload parameters. Additional experiments consider how well the routing algorithms perform under the non-uniform traffic patterns that can arise in realistic multicomputer applications.

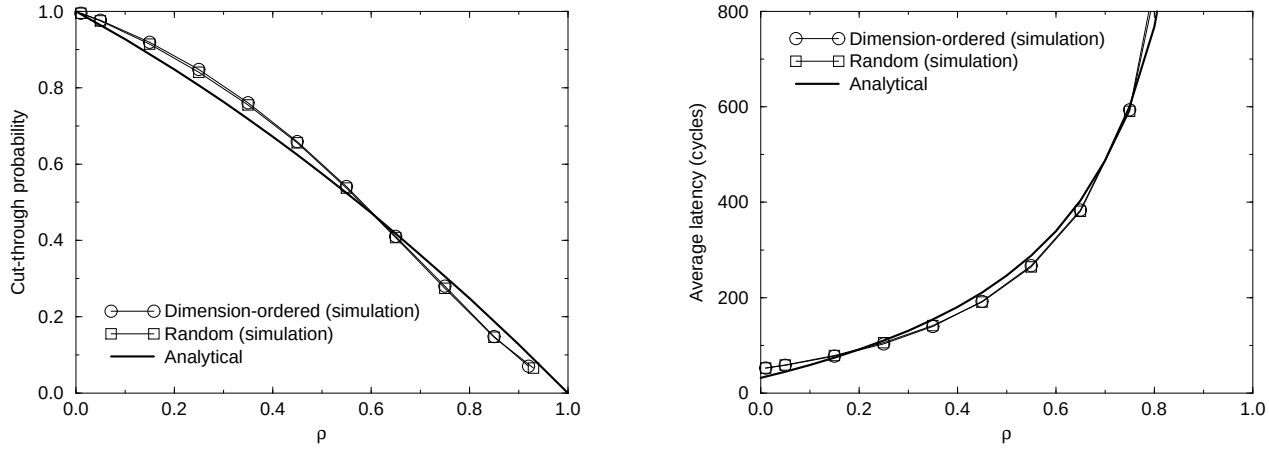
4.1 Uniform Traffic

To enable comparisons with the analytical model, the simulation experiments evaluate a router architecture that consists of a crossbar switch connecting the input ports to unbounded output queues, using the **pp-mess-sim** multicomputer network simulator [38]. Each link serves packets in first-in-first-out order, giving preference to buffered packets over new incoming packets. Every packet has a one-byte routing header to identify the destination node in a 16×16 torus (16-ary 2-cube) network; since the analysis does not include the header processing delay at each node, the analytical expressions slightly underestimate average latency at low loads. The simulation model describes the flow of each packet through the network, capturing contention for access to the outgoing links and the port to the local processor. In these preliminary experiments, each node independently generates packets with exponentially distributed inter-arrival times (with $\overline{\ell} = 64$ bytes) and uniform random selection of destination nodes, following the workload assumptions in Table 2.

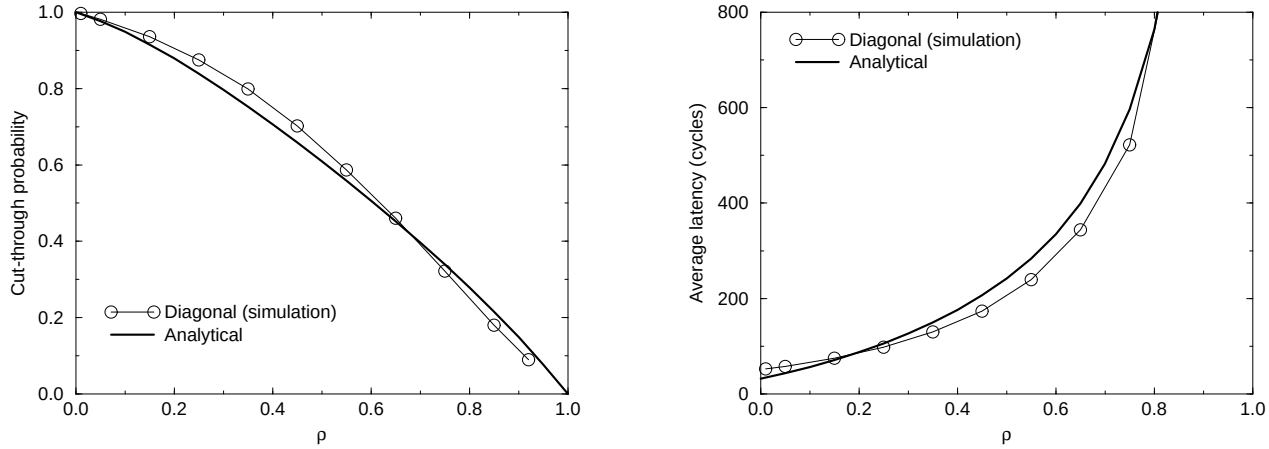
Figure 7 compares the analytical models to the simulation results for packets traveling five hops in a 16×16 torus network. The analytical expressions closely match the simulation results. In Figure 7(a), oblivious routing performs slightly better when a router selects outgoing links in dimension order, instead



