Randomized Throughput-Optimal Oblivious Routing for Torus Networks

Rohit Sunkam Ramanujam, *Student Member, IEEE,* and Bill Lin, *Member, IEEE*

Abstract—In this paper, we study the problem of optimal oblivious routing for one and two dimensional torus networks. We introduce a new closed-form oblivious routing algorithm called W2TURN that is worst-case throughput optimal for 2D torus networks. W2TURN is based on a weighted random selection of paths that contain at most two turns. Restricting the maximum number of turns in routing paths to just two enables a simple deadlock-free implementation of W2TURN. In terms of average hop count, W2TURN outperforms the best previously known closed-form worst-case throughput optimal routing algorithm called IVAL [19]. When the network radix is odd, W2TURN achieves the minimum average hop count that can be achieved with 2-turn paths while remaining worst-case throughput optimal. When the network radix is even, W2TURN comes very close to achieving the minimum average hop count while remaining worst-case throughput optimal, within just 0.72% on a 12×12 torus. We also describe another routing algorithm based on weighted random selection of paths with at most two turns called I2TURN and show that I2TURN is equivalent to IVAL. However, I2TURN eliminates the need for loop removal at runtime and provides a closed-form analytical expression for evaluating the average hop count. The latter enables us to demonstrate analytically that W2TURN strictly outperforms IVAL (and I2TURN) in average hop count. Finally, we present a new optimal weighted random routing algorithm for rings called WRD (Weighted Random Direction). WRD provides a closed-form expression for the optimal distribution of traffic along the minimal and non-minimal directions in a ring topology to achieve minimum average hop count while guaranteeing optimal worst-case throughput. Based on our evaluations, in addition to being worst-case throughput optimal, W2TURN and WRD also perform well in the average-case, and outperform the best previously known worst-case throughput optimal routing algorithms with closed-form descriptions in latency and throughput over a wide range of traffic patterns.

Index Terms—On-chip Networks, Interconnection Networks, Torus Networks, Oblivious Routing.

1 Introduction

Interconnection networks are used in a variety of applications, including packet routing [2], processor-memory interconnect [10], I/O interconnect [6], and on-chip interconnect [3], [7], [11], [21]. Torus networks, or k-ary n-cubes [5], are an important class of interconnection networks and are popular in all these application domains. In this paper, we study the problem of optimal oblivious routing for one and two dimensional torus topologies. Throughput and latency are important performance metrics in the design of routing algorithms. In many throughput-sensitive applications such as packet routing and throughput-driven applications running on general-purpose chip multiprocessors, the network traffic is not known a priori at design time and it is important for the interconnection network to guarantee certain throughput even under the most adversarial traffic. Hence, maximizing the worst-case throughput is a vital objective in oblivious routing algorithm design. In addition to maximizing worst-case throughput, sustaining high throughput in the average-case over a large set of traffic patterns is also important in many application domains. Another performance metric that often conflicts with the goal of maximizing worst-case throughput is that of minimizing packet latency. Although dimension-ordered routing (DOR) [17] can achieve minimal-length routing on torus networks, it suffers from poor worst-case throughput as it offers no route diversity. On the other hand, it is well known that Valiant routing (VAL) [20] can achieve optimal worst-case throughput by load-balancing globally across the entire network, but it does so at the expense of destroying locality and increasing latency. Other oblivious routing algorithms such as ROMM [8] and RLB [15] have good locality, but they fail to achieve optimal worst-case throughput.

To the best of our knowledge, among the closed-form oblivious routing algorithms that can guarantee optimal worst-case throughput, an improved Valiant routing algorithm called IVAL [19] achieves the lowest average hop count. Like two-phase Valiant routing, IVAL load-balances packets to a randomly chosen intermediate node, but reverses the order of traversal of dimensions between the two routing phases (e.g., XY routing, followed by YX routing). In doing so, loops are often formed, and IVAL improves over Valiant routing by removing such loops at runtime.

In this paper, we introduce a new closed-form oblivious routing algorithm called W2TURN that achieves optimal worst-case throughput for 2D-torus networks. W2TURN is based on a weighted random selection of paths with at most two turns. The restriction imposed on the number of allowed turns results in a simple deadlock-free implementation. In comparison to IVAL, W2TURN achieves lower average hop count and higher average-case throughput. We also present

- *This work was presented in part at ACM/IEEE Symposium on Architectures for Networking and Communications Systems [9], Princeton, New Jersey, USA.*
- *R. Sunkam Ramanujam and B. Lin are with the Department of Electrical and Computer Engineering, University of California, San Diego. E-mail: {rsunkamr, billlin}@ucsd.edu*

Digital Object Indentifier 10.1109/TC.2011.226

another weighted random routing algorithm based on selecting paths with at most two turns, called I2TURN. We show that I2TURN is in fact equivalent to IVAL in the sense that packets are routed over the same set of paths with the same probabilities. However, I2TURN eliminates the need for loop removal at runtime and provides a closed-form analytical expression for evaluating the average hop count. The latter enables us to demonstrate analytically that W2TURN does indeed strictly outperform IVAL (and I2TURN) in average hop count.

W2TURN also performs well in comparison to optimization-based solutions. Optimal routing for 2D-torus networks has been formulated as a multicommodity flow problem [19], which can be expressed as linear programs. Using this formulation, worst-case throughput optimal routing with minimum average hop count can be computed. However, it is difficult to guarantee deadlock-free operation for this approach since the resulting solution may include arbitrary paths with arbitrary number of turns. Motivated in part by this difficulty, Towles et al. proposed a modified formulation called 2TURN that guarantees optimal routing when the choice of routing paths is restricted to those with at most two turns. As noted in [19], the key advantage of 2TURN over the optimal solution is the fact that its paths can be described in simple terms, allowing for a simple deadlock-free implementation. However, like the optimal solution, 2TURN does not have a closed-form description, thus requiring a separate linear program for each instance of network size. These linear programs grow quickly, making them difficult to scale to large networks[1]. When the network radix is odd, W2TURN achieves the same average hop count as optimal-2TURN, but this optimal result is achieved with a closed-form algorithm without the issues mentioned above. When the network radix is even, W2TURN comes very close to optimal-2TURN in terms of average hop count, within just 0.72% of optimal-2TURN on a 12×12 torus.

We also present a new weighted random oblivious routing algorithm for one-dimensional rings called WRD (Weighted Random Direction). WRD offers both optimal worst-case throughput and the minimum average hop count achievable while remaining worst-case throughput optimal for ring networks. We are unaware of any previous oblivious routing algorithms for rings that can achieve these optimality conditions.

Finally, we present detailed evaluations comparing the performance of WRD and W2TURN with the best previously known worst-case throughput optimal routing algorithms with closed-form descriptions for ring and torus topologies, respectively. In this regard, we compare W2TURN with IVAL/I2TURN in terms of average hop count, average-case throughput and throughputs under a wide range of benign and adversarial traffic patterns. Similarly, we compare WRD with RLB over the same set of performance metrics. We observe that W2TURN can achieve up to 13.4% reduction in hop count and a similar increase in throughput over I2TURN under uniform random traffic. WRD can significantly reduce average hop count over RLB by up to 25% when the network radix is even. Using cycle-accurate flit-level simulations, we also demonstrate that the reduction in hop count and the corresponding improvement in saturation throughput achieved by W2TURN and WRD can translate into significant latency reductions under moderate to high network loads over a wide range of traffic patterns.

The rest of this paper is organized as follows: Section 2.1 provides a brief background on the torus topology. Section 2.2 discusses the techniques used to evaluate the performance of a routing algorithm. Section 3 then presents our optimal routing algorithm, WRD, for the case of rings. Section 4 describes the I2TURN routing algorithm and shows its equivalence to IVAL. Section 5 describes W2TURN for the case of 2D-torus networks. Finally, Section 6 evaluates the performance of WRD and W2TURN and Section 7 concludes the paper.

2 Background

In this section, we first provide a brief overview of the torus topology, which is the network topology considered this paper. We then follow it up with some preliminaries about computing the worst-case and average-case throughput of routing algorithms.

2.1 Torus Networks: A candidate On-Chip Network topology

Torus networks can be described as k-ary n-cubes, where k is the number of nodes along each dimension and n is the number of dimensions. Rings belong to the torus family of network topologies denoted as k-ary 1-cubes and have been often used as the interconnection fabric in commercial multi-core chips [7], [11]. One and two dimensional torus topologies are well suited for on-chip networks as they map well to a planar substrate. A 2D torus has to be physically arranged in a folded form to equalize wire lengths (as shown in Figure 1) and avoid employing long wrap-around links between edge nodes. A torus is a regular topology with symmetric links, which makes it easier to load-balance traffic over all links in the network. This is different from a network with asymmetric links like a 2D mesh, where links at the center of the network are generally more heavily loaded compared to the links at the edge of the network, even under uniform traffic.

2.2 Preliminaries

The worst-case throughput of a routing algorithm is typically defined relative to the capacity of a network, which is in turn defined by the maximum channel load γ^* that a channel at the bisection of the network needs to sustain under uniform traffic. For any n-dimensional tori with radix k, using the results in [4],

$$\gamma^* = \begin{cases} \dfrac{k}{8} & k \text{ is even} \\[2ex] \dfrac{k}{8} - \dfrac{1}{8k} & k \text{ is odd} \end{cases}$$

[1] The largest 2D-torus networks solved in [19] had $k = 11$ and $k = 13$, respectively, for optimal and optimal-2TURN routing, where k is the network radix. Interconnection networks with thousands of nodes are already in use today. Although larger instances may be solved with increasing computing power, the size of interconnection networks continues to grow as well.

