Path-Based, Randomized, Oblivious, Minimal Routing

Myong Hyon Cho, Mieszko Lis, Keun Sup Shim, Michel Kinsy and Srinivas Devadas
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA
{mhcho, mieszko, ksshim, mkinsy, devadas}@mit.edu

ABSTRACT

Path-based, Randomized, Oblivious, Minimal routing (PROM) is a family of oblivious, minimal, path-diverse routing algorithms especially suitable for Network-on-Chip applications with $n \times n$ mesh geometry. Rather than choosing among all possible paths at the source node, PROM algorithms achieve the same effect progressively through efficient, local randomized decisions at each hop. Routing is deadlock-free in all PROM algorithms when the routers have at least two virtual channels.

While the approach we present can be viewed as a generalization of both ROMM and O1TURN routing, it combines the low hardware cost of O1TURN with the routing diversity offered by the most complex n-phase ROMM schemes. As all PROM algorithms employ the same hardware, a wide range of routing behaviors, from O1TURN-equivalent to uniformly path-diverse, can be effected by adjusting just one parameter, even while the network is live and continues to forward packets. Detailed simulation on a set of benchmarks indicates that, on equivalent hardware, the performance of PROM algorithms compares favorably to existing oblivious routing algorithms, including dimension-ordered routing, two-phase ROMM, and O1TURN.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network communications

1. INTRODUCTION AND BACKGROUND

Deterministic oblivious routing algorithms are widely used in Network-on-Chip (NoC) designs because they are easy to implement in hardware. Dimension-order routing (DOR), which routes packets by following a straight path to the destination coordinate one dimension at a time, has the simplest router implementation, and, since no path exceeds the minimum number of hops,[1] offers low latency. Its throughput, however, can be poor even for local traffic since it offers no routing flexibility.

[1]a feature known as "minimal routing"

NoCArc '09, December 12, 2009, New York City, New York, USA

Several schemes have attempted to address this shortcoming. Valiant [17], which routes each packet via a random intermediate node, has provably optimal worst-case throughput,[2] but its low average-case throughput and high latency have prevented wide adoption. ROMM [9, 10] reduces this latency by confining the intermediate nodes to the minimal routing region; although it outperforms DOR in many cases, the worst-case performance of the most popular (2-phase) variant on 2D meshes and tori has been shown to be significantly worse than optimal [15, 11], while the overhead of n-phase ROMM has hindered real-world use. O1TURN [11] on a 2D mesh selects one of the DOR routes (XY or YX) uniformly at random, and offers performance roughly equivalent to 2-phase ROMM over standard benchmarks combined with near-optimal worst-case throughput; however, its limited path diversity limits performance on some traffic patterns.

We therefore set out to develop a routing scheme with low latency, high average-case throughput, and path diversity for good performance across a wide range of patterns. The PROM family of algorithms we present here is significantly more general than existing oblivious routing schemes with comparable hardware cost (e.g., O1TURN). Like n-phase ROMM, PROM is maximally diverse on an $n \times n$ mesh, but requires less complex routing logic and needs only two, rather than n, virtual channels to ensure deadlock freedom.

In what follows, we describe PROM in Section 2, and show how to implement it efficiently on a virtual-channel router in Section 3. Section 4 summarizes related routing algorithms. In Section 5, through detailed network simulation, we show that PROM algorithms are competitive with existing oblivious routing algorithms (DOR, 2-phase ROMM, and O1TURN) on equivalent hardware. We conclude the paper in Section 6.

2. PROM ROUTING

Given a flow from a source to a destination, PROM routes each packet separately via a path randomly selected from among all minimal paths. The routing decision is made *lazily:* that is, only the next hop (conforming to the minimal-path constraint) is randomly chosen at any given switch, and the remainder of the path is left to the downstream nodes. The local choices form a random distribution over all possible minimal paths, and specific PROM routing algorithms differ according to the distributions from which the random paths are drawn. In the interest of clarity, we first describe a specific instantiation of PROM, and then show how to parametrize it into a family of routing algorithms.

2.1 Coin-toss PROM

[2]where worst-case throughput is defined as the minimum throughput over all traffic patterns

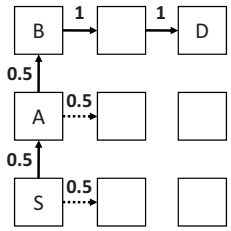


Figure 1: Choosing a minimal route randomly in PROM.