(a) Cut-through probability and average latency for oblivious routing

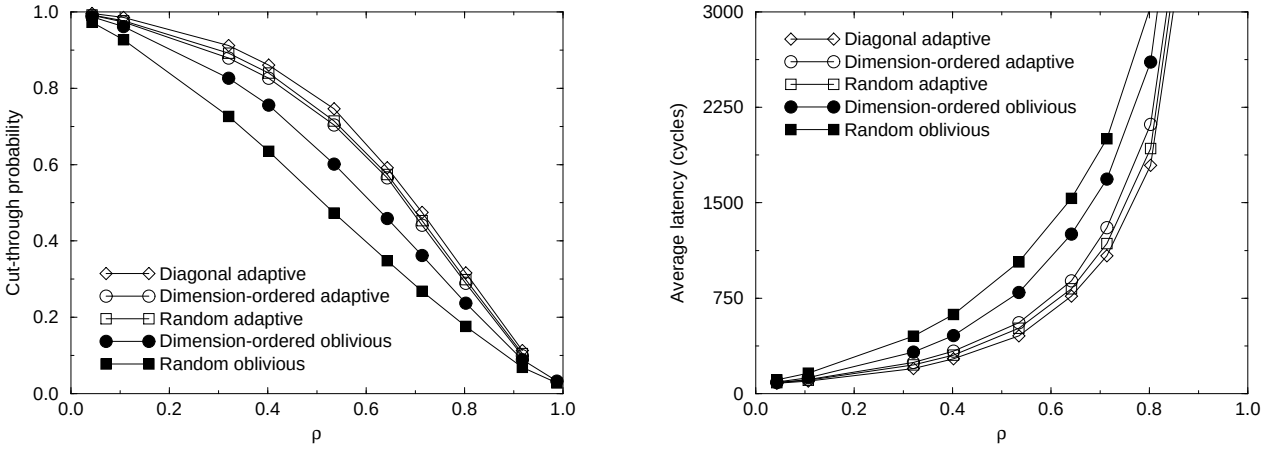


(b) Cut-through probability and average latency for passive adaptive routing



(c) Cut-through probability and average latency for diagonal adaptive routing

Figure 7: **Verifying the Analytical Model:** This figure plots the performance of the routing algorithms, based on both the analytical model and the simulation experiments, as a function of load ρ. The experiments evaluate a 16×16 torus with $\overline{\ell} = 64$ bytes and $h = 5$ hops.

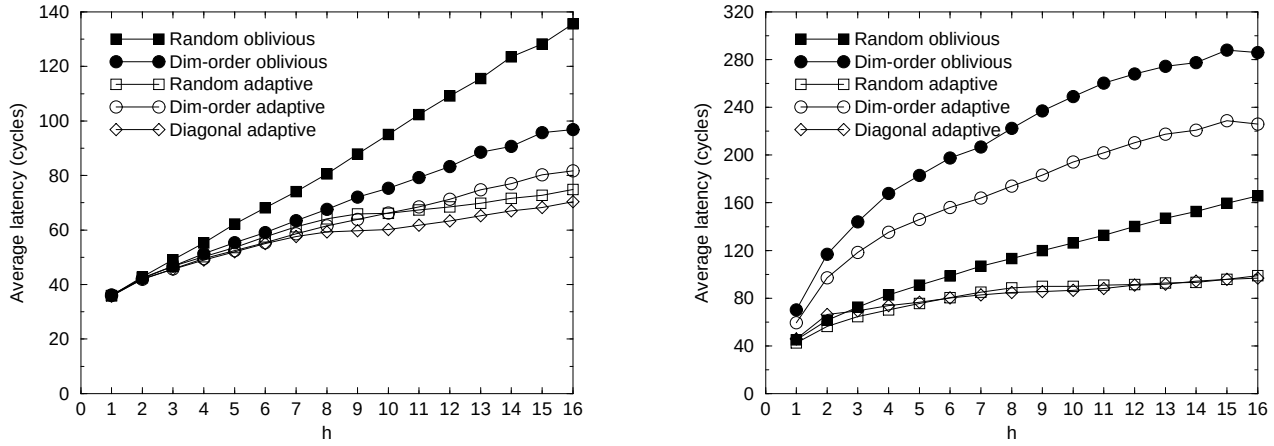


(a) Cut-through probability (b) Average latency

Figure 8: **Larger Hop-Counts:** This figure plots the performance of the routing algorithms in a 16×16 torus with uniform traffic load. The graphs show results for packets traveling $h = 12$ hops in the simulated network.

of random order. This occurs because, under dimension-ordered routing, a packet entering a node in one direction generally exits the node traveling in the same direction. This, in turn, reduces the likelihood that packets from different incoming links contend for the same output port [39, 40]. As a result, dimension-ordered routing has a higher cut-through probability and lower average latency, although the analytical model predicts that the dimension-ordered and random selection functions should have the same performance. As shown in Figure 8, the benefit of dimension-order routing becomes more significant for larger values of h, since packets travel through more nodes in each direction, changing dimensions only once.

In contrast, Figure 7(b) shows little difference between the dimension-ordered and random selection functions under *adaptive* routing, since the extra routing flexibility is sufficient to resolve these link conflicts. The diagonal algorithm exhibits the best performance by actively increasing the number of routing options, particularly at later nodes in a packet's route; still, diagonal routing does not significantly outperform the other two adaptive algorithms under uniform traffic loads and moderate values of h, as seen in Figure 8. Under low load, the analytical expressions slightly underestimate packet latency, since the analysis does not capture the small delay for processing the one-byte header at each hop in the route. Although the analytical and simulation results match fairly closely, the plots exhibit some consistent discrepancies, characterized further in Section 5.



(a) "Normal" packets (b) "Hot" packets

Figure 9: **Non-Uniform Traffic:** These figures plot the performance of the routing algorithms when 5% of the packets are destined for a single "hot-spot" node. The experiment evaluates a 16×16 torus network with a background load of "normal" uniform random traffic that consumes 20% of the available link bandwidth. Adaptive routing can significantly reduce average latency for both types of traffic, particularly when packets travel long distances in the network.

4.2 Non-Uniform Traffic

While the analytical model assumes that traffic load is uniform throughout the network, common multicomputer constructs, such as synchronization or multicast operations, can induce "hot-spots" of heavily-utilized nodes and links. Figure 9 compares the performance of the three routing algorithms in a simulated 16×16 torus network under non-uniform load. A single "hot" node receives 5% of the traffic, while the remaining packets select a destination at random; the "normal" uniform traffic loads the network to 20% capacity, with the non-uniform traffic generating a congested region near the hot-spot node. The plots in Figure 9 show average end-to-end latency for packets traveling h hops. For low values of h, normal packets do not experience significant delay, since most packets do not encounter the congested region.