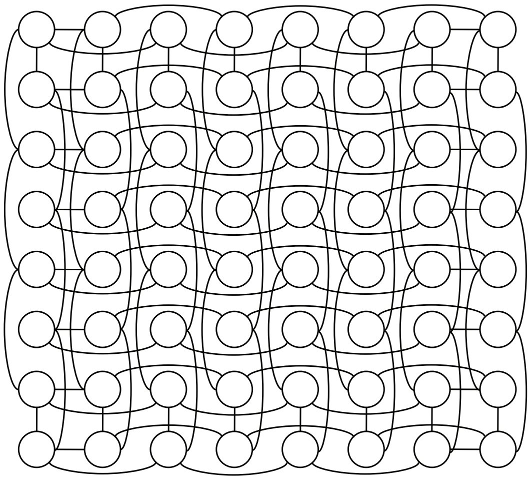


Fig. 1. Layout of a 8×8 folded torus.

The network capacity is the inverse of γ^*.

The maximum channel load[2] $\gamma(R,\Lambda)$ for a routing algorithm R and traffic matrix Λ is the expected traffic load crossing the most heavily loaded channel under R and Λ, and the worst-case channel load $\gamma_{wc}(R)$ is the maximum channel load that can be caused by any admissible traffic. Admissible traffic is defined to be any doubly sub-stochastic matrix Λ with all row and column sums bounded by 1. Suppose a network consists of N nodes, a traffic matrix $\Lambda = (\lambda_{ij})$ is an $N \times N$ matrix where λ_{ij} represents the expected traffic from node i to node j. The traffic matrix Λ is doubly sub-stochastic and hence admissible, if

$$\sum_{i=1}^{N} \lambda_{ij} \leq 1, \forall j \text{ and } \sum_{j=1}^{N} \lambda_{ij} \leq 1, \forall i$$

and it is said to be doubly stochastic if

$$\sum_{i=1}^{N} \lambda_{ij} = 1, \forall j \text{ and } \sum_{j=1}^{N} \lambda_{ij} = 1, \forall i$$

As shown in [18], the worst-case channel load for a routing algorithm R over all admissible traffic matrices can be found by solving a derived maximum weighted matching problem for each channel in the network. The worst-case saturation throughput for a routing algorithm R is the inverse of the worst-case channel load. Further, the normalized worst-case saturation throughput, $\Theta_{wc}(R)$, is defined as the worst-case saturation throughput normalized to the network capacity:

$$\Theta_{wc}(R) = \frac{\gamma^*}{\gamma_{wc}(R)} \tag{1}$$

Valiant routing (VAL) [20] is known to be worst-case throughput optimal with $\Theta_{wc}(\text{VAL}) = 0.5$. Therefore, to show that a routing algorithm $\hat{R}$ is worst-case throughput optimal for a torus network with radix k, it is sufficient to show that the maximum channel load under the worst-case traffic pattern identified using maximum weighted matching is at most

$$\gamma_{wc}(\hat{R}) = \frac{\gamma^*}{0.5} = \begin{cases} \dfrac{k}{4} & k \text{ is even} \\ \dfrac{k}{4} - \dfrac{1}{4k} & k \text{ is odd} \end{cases} \tag{2}$$

which could be demonstrated analytically or empirically over a wide range of network sizes.

In order to show that a routing algorithm $\hat{R}$ provides the minimum average hop count achievable while remaining worst-case throughput optimal, we use the multicommodity flow formulation proposed by Towles et al. [19] to derive worst-case throughput optimal routings with minimum hop count over a range of network sizes. We then compare the average hop counts of $\hat{R}$ with those of the optimal routing solutions over the same range of network sizes.

Finally, along with worst-case throughput, average-case throughput is also an important performance metric for routing algorithms. Using the methodology used in [12], [18], the average-case throughput of a routing algorithm R can be computed by averaging the throughput over T, a large set of random traffic patterns:

$$\Theta_{avg}(R) = \frac{1}{|T|} \sum_{\Lambda \in T} \left(\frac{\gamma(R,\Lambda)}{\gamma^*} \right)^{-1} \tag{3}$$

3 Optimal Routing on Rings with WRD

In this section, we consider the optimal oblivious routing problem for one-dimensional rings. Our proposed algorithm called WRD works as follows. Suppose source s sends traffic to destination d, the minimal distance around the loop is given as:

$$\Delta(s,d) = \min(|s-d|, k - |s-d|) \tag{4}$$

where k is the number of nodes in the ring. When there is no confusion, we will simply refer to $\Delta(s,d)$ as Δ. We consider two cases: first when k is odd, then when k is even.

For odd k, WRD routes traffic in the minimal and non-minimal directions with the following probabilities:

$$P_{odd} = \begin{cases} \dfrac{k-\Delta}{k} & \text{in minimal direction} \\ \dfrac{\Delta}{k} & \text{in non-minimal direction} \end{cases} \tag{5}$$

This is precisely what the RLB algorithm [15] does in the case of rings, and this has already been shown to be worst-case throughput optimal. Given the above routing probabilities and the fact that the minimal direction has Δ hops while the non-minimal direction has $k - \Delta$ hops, the average hop count for the odd-radix case can be computed as follows:

$$H_{odd}(\text{WRD}) = E\left[\frac{2\Delta(k-\Delta)}{k}\right] = \frac{k}{3} - \frac{1}{3k} \tag{6}$$

where $E[.]$ denotes the expectation operator over all possible destination nodes for a given source.

[2] Channels and links are used interchangeably in this paper.

For even k, WRD routes traffic using the following probabilities when $\Delta > 0$ and $k > 2$:

$$P_{even} = \begin{cases} \dfrac{k-\Delta-1}{k-2} & \text{in minimal direction} \\[2ex] \dfrac{\Delta-1}{k-2} & \text{in non-minimal direction} \end{cases} \tag{7}$$

When $\Delta = 0$ (i.e., $s = d$), no routing is necessary. When $k = 2$ and $\Delta > 0$, WRD routes in both directions at equal distance with equal probability, which is the same as RLB. Note that the traffic distribution in the minimal and non-minimal directions for the even radix case when $k > 2$ is different from RLB. The average hop count of this route distribution can be computed as follows:

$$H_{even}(\text{WRD}) =$$
$$E\left[\left(\frac{\Delta(k-\Delta-1)}{k-2} + \frac{(k-\Delta)(\Delta-1)}{k-2}\right)\middle|\, \Delta > 0\right] P(\Delta > 0)$$
$$= \frac{k}{3} \times \left(\frac{k-1}{k}\right) = \frac{k}{3} - \frac{1}{3}$$

WRD achieves lower average hop count than RLB when the network radix k is even and when $k > 2$ since

$$\frac{k}{3} - \frac{1}{3} < \frac{k}{3} - \frac{1}{3k} \qquad \forall k > 1$$

We next show that WRD indeed achieves optimal worst-case throughput for all network radices.

Claim 3.1: WRD is worst-case throughput optimal.

Proof: For the odd-radix case, WRD is the same as RLB, which has already been shown to be worst-case throughput optimal in [14]. For the even-radix case, we use the same proof methodology that was used in [14] for showing RLB is worst-case throughput optimal on a ring. The proof uses the method in [18] to identify a worst-case traffic pattern for WRD. We then verify that the maximum channel load using WRD on this worst-case traffic pattern is indeed at most $k/4$, as shown necessary and sufficient for worst-case throughput optimality for even k in Equation 2 of Section 2. Using the technique described in [18], a worst-case traffic pattern for WRD is Tornado traffic [4]. Suppose under the tornado traffic pattern, each node sends all its traffic to a node $k/2-1$ hops away in the clockwise direction ($\Delta = k/2 - 1$), the corresponding load on every clockwise channel is given by the sum of the contributions from the $k/2-1$ nodes preceding the channel. Using the probability for routing in the minimal direction from Equation 7, each of these $k/2-1$ preceding nodes route in the clockwise (minimal) direction with a probability of $k/2(k-2)$. Therefore, the maximum channel load on a clockwise channel, $\gamma_{clk}(\text{WRD})$, is given as:

$$\gamma_{clk}(\text{WRD}) = \frac{k/2}{k-2} \times \left(\frac{k}{2} - 1\right) = \frac{k}{4} \quad \forall k > 2$$

Similarly, the maximum channel load on a counter-clockwise channel can be computed as the sum of the contributions from $(k/2+1)$ nodes preceding the channel. Using the probability of routing in the non-minimal direction from Equation 7, each of these nodes contribute $(k-4)/2(k-2)$ of their traffic, and the sum of their contributions, $\gamma_{cclk}(\text{WRD})$, is given as:

$$\gamma_{cclk}(\text{WRD}) = \frac{(k+2)(k-4)}{4(k-2)} < \frac{k}{4} \quad \forall k > 2$$

The worst-case channel load with WRD for even k is then given as:

$$\gamma_{wc}(\text{WRD}) = max(\gamma_{clk}(\text{WRD}), \gamma_{cclk}(\text{WRD})) = \frac{k}{4} \quad \forall k > 2$$

Hence, WRD is worst-case throughput optimal.

□

Claim 3.2: WRD achieves the minimum average hop count achievable while remaining worst-case throughput optimal.

Proof: Using the methodology discussed in Section 2, we have verified this claim by comparing the average hop counts of WRD with those of optimal routing, which were computed using a multicommodity flow formulation [19].

