Weighted Random Oblivious Routing on Torus Networks*

Rohit Sunkam Ramanujam
University of California, San Diego
rsunkamr@ucsd.edu

Bill Lin
University of California, San Diego
billlin@ece.ucsd.edu

ABSTRACT

Torus, mesh, and flattened butterfly networks have all been considered as candidate architectures for on-chip interconnection networks. In this paper, we study the problem of optimal oblivious routing for one of these architecture classes, namely, the torus network. 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 results in 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 [14]. We also provide another routing algorithm based on the weighted random selection of paths with at most two turns called I2TURN and show that it 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 the optimal distribution of traffic along the minimal and non-minimal directions in a ring topology to achieve minimum average hop count under maximum worst-case throughput.

Categories and Subject Descriptors

C.1.2 [**Processor Architectures**]: Multiple Data Stream Architectures (Multiprocessors)—*Interconnection Architectures*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Routing and Layout*

*A short version of this paper appears in *Computer Architecture letters* [7].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ANCS'09, October 19-20, 2009, Princeton, New Jersey, USA.
Copyright 2009 ACM 978-1-60558-630-4/09/0010...$10.00.

General Terms

Algorithms, Design, Performance

Keywords

Interconnection Networks, Torus Networks, Oblivious Routing.

1. INTRODUCTION

Several network architectures have been considered as candidates for on-chip interconnection networks, including torus, mesh, and flattened butterfly networks [2, 4, 12, 16]. In this paper, we study the problem of optimal oblivious routing for one of these architecture classes, namely, the torus network. In the design of routing algorithms, both throughput and latency are important metrics. Although dimension-ordered routing (DOR) [11] can achieve minimal-length routing on torus networks, it suffers from poor worst-case throughput because it offers no route diversity. On the other hand, it is well-known that Valiant routing (VAL) [15] can achieve optimal worst-case throughput by load-balancing globally across the entire network, but it does so at the expense of destroying locality. Other oblivious routing algorithms such as ROMM [6] and RLB [9] 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 [14] achieves the lowest average hop count. It works by reversing 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 offers optimal worst-case throughput for 2D-torus networks. W2TURN is based on a weighted random selection of paths with at most two turns and it has a simple deadlock-free implementation. In comparison to IVAL, W2TURN achieves better performance in terms of average hop count. We also present another weighted random routing algorithm called I2TURN that is shown to be 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 [14], 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 [14], 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, within just 0.72% in average hop count on a 12×12 torus.

Finally, we 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. We are unaware of any previous oblivious routing algorithms for rings that can achieve these optimality conditions.

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 and evaluates its performance. Section 6 concludes the paper.

2. BACKGROUND

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 used as the interconnection fabric in commercial multi-core chips [5, 8]. A torus network is edge-symmetric, which makes it easier to load-balance traffic over all channels in the network. 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) in order to avoid employing long wrap-around links between edge nodes.

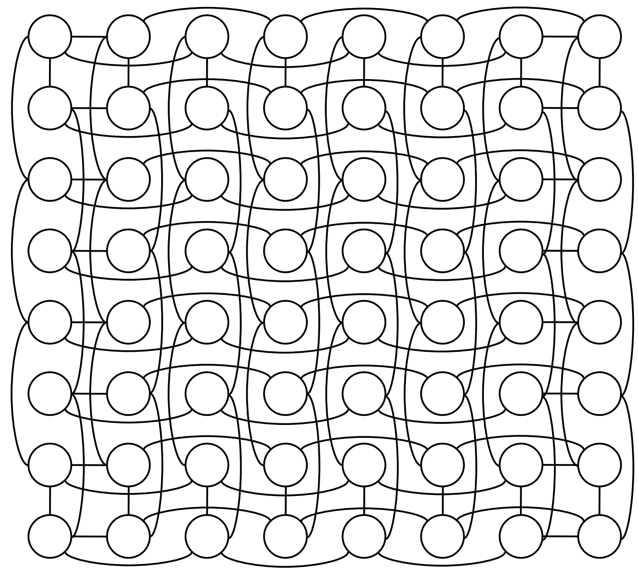

Figure 1: Layout of a 8×8 folded torus.

