Mesh and Torus Chaotic Routing

Kevin Bolding and Lawrence Snyder
Department of Computer Science and Engineering
University of Washington, Seattle

Abstract

The chaos router is an adaptive nonminimal message router for multicomputers that is simple enough to compete with the fast, oblivious routers now in use in commercial machines. It improves on previous adaptive routers by using randomization, which eliminates the need for complex livelock protection and speeds the router.

The two-dimensional chaos router is shown to be theoretically sound and physically realizable. Extensive simulation studies compare chaos routing with oblivious and deflection routing in mesh and torus networks. Chaos routing is shown to be competitive for mesh networks and superior for torus networks. This high performance is, perhaps, unexpected for the mesh since there is no finite bound on the delivery time of any message.

1 Introduction

Chaotic routing is a randomizing, adaptive, message routing technique that has previously been shown (in simulations) to be effective for the binary n-cube (hypercube) topology [KS91]. The technique is *nonminimal*, i.e. messages do not necessarily take minimal paths to their destinations, and randomization plays a critical role in preventing *livelock*, i.e. in preventing messages from continually circulating in the network without being delivered [KS90]. Though the principles apply as well to batched message communication, chaotic routing assumes a continuous workload where messages are presented at the nodes for injection into the network at random real (or fine-grain discrete) times. Routing decisions are made locally in the routing nodes based on the destination address stored in the headers of the messages and the availability of outgoing channels. Messages can "cut-through" nodes if an outgoing channel is immediately available, but they may also be stored in the node if all outgoing channels are blocked, motivating short, e.g. 20 flit, messages.

Chaotic routing's success on the hypercube suggests that it might be effective for other topologies. Networks of low dimension are important because the trend in parallel computer design is towards mesh and torus based communication structures such as in the Intel Paragon, the Tera computer, and the Caltech Mosaic. But applying "chaos" in two dimensions poses several problems. First, chaotic routing relies on theoretical foundations, the theorems of which have only been proved for hypercubes. This is easily remedied by the results in Appendix B. The second problem is subtle and applies to any nonminimal adaptive router.

In a mesh with uniform random traffic, there is a "hot spot" in the center of the mesh. That is, the shortest message paths between two points tend to cross the center of the mesh, causing the resources in the center of the network (wires, buffers, etc.) to be more heavily used (see Figure 1). All adaptive routers will try to use these paths, but nonminimal adaptive routers, when they encounter congestion, will try to "deroute" a message away from the congestion. In such cases the hot spot can act as a barrier off of which messages can "bounce": That is, the forward paths are all congested, the message is derouted "backwards", and starts forward again, not having moved (or been able to move) away from the congestion. Though all nonminimal adaptive routers are subject to this type of behavior, the affect on performance varies depending on the type of router.

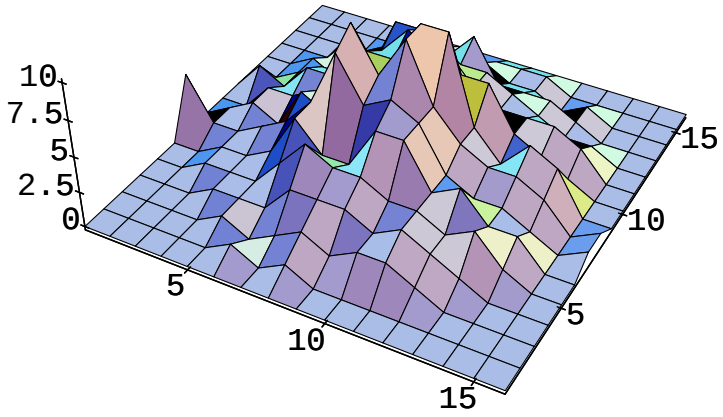

Figure 1: Average injection delay for a 256-node mesh.

Priority adaptive routers time stamp each message and routing is governed by the rule: oldest message first. Thus, messages bouncing off the hot spot will eventually age enough to be routed through it. Matters are not so certain for the chaos router, however. The primary advancement of chaotic routers [KS90] is that they eliminate the time stamping and the time consuming prioritization, replacing it with a reliance on randomization. But there is no mechanism that can assure the delivery of a message in a fixed finite time. A message can continue bouncing off the hot spot for an arbitrarily long period of time. Because of the probabilistic livelock freedom proved in Appendix B we know that the probability that the message has not been delivered in t steps goes to zero as t increases. So we can be confident that the message will be delivered eventually. But it could take a very very long time, leaving us with the question: Does chaotic routing work for the mesh?

In this paper, besides proving the "necessary theorems" for two-dimensional chaotic routing, we present simulation results comparing chaotic routers with oblivious routers and deflection, or "hot potato," routers on the mesh and torus topologies of sizes 64, 256 and 1024 nodes for both uniform random and hot spot loads. Three highlights are worth noting:

- On the mesh, chaotic routing performs as well as oblivious and deflection routing in throughput and latency for uniform traffic.

Thus, chaotic routing does work on a mesh, and in fact works about as well as other routers when the traffic is uniform.

- On the mesh chaotic routing performs better than oblivious and deflection routing in throughput and latency for nonuniform hot spot traffic.

Since it is likely that programs exhibit nonuniform traffic patterns the performance in such cases is perhaps more significant.

- On the torus chaotic routing is decidely superior to oblivious and deflection routing in throughput and latency.

The torus has better bisection bandwidth and better worst case path length than a mesh of similar size at the cost of a few extra wires. Its *vertex transitive* property aids the chaos router in giving it significantly better performance. The chaotic torus router is the best two dimensional packet router to our knowledge and thus a candidate for the next generation parallel computers.

2 Relationship to Previous Research

Borodin and Hopcroft [BH85] use the term *oblivious* to refer to routers for which the path of any message is completely determined by its [source, destination] pair. They proved that oblivious routers in an N node, d degree topology require $\sqrt{N}/d^{3/2}$ steps to route some permutations. The poor worst case performance and their fault intolerance would doom oblivious routers for use in multicomputers were it not for the fact that they are extremely simple, and thus fast. Accordingly, oblivious routers are the state-of-the-art for MIMD multicomputers such as those built by Intel, Ametech and NCUBE. Dally, Seitz and Flaig introduce oblivious routers of the type considered here for the mesh [Fla87] and torus [DS86] topologies.

Randomization was first applied in the context of message routing by Valiant and Brebner [VB81], though in a way quite different from the chaotic approach. Their technique — select a random intermediate destination for every message, route the message to that destination and then on to the true destination — was applied to batched routing in a hypercube. It could obviously be applied continuously [CS86] and in two-dimensional topologies. The main difficulty with this type of randomization is that it doubles the expected path length of any message.

An adaptive mesh router was proposed by Ngai and Seitz [NS89], but it differs from the chaotic approach by using timestamps and prioritization to prevent against livelock. Comparisons between prioritized and chaotic routers have been performed [KS91]. Adaptive wormhole routing using virtual channels has been studied by Duato [Dua91].

"Hot potato" or deflection routing is another scheme capable of adaptive routing [Smi81, Max89, FS91, Smi89]. The approach is synchronous and the time step is long enough to transmit an entire packet. At each step the incoming messages are paired with outgoing channels and are transmitted in the next step. The pairing is done in a variety of ways: Certain deterministic algorithms attempt to maximize the number of messages sent out productive channels, while others use a greedy algorithm with random selection. Those messages not receiving a productive channel are "deflected," *i.e.* derouted, out any available channel. Deflection routing differs from chaotic routing in several ways: Chaotic routing is not batched, *i.e.* does not require all headers to be present at once, thus permitting fast self-timed or high clock rate implementations, and better utilization of channels since messages can cut through, i.e. messages can be "in" multiple routers at once. Chaotic routing permits messages to wait for forward traffic to clear, thus reducing time consuming deroutes which necessarily delay the packet at least two "message times". Pausing for traffic to clear cushions the affects of bursts. Finally, chaotic routing resorts to derouting only under conditions of high load, when slower performance is inevitable.