□

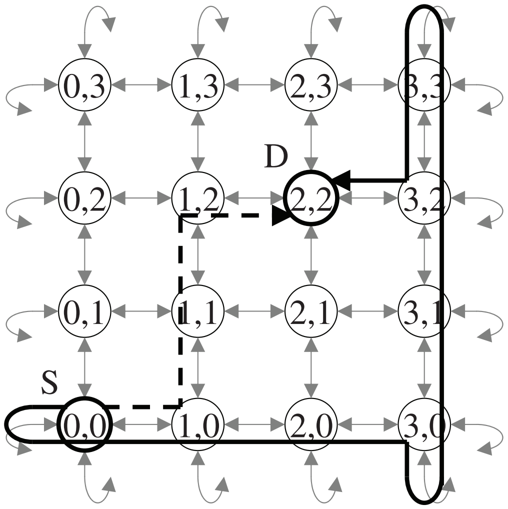


Fig. 2. Routing with 2-turn paths. Both the solid and dotted lines represent different 2-turn paths between (0,0) and (2,2).

4 The I2TURN Routing Algorithm

In this section, we describe the I2TURN routing algorithm for two-dimensional torus networks. As the name suggests, I2TURN considers routing paths with at most two turns, as shown in Figure 2. The dashed line shows an XYX 2-turn path that starts from $(0,0)$, makes the first turn at $(1,0)$, makes the second turn at $(1,2)$, and goes finally to $(2,2)$. The solid line shows an alternative XYX 2-turn path that starts from $(0,0)$, loops left and around to first turn at $(3,0)$, loops down and around to $(3,2)$, and goes finally to $(2,2)$.

The idea of using 2-turn paths was proposed in [19] in their optimal 2TURN algorithm. However, the proposed 2TURN algorithm does not have a closed-form algorithmic description. It only has a closed-form description of the possible paths that a packet may take through the network, but requires solving a separate linear program to determine the path distribution for each given network radix. The size of these linear programs grow quickly making them difficult to scale to large networks. The I2TURN algorithm and the W2TURN algorithm, which is described next, have closed-form descriptions and can be easily extended to arbitrarily large networks.

We first consider a version of I2TURN that only uses XYX 2-turn paths. Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination. The three segments of the XYX 2-turn paths are generated as follows:

1) X-segment: Choose at uniform random an X position $x^* \in [0, k-1]$ and route in the X dimension from (x_1, y_1) to (x^*, y_1) in the minimal direction.
2) Y-segment: Next, route in the Y dimension from (x^*, y_1) to (x^*, y_2) in the minimal and non-minimal directions with the following probabilities:
$$P = \begin{cases} \dfrac{k - \Delta y}{k} & \text{in minimal direction} \\ \\ \dfrac{\Delta y}{k} & \text{in non-minimal direction} \end{cases}$$
Here Δy is the minimum distance in Y from (x^*, y_1) to (x^*, y_2) computed using Equation 4. The routing along the Y dimension is identical to RLB and WRD for the odd-ring case (Equation 5).
3) X-segment: Finally, route in the X dimension from (x^*, y_2) to (x_2, y_2) in the minimal direction.

There are several degenerate cases. When $x^* = x_1$, there is no need to route on the first X-segment. Similarly, when $x^* = x_2$, then there is no need to route on the last X-segment. When $y_1 = y_2$, packets only need to be routed along the X dimension with no turns and any loop formed as a result of two-phase minimal routing on the X ring can be removed. In this case, the packet is routed with probability $(k - \Delta x)/k$ in the minimal direction, where Δx is the minimum distance in X between x_1 and x_2, and with probability $(\Delta x)/k$ in the non-minimal direction. Finally, when the source and destination are the same, no routing is necessary. Unless otherwise noted, we will regard these “degenerate” cases as “2-turn” paths as well.

4.1 Equivalence to IVAL

Claim 4.1: I2TURN routes packets using the same statistical distribution of paths as IVAL.

Proof: IVAL routes packets from the source to the destination via a random intermediate node, using minimal XY and YX routing in the two phases. IVAL identifies and removes any loop formed at runtime, resulting in the construction of loop-free 2-turn XYX paths. To show that IVAL and I2TURN route packets along the same paths with the same probabilities, we consider three cases. Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination. In the first case, when the source and destination are the same, no routing occurs in both IVAL and I2TURN.

In the second case, when $y_1 \neq y_2$, I2TURN chooses at uniform random the intermediate X position x^*. The routing in the X dimension from x_1 to x^* and x^* to x_2 are each unique in the corresponding minimal directions. For the Y segment, the packet is routed in either the minimal or non-minimal Y direction.

In IVAL, there are k^2 possible intermediate nodes (x_i, y_i) that can be chosen at uniform random. It follows that the probability of choosing an intermediate node with $x_i = x^*$ is uniformly $1/k$. As in I2TURN, the routing for IVAL in the X dimension from x_1 to x^* and x^* to x_2 are in the same corresponding minimal directions. Since IVAL paths are loop-free after runtime loop removal, we are guaranteed that the path will be a 2-turn XYX path where the packet will be routed in the Y dimension at X position x^* in either the minimal or non-minimal direction. Since routing in the X dimension is equivalent, we can reduce the proof to equivalent path selection on the Y ring.

For I2TURN, there are two possible acyclic paths on the Y ring – a minimal path in the short direction with a distance of Δy that is chosen with probability $(k - \Delta y)/k$, and a non-minimal path in the long direction with a distance of $(k - \Delta y)$ that is chosen with probability $\Delta y/k$. For IVAL, any of the k nodes on the Y ring can be chosen as the intermediate node. Since the definition of minimal and non-minimal paths is relative to y_1 and y_2, it suffices to consider the case where $y_1 = 0$ and $y_2 = \Delta y$, effectively shifting the origin to the coordinates of the source. By definition of minimal distance, $\Delta y \leq k/2$.

In the subsequent discussion, minimal and non-minimal directions (paths) refer to the short and long paths, respectively, between the source and the destination. Let i be the Y coordinate of the intermediate node chosen by IVAL. There are two situations when a packet is guaranteed to be routed along the minimal path after loop removal: when $0 \leq i < k/2$ or when $(i - \Delta y) > k/2$. There are $\lceil k/2 \rceil$ possible intermediate nodes that satisfy $0 \leq i < k/2$, and there are $\lceil k/2 \rceil - \Delta y - 1$ possible intermediate nodes that satisfy $(i - \Delta y) > k/2$, with a combined total of $\lceil k/2 \rceil + \lceil k/2 \rceil - \Delta y - 1$ intermediate nodes that will always result in IVAL routing along the minimal path after loop removal. When k is even and $i = k/2$, the distance between $y_1 = 0$ and i will be the same in both directions, giving it a 50% chance that a packet will be routed in the minimal direction after loop removal. Similarly, when k is even and $i - \Delta y = k/2$, the distance between i and $y_2 = \Delta y$ will be the same in both directions, again giving it a 50% chance that a packet will be routed in the minimal direction following loop removal. Assuming all intermediate nodes are chosen with equal probability, the total probability of choosing the minimal path along the Y ring in IVAL is given by $(\lceil k/2 \rceil + \lceil k/2 \rceil - \Delta y - 1)/k$ when k is odd and $(\lceil k/2 \rceil + \lceil k/2 \rceil - \Delta y)/k$ when k is even. When k is odd, $\lceil k/2 \rceil + \lceil k/2 \rceil = k + 1$. When k is even, $\lceil k/2 \rceil + \lceil k/2 \rceil = k$. Therefore, the probability of choosing the minimal path is equal to $(k - \Delta y)/k$ when k is either even or odd.

Finally, in the third case, when $y_1 = y_2$, but $x_1 \neq x_2$, the proof reduces to showing that IVAL will choose the same loop-free path on the X dimension as I2TURN. The same analysis presented above for the Y ring can be applied to the X dimension to show that both IVAL and I2TURN will choose the minimal (and non-minimal) paths with the same probability.

□

Claim 4.2: I2TURN is worst-case throughput optimal.

Proof: Follows from its equivalence to IVAL.

□

The above discussion applies to I2TURN with XYX 2-turn paths. I2TURN can be equivalently defined using YXY 2-turn paths by swapping dimensions. Also, I2TURN can be implemented using a randomization of XYX and YXY paths. When XYX and YXY routings are used with equal probability, I2TURN routing is symmetric and balances load equally between the X and Y channels.

4.2 Average Hop Count for I2TURN

The I2TURN routing algorithm for torus networks is described in terms of weighted random selection of 2-turn paths. Although I2TURN is equivalent to IVAL, its description of 2-turn paths based on probabilities makes it easier to derive an analytical expression for its average hop count. The average hop count for I2TURN can be expressed as the sum of the average number of hops for the three routing segments of the 2-turn paths: minimal routing on the first and last X segments, and weighted random routing on the middle Y segment. Let H_{min} denote the average hop count for minimal routing on a ring [4].

$$H_{min} = \begin{cases} \dfrac{k}{4} & k \text{ is even} \\ \\ \dfrac{k}{4} - \dfrac{1}{4k} & k \text{ is odd} \end{cases}$$

Since the routing on the middle Y segment is same as the WRD algorithm for the odd-ring case, the average hop count for this segment can be computed using Equation 6. As analyzed in the proof of Claim 4.1, we have to consider 1-in-k cases when the source and destination have the same Y coordinate. For these cases, loops formed on the X ring can be removed and the routing along the X ring becomes identical to the weighted random routing used in the Y dimension. Taken together, the average hop count for I2TURN is given as follows:

$$H_{avg}(\text{I2TURN}) = H_x(\text{I2TURN}) + H_y(\text{I2TURN})$$

$$H_x(\text{I2TURN}) = \left(1 - \frac{1}{k}\right) 2H_{min} + \left(\frac{1}{k}\right)\left(\frac{k}{3} - \frac{1}{3k}\right)$$

$$H_y(\text{I2TURN}) = \frac{k}{3} - \frac{1}{3k}$$

$$H_{avg}(\text{I2TURN}) = 2\left(1 - \frac{1}{k}\right) H_{min} + \left(1 + \frac{1}{k}\right)\left(\frac{k}{3} - \frac{1}{3k}\right)$$

It must be noted here that I2TURN routing described using YXY routing paths or a randomization of XYX and YXY routings will have the same average hop count as computed above.