Figure 1 illustrates the choices faced by a packet routed under a PROM scheme where every possible next-hop choice is decided by a fair coin toss. At the source node S, a packet bound for destination D randomly chooses to go north (bold arrow) or east (dotted arrow) with equal probability. At the next node, A, the packet can continue north or turn east (egress south or west is disallowed because the resulting route would no longer be minimal). Finally, at B and subsequent nodes, minimal routing requires the packet to proceed east until it reaches its destination. Note that the routing is oblivious and next-hop routing decisions can be computed locally at each node based on local information and the relative position of the current node to the destination node; nevertheless, the scheme is maximally diverse in the sense that each possible minimal path has a non-zero probability of being chosen. Observe, however, that the coin-toss variant does *not* choose *paths* with uniform probability;[3] next, we show how to parametrize PROM and create a uniform variant.

2.2 PROM Variants

Although all the next-hop choices in Figure 1 were 50–50 (whenever a choice was possible without leaving the minimum path), the probability of choosing each egress can be varied for each node and even among flows between the same source and destination. On a 2D mesh under minimum-path routing, each packet has at most two choices: continue straight or turn;[4] how these probabilities are set determines the specific instantiation of PROM:

O1TURN-like PROM.

O1TURN [11] randomly selects between XY and YX routes, i.e., either of the two routes along the edges of the minimal-path box. We can emulate this with PROM by configuring the source node to choose each edge with probability $\frac{1}{2}$ and setting all intermediate nodes to continue straight with probability 1 until a corner of the minimal-path box is reached, turning at the corner, and again continuing straight with probability 1 until the destination.[5]

Uniform PROM.

Uniform PROM weighs the routing probabilities so that each possible minimal *path* has an equal chance of being chosen. Since only minimal paths are considered, the local routing decision at each switch S depends only on the position relative to the destination node, and each path must be chosen with probability $\frac{x!\cdot y!}{(x+y)!}$ (where x and y indicate the number of hops to the destination along the X and Y dimensions, respectively); that is, the packet must depart S along the X dimension with probability $\frac{x}{x+y}$ and along the Y dimension with probability $\frac{y}{x+y}$, as shown in Figure 2(a). In this configuration, PROM is equivalent to n-phase ROMM with each path being chosen at the source with equal probability.[6]

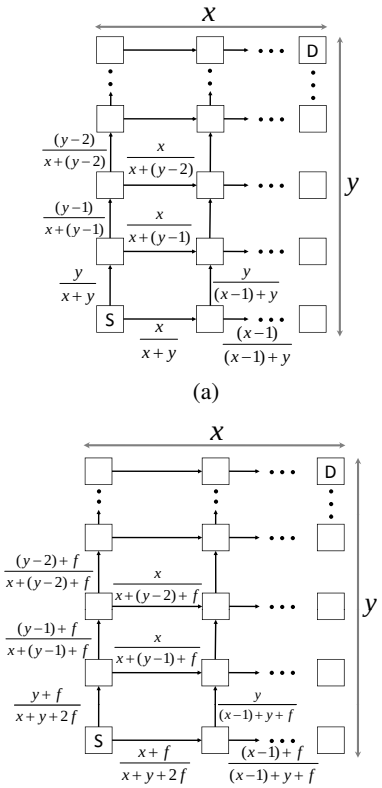


Figure 2: (a) Uniform PROM. (b) Parameterized PROM.

Parametrized PROM.

The two configurations above are, in fact, two extremes of a continuous family of PROM algorithms parametrized by a single parameter f, as shown in Figure 2(b). At the source node, the router forwards the packet towards the destination on either the horizontal link or the vertical link randomly according to the ratio $x+f : y+f$, where x and y are the distances to the destination along the corresponding axes. At intermediate nodes, two possibilities exist: if the packet arrived on an X-axis ingress (i.e., from the east or the west), the router uses the ratio of $x+f : y$ in randomly determining the next hop, while if the packet arrived on an Y-axis ingress, it uses the ratio $x : y+f$. Intuitively, PROM is less likely to make extra turns as f grows, and increasing f pushes traffic from the diagonal of the minimal-path rectangle towards the edges (see Figure 3). Thus, when $f = 0$ (Figure 3(a)), we have Uniform PROM, with most traffic near the diagonal, while $f = \infty$ (Figure 3(d)) implements the O1TURN variant with traffic routed exclusively along the edges.