3 Chaos Router Design

The chaotic router studied here is a two dimensional variant of the hypercube chaotic router [KS91]. The basic design of the chaos router is similar to a typical oblivious virtual cut-through router, with input and output frames connected by a crossbar switch, and hardware to increment or decrement the headers of messages as they pass through (see Figure 2). Two primary distinctions exist, though. The first is that the routing relation no longer specifies a single channel to traverse next, but instead a set of equally profitable channels. The first available profitable channel will be chosen for routing. The second distinction is the addition of a small (5 message) buffer, the *MultiQueue*, which holds messages for which no profitable channels are immediately available. Since

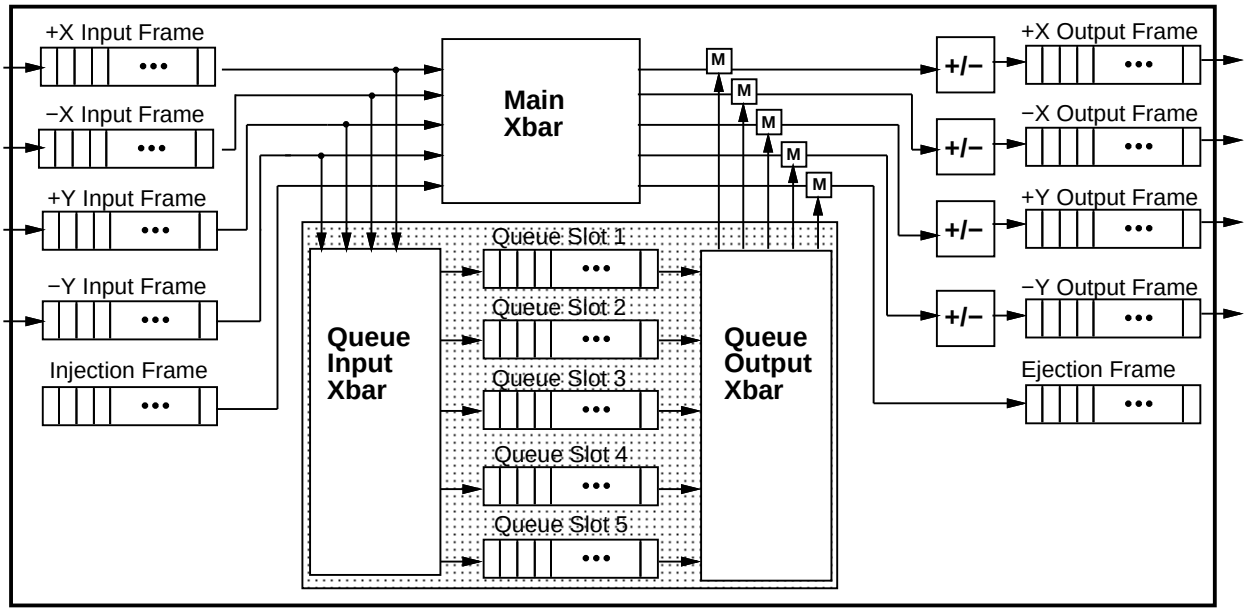

Figure 2: Two-dimensional chaos router diagram.

the buffer space is off of the critical resource path, messages in the queue do not block messages behind them. Messages enter the queue along a separate crossbar whenever they have been denied access to any profitable channel long enough for the entire message body to have arrived in the input frame. Also, in order to prevent deadlock, when a message is read from the queue into the output frame for channel i and the input frame for channel i is full, the message in the input frame is immediately read into the queue. Messages cannot enter the queue from the injection frame and messages which are awaiting the availability of the ejection buffer do not enter the queue, as they will be consumed by the processor. Whenever an output channel that is profitable for a message in the queue becomes available, the first message in the queue which can use that channel is sent from the queue through another crossbar to the output frame for that channel. When several messages can profitably use a channel at the same time, priority is given to messages in the queue (in FIFO order); among competing input frames messages are chosen randomly.

A critical situation occurs when a message is specified to be sent to the queue, but the queue is completely full. In such a situation, a message is randomly selected from the queue to be *derouted* along the first available channel so that room will be created in the queue for the newly arriving message. Derouting provides an additional factor of adaptivity to the chaos router and allows the use of a packet-exchange protocol for deadlock prevention [NS89].

4 The Network Model

The performance of different routing schemes varies much according to the model of the network being studied. For the studies of chaos routing, we use the following network model:

The network is a regular two-dimensional network of bi-connected nodes. Between each pair of

adjacent nodes in the network there is a *channel* consisting of control wires and a single data bus. The data bus is shared between the two directions, with arbitration occurring between message boundaries. The messages are fixed-size packets consisting of a header and several data words. The width of the data bus determines the size of a *flit* which is the amount of data which can be transmitted over the data bus in one cycle. We parameterize the packet size in our studies in terms of the number of flits per packet, L. Thus, for a 16 bit wide bus, a 20-flit message would contain a 16-bit header and 304 bits of data. We constrain our experiments to messages of size 20 flits, which is consistent with existing multicomputer designs.

We study the two-dimensional mesh and the two-dimensional torus in this investigation. To judge changes in performance with network size, we compare networks of 64, 256, and 1024 nodes.

5 Routers Studied

We study three routers in this paper: an oblivious router, the chaos router, and a deflection router. Most current multicomputers use some variant of oblivious routing. We chose a virtual cut-through oblivious router with input and output queueing to provide a baseline for current routing techniques. We provide results from a deflection router based on [FS91] to provide another baseline for comparison. Finally, we study the chaos router as presented in Section 3.

5.1 Oblivious Router

The oblivious router studied here is based upon the Kermani and Kleinrock [KK79] virtual cut-through router. Specifically, the router consists of a set of input and output frames and a crossbar switch which connects each input frame to every output frame. Each channel has one input frame and one output frame, each capable of holding exactly one fixed-size message[1]. The injection and delivery channels also have an input frame and an output frame, respectively, which are connected to the crossbar as well. Operation of the router proceeds in virtual cut-through fashion: whenever a message arrives in an input frame, it is immediately routed to the output frame for the next channel on its path to its destination,[2] if that output frame is available. It is not necessary to receive the entire message in an input frame before the header is sent to the output frame. If the output frame is not immediately available, the message will wait in the input frame until it becomes available, blocking any messages behind it if necessary. Operation of the channels proceeds in a similar demand-driven fashion.

5.2 Deflection Router

Deflection routing is an adaptive routing scheme in which messages arriving at a node are guaranteed to leave the node in the next routing cycle. An attempt to assign each message to a channel which reduces it distance to its destination is made, giving preference to messages with only a single profitable direction, followed by randomly assigning any remaining messages to the remaining free outgoing channels. The scheme does not quite fit the network model presented in Section 4,

[1] For the oblivious torus router, virtual channels are implemented by giving each physical channel two input and output frames.

[2] Since this is an oblivious router, there will be only one possible output channel at each routing step. In order to prevent deadlock, the channels must be traversed in order of dimension.

as channels must always be available and, thus, cannot be shared. We compensate for this by dividing each deflection routing channel into two uni-directional channels of one half the width of the chaos and oblivious routers. Also, the deflection protocol requires that all the headers of incoming messages arrive at the same moment, which is generally accomplished in their network models by using very high bandwidth channels capable of transmitting an entire message in each flit. Since our model includes multiple-flit messages, a routing decision may occur only once an entire message arrives at a node, resulting in a store-and-forward technique without virtual cut-through. Finally, in the analytical models presented in [FS91], all messages which arrive at a single destination node are removed from the network at once. In our simulations, we limit the delivery capability to the bandwidth of a standard network channel (one flit per cycle), as would be required in a realistic implementation.

6 Traffic Models

In order to compare the relative performance of the different routing schemes, a synthetic workload is applied to the simulated network and performance measurements are taken. The choice of the workload is critical when trying to compare the schemes. We provide simulation results for two workloads: *uniform random* and *hot spot* traffic.