However, for dimension-ordered oblivious routing, average delay rises steeply as packets travel further in the network. In contrast, the plots for adaptive routing stay relatively flat, since packets reduce their average latency by circumventing busy links and nodes. As h increases, the adaptive algorithms have more routing flexibility, since packets typically have more internal nodes. This improves the scalability of virtual cut-through switching as network sizes and packet distances increase. Diagonal routing performs especially well in this context, since it preserves routing flexibility even as packets near their destination nodes, clearing congestion near the hot-spot region. Packets destined for "normal" destinations avoid the nodes and links near the hot spot, significantly reducing the delay for hot packets that must enter

the congested region.

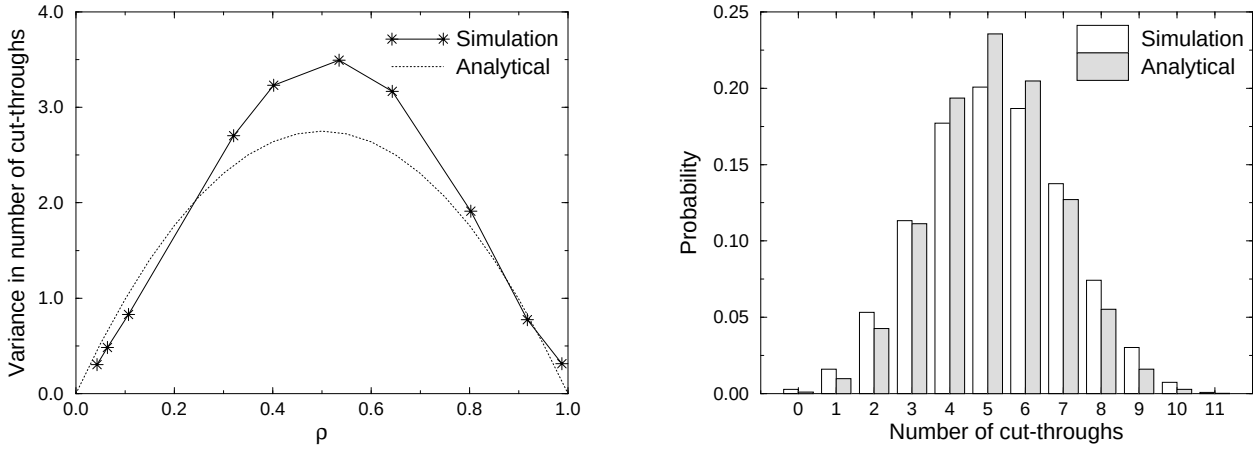
The random and dimension-ordered selection functions exhibit quite different performance under hot-spot traffic, in contrast to the expectations of the analytical model. For the "normal" traffic, dimension-ordered routing has lower average latency, since packets arriving on different incoming links tend to head to different outgoing links, as in Figure 8. However, the random routing algorithms significantly outperform dimension-ordered routing for the "hot" traffic. This occurs because dimension-ordered routing forces most hot-spot packets to enter the hot-spot node on one of the y-direction links; this results in heavy congestion on the two y-directions links, even though the x-direction links have a relatively light load. In contrast, the random selection function better balances the traffic load across the various links near the hot-spot node. Through further experimentation with non-uniform traffic, we have shown that the router's selection function(s) can have a dramatic impact on network performance, particularly for communication patterns in scientific applications [41]. These results suggest that multicomputer networks could potentially benefit from support for multiple selection functions, each tailored to a specific type of traffic, particularly if applications generate non-uniform workloads.

5 Inter-Node Dependencies

Although the simulation results match the expectations of the analytical model fairly well, the plots in Figure 7 show consistent performance differences. Unlike the analytical expressions, the simulation experiments capture the effects of dependencies between adjacent nodes in the network. This section isolates the effects of the independence assumption through a detailed comparison of the analytical and simulation results for the oblivious routing algorithms. Additional experiments show that realistic router architectures and application workloads exacerbate the effects of inter-node dependencies. New routing algorithms and task-allocation schemes can capitalize on these natural dependencies between adjacent nodes to improve network performance.

5.1 Cut-Through Correlation

To characterize the impact of the independence assumption, Figure 10 considers the performance of random oblivious routing under the architectural and workload parameters in Table 2. While Figure 8 plotted the *average* values for the cut-through probability and communication latency, Figure 10 considers the *distribution* of the number of cut-throughs. As discussed in Section 2, the number of cut-throughs should obey a binomial distribution, where a packet has cut-through probability $1-\rho$ at each of the $h-1$ intermediate nodes in its route. As a result, packets should experience an average of



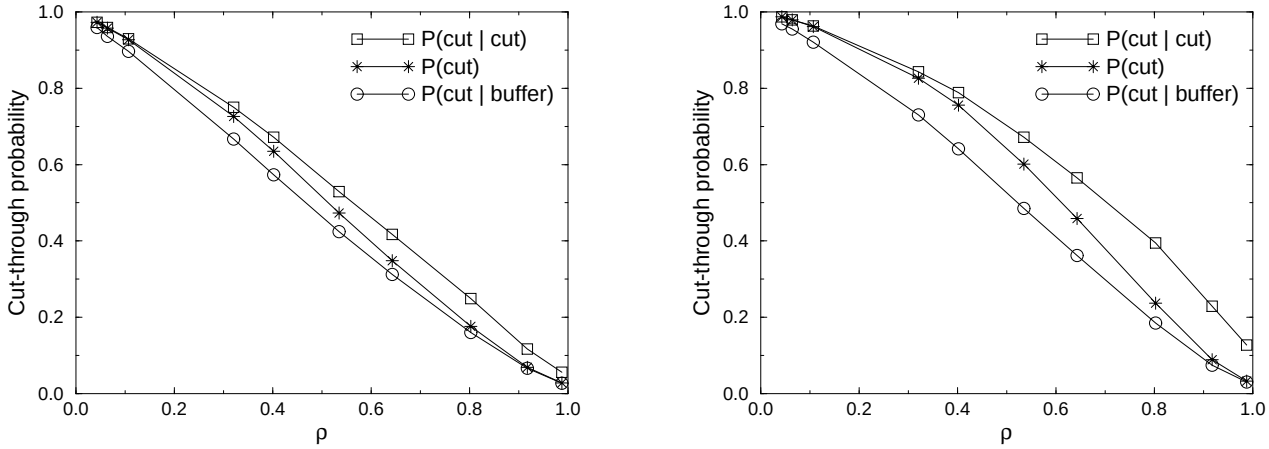
(a) Variance in number of cut-throughs (b) Number of cut-throughs ($\rho = 0.535$)