Variable Parameterized PROM (PROMV).

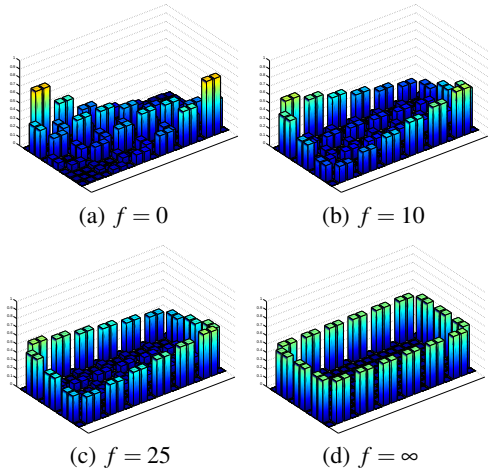
While more uniform (low f) PROM variants offer more path diversity, they tend to increase congestion around the center of the

[3]For example, while uniform path selection in Figure 1 would result in a probability of $\frac{1}{6}$ for each path, either border path (e.g., $S \to A \to B \to \cdots \to D$) is chosen with probability $\frac{1}{4}$, while each of the four paths passing through the central node has only a $\frac{1}{8}$ chance.

[4]While PROM routers also support a host of other, non-minimal schemes out of the box, this paper focuses on minimal-path routing.

[5]This slightly differs from O1TURN in virtual channel allocation, as described in Section 2.3.

[6]again, modulo differences in virtual channel allocation



(a) $f = 0$ (b) $f = 10$

(c) $f = 25$ (d) $f = \infty$

Figure 3: Probability distributions of PROM routes in a 4-by-6 minimal-path rectangle for various values of f

mesh, as most of the traffic is routed near the diagonal. Meanwhile, rectangle edges are underused especially towards the edges of the mesh, where the only possible traffic comes from the nodes on the edge.

Variable Parametrized PROM (PROMV) addresses this shortcoming by using different values of f for different flows to balance the load across the links. As the minimal-path rectangle between a source–destination pair grows, it becomes more likely that other flows within the rectangle compete with traffic between the two nodes. Therefore, PROMV sets the parameter f proportional to the minimal-path rectangle size divided by overall network size so traffic can be routed more toward the boundary when the minimal-path rectangle is large. When x and y are the distance from the source to the destination along the X and Y dimensions and N is the total number of router nodes, f is determined by the following equation: [7]

$$f = f_{max} \cdot \frac{xy}{N} \quad (1)$$

This scheme ensures efficient use of the links at the edges of the mesh and alleviates congestion in the central region of the network.

2.3 Virtual Channel Assignment

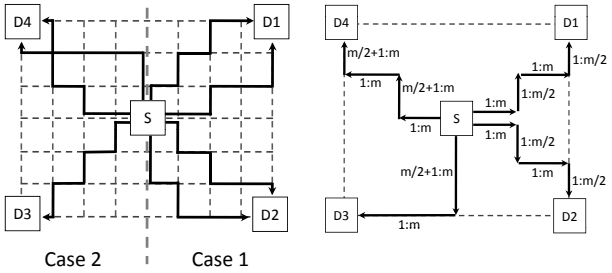
PROM requires only two virtual channels for deadlock-free routing. The virtual channel assignment depends on the relative position of the source node S and destination node D, and is the same for all flows traveling from S to D:

1. if D lies to the east of S, vertical links use the first VC;
2. if D lies to the west of S, vertical links use the second VC;
3. if D lies directly north or south of S, both VCs are used;
4. all horizontal links may use all VCs.

(When there are more than two virtual channels, they are split into two sets and assigned similarly). Figure 4 illustrates the division between eastbound and westbound traffic and the resulting allocation for m virtual channels.