6.1 Uniform Random Traffic

For uniform random traffic, each node presents a message to the network with a destination chosen uniformly randomly from each of the nodes in the network. The time between the presentation of messages is chosen randomly with a mean time based on the simulated applied load. The load is presented as a fraction of the maximum load the network could handle if there were no resource conflicts. This is computed as the point at which the utilization of channels cut by a bisection of the network reaches 100% assuming each message crosses this bisection with probability 0.5. If all channels of the network were utilized 100% of the time and all messages traveled on the shortest paths available, this maximum throughput would be obtained under uniform random traffic. The maximum applied load is then computed as the minimum inter-injection period for each network.

For the network model presented in Section 4, where one flit can be transmitted across a channel in one cycle, the minimum inter-injection period per node is $\frac{1}{2}\sqrt{N}L$ cycles for N-node meshes and $\frac{1}{4}\sqrt{N}L$ for N-node tori with messages of length L flits.

6.2 Hot Spot Traffic

Although uniform random traffic is a natural model of network traffic, many applications used on multicomputers create message traffic which has several *hot spot* nodes that receive considerably more traffic than the rest of the network. We attempt to model an abstract system by a synthetic load consisting of the same injection load as uniform random traffic, but with the destination distribution skewed in the following manner: ten "hot" nodes are chosen at the beginning of the simulation, each being four times as likely as the other N-10 nodes to be the destination of a message. Thus, these nodes become hot spots which could represent nodes that are used for synchronization or locking in a multicomputer application. The total loading of the network is the same as for uniform random traffic.

7 Simulations

Simulations for the networks and routers studied were conducted using a flit-based simulator written in C. The simulations were based on the *cycle* time unit. One cycle is the time necessary to transmit a single flit across a channel. Routing decisions can be made in a single cycle. Thus, if a message header enters a router at cycle t, it may enter the next router as early as cycle $t + 1$[3].

Simulations were run by applying the simulated load to the network in a continuous manner. Statistics were computed in intervals in which each node of the network has injected at least 50 messages. Average throughput and average latency were computed for each statistics interval and convergence was determined when the standard deviation of both of the measures over the most recent five intervals were less than 3%. The results presented here represent the averages and standard deviations of 3 to 5 runs.

The statistics reported here are the average throughput of the network normalized to the maximum throughput under uniform random load and the average latency of messages in cycles. We define latency as the time from presentation of a message to the network until the message has been completely removed from the network at its destination (notice that this does *not* include source queueing time).

8 Simulation Results

As described earlier, simulations were performed on mesh- and torus-connected networks of 64, 256, and 1024 nodes using random traffic and "hot spot" traffic for each of the three routing schemes studied. The average throughputs and average latencies are reported here.

To gauge performance, we concentrate on the high-load throughput and medium-load latencies. For low loads, all routing schemes are able to deliver the entire applied load without difficulty. The point at which the network saturates and the network is not able to keep up with the applied load is the interesting point in this case. Also, the shape of the throughput curve above saturation is important – *i.e.* does throughput ever decrease with increasing applied load? Latency is a more critical issue during lower load periods. At loads above saturation, since the network cannot keep up with the load applied, the latency of messages which do get through becomes of only peripheral interest. However, when the network is operating below saturation, it is latency that is the critical figure of merit. Thus, we will consider throughput saturation points and below-saturation latencies as the figures of merit for the networks studied.

We present full throughput and latency curves for 256-node networks with chaos and oblivious routing. We graph only throughput data for deflection routing because the store-and-forward nature of the router results in especially high latency figures. Since the shapes of the curves do not differ appreciably over different network sizes, we present only the 100% load throughput and 50% load latency points for other size networks. The raw data is given for all networks in Appendix A.

8.1 Mesh networks

For mesh networks under uniform random traffic, all three routing schemes give similar throughput results (Figures 3 and 4). The throughput reaches 80–90% of the maximum throughput in each

[3]For the deflection router, since the entire message must be received before transmission can begin, a message entering a router at cycle t will not leave until cycle $t + L$.

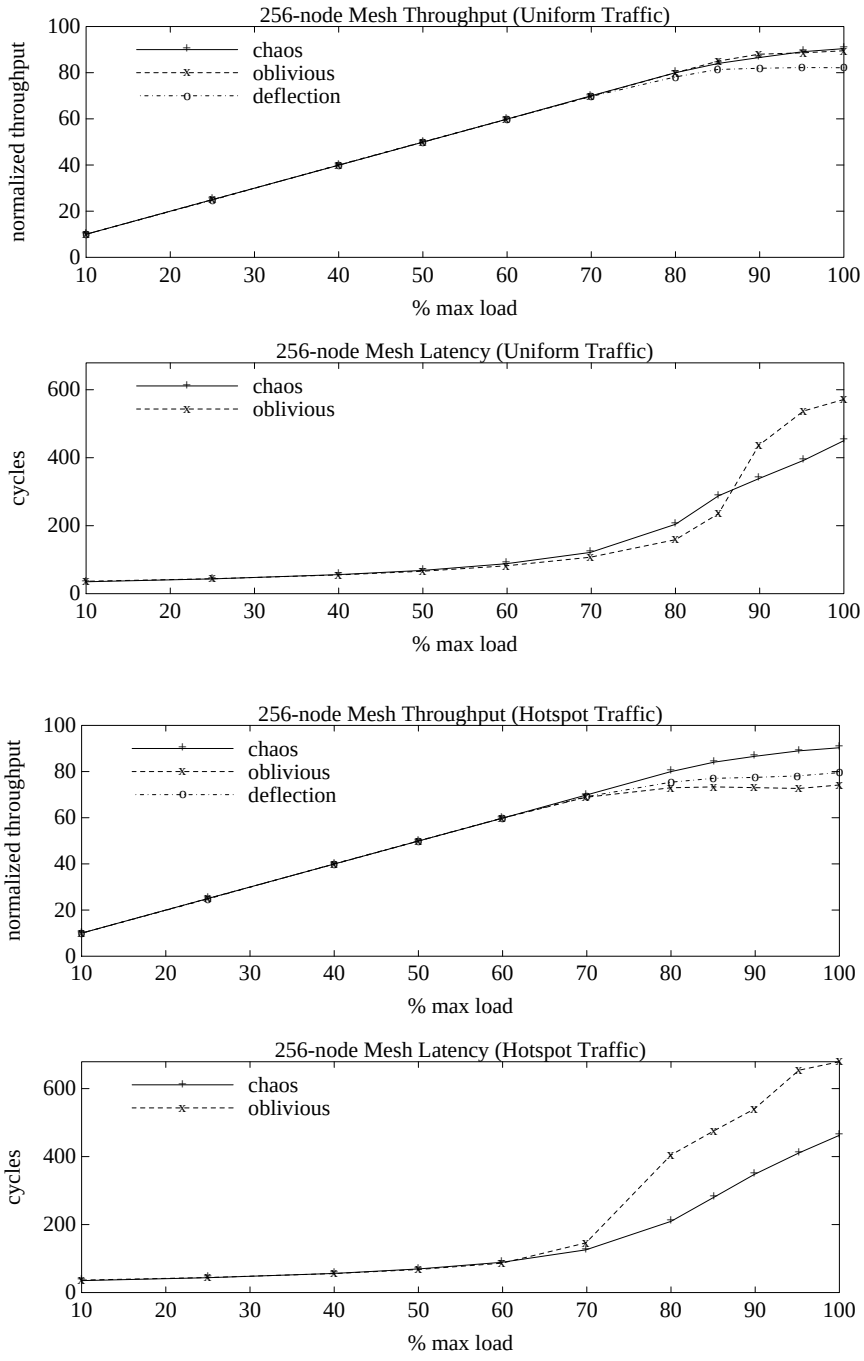

Figure 3: 256-node mesh results

case and there is no decline in throughput as the load approaches 100%. The latencies for the oblivious and chaos routers remain very close throughout all load ranges, with the chaos router giving slightly better values. The performance of the adaptive schemes is actually lower than would be expected: the additional hardware gives little or no benefit under random loads. This is the re-