Figure 10: **Number of Cut-Throughs:** These graphs plot the distribution of the number of cut-throughs for packets traveling $h = 12$ hops in a 16×16 torus under random oblivious routing. Based on the analytical model, the number of cut-throughs should follow a binomial distribution; however, the simulation results show that more packets cut through a very large or very small number of intermediate nodes.

$(1-\rho)(h-1)$ cut-throughs, with a variance of $\rho(1-\rho)(h-1)$. Instead, the simulated network exhibits *greater* variability in the number of cut-throughs, particularly for moderate values of ρ. As shown in Figure 10(b), the simulated network has a *flatter* distribution for the number of cut-throughs, with heavier tails than the analytical model suggested.

In particular, the simulated network has more packets that experience a large number of cut-throughs or a small number of cut-throughs; in Figure 10(b), this is shown by the taller "simulation" bars for both small and large values on the x-axis. This, in turn, translates into greater variability in end-to-end packet latency [42], even for an idealized router model. A closer look at the simulation data reveals the cause of this variability. Figure 11(a) shows the packet cut-through probability, conditioned on the switching decision at the packet's previous hop. In theory, the likelihood of cutting through an intermediate node should be independent of the packet's experience at previous hops in its route. However, Figure 11(a) shows significant variation in the cut-through probability, depending on whether or not a cut-through occurred on the packet's previous hop; the middle curve is effectively the weighted average of the two conditioned plots.

A prior cut-through encourages an additional cut-through, while buffering a packet is more likely to lead to future packet bufferings. As a result, some packets cut through several nodes in a row, while other packets consistently buffer at intermediate nodes. At low loads, this phenomenon increases the likelihood of cut-throughs, since the frequent cut-throughs perpetuate themselves, while the reverse

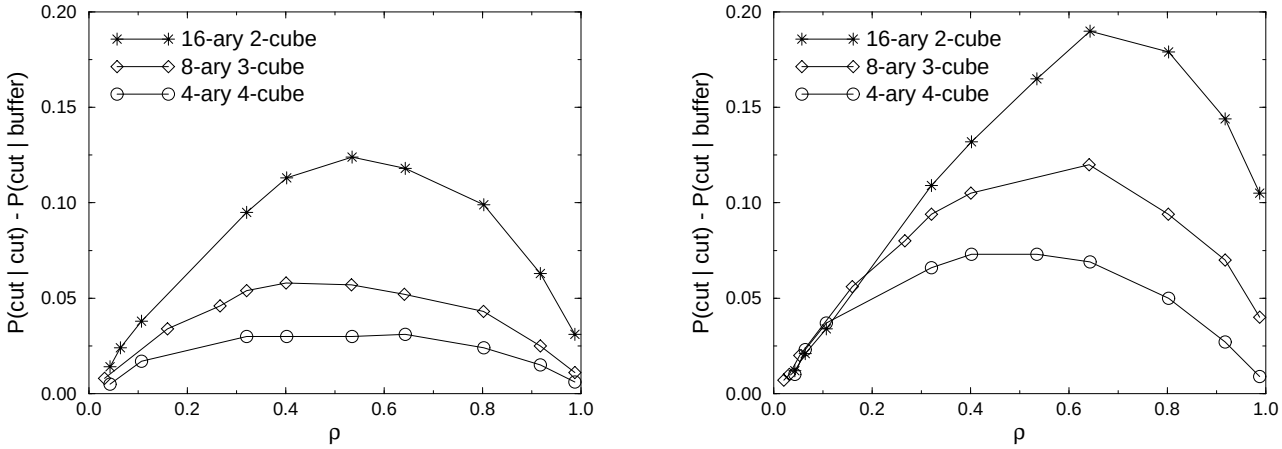


(a) Random oblivious routing (b) Dimension-ordered oblivious routing

Figure 11: **Conditional Cut-Through Probability:** These graphs show the influence of packet history on network performance for random and dimension-ordered oblivious routing. Using simulation results for a 16×16 torus network, the graphs plot the cut-through probability, conditioned on the switching decision at the previous hop in the packet's route, for traffic traveling $h = 12$ hops. Cutting through one link increases the likelihood that the packet cuts through the subsequent link, especially under dimension-ordered routing.

occurs at high loads. Hence, for small ρ, the simulated packets consistently experience slightly more cut-throughs than the analytical model predicts, while the opposite is true for large ρ, as seen in Figure 7(a). This type of dependency seems unusual, since random oblivious routing considers exactly one outgoing link at each node, irrespective of a packet's past history. Experiments with fixed-size packets and different average packet lengths show the same dependencies on past history.

Correlation in the cut-through probability occurs because in an actual system, as in the simulator, nodes are not truly independent. In a real network, if a packet buffers at one node, then it eventually exits the router behind at least one other packet. This increases the likelihood that the packet's next outgoing link is busy, too. Likewise, if a packet cuts through a node, then the link it uses has been idle for some length of time, increasing the likelihood that the packet encounters a light load at the subsequent node. Although such inter-node correlations also affect packet-switched networks, the performance of cut-through networks is much more sensitive to the probability of encountering idle outgoing links. As shown in Figure 11(b), the dependencies between nodes are even more pronounced under dimension-ordered routing, due to the tighter coupling between incoming and outgoing links in the same dimension of the network.



