Minimal vs. non Minimal Adaptive Routing on k-ary n-cubes

Fabrizio Petrini and Marco Vanneschi
Dipartimento di Informatica, Università di Pisa,
Corso Italia 40, 56125 Pisa, Italy,
tel +39 50 887248, fax +39 50 887226
e-mail: {petrini,vannesch}@di.unipi.it

Abstract

There is a common agreement in the scientific community that adaptive routing algorithms will eventually replace the deterministic ones that are currently in use in multicomputer networks. An open issue in this research area is to determine the optimal degree of adaptivity that the routing algorithm should provide. Through simulation on a 16-ary 2-cube, we analyze some minimal wormhole algorithms based on Duato's methodology and a non minimal cut-through algorithm, the Chaos routing. The experimental results show that the Chaos routing outperforms the minimal adaptive algorithms under the uniform traffic, but suffers from post saturation problems with the complement, transpose and bit reversal traffics. From these results we argue that non minimal algorithms should pose an upper bound on the degree of adaptivity in order to avoid performance degradations due to excessive misrouting.

Keywords: Interconnection networks, k-ary n-cubes, adaptive routing, wormhole, cut-through

1 Introduction

Current multicomputers that adopt k-ary n-cubes as their interconnection networks typically use deterministic routing algorithms, where a packet's path depends only on the addresses of both source and destination. An example is the *dimension order* routing, where a packet corrects the differences between the two addresses in a strict increasing dimension order. However, deterministic routing performance degrades quickly in the presence of congestion.

Adaptive routing is an alternative to deterministic routing. In adaptive algorithms, the path taken by a packet may depend on the network status and on the packets it encounters. Adaptive algorithms can potentially use all the available paths to relieve congestion. In *minimal* adaptive routing the path taken is a shortest one, while in *non minimal* adaptive routing a packet may take any path between its source and destination, possibly making moves in the network that place it farther from the destination. When a packet makes such a move, the packet is said to have been *derouted*. Non minimal routing allows more flexibility in choosing paths, but at the cost of a more complex router structure to avoid *livelock*, the situation in which a packet never reaches its destination because it is derouted frequently.

Most existing adaptive routing algorithms guarantee deadlock-freedom by defining a set of cycle free routes on a set of virtual networks implemented with virtual channels. These algorithms display interesting trade-offs between their degree of adaptivity and the number of virtual channels needed. If the dimensions are routed strictly in order, as in row column or e-cube routing, no deadlocks can result in a mesh and binary n-cube, even with one virtual channel for physical channel. Introducing adaptivity usually requires an increase in the number of virtual channels. For example, Linder and Harden [18] describe fully adaptive minimal algorithms for k-ary n-cubes with unidirectional and bidirectional links. For the unidirectional case, $(n + 1)$ virtual channels are needed for physical channel. For k-ary n-cubes with bidirectional links, 2^{n-1} virtual channels are needed per physical link in each direction, if the network has no wrap around links. With toroidal cubes the number increases to $(n + 1)2^{n-1}$.

Chien and Kim [6] presented an approach to trade-off adaptivity against the number of virtual channels. Their algorithm, called *planar adaptive routing*, is minimal and partially adaptive. This approach involves examining the routing dimensions in pairs, and constraining the routing choices at any time to one or two dimensions. This, in general, is less flexible than the fully adaptive algorithm of Linder and Harden, but requires only a constant number of virtual channels, regardless of the network dimension. For example, in a k-ary n-cube without wrap around connections, only three virtual channels for each physical link are required.

The *turn model* [15] prevents some of the transitions between dimensions, and generalizes to multidimensional meshes and binary n-cubes. This scheme requires only a virtual channel per physical channel, is non minimal, and partially adaptive. However it only works for meshes; tori require additional virtual channels for deadlock prevention.

The turn model can be used in conjunction with virtual channels to increase adaptivity and to generalize to torus connected k-ary n-cubes. Thus, instead of prohibiting some turns, the packet can be switched to a different virtual channel upon taking such a turn. Dally and Aoki [7] used this idea to design partially adaptive non minimal routing algorithms for the class of k-ary n-cubes.