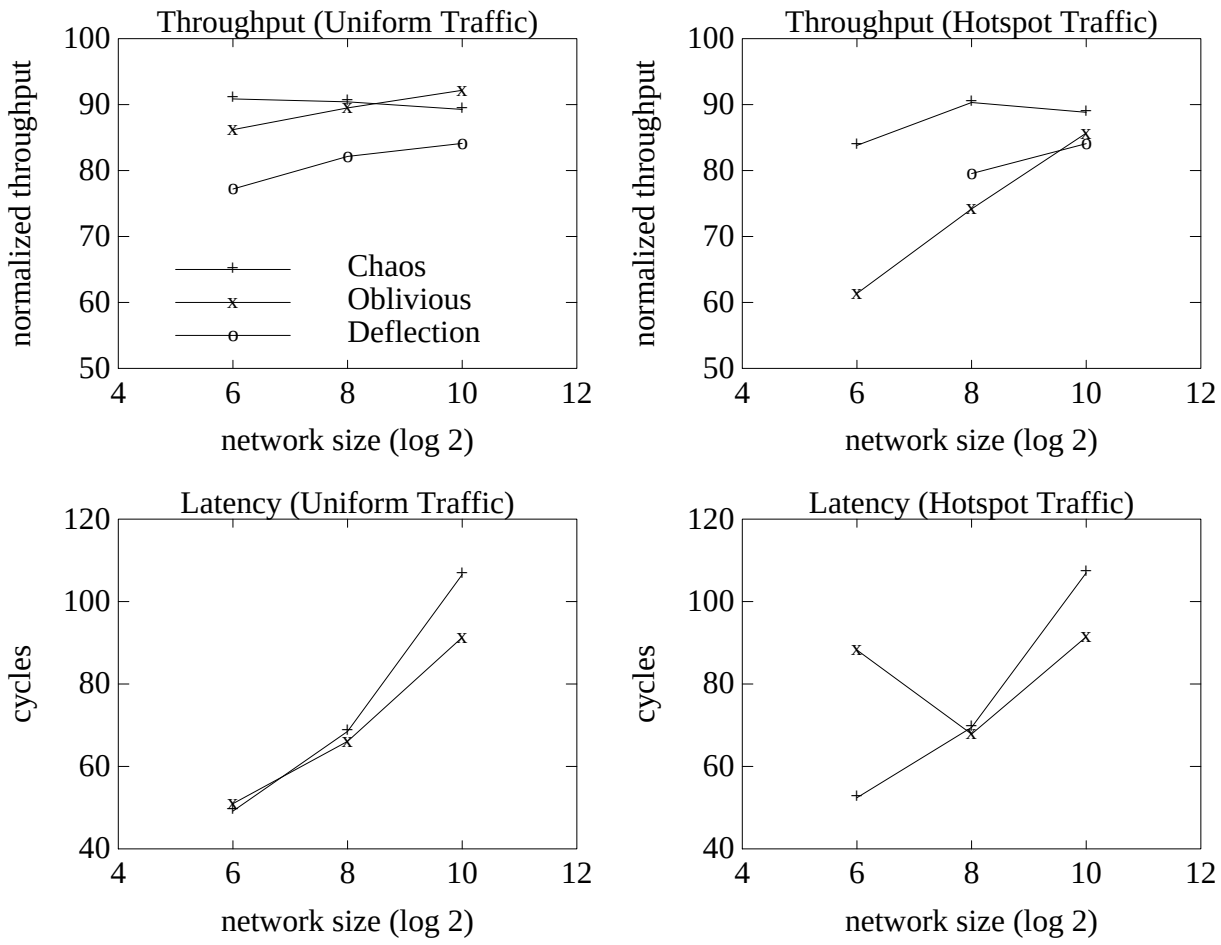

Figure 4: Mesh throughput (100% applied load) and latency (50% applied load) vs. network size

sult of the large hot spot inherently present in the center of mesh-connected networks (Figure 1) which creates a substantial barrier to cross-network traffic. While the oblivious router sends messages straight through the hot spot, even if slowly, the adaptive routers attempt to route messages around the congested center. However, since the area is so large, messages tend to bounce around the periphery for long periods of time, resulting in very long paths from source to destination.

When hot spots are added to the mesh, chaos routing becomes distinctly better than oblivious and deflection routing for small networks, with the benefit declining as network size decreases (Figures 3 and 4)[4]. This can be seen as the oblivious throughput increases with network size while the chaos throughput remains relatively stable. For small networks the oblivious throughput is especially low, resulting in high latencies from the additional congestion. This behavior is due to the fact that the central hot spot presents a more formidable barrier in larger networks – for small networks the ten hot spots influence the traffic greatly and the chaos router performs better, but as the network grows, the central hot spot dominates the traffic flow and the oblivious router performance improves. Thus, for smaller network sizes, the adaptivity of the chaos router proves

[4]Data is not currently available for the 64-node deflection router

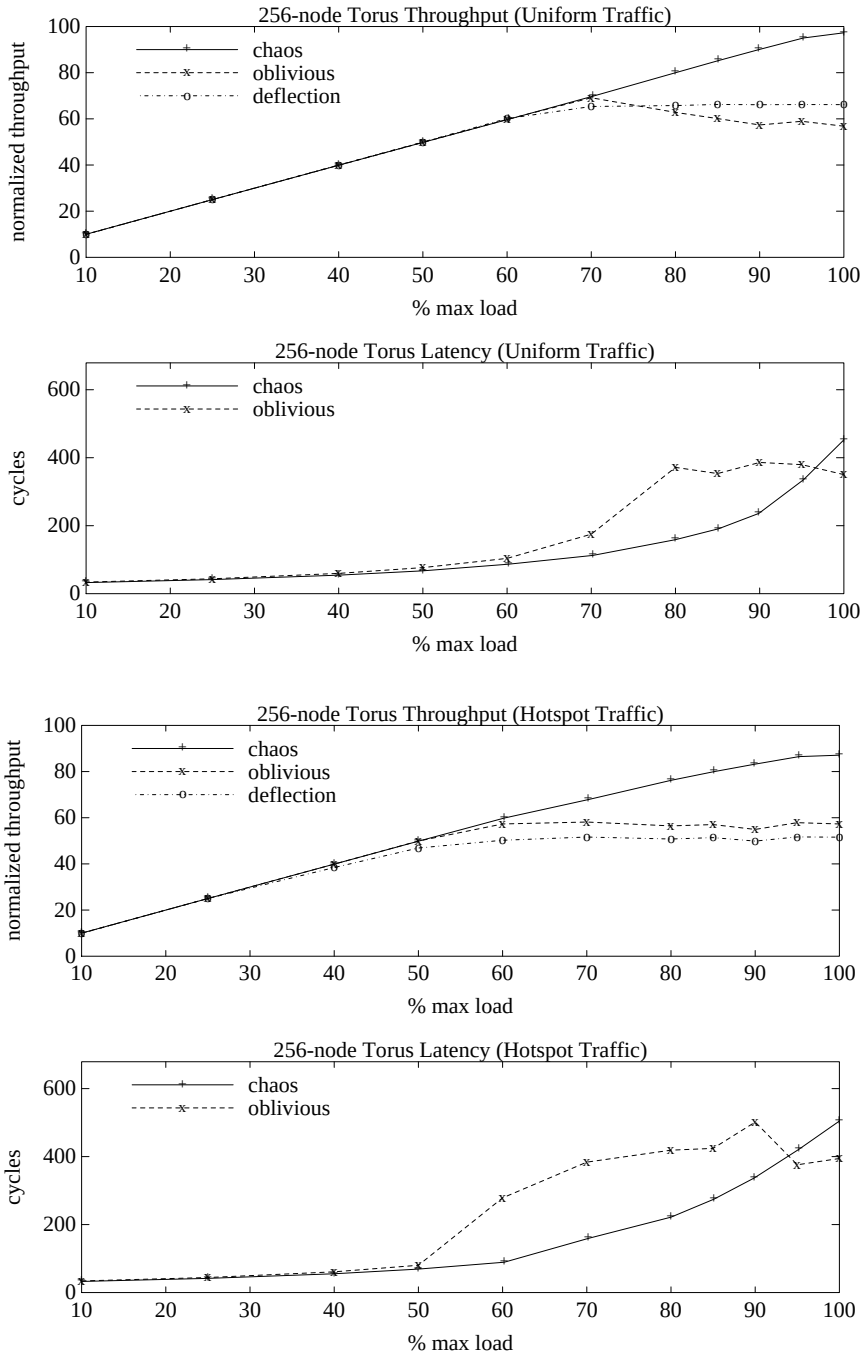

Figure 5: 256-node torus results

useful in the mesh, but this advantage diminishes with increasing network size.