(a) Random oblivious routing

(b) Dimension-ordered oblivious routing

Figure 12: **Network Topology:** These graphs highlight the influence of network topology on inter-node dependencies for random and dimension-ordered oblivious routing. The figures show the difference between the two conditional cut-through probabilities, effectively subtracting the top and bottom curves in Figure 11. These simulation results for packets traveling $h = 6$ hops show that low-dimensional topologies exhibit a stronger dependence on packet history.

5.2 Network Topology and Non-Uniform Traffic

In addition, modern cut-through networks typically employ low-dimensional topologies that limit the mixing of incoming traffic streams; a larger number of incoming and outgoing links would ensure more traffic mixing, reducing the dependencies between adjacent nodes [31]. Figure 12 illustrates this effect by comparing cut-through correlation in three k-ary n-cube topologies with different dimensions n, where the network has k nodes along each of n dimensions for a total of k^n nodes. The graph plots the difference

$$P[\text{cut-through} \mid \text{cut-through on previous hop}] - P[\text{cut-through} \mid \text{buffering on previous hop}]$$

for packets traveling 6 hops in the simulated networks. In the absence of inter-node dependencies, this metric should equal zero, independent of network load. While the metric consistently exceeds zero for all three topologies, the 16-ary 2-cube network exhibits a much stronger dependence on past packet history, particularly under dimension-ordered routing.

Similarly, many common communication patterns, such as scientific permutations, limit the interaction of traffic from different incoming links. Figure 13 shows the cut-through performance of dimension-ordered routing under a *bit-reversal* permutation, where a packet travels to the destination node whose binary address is the reverse of the source identifier; for example, in a 16×16 torus, node

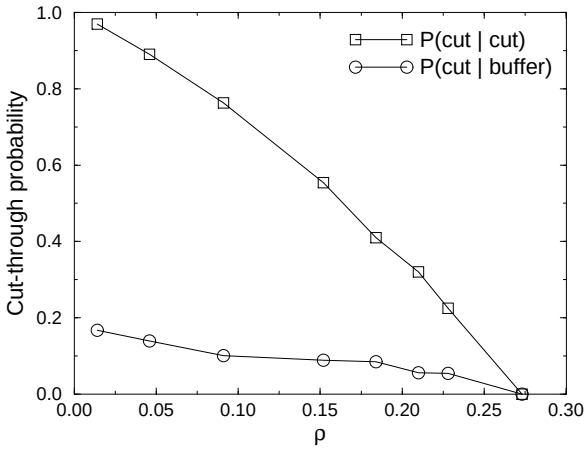


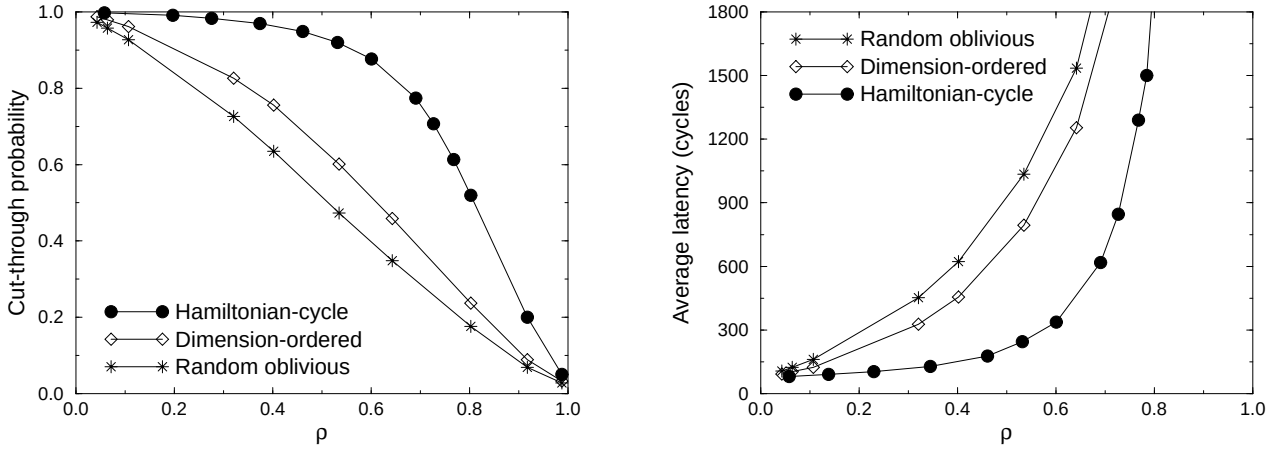
Figure 13: **Bit-Reversal Traffic:** This graph plots the influence of packet history under a non-uniform, bit-reversal traffic pattern. In these simulation experiments, packets employ dimension-ordered oblivious routing in a 16×16 torus network. The bit-reversal traffic limits the mixing of packets from different incoming links, exacerbating the dependencies between neighboring nodes.

$(3, 10)$ communicates with node $(12, 5)$. This generates a non-uniform traffic load[2] as seen by the small peak link utilization in Figure 13. Under this traffic pattern, nodes in a common row (same y coordinate) of the torus communicate with destinations in the same column (same x coordinate), and vice versa. Consequently, packets that contend for the same outgoing link tend to share several links in common during the remainder of their routes; as a result, a blocked packet must repeatedly buffer behind other traffic, significantly reducing the cut-through probability, even at low network loads.

5.3 Hamiltonian-Cycle Routing

To further demonstrate the influence of traffic mixing on cut-through performance, we consider an extreme example of a routing algorithm that prohibits the mixing of traffic from different incoming links. In particular, *Hamiltonian-cycle* routing decomposes the underlying topology into link-disjoint cycles that include all nodes in the network; for example, a 4×4 torus (4-ary 2-cube) has four disjoint, unidirectional Hamiltonian cycles, as shown in Figure 15(a). By forcing each packet to stay in a single cycle, or "dimension," for its entire route, H-cycle routing associates each input link with exactly one output link. Packets arriving on different incoming links never compete for the same outgoing link; likewise, packets arriving on the same incoming link never route to different outgoing links. We assume that each packet travels on the shortest Hamiltonian path toward its destination. Although H-cycle routing can result in extremely long routes, compared to shortest-path algorithms, the experiments with

[2]Since the various parts of the network experience different traffic loads, the experiment focuses on a single pair of adjacent links; other link pairs show similar cut-through correlation effects. The non-uniform traffic load limits the average link utilization to under 30% of the available bandwidth.