Virtual channels can be expensive because they complicate routing decision and channel control, increasing the the router complexity and delay significantly, with a consequent cost/performance trade-off [5].

To reduce the number of virtual channels, non monotonic allocation strategies have been proposed [12] [2]. Duato has summarized these strategies in a necessary and sufficient condition for deadlock-free adaptive routing [13]. Virtual channels are divided into two classes: some are specifically dedicated to the prevention of deadlocks while the others allow fully adaptive routing.

In this paper we compare through simulation the performance of a family of minimal adaptive algorithms, based on Duato's methodology [14], with the Chaos routing [3], a non minimal cut-through version of the hot potato routing. Our simulation results conducted on a bi-dimensional torus with 256 nodes show that the non minimal routing outperforms the minimal adaptive and the deterministic routing with uniform traffic, but suffers from post saturation problems with the complement, transpose and bit reversal traffic patterns. When the network saturates, the non minimal algorithm tends to deroute many packets towards the less loaded regions of the topology. These packets experience a sharp increase of the network latency.

The rest of this paper is organized as follows. Section 2 describes the routing algorithms. Section 3 presents a detailed simulation model for studying the performance characteristics of the routing algorithms. The communication patterns adopted as benchmarks in the experimental evaluation are described in section 4. Using the simulation model and the benchmarks, in section 5 we evaluate the communication performance of the algorithms on a bi-dimensional torus with 256 nodes. An overview of the experimental results and some concluding remarks are given in section 6.

2 The routing algorithms

In this section we briefly review the deterministic, Duato and Chaos routing algorithms.

The deterministic algorithm is a dimension order routing based on a static channel dependency graph [10]. Packets are sent to their destination along a unique minimal path. The potential deadlocks caused by the wrap-around connections are avoided doubling the number of virtual channels and creating two distinct virtual networks. Packets enter the first virtual network and switch to the second network upon crossing a toroidal connection. Our version uses two virtual channels for each physical link.

Rather than using a static channel dependency graph, Duato's methodology only requires the absence of cyclic dependencies on a connected channel subset. The remaining channels can be used in almost any way. The methodology starts from a basic deadlock-free routing algorithm and then adds virtual channels where packets can be routed in any minimal path. Our algorithm uses an improved deterministic algorithm [14] with two virtual channels. In our experiments we add two and four more virtual channels that are used for fully minimal adaptive routing.

The Chaos routing operates as a minimal adaptive routing under normal conditions. A packet is moved into the central buffer, called *multiqueue*, if it is potentially in a deadlock loop or if it has stalled at an input buffer[1]. When an output buffer becomes available, packets in the multiqueue desiring that buffer have priority over packets in the input buffers. If more than a packet in the multiqueue desires the channel, the output buffer is allocated according to a FIFO ordering. If a packet must be moved to a full multiqueue, a packet in the multiqueue is chosen at random and is derouted to an available output buffer.

3 Relevant details of the simulated model

This section presents a router model and a simulation environment, that are used in the following sections to analyze the performance of the k-ary n-cubes with some communication benchmarks.

Figure 1 outlines the internal structure of the router. We can distinguish the external channels or links, the input and the output buffers that implement the virtual channels

[1] A packet is considered stalled if it is fully contained in an input buffer.