8.2 Torus networks

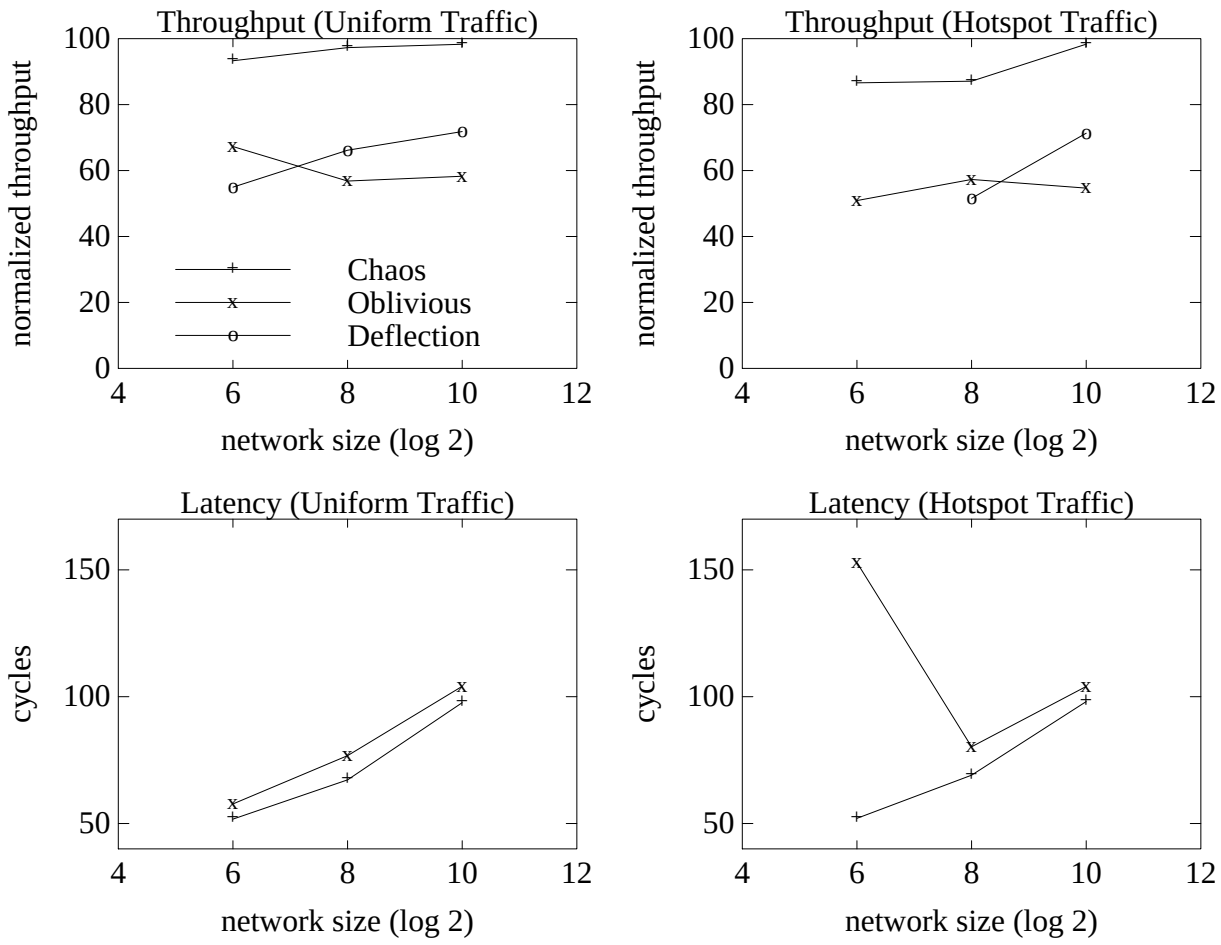

Figure 6: Mesh throughput (100% applied load) and latency (50% applied load) vs. network size

For torus-connected networks, the chaos router performs significantly better than both the oblivious and deflection routers in all respects (Figures 5 and 6). Since a torus is *vertex-transitive*, *i.e.* the network appears the same to every node, traffic is uniformly distributed throughout the network, unlike the mesh. This translates into a performance advantage for the chaos router, which allows messages to use the entire network without the constraints of oblivious dimension-order routing. The chaos router achieves near-maximum throughput under random traffic for all network sizes considered, while the oblivious and deflection routers top out at 55–70% performance. Again, the latencies remain low for low to medium loads, indicating very superior performance.

A disturbing property of the torus oblivious router is that the maximum throughput is achieved at less than the maximum load. This is due to a disturbance of the vertex-transitivity of the network introduced by the addition of deadlock prevention. Since the virtual-channel deadlock prevention scheme applied [DS87] distinguishes certain nodes as "special" in order to break cycles, the uniformity of the network is broken and hot spots are introduced at high loads. This results in the degradation of throughput as load is increased. The chaos and deflection routers preserve the uniformity of the network and do not exhibit this behavior.

For hot spot traffic, the adaptive routers perform well and the oblivious router suffers an earlier leveling of throughput than with random traffic[5]. The advantage of chaos routing is clearly apparent here, as throughput and latency are only minimally affected by the non-uniform traffic load. Overall, the chaos router is clearly superior to the oblivious and deflection routers for the torus network.

9 Conclusions

We have presented a two-dimensional variant of the hypercube chaos router and shown it to be a viable router. The theoretical foundations of the two-dimensional router have been presented. A working design of the chaos router has been given which is capable of competing with oblivious routers for critical-path complexity. The performance of the chaos router is comparable to oblivious routers for meshes with random traffic and better with hot spot traffic. For torus networks, the chaos router performs much better than the oblivious and deflection routers.

Acknowledgments

We wish to thank Sen-ching Cheung for running hundreds of simulations of the routers studied, Carl Ebeling for his help in designing the router, and Smaragda Konstantinidou for her original chaos design work.

This work was supported in part by the Office of Naval Research under Grant N00014-89-J-1368, the National Science Foundation under Grant MIP-9013274, and an Intel Foundation Graduate Fellowship.

References

[BH85] A. Borodin and J. E. Hopcroft. Routing, merging and sorting on parallel models of computation. *Journal of Computer and System Sciences*, 30:130–145, 1985.

[CS86] Y. Chang and J. Simon. Continuous routing and batch routing on the hypercube. In *Proceedings of Principles of Distributed Computing*, pages 272–281. ACM, 1986.

[DS86] W. Dally and C. Seitz. The torus routing chip. *Journal of Distributed Computing*, 1(3), 1986.

[DS87] W. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, C-36(5):547–553, May 1987.

[Dua91] J. Duato. Deadlock-free adaptive routing algorithms for multicomputers. *Technique et Science Informatiques*, 10(4):275–285, 1991.

[Fla87] C. Flaig. VLSI mesh routing systems. Master's thesis, California Institute of Technology, May 1987.