5 The W2TURN Routing Algorithm

Next, we describe the W2TURN routing algorithm for 2D-torus networks. Like I2TURN, W2TURN also considers different routing paths with at most two turns, as shown in Figure 2. However, the probabilities with which the 2TURN paths are chosen are different, giving W2TURN an edge over I2TURN in terms of average hop-count. The W2TURN algorithm was developed in part from examining the path distribution derived out of the optimal 2TURN formulation. W2TURN was also based on the intuition gained from studying optimal routing for the 1D ring case (WRD) and the I2TURN algorithm. The high-level idea while developing W2TURN was to distribute the traffic equally across the 1-dimensional rings (Y rings in the case of XYX routing and X rings in the case of YXY routing) and use our knowledge of routing optimally on each 1D ring. We further improvised on this high-level idea to match the traffic distribution from the optimal 2TURN solution.

In the remainder of this section, we present the W2TURN routing algorithm and analyze its worst-case throughput and average hop count. Like WRD, we consider the odd-k and even-k cases separately.

5.1 When k is odd

We first describe the weighted random selection of XYX routing paths in W2TURN. $\Delta(x_1, x_2)$ refers to the minimum distance on the X-ring between nodes having X-coordinates x_1 and x_2 and the same Y-coordinate. $\Delta(y_1, y_2)$ refers to the minimum distance on the Y-ring between nodes having Y-coordinates y_1 and y_2 and the same X-coordinate. The definition of minimum distance along a dimension follows from Equation 4.

Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination, the three segments of the XYX 2-turn paths are generated as follows:

1) X-segment: Choose at uniform random an X position $x^* \in [0, k-1]$. Then, consider two cases:
    a) Route in the minimal direction from (x_1, y_1) to (x^*, y_1) if *any* of the following conditions are satisfied:
        - $\Delta(x_1, x^*) < \lfloor \frac{k}{2} \rfloor$,
        - (x_2, y_1) is not on the minimal path from (x_1, y_1) to (x^*, y_1), or
        - $\Delta(x_1, x_2) = \lfloor \frac{k}{2} \rfloor$.
    b) Otherwise, route from (x_1, y_1) to (x^*, y_1) on the X ring with the following probabilities:

$$P = \begin{cases} \dfrac{k - \Delta(x_1, x_2)}{k} & \text{in minimal direction} \\ \dfrac{\Delta(x_1, x_2)}{k} & \text{in non-minimal direction} \end{cases}$$

    where minimal and non-minimal directions refer to the short and long paths, respectively, on the X ring from (x_1, y_1) to (x^*, y_1).

2) Y-segment: Next, route in the Y dimension from (x^*, y_1) to (x^*, y_2). We again consider two cases:
    a) Route in the minimal direction from (x^*, y_1) to (x^*, y_2) if *all* of the following conditions are satisfied:
        - $x_1 \neq x_2$,
        - $\Delta(y_1, y_2) < \lfloor \frac{k}{2} \rfloor$, and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

- $(x^* = x_1$ or $x^* = x_2)$.

b) Otherwise, route from (x^*, y_1) to (x^*, y_2) on the Y ring using WRD.

3) X-segment: Finally, route in the X dimension from (x^*, y_2) to (x_2, y_2), again with two cases:

a) Route in the minimal direction from (x^*, y_2) to (x_2, y_2) if *any* of the following conditions are satisfied:

- $\Delta(x^*, x_2) < \lfloor \frac{k}{2} \rfloor$,
- (x_1, y_2) is not on the minimal path from (x^*, y_2) to (x_2, y_2), or
- $\Delta(x_1, x_2) = \lfloor \frac{k}{2} \rfloor$.

b) Otherwise, route from (x^*, y_2) to (x_2, y_2) on the X ring with the following probabilities:

$$P = \begin{cases} \dfrac{k - \Delta(x_1, x_2)}{k} & \text{in minimal direction} \\ \dfrac{\Delta(x_1, x_2)}{k} & \text{in non-minimal direction} \end{cases}$$

where minimal and non-minimal directions refer to the short and long paths, respectively, on the X ring from (x^*, y_2) to (x_2, y_2).

There are several degenerate cases. When $x^* = x_1$, there is no need to route on the first X-segment. Similarly, when $x^* = x_2$, there is no need to route on the last X-segment. When $y_1 = y_2$, packets only need to be routed along the X dimension with no turns using WRD. Finally, when the source and destination are the same, no routing is necessary.

Given the above XYX routing algorithm, the version with YXY routing can be equivalently defined by swapping dimensions. To achieve worst-case throughput optimality for the odd k case, W2TURN requires using both XYX and YXY routing with equal probabilities.

5.2 When k is even

For the even-radix case, we first describe the weighted random selection of XYX routing paths. Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination, the three segments of the XYX 2-turn paths are generated as follows:

1) X-segment: First, choose at uniform random an X position $x^* \in [0, k-1]$. Then route minimally from (x_1, y_1) to (x^*, y_1). If the number of hops in both directions are equal, choose the direction that does not contain the node (x_2, y_1). If $x^* = x_2$, and the number of hops in both directions are equal, choose either direction with equal probability.
2) Y-segment: Route from (x^*, y_1) to (x^*, y_2) using WRD.
3) X-segment: Route minimally from (x^*, y_2) to (x_2, y_2). If the number of hops in both directions are equal, choose the direction that does not contain the node (x_1, y_2). If $x^* = x_1$, and the number of hops in both directions are equal, choose either direction with equal probability.

There are several degenerate cases. When $x^* = x_1$, there is no need to route on the first X-segment. Similarly, when $x^* = x_2$, then there is no need to route on the last X-segment. When $y_1 = y_2$, the packet only needs to be routed along the X dimension using the same algorithm described above. For this case, any loop formed as a result of an overlap in the routing paths of the two X segments should be removed. Following loop removal, the probabilities of routing in the minimal and non-minimal directions are given as follows when $\Delta(x_1, x_2) < k/2$ and $\Delta(x_1, x_2) > 0$:

$$P = \begin{cases} \dfrac{k - \Delta(x_1, x_2) - 1}{k} & \text{in minimal direction} \\ \dfrac{\Delta(x_1, x_2) + 1}{k} & \text{in non-minimal direction} \end{cases}$$

When $\Delta(x_1, x_2) = k/2$ a packet is routed in either direction with equal probability. Finally, when the source and destination are the same, no routing is necessary.

The YXY routing paths can be equivalently defined by swapping dimensions. To achieve worst-case throughput optimality for the even k case, W2TURN requires interpolating over the following four routings with the corresponding specified probabilities:

- XYX routing with probability $\frac{k}{2(k+1)}$
- YXY routing with probability $\frac{k}{2(k+1)}$
- Dimension-ordered XY routing with probability $\frac{1}{2(k+1)}$
- Dimension-ordered YX routing with probability $\frac{1}{2(k+1)}$

5.3 Throughput Optimality

In this section, we show that W2TURN is indeed worst-case throughput optimal.

Claim 5.1: W2TURN is worst-case throughput optimal.

Proof: We again use the same proof methodology that was used in [14], which uses the method in [18] for identifying a worst-case traffic pattern. We then show that the maximum channel load using W2TURN on this worst-case traffic pattern is indeed at most $k/4$ when k is even and at most $(k/4 - 1/(4k))$ when k is odd, as shown necessary and sufficient in Equation 2. For a network with radix k, a worst-case traffic pattern for W2TURN is shown as follows:

$$\text{Node}(x, y) \text{ sends packets to } (x + \lfloor k/2 \rfloor, y + \lfloor k/2 \rfloor)$$

The above traffic pattern is same as Tornado traffic [4] when k is odd. Using worst-case load analysis, the maximum channel load for the worst-case traffic pattern was found to be the same as Equation 2 for all values of k analyzed[3]. □

5.4 Latency Analysis

In this section, we express the average hop count of W2TURN in terms of the average hop count expressions derived for WRD and the network radix k. Later, in Section 6 we show that W2TURN indeed outperforms I2TURN in average hop count. We treat the even and odd k cases separately.

[3] Maximum channel load was verified for k up to 40.

5.4.1 Odd k

The average hop count of XYX routing is presented in this section. YXY routing will have identical hop count due to symmetry. The average hop count for XYX routing is given by the sum of the average hop counts of the two X segments and the Y segment.