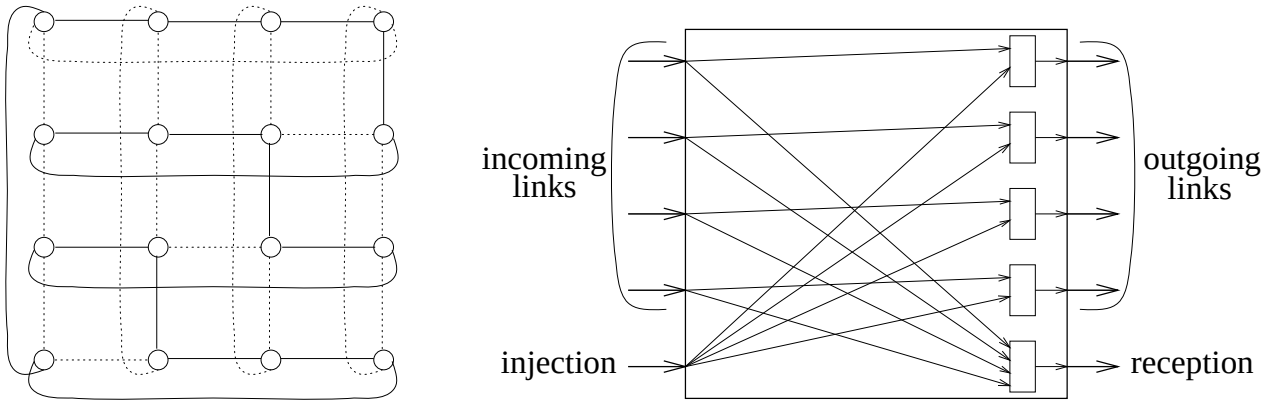


(a) Cut-through probability p_c (b) Average packet latency

Figure 14: **Hamiltonian-Cycle Routing:** This figure evaluates the performance of *Hamiltonian-Cycle* routing, which forces each packet to travel on one of four distinct cycles in the network. The experiments compare this non-minimal scheme to random and dimension-ordered oblivious routing, as a function of link utilization ρ. Note that H-cycle routing generates longer routes in the network, resulting in a higher value of ρ for the same packet generation rate; still, the algorithm has the potential to improve network performance by increasing the cut-through probability.

H-cycle routing illustrate how limiting traffic mixing can dramatically reduce network contention, as shown by the graphs in Figure 14.

The experiments compare three oblivious routing algorithms across a range of ρ values, where H-cycle routing has a smaller injection rate λ to compensate for the larger average path length. For the same link utilization, H-cycle routing has a significantly higher cut-through probability, and much lower latency, than random and dimension-ordered routing by restricting the mixing of traffic from different incoming links. Although the long routes limit the general utility of H-cycle routing, the network can reduce the effects of long routes by placing communicating tasks on nodes that are near each other on one of the underlying Hamiltonian cycles. In addition, Hamiltonian-cycle routing facilitates a simple and efficient router implementation by reducing the complexity of the switch that connects the incoming and outgoing links. The reduced connectivity between links also simplifies the buffer architecture for storing blocked packets, permitting the router to place packet queues at the incoming links without introducing contention between traffic headed to different outgoing links. Alternatively, a network could support a mixture of H-cycle and shortest-path routing, resorting to traditional algorithms when the H-cycle path is prohibitively long.



(a) Four unidirectional cycles (b) Two-dimensional router

Figure 15: **H-Cycle Routing:** This figure shows two disjoint, bidirectional Hamiltonian cycles (for a total of four unidirectional cycles) in a 4×4 torus network, along with a simplified router architecture for H-cycle routing.

6 Related Work

A variety of analytical models have been developed to evaluate communication performance in virtual cut-through [19–26] and wormhole [40, 43–45] networks. Although wormhole and virtual cut-through switching are conceptually similar, they typically require different analysis techniques. Models of wormhole switching typically focus on capturing the effects of stalled packets on the performance of other traffic. Analysis of virtual cut-through switching began with the work in [5], which motivates our approach in Section 2.1; we extend this model to consider adaptive routing and the effects of inter-node dependencies. Other recent work on analytical models of virtual cut-through switching focus on the delivery-time distribution [21, 42] and the impact of fixed-length packets [22] under oblivious routing. In addition, several recent studies [23–25] have investigated the use of virtual cut-through switching within a multistage ATM switch, which introduces new traffic models and performance requirements, including bursty traffic with delay constraints.

A variety of simulation-based studies have evaluated adaptive routing in wormhole and virtual cut-through networks. Recent work in [26] presents an analytic model of adaptive routing under virtual cut-through switching, based on enumerating routing choices at each hop in the route, similar to the P_2 parameter defined in Section 2.2. In contrast to the model in [26], we focus on several different selection functions and present a closed-form analytic expression for dimension-ordered and random routing, instead of tabulating P_2 numerically by enumerating all possible paths; in addition, we derive an efficient recurrence for computing P_2 for the diagonal selection function. The work in [46] includes simulation experiments that evaluate several selection functions, including the diagonal scheme. In

contrast to our results, the experiments in [46] find that diagonal routing performs *worse* than the other approaches, due to the nature of the deadlock-avoidance technique; the proposed *dimension-reversal* scheme restricts a packet's access to certain virtual channels whenever the packet travels from a higher dimension to a lower dimension, which happens frequently in diagonal routing.