[5]The delivery capability of the routers was increased to 4X for the 64-node oblivious and chaos torus networks with hot spot traffic. Data is not currently available for the 64-node deflection router

[FS91] Chien Fang and Ted Szymanski. An analysis of deflection routing in multi-dimensional regular mesh networks. In *Proceedings of IEEE INFOCOM '91*, pages 859–868. IEEE, April 1991.

[KK79] P. Kermani and L. Kleinrock. Virtual cut-through: A new computer communication switching technique. *Computer Networks*, 3:267–286, 1979.

[KS90] Smaragda Konstantinidou and Lawrence Snyder. The chaos router: A practical application of randomization in network routing. In *Proceedings of the 2nd Symposium on Parallel Algorithms and Architectures*, pages 21–30. ACM, 1990.

[KS91] Smaragda Konstantinidou and Lawrence Snyder. Chaos router: Architecture and performance. In *Proceedings of the 18th International Symposium on Computer Architecture*, pages 212–221. IEEE, May 1991.

[Max89] N. F. Maxemchuk. Comparison of deflection and store-and-forward techniques in the manhattan street and shuffle-exchange networks. In *Proceedings of IEEE INFOCOM '89*, pages 800–809. IEEE, 1989.

[NS89] J. Y. Ngai and C. L. Seitz. A framework for adaptive routing in multicomputer networks. In *Proceedings of the Symposium of Parallel Algorithms and Architectures*, pages 1–9. ACM, 1989.

[Smi81] B. J. Smith. Architecture and applications of the HEP multiprocessor computer system. In *Proceedings of SPIE*, pages 241–248, 1981.

[Smi89] David Smitley. Design tradeoffs for a high speed network node. Technical Report SRC-TR-89-007, Supercomputing Research Center Institute for Defense Analysis, Bowie, Maryland, July 1989.

[VB81] L. G. Valiant and G.J. Brebner. Universal schemes for parallel communication. In *Proceedings of the 13th ACM Symposium on Theory of Computing*, pages 263–277, 1981.

Appendix A: Numerical Results

Data for 64, 256, and 1024-node mesh and torus networks with uniform random and hot spot traffic. Statistics presented are the means and standard deviations for normalized throughput and latency over three runs using different random number seeds.

64-Node Mesh (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.98	39.94	49.84	60.12	70.16	80.02	85.36	88.62	90.44	90.86
std xpt	0.00	0.04	0.08	0.05	0.13	0.14	0.21	0.44	1.56	0.61	0.61
mean lat	28.10	33.46	41.21	49.17	61.29	78.87	110.52	142.07	177.97	213.52	226.28
std lat	0.13	0.31	0.27	0.42	0.99	3.04	6.75	6.02	8.77	3.84	4.01

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.94	49.82	60.08	70.14	79.66	84.02	86.24	86.78	86.20
std xpt	0.00	0.00	0.08	0.07	0.16	0.15	0.36	0.60	1.45	1.60	0.70
mean lat	29.00	34.58	43.08	50.97	64.85	84.53	140.30	228.21	281.70	297.65	324.07
std lat	0.06	0.25	0.36	0.65	1.62	5.03	15.64	18.37	31.15	11.47	12.95

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.74	40.00	49.94	60.26	69.86	76.52	77.14	76.72	77.18	77.20
std xpt	0.00	0.05	0.09	0.15	0.30	0.22	1.09	1.16	0.91	1.28	0.97
mean lat	275.85	282.89	297.98	313.65	344.96	381.67	443.72	459.15	465.28	466.82	465.58
std lat	1.39	0.45	0.72	2.31	3.76	9.40	6.82	4.14	2.55	2.35	0.77

256-Node Mesh (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.92	49.90	59.80	69.90	80.00	84.02	86.54	89.12	90.42
std xpt	0.00	0.00	0.04	0.00	0.13	0.06	0.14	0.50	0.42	0.32	0.39
mean lat	35.01	43.51	56.10	68.50	88.11	121.41	204.12	287.84	338.42	392.01	450.16
std lat	0.02	0.12	0.27	0.77	0.59	2.96	11.81	7.41	12.59	15.42	11.87

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.94	49.90	59.72	69.74	79.96	85.04	88.00	88.62	89.50
std xpt	0.00	0.00	0.05	0.00	0.04	0.21	0.14	0.16	0.66	1.20	0.67
mean lat	36.54	43.92	55.16	66.08	81.98	107.29	158.84	235.17	436.44	537.33	571.43
std lat	0.09	0.10	0.21	0.51	0.72	1.75	4.47	8.50	47.99	18.90	12.16

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.80	40.06	50.02	59.92	69.70	78.16	81.38	81.92	82.22	82.14
std xpt	0.00	0.00	0.05	0.07	0.04	0.14	0.39	0.77	0.70	0.35	0.30
mean lat	489.65	498.55	515.62	533.11	568.28	635.84	745.73	817.43	885.50	906.09	914.57
std lat	1.09	1.66	1.70	2.32	2.15	4.60	10.89	15.70	9.24	5.31	8.85

1024-Node Mesh (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	40.00	49.90	59.93	69.80	79.20	82.73	85.67	87.60	89.27
std xpt	0.00	0.00	0.00	0.00	0.05	0.14	0.08	0.09	0.33	0.16	0.29
mean lat	48.39	62.44	83.58	106.55	143.91	211.04	426.91	593.96	696.56	818.04	920.75
std lat	0.04	0.04	0.21	0.89	1.03	2.25	15.36	0.73	3.27	1.46	16.89

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	40.00	49.90	59.90	69.80	79.70	85.00	89.97	92.20	92.17
std xpt	0.00	0.00	0.00	0.00	0.00	0.00	0.22	0.00	0.17	0.37	0.25
mean lat	51.32	61.30	76.29	91.31	113.02	146.76	207.53	271.99	421.14	807.24	1016.45
std lat	0.07	0.04	0.33	0.25	0.71	1.02	2.20	3.57	18.02	11.44	21.61

Deflection Routing

% load	10	20	40	50	60	70	80	85	90	95	100
mean xpt	10.00	19.90	40.00	49.98	60.06	69.68	78.32	82.10	83.94	84.48	84.12
std xpt	0.00	0.00	0.00	0.04	0.05	0.07	0.12	0.41	0.29	0.20	0.10
mean lat	915.94	921.98	949.80	975.49	1032.79	1173.05	1381.15	1546.98	1696.88	1769.75	1815.87
std lat	0.85	0.71	1.41	0.85	2.35	7.26	4.49	29.23	23.93	7.43	3.34

64-Node Mesh (Hot Spot Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.98	39.90	49.94	60.00	69.94	79.16	80.66	82.32	85.32	83.82
std xpt	0.00	0.04	0.00	0.08	0.21	0.54	2.41	3.53	4.16	4.78	5.89
mean lat	28.24	34.06	42.23	52.41	67.38	95.31	148.70	205.85	238.93	260.89	307.58
std lat	0.20	0.20	0.65	0.88	5.27	15.63	30.15	56.41	56.55	42.80	67.44

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.98	39.94	48.70	57.34	60.44	59.98	62.04	61.92	63.00	61.36
std xpt	0.00	0.04	0.05	2.60	5.98	5.59	7.42	6.63	6.97	6.47	6.85
mean lat	29.04	35.27	46.74	88.24	146.34	256.72	346.68	360.21	388.57	399.17	397.32
std lat	0.17	0.43	3.08	64.80	87.60	56.14	70.48	25.55	39.23	18.99	20.95

256-Node Mesh (Hot Spot Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.94	49.90	59.72	69.80	79.98	84.06	86.58	88.98	90.32
std xpt	0.00	0.00	0.05	0.00	0.04	0.00	0.12	0.48	0.38	0.54	0.65
mean lat	35.07	43.71	56.29	69.47	88.99	125.68	209.44	280.39	347.62	410.86	462.55
std lat	0.04	0.16	0.44	0.55	2.14	1.95	13.81	5.85	15.35	13.42	11.05