We first consider the average hop count for the X segments. Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination, when $y_1 = y_2$, which is one of the degenerate cases, WRD is used to route on the X ring. The average hop count for this case is equal to the average hop count of WRD with odd radix.

$$H_{case1} = \frac{k}{3} - \frac{1}{3k}$$

When $y_1 \neq y_2$, minimal routing is used on both X segments if either of the three conditions stated in Section 5.1 are satisfied. In this case, we first compute the probability of routing non-minimally in the X dimension (when all three conditions are not satisfied) and multiply it by the extra hops added (over minimal) as a result of non-minimal routing. We denote this penalty paid over minimal routing by routing non-minimally as $H_{penalty}$.

$$H_{penalty} = \frac{2}{k}\left[\frac{1}{k}\sum_{\Delta(x_1,x_2)=0}^{\lfloor\frac{k}{2}\rfloor-1}\frac{\Delta(x_1,x_2)}{k}\right]$$

The latency of each X segment is then given as:

$$H_{case2} = H_{min} + H_{penalty}$$

H_{case1} denotes the combined latency of the two X segments when $y_1 = y_2$ and H_{case2} denotes the average latency of each X segment when $y_1 \neq y_2$. Hence, the average latency of the two X segments can be expressed as follows:

$$H_x(\text{XYX}) = \frac{1}{k}H_{case1} + \frac{(k-1)}{k}(2 \times H_{case2}) \tag{8}$$

Next, we consider the average hop count of the Y segment. In the Y dimension, a packet is routed minimally if all the conditions stated in Section 5.1 are satisfied. Else, it is routed using WRD. The probability that the first condition is true, i.e. $x_1 \neq x_2$ is $(k-1)/k$ and the probability that the third condition is true, i.e. $x^* = x_1$ or $x^* = x_2$, given $x_1 \neq x_2$ is $2/k$. Using these results, the average hop count savings by routing minimally in the Y dimension instead of using WRD when all three conditions are true is given as follows:

$$H_{savings} = \frac{2}{k}\frac{(k-1)}{k}\left[\frac{2}{k}\sum_{\Delta(y_1,y_2)=0}^{\lfloor\frac{k}{2}\rfloor-1}\frac{\Delta(y_1,y_2)}{k}(k-2\Delta(y_1,y_2))\right]$$

The average latency in the Y dimension can then be expressed as:

$$\begin{aligned} H_y(\text{XYX}) &= H_{odd}(\text{WRD}) - H_{savings} \\ &= \frac{k}{3} - \frac{1}{3k} - H_{savings} \end{aligned} \tag{9}$$

From Equations 8 and 9 and using the fact that YXY routing will have the same average hop count as XYX routing,

$$H_{odd}(\text{W2TURN}) = H_x(\text{XYX}) + H_y(\text{XYX})$$

5.4.2 Even k

W2TURN routing for an even network radix is an interpolation of four different routings - XYX, YXY, XY and YX. We first consider the average hop count for the XYX routing paths. When $y_1 = y_2$, no routing is necessary along the Y dimension and there is a possibility of loop removal on the X ring after two phases of X routing. Following loop removal, using the probabilities described in Section 5.2 the combined average hop count of the two X segments is given as follows:

$$H_{case1} = \frac{1}{2} + \frac{k}{3} - \frac{4}{3k}$$

For the case when $y_1 \neq y_2$, a packet is routed in the minimal direction on both the X segments. Hence, the average hop count for each X segment in this case is given as:

$$H_{case2} = H_{min} = \frac{k}{4}$$

Therefore, the average hop count for the two X segments of the XYX routing paths can be expressed as:

$$H_x(\text{XYX}) = \frac{1}{k}H_{case1} + \frac{(k-1)}{k}(2 \times H_{case2}) \tag{10}$$

Since WRD is used along the Y dimension, the average hop count along this dimension is given as:

$$H_y(\text{XYX}) = H_{even}(\text{WRD}) = \frac{(k-1)}{3} \tag{11}$$

From Equations 10 and 11,

$$H(\text{XYX}) = H_x(\text{XYX}) + H_y(\text{XYX})$$

The hop count for YXY routing is identical to XYX routing due to symmetry. The average hop counts for minimal XY and YX routings are given as:

$$H(\text{XY}) = H(\text{YX}) = \frac{k}{2}$$

The average hop count of W2TURN when the network radix is even, H_{even}(W2TURN), is given by the weighted mean of the average hop counts of XYX, YXY, XY and YX routings with weights $k/2(k+1)$, $k/2(k+1)$, $1/2(k+1)$, and $1/2(k+1)$, respectively.

5.5 Deadlock-Free Implementation

W2TURN uses the same set of 2-turn paths as the optimal 2TURN formulation proposed in [19]. When k is odd, W2TURN distributes traffic over these 2-turn paths with the same probabilities as optimal 2TURN. When k is even, W2TURN also uses the same set of 2-turn paths, although with different probabilities. Since W2TURN uses the same set of 2-turn paths as optimal-2TURN, a deadlock-free implementation requires exactly the same number of virtual channels, which has been shown to be four virtual channels per physical channel (the same requirement for

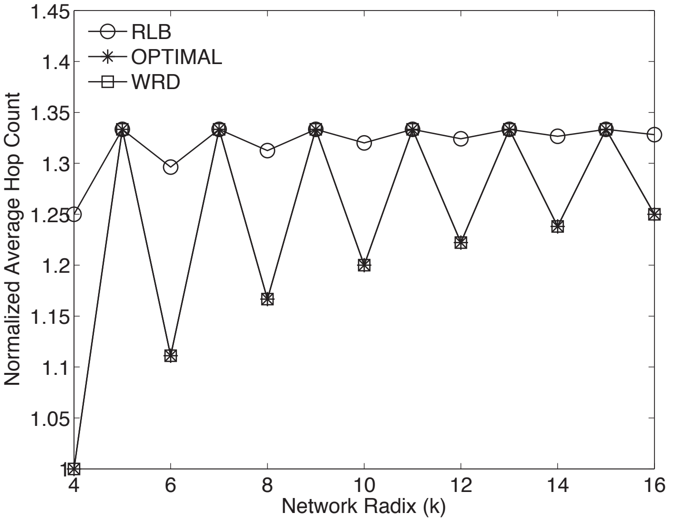


Fig. 3. Comparison of the average hop counts of WRD with RLB [15] and optimal routing [19] on ring networks with different radices.

VAL [20]). In particular, W2TURN can be made deadlock-free by incrementing a packet's virtual channel set after each turn from the Y to the X dimension. Since any 2-turn path has at most one turn from Y to X, this approach requires two virtual channel sets. Each set requires two virtual channels to resolve intra-dimension deadlocks, therefore requiring four virtual channels per physical channel in total. As stated earlier, the key advantage of W2TURN over optimal-2TURN is that W2TURN provides a closed-form algorithm that can achieve comparable performance with the same simple deadlock-free implementation. Further, the W2TURN algorithm only involves simple conditional checks and probability calculations that can be readily implemented in parallel.

6 Performance Evaluation

In this section, we evaluate the performance of WRD and W2TURN in terms of latency and throughput. Both WRD and W2TURN have already been shown to be worst-case throughput optimal for one-dimensional and two-dimensional torus topologies, respectively. This section focuses on other throughput metrics like average-case throughput and throughputs under different benign and adversarial traffic patterns.

6.1 Evaluation of WRD

6.1.1 Hop count analysis

Figure 3 compares the average hop count of WRD with RLB [15] and optimal routing [19]. All three routing algorithms considered achieve optimal worst-case throughput for ring networks. WRD and RLB have closed-form algorithmic descriptions, while the optimal routing results were obtained using the multicommodity flow formulation proposed in [19]. The average hop counts are normalized to minimal routing. As shown in Figure 3, WRD achieves the same hop-count as optimal routing for all network radices.

When k is odd, WRD and RLB are equivalent and therefore have the same hop count, as shown in Figure 3. When k is even, WRD outperforms RLB because WRD routes in the minimal direction more often. In this case, WRD can achieve up to 25% reduction in average-hop count over RLB.

6.1.2 Throughput evaluation

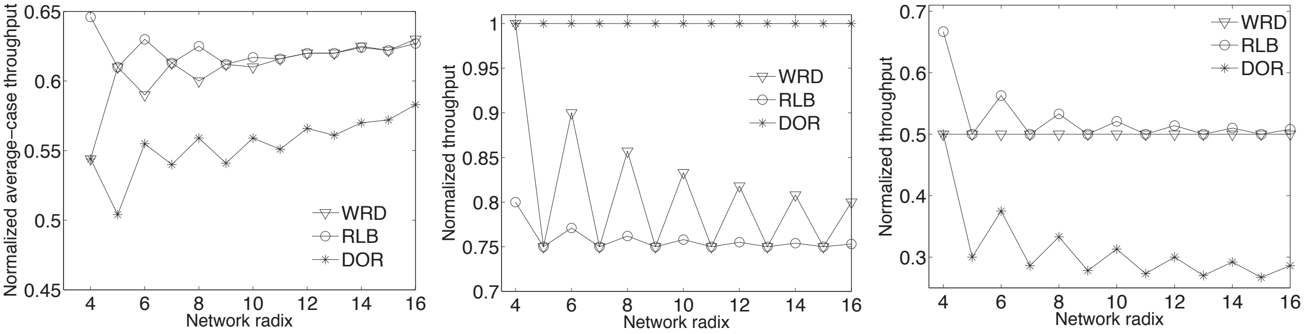
WRD is optimal in terms of worst-case throughput, but is not minimal in terms of latency as it employs non-minimal routing paths. Here, we compare the throughput of WRD with two other routing algorithms for rings with closed-form descriptions, namely, RLB and DOR. RLB also achieves the same optimal worst-case throughput as WRD but has a higher average hop count when the network radix is even. DOR, on the other hand, achieves minimal hop count while sacrificing worst-case throughput. The throughput metrics used in this section are average-case throughput and throughput under uniform random traffic and tornado traffic [4]. Uniform random traffic is a benign traffic pattern which is easy to route since it is inherently load-balanced. In contrast, tornado traffic is an adversarial pattern, which is also a worst-case traffic pattern for both DOR and WRD.

The throughput analysis is carried out in two steps. Initially, we perform a simplified throughput analysis for a range of network sizes from 4 nodes to 16 nodes. This analysis assumes ideal single-cycle routers with infinite buffers. In the next section, we back these results with more realistic flit-level simulations for an 8-node ring topology. All throughput results presented subsequently are normalized to the network capacity (refer Section 2).