2.2 Preliminaries

The worst-case throughput of a routing algorithm is typically defined relative to the capacity of a network, which is 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 [3],

$$\gamma^* = \begin{cases} \frac{k}{8} & k \text{ is even} \\ \frac{k}{8} - \frac{1}{8k} & k \text{ is odd} \end{cases}$$

The network capacity is the inverse of γ^*.

The maximum channel load $\gamma(R, \Lambda)$ 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

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

[1]The largest 2D-torus networks solved in [14] 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.

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 [13], 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) [15] 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 worst-case 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 network sizes.

Finally, 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. [14] to derive worst-case throughput optimal routings with minimum hop count over a wide 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.

3. OPTIMAL ROUTING ON RINGS WITH WRD

This section considers the optimal oblivious routing problem for one-dimensional rings.

3.1 WRD Algorithm

Our proposed algorithm called WRD works as follows. Suppose source s sends traffic to destination d. Then the minimal distance around the loop is given as:

$$\Delta(s, d) = \min(|s - d|, k - |s - d|) \tag{3}$$

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

$$P_{odd} = \begin{cases} \dfrac{k - \Delta}{k} & \text{in the minimal direction} \\ \dfrac{\Delta}{k} & \text{in the non-minimal direction} \end{cases} \tag{4}$$

This is precisely what the RLB algorithm [9] 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:

$$H_{odd}(\text{WRD}) = E\left[\frac{2\Delta(k - \Delta)}{k}\right] = \frac{k}{3} - \frac{1}{3k} \tag{5}$$

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

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

$$P_{even} = \begin{cases} \dfrac{k - \Delta - 1}{k - 2} & \text{in the minimal direction} \\ \dfrac{\Delta - 1}{k - 2} & \text{in the non-minimal direction} \end{cases} \tag{6}$$

When $\Delta = 0$ (i.e., $s = d$), or when $k = 1$, no routing is necessary. When $k = 2$ and $\Delta > 0$, then WRD routes in both directions at equal distance with equal probability, which is the same as RLB. When $\Delta = k/2$, using Equation 6, WRD again routes in both directions with equal probability. Note that these probabilities for the even case when $k > 2$ are different from those used by RLB. The average hop count of this route distribution can be computed as follows:

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

WRD achieves a lower average hop count than RLB for even radix when $k > 2$ since

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

We next show that WRD indeed achieves optimal worst-case throughput.

Claim 3.1. *WRD is worst-case throughput optimal.*

Proof. For the odd case, WRD is the same as RLB, which has already been shown to be worst-case throughput optimal. For the even case, we use the same proof methodology that was used in [10] for showing RLB is worst-case throughput optimal on a ring, which uses the method in [13] for identifying a worst-case traffic pattern. 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 in Equation 2 of Section 2 for even k. A worst-case traffic pattern for WRD is Tornado traffic [3]. 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 each clockwise channel is given by the sum of the contributions from the $k/2 - 1$ nodes preceding the channel. Since each of these $k/2-1$ nodes route in the clockwise (minimal) direction with a probability of $\frac{k/2}{k-2}$, the maximum channel load on a channel along the clockwise direction, γ_c(WRD), is given as:

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

Similarly, the maximum channel load on a counter-clockwise channel can 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 6, each of these nodes contribute $\frac{k-4}{2(k-2)}$ of their traffic and the sum of their contributions, γ_{cc}(WRD), is given as:

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

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

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

Therefore, 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 [14]. □

3.2 Performance Evaluation

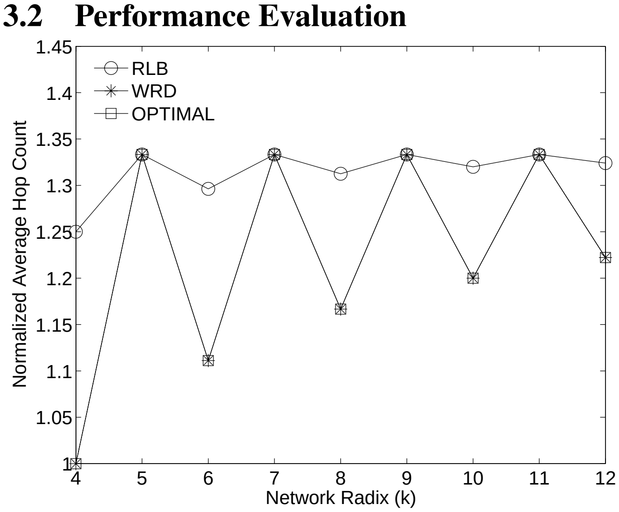

Figure 2: Comparing average hop counts of WRD with RLB [9] and optimal routing [14] on different size rings.

Figure 2 compares the average hop count performance of WRD with RLB [9] and optimal routing [14]. The average hop counts are normalized with respect to minimal routing. The optimal routing results were obtained using the multi-commodity flow formulation proposed in [14]. As can be seen

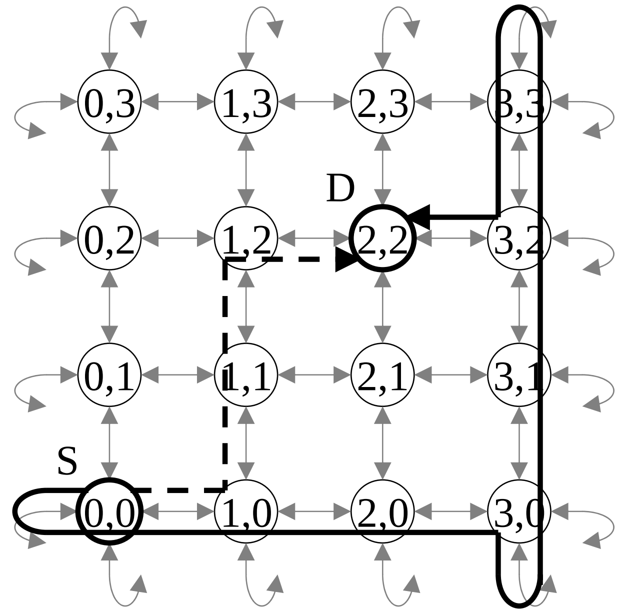

Figure 3: Routing with 2-turn paths.

in Figure 2, WRD achieves the same performance as optimal routing, but using a closed-form algorithmic description.

When k is odd, WRD and RLB are equivalent, and therefore they have the same performance, as shown in Figure 2. When k is even, WRD outperforms RLB because WRD routes in the minimal direction more often.

4. THE I2TURN ROUTING ALGORITHM

This section describes the I2TURN routing algorithm for 2D-torus networks. As the name suggests, I2TURN considers routing paths with at most two turns, as shown in Figure 3. 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 [14] in their optimal 2TURN algorithm. However, their 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 path are generated as follows:

1. X-segment: Choose at uniform random an X position $x^* \in [0, k-1]$ and route the packet in the X dimension from (x_1, y_1) to (x^*, y_1) in the minimal direction.

2. Y-segment: Next, route the packet in the Y dimension from (x^*, y_1) to (x^*, y_2) in the minimal and non-minimal directions with the following probabilities.

$$P = \begin{cases} \dfrac{k - \Delta y}{k} & \text{in the minimal direction} \\ \\ \dfrac{\Delta y}{k} & \text{in the non-minimal direction} \end{cases}$$

   This is the same weighted random routing as WRD for the odd-ring case (as in Equation 4), where Δy is the minimum distance in Y from (x^*, y_1) to (x^*, y_2) (as in Equation 3).