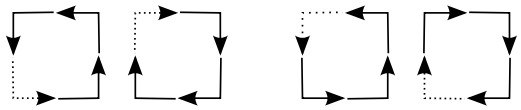
To show that this assignment is deadlock-free, we invoke the turn model [5], a systematic way of generating deadlock-free routes. Figure 5 shows two different turn models that can be used in a 2D mesh: each model disallows two of the eight possible turns,

[7] the value of f_{max} was fixed to the same value for all our experiments.



(a) East- and westbound routes (b) VC set allocation

Figure 4: Virtual channel assignment under PROM



(a) West-First (rotated 180°) (b) North-Last (rotated 270°)

Figure 5: Permitted (solid) and forbidden (dotted) turns in two turn models on a 2D mesh

and, when all traffic in a network obeys the turn model, deadlock freedom is guaranteed. For PROM, the key observation[8] is that minimal-path traffic always obeys one of those two turn models: eastbound packets never turn westward, westbound packets never turn eastward, and packets between nodes on the same row or column never turn at all. Thus, westbound and eastbound routes always obey the restrictions of Figures 5(a) and 5(b), respectively, and placing them on different virtual networks ensures deadlock freedom. Traffic over horizontal links and traffic between nodes on the same column simultaneously conform to both models, and may use both virtual networks.[9]

Note that the correct virtual channel allocation for a packet can be determined *locally* at each switch, given only the packet's destination (encoded in its flow ID), and which ingress and virtual channel the packet arrived at. For example, any packet arriving from a west-to-east link and turning north or south must be assigned the first VC (or VC set), while any packet arriving from an east-to-west link and turning must get the second VC; finally, traffic arriving from the north or south stays in the same VC it arrived on.

Note that the virtual channel assignment in PROM differs from that of both O1TURN and n-phase ROMM even when the routing behavior itself is identical. While PROM with $f = \infty$ selects VCs based on the overall direction as shown above, O1TURN chooses VCs depending on the initial choice between the XY and YX routes at the source node; because all traffic on a virtual network is either XY or YX, no deadlock results. ROMM, meanwhile, assigns a separate VC to each *phase*; since each phase uses exclusively one type of DOR (say XY), there is no deadlock, but the assignment is inefficient for general n-phase ROMM which uses n VCs where two would suffice.

3. IMPLEMENTATION COST

Other than a randomness source, a requirement common to all randomized algorithms, implementing any of the PROM algorithms requires almost no hardware overhead over a classical obliv-

[8] due to Shim et al. [12]

[9] PROM does not *explicitly* implement turn model restrictions, but rather forces routes to be minimal, which automatically restricts possible turns; thus, we only use the turn model to show that VC allocation is deadlock-free.

ious virtual channel router [4]. As with DOR, the possible next-hop nodes can be computed directly from the position of the current node relative to the destination; for example, if the destination lies to the northwest on a 2D mesh, the packet can choose between the northbound and westbound egresses. Similarly, the probability of each egress being chosen (as well as the value of the parameter f in PROMV) only depends on the location of the current node, and on the relative locations of the source and destination node, which usually form part of the packet's flow ID.

As discussed in Section 2.3, virtual channel allocation also requires only local information already available in the classical router: namely, the ingress port and ingress VC must be provided to the VC allocator and constrain the choice of available VCs when routing to vertical links, which, at worst, requires simple multiplexer logic. This approach ensures deadlock freedom, and eliminates the need to keep any extra routing information in packets.

The routing header required by most variants of PROM needs only the destination node ID, which is the same as DOR and O1TURN and amounts to $2\log_2(n)$ bits for an $n \times n$ mesh; depending on the implementation chosen, PROMV may require an additional $2\log_2(n)$ bits to encode the source node if it is used in determining the parameter f. In comparison, packets in canonical k-phase ROMM carry the IDs for the destination node as well as the $k-1$ intermediate nodes in the packet, an overhead of $2k\log_2(n)$ on an $n \times n$ mesh, although one could imagine a somewhat PROM-like version of ROMM where only the next intermediate node ID (in addition to the destination node ID) is carried with the packet, and the $k+1$st intermediate node is chosen once the packet arrives at the kth intermediate destination.

Thus, PROM hardware offers a wide spectrum of routing algorithms at an overhead equivalent to that of O1TURN and smaller than even 2-phase ROMM.