The evaluation of the inter-node dependencies in Section 5 complements earlier work on the effects of Kleinrock's independence assumption [31]. Although the independence assumption has been studied in the context of packet-switched networks [31–34], to the best of our knowledge, no previous study has revisited this issue in the context of virtual cut-through switching, which is an important scheme widely used in modern multicomputer systems. The initial work on virtual cut-through switching in [5] justifies the use of the independence assumption based on earlier studies of packet-switched networks. Although the assumption remains fairly accurate under virtual cut-through switching, as shown in Figure 7, inter-node dependencies have a stronger effect in cut-through networks by affecting the likelihood that a packet can cut through intermediate nodes in its route. Moreover, the traffic patterns and topologies in modern multicomputer networks serve to exacerbate the effects by limiting the mixing of traffic from different incoming links, as shown in Section 5.2.

7 Conclusions and Future Work

In modern multicomputer networks, communication performance is sensitive to the subtle interplay between routing algorithms and switching schemes. This paper compares the cut-through performance of several oblivious and adaptive routing algorithms, with different selection functions, based on both analytical models and event-driven simulations. The analytical model exploits the concept of *internal* and *border* nodes to develop recurrences for computing the likelihood that a packet has two routing choices at intermediate nodes in its route. Based on these recurrences, we derived a closed-form expression for evaluating random and dimension-ordered adaptive routing, as well as a simplified recurrence for studying diagonal routing, as a function of link utilization and the distance packets travel in the network. As a result, the analytical model can efficiently predict the performance of large networks to help weigh the cost-performance trade-offs of the various routing algorithms.

For the sake of tractability, the analytical model relies on a collection of simplifying assumptions about the network architecture and the communication workload. Through simulation experiments on an idealized router model, we verify the "trends" in the analysis and highlight the influence of more realistic traffic patterns. Despite the close agreement between the analytical and simulation results, the experiments show consistent discrepancies due to the dependencies between adjacent nodes in cut-through networks. Ultimately, cut-through probability is sensitive to whether a packet could establish

a cut through at the previous node in its route. This dependence on past history can cause analytical models to consistently underestimate or overestimate cut-through probability and, in turn, other metrics like the mean and variance of end-to-end packet latency. By comparing analytical and simulation results, the paper characterizes these correlation effects under different routing algorithms, network topologies, and traffic patterns.

The simulation experiments show that the network policies and communication workloads in a modern multicomputer system exacerbate inter-node dependencies by limiting traffic mixing. These results motivate the development of novel routing algorithms and task-allocation schemes that exploit the natural dependencies between neighboring nodes. For example, Hamiltonian-cycle routing can increase cut-through probability by limiting contention between incoming packets; coupled with an effective algorithm for assigning application tasks to processing nodes, H-cycle routing has the potential to improve packet latency and network throughput. Other task-allocation schemes could strive to minimize the number of border nodes between communicating tasks; this would increase the likelihood that adaptive routing algorithms could consider multiple outgoing links for incoming packets.

Future work will evaluate cut-through networks under a wider range of traffic patterns and network policies to isolate the scenarios that increase or decrease the dependencies between adjacent nodes. This knowledge can aid the construction of new routing algorithms and flow-control schemes that improve a packet's chance of avoiding delay at intermediate nodes. In particular, the model in Section 2 does not capture the effects of finite packet buffers in the router, which can have significant performance implications, particularly under heavy load. Under these constraints, the flow-control schemes necessary to avoid buffer overflow are likely to introduce additional dependence between the traffic load at nearby nodes. In addition, realistic multicomputer applications generate traffic with a mixture of different packet sizes and interarrival times, which may influence the relationship between packet history and cut-through performance. Based on these results, future research could consider flexible network architectures that support these diverse communication patterns and exploit the natural dependencies between neighboring nodes.

References

[1] W. Athas and C. Seitz, "Multicomputers: Message-passing concurrent computers," *IEEE Computer Magazine*, pp. 9–24, August 1988.

[2] X. Zhang, "System effects of interprocessor communication latency in multicomputers," *IEEE Micro*, pp. 12–15, 52–55, April 1991.

[3] L. Ni and P. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Computer Magazine*, pp. 62–76, February 1993.

[4] W. J. Dally and C. L. Seitz, "The torus routing chip," *Journal of Distributed Computing*, vol. 1, no. 3, pp. 187–196, 1986.

[5] P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique," *Computer Networks*, vol. 3, pp. 267–286, September 1979.

[6] J. Ngai and C. Seitz, "A framework for adaptive routing in multicomputer networks," in *Proceedings of the Symposium on Parallel Algorithms and Architectures*, pp. 1–9, June 1989.

[7] J. Rexford and K. G. Shin, "Support for multiple classes of traffic in multicomputer routers," in *Proceedings of the Parallel Computer Routing and Communication Workshop*, pp. 116–130, May 1994.

[8] K. G. Shin and S. W. Daniel, "Analysis and implementation of hybrid switching," *IEEE Transactions on Computers*, vol. 45, pp. 684–692, June 1996.

[9] A. Nowatzyk, M. Browne, E. Kelly, and M. Parkin, "S-connect: From networks of workstations to supercomputer performance," in *Proceedings of the International Symposium on Computer Architecture*, pp. 71–82, June 1995.

[10] N. Boden, D. Cohen, R. Felderman, A. Kulawik, C. Seitz, J. Seizovic, and W.-K. Su, "Myrinet: A gigabit-per-second local area network," *IEEE Micro*, pp. 29–36, February 1995.

[11] A. L. Davis, "Mayfly: A general-purpose, scalable, parallel processing architecture," *Lisp and Symbolic Computation*, vol. 5, pp. 7–47, May 1992.

[12] S. S. Owicki and A. R. Karlin, "Factors in the performance of the AN1 computer network," in *Proceedings of ACM SIGMETRICS*, pp. 167–180, June 1992.

[13] Y. Tamir and G. Frazier, "Dynamically-allocated multi-queue buffers for VLSI communication switches," *IEEE Transactions on Computers*, vol. 41, pp. 725–737, June 1992.

[14] K. Bolding, S.-C. Cheung, S.-E. Choi, C. Ebeling, S. Hassoun, T. A. Ngo, and R. Wille, "The Chaos router chip: Design and implementation of an adaptive router," in *Proceedings of the IFIP International Conference on VLSI*, pp. 311–320, September 1993.