3. X-segment: Finally, route the packet in the X dimension from (x^*, y_2) to (x_2, y_2) in the minimal direction.

There are several degenerate cases. When $x^* = x_1$, then 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 with no turns. In this case, the packet is routed with probability $\frac{k-\Delta x}{k}$ in the minimal direction, where Δx is the minimum distance in X between x_1 and x_2, with probability $\frac{\Delta x}{k}$ 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 where $y_1 \neq y_2$, 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. Thus, there are exactly two 2-turn paths through x^*.

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 $\frac{1}{k}$. 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. Therefore, we can reduce the proof to equivalent path distribution 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 $\frac{k-\Delta y}{k}$, and a non-minimal path in the long direction with a distance of $(k - \Delta y)$ that is chosen with probability $\frac{\Delta y}{k}$. For IVAL, any of the k nodes on the Y ring can be chosen as the intermediate node. Since the definition of a minimal or non-minimal path 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 of the source to location 0. By definition of minimal distance, $\Delta y \leq \frac{k}{2}$.

In this 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 intermediate node chosen by IVAL. There are two situations where a packet is guaranteed to be routed along the minimal path after loop removal: when $0 \leq i < \frac{k}{2}$ or when $(i - \Delta y) > \frac{k}{2}$. There are $\lceil \frac{k}{2} \rceil$ possible intermediate nodes that satisfy $0 \leq i < \frac{k}{2}$, and there are $\lceil \frac{k}{2} \rceil - \Delta y - 1$ possible intermediate nodes that satisfy $(i - \Delta y) > \frac{k}{2}$, with a combined total of $\lceil \frac{k}{2} \rceil + \lceil \frac{k}{2} \rceil - \Delta y - 1$ intermediate nodes that will always result in IVAL routing over the minimal path after loop removal. When k is even and $i = \frac{k}{2}$, 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 $i - \Delta y = \frac{k}{2}$, 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 the intermediate nodes are chosen with equal probability, the total probability of choosing the minimal path is given by $(\lceil \frac{k}{2} \rceil + \lceil \frac{k}{2} \rceil - \Delta y - 1)/k$ when k is odd and $(\lceil \frac{k}{2} \rceil + \lceil \frac{k}{2} \rceil - \Delta y)/k$ when k is even. When k is odd, $\lceil \frac{k}{2} \rceil + \lceil \frac{k}{2} \rceil = k + 1$. When k is even, $\lceil \frac{k}{2} \rceil + \lceil \frac{k}{2} \rceil = k$. Therefore, the probability of choosing the minimal path reduces to $\frac{k-\Delta y}{k}$ in both cases.

Finally, in the third case where $y_1 = y_2$, 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 above for the Y ring can be applied to the X dimension to show that both IVAL and I2TURN will choose the minimal path with the same probability. □

CLAIM 4.2. *I2TURN is worst-case throughput optimal.*

PROOF. Follows from its equivalence to IVAL. □

I2TURN can be similarly 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 among the X and Y channels.

4.2 Average Hop Count for I2TURN

The average hop count for I2TURN can be expressed as the sum of the average hops for the three corresponding routing segments: minimal routing for the first and last X segments, weighted random routing for the middle Y segment. Let H_{min} denote the average hop count for minimal routing on a ring [3].

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

Since the routing on the middle Y segment uses the same weighted distribution of traffic on the minimal and non-minimal paths as the WRD algorithm for the odd-ring case, the average hop count for this segment is the same as Equation 5. As analyzed in the proof of Claim 4.1, we have to consider 1-in-k cases where 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 is identical to the routing used in the Y dimension. Taken together, the average hop count for I2TURN is given as follows:

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

5. THE W2TURN ROUTING ALGORITHM

In this section, 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 3. 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 that we gained from studying the optimal routing in the 1D ring case (WRD) and the I2TURN algorithm.

In the remainder of this section, we present the W2TURN 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 algorithm using an XYX routing. $\Delta(x_1, x_2)$ 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 follows from Equation 3.

Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination, the three segments of the XYX 2-turn path 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 *either* of the following conditions are satisfied:
      - $\Delta(x_1, x^*) \neq \lfloor \frac{k}{2} \rfloor$,
      - (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 the packet from (x_1, y_1) to (x^*, y_1) on the X ring with the following probabilities:

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

   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 the packet 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
      - $(x^* = x_1 \text{ or } x^* = x_2)$.