4. RELATED WORK

Dimension-ordered routing (DOR) is an extremely simple routing algorithm for a broad class of networks that include 2D mesh networks [3]. Packets simply route along one dimension first and then in the next dimension. This simplicity comes at the cost of poor worst-case and average-case throughput for mesh networks. However, its simplicity is also its strength as it enables low complexity implementations.

ROMM [9, 10] randomly chooses an intermediate node within the minimum rectangle defined by the source and destination nodes and routes packets via the intermediate node. ROMM can have two to n phases in an $n \times n$ mesh, with each of the two phases (i.e., from source node to intermediate node and from intermediate node to destination node) may use some variation of DOR (i.e., XY-order or YX-order). It has been demonstrated that ROMM may saturate at a lower throughput than DOR in 2-D torus networks [15] and 2-D mesh networks [11]. Two-phase ROMM does not have much path diversity and therefore its load balancing properties are not strong. While increasing the number of phases typically reduces congestion, it comes at the cost of increased hardware complexity, for example in the form of additional bits in the routing header (cf. Section 3); further, more virtual channels are required, and a virtual channel must be assigned to each phase. The packet or the router needs to know/check what phase the packet is in. Uniform PROM is equivalent to n-phase ROMM while being significantly more efficient in its hardware implementation.

Valiant proposed a routing algorithm that randomly chooses a node in the network and routes via that node [17]. ROMM is similar to Valiant in that both use two-phase routing. While ROMM chooses the intermediate node from within the minimum rectangle, Valiant may choose an intermediate node from anywhere within the network. Consequently, Valiant is a non-minimal routing algorithm. Though Valiant achieves optimal worst-case throughput, it sacrifices average-case behavior and latency (due to non-minimal routing).

In O1TURN [11], Seo et al show that simply balancing traffic between XY and YX routing can guarantee provable worst-case throughput. O1TURN matches the average case behavior of ROMM for both global and local traffic. However, O1TURN's load balancing capability is not as good as PROM or PROMV since the path diversity in O1TURN is quite low.

We note that randomized routing algorithms such as ROMM, Valiant, O1TURN and PROM can result in out-of-order packet arrivals at the destination node, unlike DOR. This means that the destination node has to have a large enough buffer such that packets can be reordered to be processed in order in the processing element.

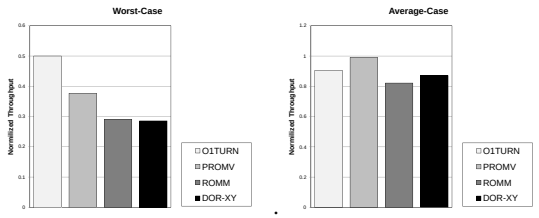
Classic adaptive routing schemes include the turn routing methods [5] and odd even routing [1]. These are general schemes that allow packets to take different paths through the network while ensuring deadlock freedom but do not specify the mechanism by which a particular path is selected. An adaptive routing policy determines what path a packet takes based on network congestion. Many policies have been proposed (e.g., [2, 7, 13, 14, 6]. PROM routing is oblivious routing and PROM achieves load balancing through randomization. The hardware cost for PROM is significantly lower than for adaptive algorithms that require local or global intelligence to adapt routes and also require routing logic to ensure that paths are selected to avoid deadlock. PROM avoids deadlock quite simply through appropriate virtual channel assignment, utilizing an observation first made in [12].

5. EXPERIMENTAL RESULTS

To evaluate the potential of PROM algorithms, we compared variable parametrized PROM (PROMV, described in Section 2.2) on a 2D mesh against two path-diverse algorithms with comparable hardware requirements, O1TURN and 2-phase ROMM, as well as dimension-order routing (DOR). First, we analytically assessed throughput on worst-case and average-case loads; then, we examined the performance in a realistic router setting through extensive simulation.

5.1 Ideal Throughput

To evaluate how evenly the various oblivious routing algorithms distribute network traffic, we analyzed the ideal throughput[10] in the same way as [15] and [16], both for worst-case throughput and for average-case throughput.