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.92	49.88	59.74	68.86	72.98	73.36	73.04	72.68	74.18
std xpt	0.00	0.00	0.04	0.04	0.08	2.03	3.06	3.92	3.15	4.19	4.22
mean lat	36.65	44.20	55.89	67.85	86.33	146.17	404.74	475.77	540.07	654.20	679.07
std lat	0.09	0.10	0.32	0.97	1.65	51.82	20.18	31.20	31.55	55.79	34.17

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.80	40.04	50.00	59.90	69.14	75.40	77.06	77.48	78.12	79.54
std xpt	0.00	0.00	0.05	0.09	0.00	0.72	3.22	3.38	5.34	4.40	1.40
mean lat	491.79	501.02	519.76	541.66	580.57	685.81	826.77	870.31	904.49	917.06	914.57
std lat	1.43	1.56	2.54	1.34	10.91	64.37	92.47	63.54	85.82	70.19	18.42

1024-Node Mesh (Hot Spot Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	40.00	49.90	59.90	69.70	79.30	82.63	85.17	87.83	88.83
std xpt	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.19	0.33	0.25	0.34
mean lat	48.46	62.68	83.90	106.97	144.48	212.02	433.12	593.56	701.25	821.30	927.16
std lat	0.07	0.04	0.17	0.65	0.45	3.85	14.46	5.52	5.35	6.58	11.48

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	40.00	49.90	59.90	69.80	79.90	84.07	85.17	84.53	85.67
std xpt	0.00	0.00	0.00	0.00	0.00	0.00	0.22	0.86	2.37	1.97	0.78
mean lat	51.31	61.30	76.18	91.44	113.14	148.51	211.49	418.36	661.83	952.56	1031.76
std lat	0.11	0.19	0.31	0.23	0.55	0.61	1.28	95.93	113.84	5.30	8.34

Deflection Routing

% load	10	20	40	50	60	70	80	85	90	95	100
mean xpt	10.00	19.90	39.92	50.00	60.06	69.56	78.00	81.62	83.58	84.14	84.08
std xpt	0.00	0.00	0.21	0.00	0.05	0.08	0.30	0.42	0.25	0.28	0.28
mean lat	916.79	922.19	948.61	976.43	1035.11	1179.02	1388.08	1521.52	1672.59	1752.57	1800.90
std lat	1.32	0.88	2.25	1.77	3.78	6.19	3.58	22.72	19.49	11.62	8.46

64-Node Torus (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	91	95	100
mean xpt	10.00	24.92	39.80	49.78	59.70	70.38	79.80	85.28	90.48	92.74	93.30
std xpt	0.00	0.04	0.00	0.15	0.17	0.16	0.30	0.50	1.10	1.29	1.57
mean lat	27.22	33.42	42.34	51.73	64.62	82.74	113.27	133.73	183.85	193.62	217.24
std lat	0.10	0.36	0.60	1.09	1.07	3.69	5.40	3.44	9.65	4.56	8.05

Oblivious Routing

% load	10	25	40	50	60	70	80	85	91	95	100
mean xpt	10.00	24.96	39.82	50.02	59.68	69.86	68.86	66.48	65.94	67.08	67.28
std xpt	0.00	0.05	0.07	0.07	0.07	0.64	1.09	0.88	1.30	1.41	0.69
mean lat	27.94	34.75	46.18	57.72	77.29	136.69	206.29	212.02	213.10	215.97	224.85
std lat	0.06	0.31	0.64	1.59	3.23	10.57	5.77	3.49	5.61	1.61	6.59

Deflection Routing

% load	10	25	40	50	60	70	80	85	91	95	100
mean xpt	9.92	25.04	39.96	49.92	54.72	56.38	55.70	55.78	55.84	55.62	54.96
std xpt	0.04	0.10	0.05	0.19	0.56	1.30	0.58	1.12	0.94	1.29	0.98
mean lat	226.81	239.56	264.54	307.13	354.75	357.24	359.80	359.32	358.15	360.45	361.70
std lat	0.80	1.09	0.83	7.45	5.17	2.20	3.19	0.82	1.09	3.15	3.21

256-Node Torus (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.80	59.84	69.82	79.98	85.24	90.00	95.04	97.28
std xpt	0.00	0.00	0.00	0.00	0.19	0.35	0.16	0.23	0.21	0.10	1.03
mean lat	32.60	41.41	54.49	67.21	86.82	112.63	158.89	190.83	236.16	335.06	452.56
std lat	0.09	0.08	0.49	0.68	1.47	0.57	2.40	4.22	10.76	6.39	12.81

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.78	59.74	69.14	62.80	60.12	57.36	58.94	56.82
std xpt	0.00	0.00	0.00	0.04	0.19	0.99	2.68	4.58	2.45	2.84	1.26
mean lat	34.11	43.78	59.69	76.75	103.82	175.07	371.54	353.40	386.08	379.63	350.29
std lat	0.03	0.07	0.56	1.14	2.03	9.58	54.28	25.24	22.03	42.25	31.39

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	9.92	25.02	40.00	49.96	60.12	65.36	65.76	66.24	66.12	66.20	66.16
std xpt	0.04	0.04	0.00	0.05	0.12	0.49	0.59	0.62	0.10	0.48	0.58
mean lat	384.95	397.64	419.45	446.87	499.89	585.66	598.01	600.92	604.44	603.48	603.96
std lat	0.83	0.58	0.59	1.41	4.54	5.22	3.10	1.24	2.60	2.01	2.56

1024-Node Torus (Uniform Random Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.90	59.63	69.33	79.90	85.03	89.93	95.20	98.30
std xpt	0.00	0.00	0.00	0.00	0.05	0.05	0.08	0.05	0.12	0.29	0.50
mean lat	43.23	57.36	77.50	97.64	126.13	170.39	245.77	304.39	384.02	526.87	898.34
std lat	0.01	0.10	0.16	0.43	1.27	0.74	0.44	1.37	3.34	13.05	39.15

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.80	59.60	69.43	72.73	54.67	54.67	55.40	58.27
std xpt	0.00	0.00	0.00	0.00	0.00	0.33	1.18	3.63	2.25	2.77	5.78
mean lat	45.64	58.80	80.65	104.09	137.68	199.66	441.70	640.12	738.73	788.16	605.01
std lat	0.03	0.15	0.20	0.48	0.92	1.36	9.97	28.51	131.94	146.23	117.55

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.80	40.03	50.00	59.93	69.63	71.67	71.67	71.57	71.90	71.83
std xpt	0.00	0.14	0.05	0.00	0.05	0.12	0.09	0.09	0.05	0.16	0.17
mean lat	704.67	717.80	742.87	772.87	821.82	956.19	1086.06	1097.57	1104.39	1106.64	1110.12
std lat	0.15	1.38	0.50	1.26	0.49	5.61	1.65	1.21	0.67	1.27	0.77

64-Node Torus (Hot Spot Traffic)

Chaos Routing (4X delivery rate)

% load	10	25	40	50	60	70	80	85	91	95	100
mean xpt	10.00	24.93	39.97	49.70	59.57	70.00	79.30	83.33	83.60	86.23	86.60
std xpt	0.00	0.05	0.12	0.16	0.25	0.45	1.30	2.09	2.61	3.71	2.13
mean lat	26.78	32.90	41.86	52.04	65.77	87.91	126.36	146.04	172.60	185.32	207.55
std lat	0.05	0.10	0.22	1.20	2.73	4.46	7.55	9.70	12.09	11.23	6.20

Oblivious Routing (4X delivery rate)

% load	10	25	40	50	60	70	80	85	91	95	100
mean xpt	10.00	24.93	40.00	46.43	46.50	47.40	47.30	47.83	49.40	47.27	50.90
std xpt	0.00	0.05	0.08	3.07	4.81	4.62	5.53	2.01	2.84	1.60	3.21
mean lat	28.10	36.25	52.77	152.88	251.78	280.01	282.25	283.16	270.38	315.90	287.63
std lat	0.04	0.22	1.86	53.41	16.69	27.64	26.12	7.70	27.60	14.48	18.87