Figure 4(a) compares the average-case throughput of WRD with RLB and DOR. As discussed in Section 2, average-case throughput of a routing algorithm is computed by averaging the throughput over a large set of randomly generated permutation traffic patterns[4]. In this paper, we average the throughput over a set of 10,000 randomly generated permutation traffic matrices. WRD is identical to RLB when the network radix is odd and this is reflected in the average-case throughput results as well. RLB is slightly better than WRD for even radices, when the network radix is low. For larger topologies (beyond 10 nodes), the difference in throughputs is negligible and by radix 16, WRD even slightly outperforms RLB. Both WRD and RLB outperform DOR in terms of average-case throughput. On average, over all network radices evaluated, WRD outperforms DOR by around 9.8% in average-case throughput.

As shown in Figure 4(b), DOR is the best routing algorithm under uniform random traffic, where traffic from a node is equally distributed to all nodes in the network. Non-minimal routing in WRD and RLB prevent these algorithms from sustaining throughputs as high as DOR when the traffic is uniform. However, for even network radices, the lower average hop count of WRD compared to RLB directly translates into a corresponding gain in throughput. For even radices, the throughput of WRD is 12.3% higher than RLB on average under uniform traffic.

[4] A permutation traffic matrix is one in which a source sends all its traffic to a single destination, obtained from a one-to-one mapping between pairs of nodes in the network [4].



(a) Average-case throughput

(b) Uniform random

(c) Tornado

Fig. 4. Throughput evaluation of WRD for different network sizes.

Finally, Figure 4(c) proves that the throughput of a minimal routing algorithm like DOR can degrade tremendously under an adversarial traffic pattern like tornado traffic. Therefore guaranteeing a level of worst-case performance using optimal routing algorithms like WRD is important. As discussed in Section 3, tornado traffic is a worst-case traffic pattern for WRD, which sustains optimal worst-case throughput of half the network capacity under the tornado traffic pattern. On an average over all radices, this is 64% higher than what DOR can sustain under the same traffic pattern. RLB performs comparably to WRD and its throughput converges to half the network capacity for high radices.

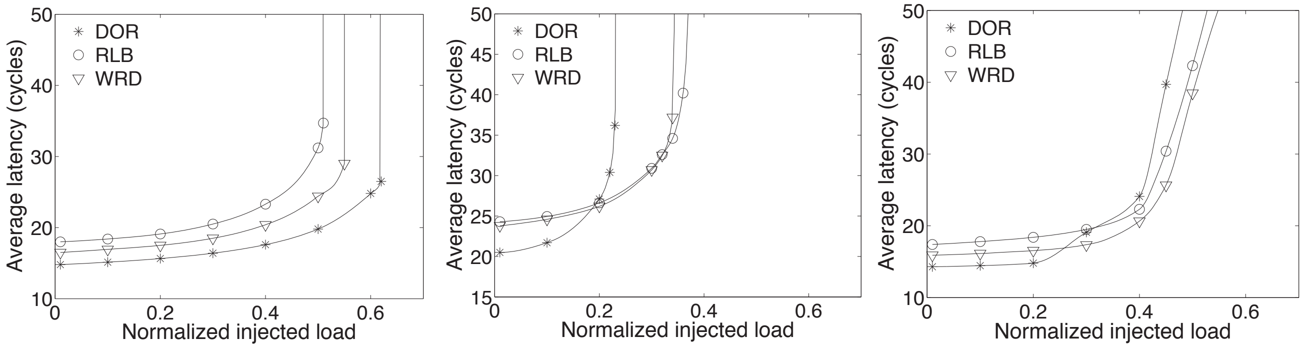
6.1.3 Flit-level simulations

The results obtained using ideal throughput analysis represent upper bounds to the actual achievable throughput because it assumes ideal single-cycle routers with infinite buffers and ignores issues like flow control and contention in switches. Hence, we use cycle-accurate flit-level simulations to gain more realistic insights into the performance of the routing algorithms. We restrict ourselves to 8-node and 16-node ring topologies. A topology with odd radix is not considered because WRD is identical to RLB for odd radices.

In our evaluations, we specifically consider the application of the proposed routing algorithms in the context of on-chip interconnection networks. We modified the PopNet [13] on-chip network simulator to perform flit-level simulations. PopNet models a typical input-buffered VC router with five pipelined stages. Route computation is performed in the first stage followed by VC allocation, switch arbitration, switch traversal and link traversal. The head flit of a packet proceeds through all five stages while the body and tail flits bypass the first two stages and inherit the output port and output VC reserved by the head flit. Credit-based flit-level flow control is used between adjacent routers. We assume 8 virtual channels (VCs) per physical channel, each 5 flits deep. For ring topologies, two virtual channels are sufficient to avoid intra-dimension deadlocks. However, it is well known that VCs improve the throughput of any routing algorithm by reducing head-of-line blocking and enabling better statistical multiplexing of flits. So, having a reasonably large number of VCs lets us compare the best performance of all routing algorithms. 3-flit packets are injected into the network and we use PopNet to evaluate the average routing delays under different injection loads. For each simulation, we ran the simulator for 500,000 cycles. The latency of a packet is measured as the delay between the time the head flit is injected into the network and the time the tail flit is consumed at the destination.

Uniform random traffic and tornado traffic are used for comparing WRD with RLB and DOR under benign and adversarial traffic conditions, respectively. In order to capture the average-case performance of the routing algorithms, we generated 250 random permutation traffic patterns and ran the simulation on each pattern for 20,000 cycles. Finally, we report the average delay over the 250 patterns under different injection loads. For the 8-node ring and 16-node ring topologies, the maximum latency used for averaging is clipped at 75 cycles and 125 cycles (around 3 times the average zero-load latency), respectively, in order to keep the impact of a single observation on the computed average within bounds. We refer to this traffic pattern as *dynamic random* traffic as the *average* performance over the set of traffic patterns can be considered to be equivalent to the performance of the routing algorithms under a dynamically changing traffic matrix where the traffic pattern changes every 20,000 cycles. Although more accurate modeling of on-chip network traffic exists in literature [1], [16], in this paper, we restrict ourselves to comparing three important performance characteristics of the routing algorithms: performance under adversarial traffic, performance under benign traffic, and average-case performance over a set of random traffic patterns, just as we did for ideal analysis.

Figures 5 and 6 present the flit-level simulation results for the three traffic patterns on the 8-node and 16-node ring topologies, respectively. The actual throughput obtained is around 60-70% of the throughput predicted using ideal analysis, but the saturation throughput trends remain unchanged. As expected from the ideal throughput analysis, DOR outperforms WRD, which in turn outperforms RLB in terms of saturation throughput under uniform random traffic. The latency of WRD under low loads is 8.3% lower than RLB and 11.5% higher than DOR for the 8-node topology. These numbers are slightly less than the corresponding numbers obtained using hop count analysis, where the hop count of WRD is 11.2% lower than RLB and 16.5% higher than DOR. This is because packet

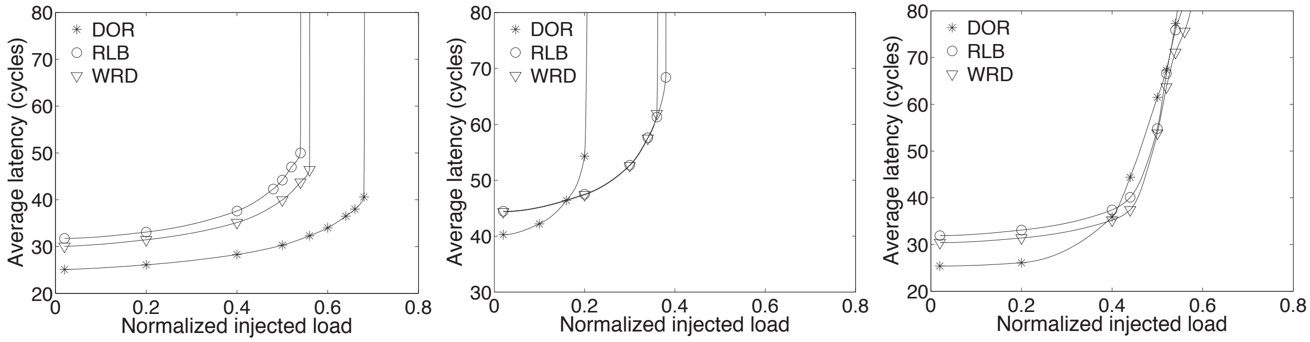


(a) Uniform random

(b) Tornado

(c) Dynamic random

Fig. 5. Performance of WRD on a 8-node ring.



(a) Uniform random

(b) Tornado

(c) Dynamic random

Fig. 6. Performance of WRD on a 16-node ring.

delays in flit-level simulations include the transmission delay of multi-flit packets and the router pipeline delay at the destination, which are ignored while measuring hop count. For the larger 16-node topology, the average latency of WRD is 5.3% lower than RLB and about 20% higher than DOR under uniform random traffic.

For tornado traffic, although DOR has lower latency under very low loads, it saturates much earlier compared to WRD and RLB. WRD and RLB have comparable saturation throughput and latency under tornado traffic. Finally, for dynamic random traffic, we observe that WRD in fact achieves 9-10% lower latency compared to RLB over the entire range of injection rates for the 8-node ring and 5-7% lower latency for the 16-node ring. WRD also starts achieving lower latency compared to DOR beyond injection loads of 25% of network capacity for the 8-node ring and 40% of capacity for the 16-node ring. Hence, in addition to being worst-case throughput optimal, WRD performs well in the average-case.