(a) Worst-Case (b) Average

Figure 6: Ideal Balanced Throughput

On worst-case traffic, shown in Figure 6(a), PROMV does significantly better than 2-phase ROMM and DOR, although it does not perform as well as O1TURN (which, in fact, has optimal throughput [11]). On average-case traffic, however, PROMV outperforms

[10]"ideal" because effects other than network congestion, such as head-of-line blocking, are not considered

the next best algorithm, O1TURN, by 10% (Figure 6(b)); PROMV wins in this case because it offers higher path diversity than the other routing schemes and is thus better able to spread traffic load across the network. Indeed, average-case throughput is of more concern to real-world implementations because, while every oblivious routing algorithm is subject to a worst-case scenario traffic pattern, such patterns tend to be artificial and rarely, if ever, arise in real NoC applications.

5.2 Simulation Setup

The actual performance on specific on-chip network hardware, however, is not fully described by the ideal-throughput model on balanced traffic. Firstly, both the router architecture and the virtual channel allocation scheme could significantly affect the actual throughput due to unfairness of scheduling and head-of-line blocking issues; secondly, balanced traffic is often not the norm: if network flows are not correlated at all, for example, flows with less network congestion could have more delivered traffic than flows with heavy congestion and traffic would not be balanced.

In order to examine the actual performance on a common router architecture, we performed cycle-accurate simulations of a 2D-mesh on-chip network under a set of standard synthetic traffic patterns, namely *transpose*, *bit-complement*, *shuffle*, and *bit-reverse* (See Table 1 for details). One should note that, like the worst-case traffic pattern above, these remain specific and regular traffic patterns and do not reflect all traffic on an arbitrary network; nevertheless, they were designed to simulate traffic produced by real-world applications [4], and so are often used to evaluate routing algorithm performance.

We focus on delivered throughput in our experiments, since we are comparing minimal routing algorithms against each other. We left out Valiant, since it is a non-minimal routing algorithm and because its performance has been shown to be inferior to ROMM and O1TURN [11]. While our experiments included both DOR-XY and DOR-YX routing, we did not see significant differences in the results, and consequently report only DOR-XY results.

Routers in our simulation were configured for 8 virtual channels per port, allocated either in one set (for DOR) or in two sets (for O1TURN, 2-phase ROMM, and PROMV; cf. Section 2.3), and then dynamically within each set. Because under dynamic allocation the throughput performance of a network can be severely degraded by head-of-line blocking [12] especially in path-diverse algorithms which present more opportunity for sharing virtual channels among flows, we were concerned that the true performance of PROM and ROMM might be hindered. We therefore repeated all experiments using Exclusive Dynamic Virtual Channel Allocation [8], a dynamic virtual channel allocation technique which reduces head-of-line blocking by ensuring that flits from a given flow can use only one virtual channel at each ingress port, and report both sets of results. Note that under this allocation scheme multiple flows *can* share the same virtual channel, and therefore it is different from having private channels for each flow, and can be used in routers with one or more virtual channels.

Characteristic	Configuration
Topology	8x8 2D MESH
Routing	PROMV($f_{max} = 1024$), DOR, O1TURN, 2-phase ROMM
Virtul channel allocation	Dynamic, EDVCA
Per-hop latency	1 cycle
Virtual channels per port	8
Flit buffers per VC	8
Average packet length (flits)	8
Traffic workload	bit-complement, bit-reverset, shuffle, transpose
Warmup / Analyzed cycles	20K / 100K

Table 1: Summary of network configuration

5.3 Simulation Results

Under conventional dynamic virtual channel allocation (Figure 7), PROMV shows better throughput than ROMM and DOR under all traffic patterns, and slightly better than O1TURN under *bit-complement* and *shuffle*. The throughput of PROMV is the same as O1TURN under *bit-reverse* and worse than O1TURN under *transpose*.

Using Exclusive Dynamic VC allocation improves results for all routing algorithms, and allows PROMV to reach its full potential: on all traffic patterns but *bit-complement*, PROMV performs best. The perfect symmetry of *bit-complement* pattern causes PROMV to have worse ideal throughput than DOR and O1TURN which have perfectly even distribution of traffic load all over the network; in this special case of the perfect symmetry, the worst network congestion increases as some flows are more diversified in PROMV.

Note that these results highlight the limitations of analyzing ideal throughput given balanced traffic (cf. Section 5.1). For example, while PROMV has better ideal throughput than O1TURN on *transpose*, head-of-line blocking issues allow O1TURN to perform better under conventional dynamic VC allocation; on the other hand, while the perfectly symmetric traffic of *bit-complement* enables O1TURN to have better ideal throughput than PROMV, it is unable to outperform PROMV under either VC allocation regime.