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

3. X-segment: Finally, route the packet 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 *either* of the following conditions are satisfied:
      - $\Delta(x^*, x_2) \neq \lfloor \frac{k}{2} \rfloor$,
      - (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 the packet from (x^*, y_2) to (x_2, y_2) on the X ring with the following probabilities:

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

   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 $x^* = x_1$, then 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 $y_1 = y_2$, then packet only needs 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 case, we again first describe the algorithm using an XYX routing. Suppose (x_1, y_1) is the source and (x_2, y_2) is the destination. The three segments of the XYX 2-turn path 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 the packet 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 $x^* = x_1$, then 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 $\Delta(x_1, x_2) \neq k/2$ and $\Delta(x_1, x_2) > 0$:

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

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 version is again defined equivalently 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 [10], which uses the method in [13] 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 $\frac{k}{4}$ when k is even and at most $\left(\frac{k}{4} - \frac{1}{4k}\right)$ when k is odd, as shown necessary and sufficient in Equation 2. A worst-case traffic pattern for W2TURN, independent of k, is again Tornado traffic [3]. Using worst-case load analysis, the maximum channel loads for Tornado traffic were found to be the same as Equation 2 for all values of k analyzed[2]. □

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 5.5 we show that W2TURN indeed outperforms I2TURN in average hop count. We treat the even and odd k cases separately.

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 for 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 (over minimal) added as a result of non-minimal routing. We denote this penalty paid over minimal routing, by routing non-minimally, as $H_{penalty}$.

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

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

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:

$$H_x(\text{XYX}) = \frac{1}{k}H_{case1} + \frac{(k-1)}{k}(2 \times H_{case2}) \quad (7)$$

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 $\frac{(k-1)}{k}$ 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 $\frac{2}{k}$. 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:

$$H_y(\text{XYX}) = H_{odd}(\text{WRD}) - H_{savings}$$
$$= \frac{k}{3} - \frac{1}{3k} - H_{savings} \quad (8)$$

From Equations 7 and 8 and from the fact that YXY routing will have an average hop count identical to 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. 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 can be expressed as:

$$H_x(\text{XYX}) = \frac{1}{k}H_{case1} + \frac{(k-1)}{k}(2 \times H_{case2}) \quad (9)$$

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

$$H_y(\text{XYX}) = H_{even}(\text{WRD}) = \frac{(k-1)}{3} \quad (10)$$

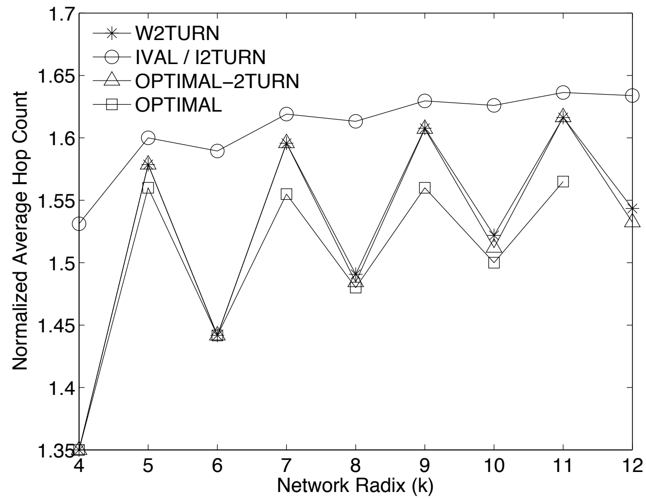

Figure 4: Comparing average hop counts for several routing methods with optimal worst case throughput on different size 2D-torus networks. Optimal routing and optimal routing with restriction to 2-turn paths are included.