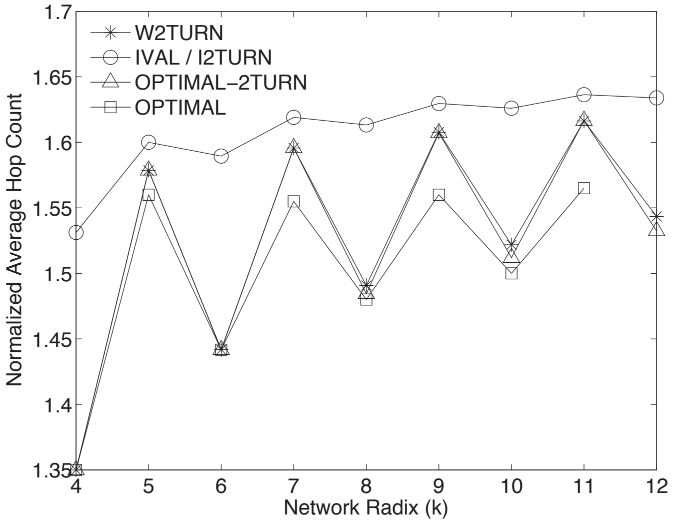


Fig. 7. Comparison of average hop counts for several routing methods with optimal worst-case throughput on 2D-torus networks with different radices. Optimal routing and optimal routing with restriction to 2-turn paths are included.

6.2 Evaluation of W2TURN

6.2.1 Hop count analysis

Figure 7 compares the average hop count of W2TURN with I2TURN, optimal 2TURN routing [19], and optimal routing [19]. The average hop counts are again normalized to minimal dimension-ordered routing. The optimal and optimal-2TURN routing results were obtained using the corresponding multicommodity flow formulations proposed in [19]. As can be seen in Figure 3, the average hop count of W2TURN is lower than I2TURN for all network radices. When the network radix is odd, W2TURN achieves the same average hop count as optimal-2TURN, but this optimal result is achieved with a closed-form algorithm. When the network radix is even, W2TURN comes very close to optimal-2TURN, within just 0.72% in average hop count for k up to 12. Also, as shown in Figure 7, the hop count of W2TURN is very close to optimal routing when k is even[5], within just 1.4% for $k = 10$. Although optimal routing performs noticeably better when k is odd, it is difficult to guarantee deadlock-free operation for optimal routing because the resulting solution may include arbitrary paths and turns.

6.2.2 Throughput evaluation

In this section, we compare the throughput of W2TURN with IVAL/I2TURN, the best previously known worst-case throughput optimal routing algorithm with a closed-form description. We evaluate a randomized version of I2TURN that uses XYX and YXY routing paths with equal probabilities. The randomization balances the load between the X and Y channels and improves the average-case throughput of I2TURN over a non-randomized version that uses just XYX (or YXY) routing paths. We also include DOR in our evaluation to compare W2TURN with a minimal routing algorithm and to emphasize the importance of worst-case throughput optimality.

Throughput analysis is again carried out in two steps. We first present results using ideal throughput analysis over a range of network radices from 4 (16 nodes) to 16 (256 nodes) and then back these results using cycle-accurate flit-level simulations for an even-radix 8×8 topology and an odd-radix 7×7 topology. The traffic patterns used for evaluating W2TURN are two-dimensional versions of the patterns used for evaluating WRD, i.e., uniform random traffic and tornado traffic. In addition, we present average-case throughput results based on averaging the throughput of the routing algorithms over 10,000 randomly generated permutation traffic patterns.

Figure 8(a) compares the average-case throughput of W2TURN with I2TURN and DOR. W2TURN performs marginally better than I2TURN over all the network sizes considered. The average-case throughput of W2TURN is slightly higher than I2TURN when the network radix is even, but the difference is negligible when the network radix is odd. Both W2TURN and I2TURN, however, significantly outperform DOR in terms of average-case throughput. On average, the average-case throughput of W2TURN is 47.3% higher than DOR. This shows that although W2TURN (and I2TURN) are designed for optimal worst-case performance, they also perform very well in the average case.

As in the case of one-dimensional rings, DOR achieves the highest throughput when the traffic is inherently load-balanced, as shown in Figure 8(b). W2TURN and I2TURN achieve lower throughputs due to their non-minimal nature. The average hop-count reduction of W2TURN over I2TURN helps it achieve a proportional increase in throughput under uniform traffic. As shown in Figure 7, the hop count reduction is higher when the network radix is even, resulting in higher throughput improvements. On average, under uniform random traffic, the saturation throughput of W2TURN is 7.75% higher than I2TURN for even-radix networks and 1.25% higher than I2TURN for odd-radix networks. Maximum improvement of up to 13.5% over I2TURN can be observed for the 4×4 topology.

Finally, tornado traffic is an adversarial traffic pattern for all three routing algorithms. In fact, it is the worst-case traffic pattern for W2TURN, I2TURN and DOR when the network radix is odd. Therefore, for odd radices, W2TURN and I2TURN achieve the optimal worst-case throughput of half the network capacity. The worst-case throughput of DOR is significantly lower. For odd network radices, W2TURN degenerates to I2TURN under tornado traffic. This can be deduced from the descriptions of the two algorithms in Sections 4 and 5.1 assuming $\Delta(x_1, x_2) = \Delta(y_1, y_2) = \lfloor k/2 \rfloor$. When the network radix is even, W2TURN can outperform I2TURN by up to 9.4% for low network radices. However, for high network radices, the throughput of W2TURN and I2TURN converge to half the network capacity for even-radix topologies.

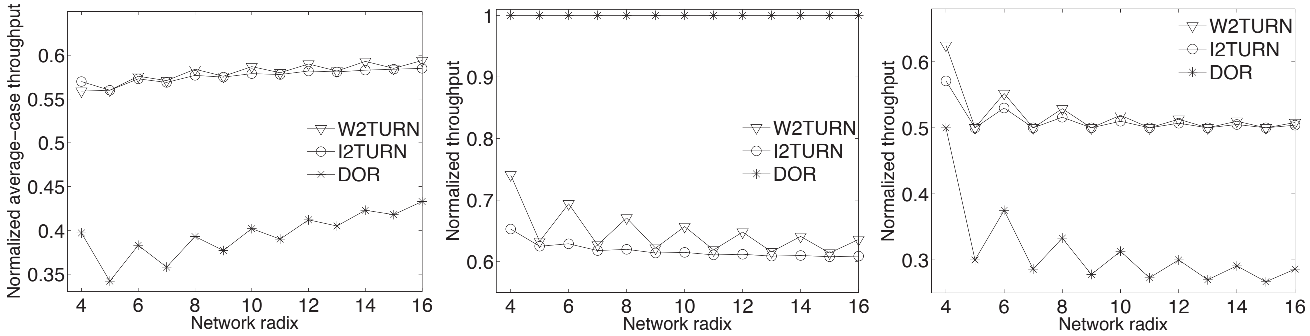
6.2.3 Flit-level simulations

Next, we compare the performance of W2TURN, I2TURN and DOR using flit-level simulations. We use an even-radix 8×8 torus topology and an odd-radix 7×7 torus topology for our experiments. We use the same cycle-accurate flit-level simulator, PopNet [13], for our evaluation of W2TURN. In this case, the simulator models 5-ported pipelined routers for two-dimensional networks. The number of VCs used is still 8 but the buffering is increased to 8 flits per VC to accommodate the increased traffic volume in two-dimensional networks. As discussed in Section 5.5, 4VCs are sufficient to avoid deadlocks in 2-turn paths. However, increasing the number of VCs helps in significantly improving the performance of all the routing algorithms.

Figures 9(a), 9(b) and 9(c) present the flit-level simulation results for a 8×8 torus topology under uniform random traffic, tornado traffic and dynamic random traffic, respectively. The actual throughput sustained is around 50-60% of the throughput predicted using ideal analysis due to the non-idealities in the routers. However, the throughput trends are consistent with the ideal results. For uniform traffic, W2TURN achieves around 6% higher saturation throughput compared to I2TURN and at the same time, the latency under low loads is reduced by 6.5%. The latency reduction is quite close to the hop count difference of 8.2% predicted in Figure 7. The observed difference in latency is lower because the transmission delay of the 3-flit packets and the router pipeline delay at the destination node are ignored while calculating hop count. Both W2TURN and I2TURN, however, pay a latency penalty of 40% and 50%, respectively, over DOR as they select both minimal and non-minimal routing paths.

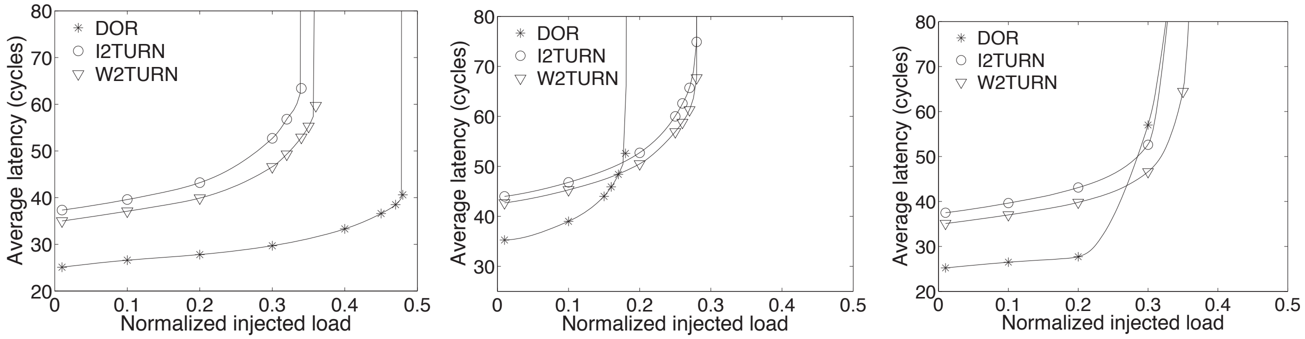
Tornado traffic is an adversarial traffic pattern for all three routing algorithms. Therefore, the maximum throughputs sustained are less than the throughputs sustained under uniform random traffic. The reduction is drastic for DOR, which does not guarantee optimal worst-case throughput. The maximum throughput sustained with DOR under tornado traffic falls by more than 60% compared to the maximum throughput sustained under uniform traffic. On the other hand, the throughput reduction for W2TURN and I2TURN are less severe at just 22% and 18%, respectively. W2TURN marginally outperforms I2TURN under tornado traffic, both

[5] The largest 2D-torus network with an even radix solved for optimal routing in [19] was $k = 10$.