While PROMV does not guarantee better performance under *all* traffic patterns (as exemplified by *bit-complement*), it offers competitive throughput performance under a variety of traffic patterns because it can distribute traffic load among many network links. Indeed, we would expect PROMV to offer higher performance on most traffic loads because it shows 10% better average-case ideal throughput of balanced traffic (Figure 6(b)), which, once the effects of head-of-line blocking are mitigated, begins to more accurately resemble real-world traffic patterns.

6. CONCLUSIONS

We have presented a parametrizable oblivious routing scheme that includes n-phase ROMM and O1TURN as its extreme instantiations. Intermediate instantiations push traffic either inward or outward in the minimum rectangle defined by the source and destination. The complexity of a PROM router implementation is equivalent to O1TURN and simpler than 2-phase ROMM, but the scheme enables significantly greater path diversity in routes, thus showing 10% better performance on average in reducing the network congestion under random traffic patterns. The cycle-accurate simulations under a set of synthetic traffic patterns show that PROMV offers competitive throughput performance under various traffic patterns. It is also shown that if the effects of head-of-line blocking are mitigated, the performance benefit of PROMV can be significant.

Going from PROM to PRAM, where A stands for Adaptive is fairly easy. The probabilities of taking the next hop at each node can depend on local network congestion. With parametrized PROM, a local network node can adaptively control the traffic distribution simply and intuitively by adjusting the value of f in its routing decision. This may enable better load balancing especially under bursty traffic and we will investigate this in the future.

7. REFERENCES

[1] G.-M. Chiu. The Odd-Even Turn Model for Adaptive Routing. *IEEE Trans. Parallel Distrib. Syst.*, 11(7):729–738, 2000.

[2] W. J. Dally and H. Aoki. Deadlock-free adaptive routing in multicomputer networks using virtual channels. *IEEE*