256-Node Torus (Hot Spot Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.84	59.86	67.88	76.20	79.96	83.16	86.48	87.10
std xpt	0.00	0.00	0.00	0.08	0.20	4.35	7.35	10.13	12.78	14.05	12.96
mean lat	32.70	41.73	55.31	69.05	89.25	159.46	221.82	274.74	337.27	421.60	504.57
std lat	0.10	0.16	0.50	0.97	1.77	84.49	106.18	134.05	146.58	108.28	82.81

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.90	57.30	58.14	56.46	57.04	54.94	57.86	57.30
std xpt	0.00	0.00	0.00	0.17	3.08	2.53	0.60	1.42	2.38	2.02	0.86
mean lat	34.19	44.21	61.00	80.25	278.57	383.71	418.91	424.15	501.58	375.87	394.41
std lat	0.06	0.22	0.22	1.22	63.39	39.07	54.33	45.40	60.84	17.15	66.11

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	9.92	25.02	38.42	46.88	50.24	51.64	50.78	51.38	49.88	51.68	51.58
std xpt	0.04	0.04	3.21	5.99	6.63	7.21	6.86	6.25	9.04	7.87	6.19
mean lat	385.55	401.10	571.94	629.85	775.44	755.68	777.72	750.42	821.97	768.35	753.30
std lat	0.63	1.76	273.81	239.34	118.24	100.44	109.44	63.96	189.35	114.82	72.11

1024-Node Torus (Hot Spot Traffic)

Chaos Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.90	59.67	69.33	79.20	85.10	90.00	95.27	98.33
std xpt	0.00	0.00	0.00	0.00	0.05	0.05	0.36	0.08	0.08	0.21	0.31
mean lat	43.21	57.35	77.55	98.07	127.03	170.21	242.79	304.91	384.79	540.51	900.73
std lat	0.04	0.11	0.29	0.36	0.88	0.44	1.22	1.99	1.30	11.49	29.34

Oblivious Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	25.00	39.90	49.80	59.60	69.90	57.23	55.37	54.80	52.60	54.67
std xpt	0.00	0.00	0.00	0.00	0.00	0.08	3.93	3.90	2.64	1.77	0.83
mean lat	45.64	58.94	81.03	104.04	140.51	225.90	640.02	670.64	632.84	851.68	696.35
std lat	0.05	0.17	0.30	0.63	1.36	12.71	107.53	114.69	78.92	111.51	117.68

Deflection Routing

% load	10	25	40	50	60	70	80	85	90	95	100
mean xpt	10.00	24.80	40.00	49.97	59.90	69.43	71.27	71.13	71.10	71.33	71.33
std xpt	0.00	0.14	0.00	0.09	0.00	0.12	0.05	0.31	0.16	0.12	0.17
mean lat	704.58	717.89	744.63	774.71	826.48	967.66	1087.28	1102.65	1108.72	1112.37	1115.66
std lat	0.70	0.98	0.51	0.62	1.05	9.94	0.77	4.56	2.36	0.65	1.83

Appendix B: Theoretical Considerations

It is necessary to show that chaotic routers are both deadlock free and livelock free. Deterministic deadlock freedom is straightforward for routers that use a message exchange protocol [NS89]. The case analysis for the two-dimensional case matches the hypercube case [KS90].

Deterministic livelock freedom, that every message is delivered after a given period of time, is not true for chaotic routers. However, probabilistic livelock freedom – the probability that a message remains undelivered after t seconds goes to zero as t increases – is true. The following sketch of th proof mirrors the hypercube argument [KS90].

The message's path through the torus network is described by a sequence of *moves*. The distance of a message from its destination is the Manhattan distance, which can be at most $\sqrt{N}-1$. Clearly, for $\sqrt{N}$ even, every move either increases or decreases the message's distance to the destination. The probability of moving closer is the probability of being routed $p \geq \epsilon$, which is established in a theorem arguing that a message remains in the multiqueue a bounded amount of time and is thus subjected to only a bounded number of random derouting decisions (see [KS90]). The probability of moving further is $q = 1 - p$.

Let us define a *game* as a sequence of $\sqrt{N}$ moves. Message M starts game i at distance a_i and finishes at distance a_{i+1}. Let l_i denote the event that M was not delivered during game i and w_i the event that M was delivered during game i.

Let $Q(i)$ be the probability that message M has not been delivered after i games. Then

$$Q(i) = P(l_i l_{i-1}..l_1) = P(l_i \mid l_{i-1}..l_1) \cdot P(l_{i-1}..l_1)$$

For simplicity, let us substitute F_k for $l_k...l_2 l_1$, $1 \leq k < i$ and let us define $P(l_1 \mid F_0) \equiv P(l_1)$ and $P(w_1 \mid F_0) \equiv P(w_1)$. Then

$$Q(i) = P(l_i \mid F_{i-1}) \cdot P(l_{i-1} \mid F_{i-2}) \cdots P(l_1) \tag{1}$$

Clearly, $P(l_j \mid F_{j-1}) = 1 - P(w_j \mid F_{j-1})$, $1 \leq j \leq i$. In the following we will estimate $P(w_j \mid F_{j-1})$. Let $S_{j,k}$ denote the event that message M starts game j at Manhattan distance k from its destination. Events $S_{j,k}$ are mutually exclusive and one of them necessarily happens. Thus

$$P(w_j \mid F_{j-1}) = P(w_j S_{j,1} \cup .. \cup w_j S_{j,\sqrt{N}} \mid F_{j-1}) \Rightarrow$$

$$P(w_j \mid F_{j-1}) = \sum_{k=1}^{\sqrt{N}} P(w_j S_{j,k} \mid F_{j-1}) \Rightarrow$$

$$P(w_j \mid F_{j-1}) = \sum_{k=1}^{\sqrt{N}} P(w_j \mid S_{j,k} F_{j-1}) \cdot P(S_{j,k} \mid F_{j-1}).$$

But $P(w_j \mid S_{j,k} F_{j-1}) \geq \epsilon^{\sqrt{N}}$, thus

$$P(w_j \mid F_{j-1}) \geq \epsilon^{\sqrt{N}} \sum_{k=1}^{\sqrt{N}} P(S_{j,k} \mid F_{j-1}).$$

Since $\sum_{k=1}^{\sqrt{N}} P(S_{j,k} \mid F_{j-1}) = 1 \Rightarrow$

$$P(w_j \mid F_{j-1}) \geq \epsilon^{\sqrt{N}} \Rightarrow$$

$$P(l_j \mid F_{j-1}) \leq 1 - \epsilon^{\sqrt{N}} \tag{2}$$

Finally, $(1), (2) \Rightarrow Q(i) \leq (1 - \epsilon^{\sqrt{N}})^i$.

Thus the probability $Q(i)$ that M will not have been delivered after i games, where $i \to \infty$ is:

$$\lim_{i \to \infty} Q(i) = (1 - \epsilon^{\sqrt{N}})^i = 0$$

The probability $P(i)$ that M will be delivered after i games, where $i \to \infty$ is:

$$\lim_{i \to \infty} P(i) = 1.$$

The essential feature of the proof is the condition that $p \geq \epsilon$. Since this condition holds for meshes on the edges (for the available edges), the theorem has the mesh topology as a corollary.

References

[KS90] Smaragda Konstantinidou and Lawrence Snyder. The chaos router: A practical application of randomization in network routing. In *Proceedings of the 2nd Symposium on Parallel Algorithms and Architectures*, pages 21–30. ACM, 1990.

[NS89] J. Y. Ngai and C. L. Seitz. A framework for adaptive routing in multicomputer networks. In *Proceedings of the Symposium of Parallel Algorithms and Architectures*, pages 1–9. ACM, 1989.