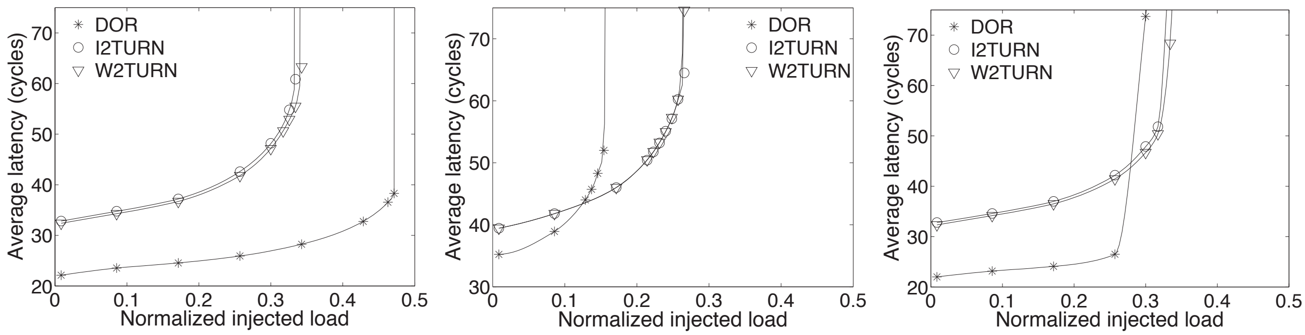
(a) Average-case throughput
(b) Uniform random
(c) Tornado

Fig. 8. Throughput evaluation of W2TURN for different network sizes.



(a) Uniform random
(b) Tornado
(c) Dynamic random

Fig. 9. Performance of W2TURN on a 8×8 torus topology.



(a) Uniform random
(b) Tornado
(c) Dynamic random

Fig. 10. Performance of W2TURN on a 7×7 torus topology.

in terms of latency and saturation throughput. W2TURN outperforms DOR significantly by 55% in terms of saturation throughput.

Figure 9(c) presents results for dynamic random traffic, which captures the average-case behavior of the routing algorithms. The results correspond to the average packet latency measured over 250 randomly generated permutation traffic patterns. The maximum latency of each observation is clipped at 125 cycles to prevent biasing the average to a few large delays observed when the network saturates or approaches saturation. Under low loads of less than 20% of the network capacity, the average delay of W2TURN is 6-7% lower than I2TURN. However, for moderate to high loads, latency reductions of 12-40% can be achieved. This is because the saturation throughput of W2TURN is higher than I2TURN for most of the traffic patterns evaluated. The packet latencies tend to increase rapidly when the network approaches saturation, resulting in much higher average delay numbers for I2TURN compared to W2TURN. W2TURN also outperforms DOR in terms of latency at and beyond injection rates of 30% of network capacity.

Finally, Figure 10 compares W2TURN with I2TURN and DOR on an odd-radix 7×7 topology. The throughput and latency differences between W2TURN and I2TURN are smaller for the odd-radix case, as expected from the ideal throughput analysis. W2TURN has marginally lower latency and marginally higher throughput under uniform and dynamic random traffic patterns. Under tornado traffic, W2TURN and I2TURN algorithms are identical.

7 Conclusion

This paper presented an optimal closed-form routing algorithm for rings called WRD and a closed-form worst-case throughput optimal routing algorithm for 2D torus networks called W2TURN. WRD can achieve the minimum average hop count on rings while remaining worst-case throughput optimal. When the network radix is even, WRD outperforms RLB, the best-known worst-case throughput optimal routing algorithm for rings, in terms of latency over a wide range of traffic patterns. W2TURN routing algorithm for 2D torus networks is based on a weighted random selection of paths with at most 2 turns, which enables a simple deadlock-free implementation with just 4 virtual channels. W2TURN is shown to achieve optimal-2TURN routing when the network radix is odd and is within just 0.72% of optimal-2TURN routing in average hop count when the network radix is even. However, unlike optimal-2TURN, which requires solving large linear programs that do not scale, W2TURN has a closed-form algorithmic description that can scale to arbitrarily large networks. The paper also presented an algorithm called I2TURN that, like W2TURN, is based on a weighted random selection of 2-turn paths. We prove that I2TURN is equivalent to IVAL and hence, is worst-case throughput optimal. We also derive analytical expressions for the average hop counts of I2TURN and W2TURN. These are used to show that the average hop count of W2TURN is strictly less than I2TURN (and IVAL), the best previously known worst-case throughput optimal algorithm with a closed-form description. Finally, we show that W2TURN outperforms I2TURN both in terms of latency and throughput over a wide range of traffic matrices.

References

[1] P. Bogdan and R. Marculescu. Non-stationary traffic analysis and its implications on multicore platform design. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 30(4):508 –519, april 2011.

[2] W. J. Dally, P. Carvey, and L. Dennison. The Avici terabit switch/router. In *Proc. Hot Interconnects 6*, Aug. 1998.

[3] W. J. Dally and B. Towles. Route packets not wires: On-chip interconnection networks. In *Proc. Design Automation Conference*, Las Vegas, Nevada, USA, June 2001.

[4] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

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

[6] IBM. IBM InfiniBand 8-port 12x switch. http://www-3.ibm.com/chips/products/infiniband/.

[7] J. A. Kahle *et al.* Introduction to the Cell multiprocessor. *IBM Journal of Research and Development*, 49(4/5), 2005.

[8] T. Nesson and S.L. Johnsson. ROMM routing on mesh and torus networks. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 275–287, Santa Barbara, CA, USA, 1995.

[9] R. Sunkam Ramanujam and B. Lin. Weighted random oblivious routing on torus networks. In *ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, Princeton, NJ, USA, Oct. 2009.

[10] S. L. Scott and G. Thorson. The Cray T3E network: Adaptive routing in a high-performance 3D torus. In *Hot Interconnects-4*, 1996.

[11] L. Seiler *et al.* Larrabee: a many-core x86 architecture for visual computing. In *SIGGRAPH*, 2008.

[12] D. Seo *et al.* Near-optimal worst-case throughput routing for two-dimensional mesh networks. In *Proc. International Symposium on Computer Architecture*, Madison, WI, USA, June 2005.

[13] L. Shang, L-S. Peh, and N. K. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *Proc. International Symposium on High Performance Computer Architecture*, Anaheim, CA, USA, Feb. 2003.

[14] A. Singh. Load-balanced routing in interconnection networks. Ph.D thesis, Stanford University, Mar. 2003.

[15] A. Singh, W.J. Dally, B. Towles, and A. K. Gupta. Locality-preserving randomized oblivious routing on torus networks. In *ACM Symposium on Parallel Algorithms and Architectures*, Winnipeg, MB, Canada, 2002.

[16] V. Soteriou, H. Wang, and L-S. Peh. A statistical traffic model for on-chip interconnection networks. In *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation*, pages 104–116, Washington, DC, USA, 2006. IEEE Computer Society.

[17] H. Sullivan, T.R. Bashkow, and D. Klappholz. A large scale, homogenous, fully distributed parallel machine. In *Proc. of the 4th Annual Symposium on Computer Architecture*, pages 105–117, 1977.

[18] B. Towles and W. J. Dally. Worst-case traffic for oblivious routing functions. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 1–8, Winnipeg, Manitoba, Canada, Aug. 2002.

[19] B. Towles and W. J. Dally. Throughput-centric routing algorithm design. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 200–209, San Diego, CA, USA, June 2003.

[20] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *ACM Symposium on Theory of Computing*, 1981.

[21] S. Vangal *et al.* An 80-tile 1.28 TFLOPS network-on-chip in 65nm CMOS. In *Proc. International Solid-State Circuits Conference*, San Francisco, CA, USA, 2007.

Rohit Sunkam Ramanujam received his B Tech degree from the Indian Institute of Technology, Kharagpur in 2006 and MS and PhD degrees in Electrical engineering from the University of California, San Diego, in 2008 and 2011, respectively. He is currently working at Juniper Networks, Sunnyvale, California. He is interested in the design of high performance routing algorithms and router architectures for on-chip interconnection networks.

Bill Lin holds a BS, a MS, and a Ph.D. degree in Electrical Engineering and Computer Sciences from the University of California, Berkeley. He is a Professor of Electrical and Computer Engineering at the University of California, San Diego, where he is actively involved with the Center for Wireless Communications (CWC), the Center for Networked Systems (CNS), and the California Institute for Telecommunications and Information Technology (Calit2) in industry-sponsored research efforts. Prior to joining the faculty at UCSD, he was the head of the System Control and Communications Group at IMEC, Belgium. IMEC is the largest independent microelectronics and information technology research center in Europe. It is funded by European funding agencies in joint projects with major European telecom and semiconductor companies. His research has led to over 150 journal and conference publications, including 2 best paper awards, 2 best paper nominations, and 2 distinguished paper citations. He has served on panels and given invited presentations at several major conferences, and he has served on over 35 program committees, including serving as the General Chair for NOCS-2009, ANSC-2010, and IWQoS-2011. He also holds 3 awarded patents.