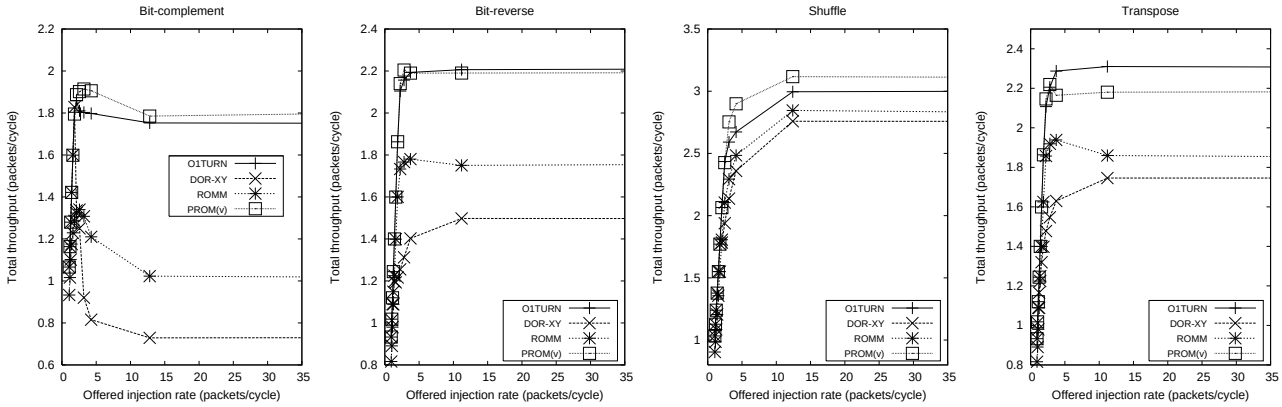


Figure 7: Dynamic VC Allocation

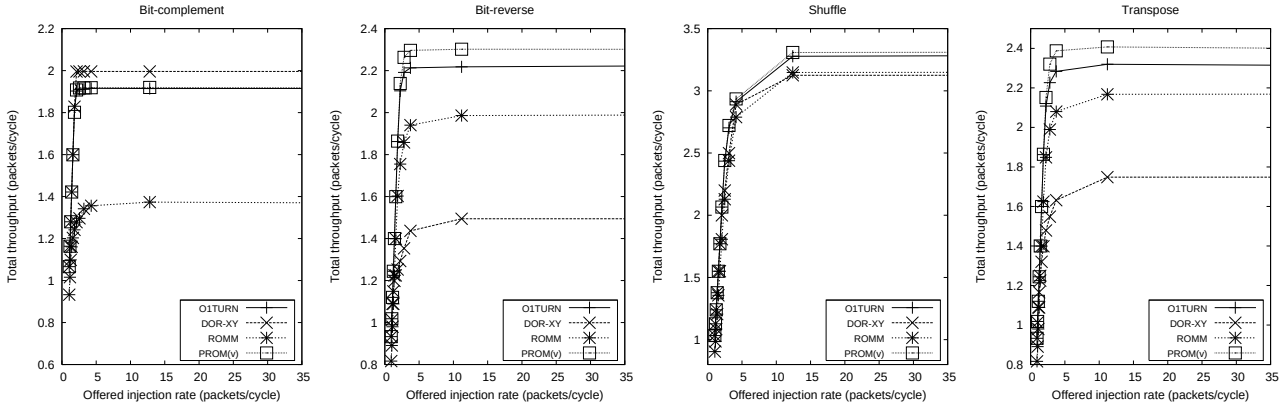


Figure 8: Exclusive-Dynamic VC Allocation

Transactions on Parallel and Distributed Systems, 04(4):466–475, 1993.

[3] W. J. Dally and C. L. Seitz. Deadlock-Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Trans. Computers*, 36(5):547–553, 1987.

[4] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.

[5] C. J. Glass and L. M. Ni. The turn model for adaptive routing. *J. ACM*, 41(5):874–902, 1994.

[6] P. Gratz, B. Grot, and S. W. Keckler. Regional Congestion Awareness for Load Balance in Networks-on-Chip. In *In Proc. of the 14th Int. Symp. on High-Performance Computer Architecture (HPCA)*, pages 203–214, Feb. 2008.

[7] H. J. Kim, D. Park, T. Theocharides, C. Das, and V. Narayanan. A Low Latency Router Supporting Adaptivity for On-Chip Interconnects. In *Proceedings of Design Automation Conference*, pages 559–564, June 2005.

[8] M. Lis, K. S. Shim, M. H. Cho, and S. Devadas. Guaranteed in-order packet delivery using Exclusive Dynamic Virtual Channel Allocation. Technical Report CSAIL-TR-2009-036 (http://hdl.handle.net/1721.1/46353), Massachusetts Institute of Technology, Aug. 2009.

[9] T. Nesson and S. L. Johnsson. ROMM Routing: A Class of Efficient Minimal Routing Algorithms. In *in Proc. Parallel Computer Routing and Communication Workshop*, pages 185–199, 1994.

[10] T. Nesson and S. L. Johnsson. ROMM routing on mesh and torus networks. In *Proc. 7th Annual ACM Symposium on Parallel Algorithms and Architectures SPAA'95*, pages 275–287, 1995.

[11] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi. Near-Optimal Worst-Case Throughput Routing for Two-Dimensional Mesh Networks. In *Proceedings of the 32nd Annual International Symposium on Computer Architecture (ISCA 2005)*, pages 432–443, 2005.

[12] K. S. Shim, M. H. Cho, M. Kinsy, T. Wen, M. Lis, G. E. Suh, and S. Devadas. Static Virtual Channel Allocation in Oblivious Routing. In *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip*, May 2009.

[13] A. Singh, W. J. Dally, A. K. Gupta, and B. Towles. GOAL: a load-balanced adaptive routing algorithm for torus networks. *SIGARCH Comput. Archit. News*, 31(2):194–205, 2003.

[14] A. Singh, W. J. Dally, B. Towles, and A. K. Gupta. Globally Adaptive Load-Balanced Routing on Tori. *IEEE Comput. Archit. Lett.*, 3(1), 2004.

[15] B. Towles and W. J. Dally. Worst-case traffic for oblivious routing functions. In *SPAA '02: Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–8, 2002.

[16] B. Towles, W. J. Dally, and S. Boyd. Throughput-centric routing algorithm design. In *SPAA '03: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 200–209, 2003.

[17] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *STOC '81: Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 263–277, 1981.