[15] H. S. Lee, H. W. Kim, J. Kim, and S. Lee, "Adaptive virtual cut-through as an alternative to wormhole routing," in *Proceedings of the International Conference on Parallel Processing*, pp. I–68–I–75, August 1995.

[16] K. Toda, K. Nishida, E. Takahashi, N. Michell, and Y. Yamaguchi, "Design and implementation of a priority forwarding router chip for real-time interconnection networks," *International Journal of Mini and Microcomputers*, vol. 17, no. 1, pp. 42–51, 1995.

[17] S. Konstantinidou, "Segment router: A novel router design for parallel computers," in *Proceedings of the Symposium on Parallel Algorithms and Architectures*, June 1994.

[18] S. Daniel, J. Rexford, J. Dolter, and K. Shin, "A programmable routing controller for flexible communications in point-to-point networks," in *Proceedings of the IEEE International Conference on Computer Design*, pp. 320–325, October 1995.

[19] M. Ilyas and H. T. Mouftah, "Towards performance improvement of cut-through switching in computer networks," *Performance Evaluation*, vol. 6, pp. 125–133, July 1986.

[20] A. Abo-Taleb and H. T. Mouftah, "Delay analysis under a general cut-through switching technique computer networks," *IEEE Transactions on Communications*, vol. COM-35, pp. 356–359, March 1987.

[21] J. W. Dolter, P. Ramanathan, and K. G. Shin, "Performance analysis of virtual cut-through switching in HARTS: A hexagonal mesh multicomputer," *IEEE Transactions on Computers*, vol. 40, pp. 669–680, June 1991.

[22] S. Abraham and K. Padmanabhan, "Performance of multicomputer networks under pin-out constraints," *Journal of Distributed Computing*, pp. 237–248, 1991.

[23] Y. S. Youn and C. K. Un, "Performance analysis of an integrated voice/data cut-through switching network," *Computer Networks and ISDN Systems*, vol. 21, no. 1, pp. 41–51, 1991.

[24] T. D. Morris and H. G. Perros, "Approximate analysis of a discrete-time tandem network of cut-through queues with blocking and bursty traffic," *Performance Evaluation*, vol. 17, pp. 207–223, May 1993.

[25] I. Widjaja, A. Leon-Garcia, and H. T. Mouftah, "The effect of cut-through switching on the performance of buffered Banyan networks," *Computer Networks and ISDN Systems*, vol. 26, pp. 139–159, 1993.

[26] W. A. Najjar, A. Lagman, S. Sur, and P. K. Srimani, "Modeling adaptive routing in k-ary n-cube networks," in *Proceedings of the International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 120–125, 1994.

[27] V. Karamcheti and A. A. Chien, "Software overhead in messaging layers: Where does the time go?," in *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 51–60, October 1994.

[28] K. Aoyama and A. Chien, "Cost of adaptivity and virtual lanes in a wormhole router," *Journal of VLSI Design*, vol. 2, no. 4, pp. 315–333, 1995.

[29] W. Dally, "Virtual-channel flow control," *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, pp. 194–205, March 1992.

[30] H. G. Badr and S. Podar, "An optimal shortest-path routing policy for network computers with regular mesh-connected topologies," *IEEE Transactions on Computers*, vol. C-38, pp. 1362–1370, October 1989.

[31] L. Kleinrock, *Communication Nets: Stochastic Message Flow and Delay*. New York: McGraw-Hill, 1964.

[32] S. Calo, "Delay properties of message channels," in *Proceedings of the International Conference on Communications*, pp. 43.5.1–43.5.4, June 1979.

[33] K. Fendick, V. Sakena, and W. Whitt, "Dependence in packet queues," *IEEE Transactions on Communications*, vol. 37, pp. 1173–1183, November 1989.

[34] W. Lau and S. Li, "Traffic analysis in large-scale high-speed integrated networks: Validation of nodal decomposition approach," in *Proceedings of IEEE INFOCOM*, pp. 1320–1329, 1993.

[35] L. Kleinrock, *Queueing systems*, vol. I: Theory. John Wiley & Sons, 1975.

[36] P. J. Burke, "The output of a queueing system," *Operations Research*, vol. 4, pp. 699–704, 1956.

[37] J. R. Jackson, "Networks of waiting lines," *Operations Research*, vol. 5, pp. 518–521, August 1957.

[38] J. Rexford, W. Feng, J. Dolter, and K. G. Shin, "PP-MESS-SIM: A flexible and extensible simulator for evaluating multicomputer networks," *IEEE Transactions on Parallel and Distributed Systems*, pp. 25–40, January 1997.

[39] S. Abraham and K. Padmanabhan, "Performance of the direct binary n-cube network for multiprocessors," *IEEE Transactions on Computers*, vol. 38, pp. 1000–1011, July 1987.

[40] A. Agarwal, "Limits on interconnection network performance," *IEEE Transactions on Parallel and Distributed Systems*, vol. 2, pp. 398–412, October 1991.

[41] W. Feng and K. G. Shin, "Impact of selection functions on routing algorithm performance in multicomputer networks," in *Proceedings of the International Conference on Supercomputing*, pp. 132–139, July 1997.

[42] J. Rexford and K. G. Shin, "Shortest-path routing in homogeneous point-to-point networks with virtual cut-through switching," Computer Science and Engineering Technical Report CSE-TR-146-92, University of Michigan, November 1992.

[43] W. J. Dally, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Transactions on Computers*, vol. 39, pp. 775–785, June 1990.

[44] J. Kim and C. Das, "Modeling wormhole routing in a hypercube," in *Proceedings of the International Conference on Distributed Computing Systems*, pp. 386–393, May 1991.

[45] J. T. Draper and J. Ghosh, "A simple analytical model for wormhole routing in multicomputer systems," *Journal of Parallel and Distributed Computing*, vol. 20, pp. 202–214, November 1994.

[46] W. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, pp. 466–475, April 1993.