From Equations 9 and 10,

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

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

5.5 Performance Evaluation

Figure 4 compares the average hop count performance of W2TURN with I2TURN, optimal 2TURN routing [14], and optimal routing [14]. The average hop counts are again normalized with respect to minimal dimension-ordered routing. The optimal and optimal-2TURN routing results were obtained using the corresponding multicommodity flow formulation proposed in [14]. As can be seen in Figure 2, W2TURN outperforms I2TURN. 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 can be seen in Figure 4, W2TURN comes very close to optimal routing when k is even[3], 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.

[3]The largest 2D-torus network with an even radix solved for optimal routing in [14] was $k = 10$.

5.6 Deadlock-Free Implementation

W2TURN uses the same set of 2-turn paths as the optimal 2TURN formulation proposed in [14]. When k is odd, W2TURN in fact 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 VAL [15]). 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, thereby 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. 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. The paper also presented an algorithm called I2TURN which, 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 closed form analytical expressions for the average hop counts of I2TURN and W2TURN and show that the average hop count of W2TURN is strictly less than I2TURN (and IVAL), the best previous closed-form worst-case throughput optimal algorithm. W2TURN is shown to achieve optimal-2TURN routing when the network radix is odd and within just 0.72% of optimal-2TURN routing in average hop count when the network radix is even. It does so with a closed-form algorithm which can scale to arbitrarily large networks unlike optimal-2TURN, which requires solving large linear programs that do not scale. Finally, we show that W2TURN can be made deadlock free using only four virtual channels which makes it attractive from an implementation perspective.

7. REFERENCES

[1] ILOG, Mountain View, CA, CPLEX 7.5 User's manual, Oct. 2001, pp. 168-177.

[2] J. Balfour, W.J. Dally. Design tradeoffs for tiled CMP on-chip networks. *International Conference on Supercomputing*, Queensland, Australia, 2006.

[3] W. J. Dally, B. Towles. Principles and practices of interconnection networks. Morgan Kaufmann, 2003.

[4] W.J. Dally, B. Towles. Route packets, not wires: On-chip interconnection networks. *Design Automation Conference (DAC)*, June 2001, Las Vegas, Nevada, USA.

[5] M. Kistler, M. Perrone, F. Petrini. Cell multiprocessor communication network: Built for speed. *IEEE MICRO*, May, 2006.

[6] T. Nesson, S. L. Johnsson. ROMM routing on mesh and torus networks. *ACM Symposium on Parallel Algorithms and Architectures*, Santa Barbara, CA, 1995.

[7] R. S. Ramanujam, B. Lin. Weighted random routing on torus networks. *IEEE Computer Architecture Letters*, vol. 8, no. 1, January 2009.

[8] L. Seiler et al. Larrabee: A many-core x86 architecture for visual computing. *SIGGRAPH Conference*, 2008.

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

[10] A. Singh. Load-balanced routing in interconnection networks. Ph.D. thesis, Stanford University, March 2005.

[11] H. Sullivan, T. R. Bashkow. A large scale, homogeneous, fully distributed parallel machine. *Annual Symposium on Computer Architecture*, ACM Press, 1977, pp. 105-117.

[12] M. B. Taylor, W. Lee, S. Amarasinghe, A. Agarwal. Scalar operand networks: On-chip interconnect for ILP in partitioned architectures. *International Symposium on High-Performance Computer Architecture (HPCA)*, pages 341-353, Anaheim, CA, 2003.

[13] B. Towles, W. J. Dally. Worst-case traffic for oblivious routing functions. *ACM Symposium on Parallel Algorithms and Architectures*, Winnipeg, MB, Canada, 2002.

[14] B. Towles, W. J. Dally, S. P. Boyd. Throughput-centric routing algorithm design. *ACM Symposium on Parallel Algorithms and Architectures*, San Diego, CA, June, 2003.

[15] L. G. Valiant, G. J. Brebner. Universal schemes for parallel communication. *ACM Symposium on the Theory of Computing*, pp. 263-277, Milwaukee, MN, 1981.

[16] S. Vangal et al. An 80-tile sub-100-W teraflops processsor in 65nm CMOS. *IEEE Journal of Solid State Circuits*, Jan 2008.