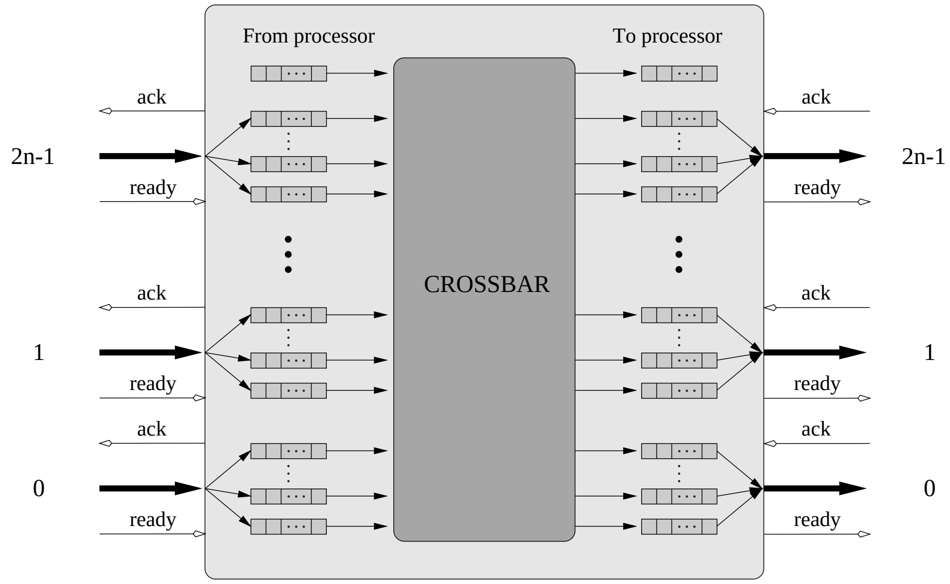

Figure 1: The internal structure of the router

and an internal crossbar. The router has $2*n$ bidirectional channels and each channel on the single direction is logically composed of three interfaces: a *data path* that transmits messages on a flit level, a *ready* line that flags the presence of a flit on the data path and specifies the virtual channel where the flit is to be stored and an *ack* line in the reverse direction that sends an acknowledgment every time buffer space is released in the input lanes. Also, there is another internal channel that interfaces the processor with router.

A flit is moved from an output lane to the corresponding input lane in a neighboring node in T_{link} cycles, when there is at least a free buffer position. Each output lane has associated a counter that is initialized with the total number of buffers in the input lane, it is decremented after sending a flit and it is incremented upon receiving an acknowledgment. When multiple lanes are enabled, an arbiter picks one of them, giving priority to the deterministic lanes. When the flit is put on the data path, the ready line signals the virtual channel where the flit must be stored. Assuming v virtual channels for each link, the ready line can be coded with $\lceil \log_2 v \rceil$ bits.

When a header flit reaches the top of an input lane, the routing algorithm tries to establish a path in the crossbar with a suitable output lane. This path will remain in action till the transmission of the tail flit of the packet. Our model allows the routing of a single header at a time every $T_{routing}$ cycles. The extra complexity of a parallel router has been shown to give little or no advantage in terms of performance in the presence of wormhole or cut-through flow control [3].

Although a physical link services in each direction at most one virtual channel every T_{link} cycles, multiple virtual channels can be active at the input and output ports of the crossbar. The internal flit propagation takes $T_{crossbar}$ cycles. Every time a flit is moved from an input lane to the corresponding output lane, a feedback is sent back to the neighboring node to update the counter of free positions.

This model is evaluated in the SMART (Simulator of Massive ARchitectures and Topologies) environment [20]. Implemented in C++, SMART is an object-oriented discrete-event simulation tool for evaluating massively parallel architectures. Configuring some shell scripts, it is possible to select the network topology, the internal router policies and the traffic pattern generated by each node. The simulator allows the definition of the packet length, interarrival times, number of virtual channels and buffers for both input and output lanes. Also, it is possible to monitor several metrics and time-dependent events, that are gathered in trace files. SMART supports three families of topologies: k-ary n-cubes, k-ary n-flies and k-ary n-trees and a node architecture with processing capabilities and a memory hierarchy.

The experiments in this paper evaluate a toroidal 16-ary 2-cube with 256 processing nodes. Each node in the network is labeled by a pair of coordinates (x, y), $0 \leq x, y < 16$, relative to some root node designated to have coordinates $(0, 0)$[2]. The link, routing and crossbar delays T_{link}, $T_{routing}$ and $T_{crossbar}$ are all equalized to a single cycle, in order to model the router as a balanced pipeline. This can be considered unrealistic with the current technology: ongoing research on router implementations would suggest that the routing delay is at least two, three times the link and the crossbar delay [14] [3] [4]. However, some aggressive implementations tend to reduce such a delay with an internal pipelined architecture [16] [1] [11] [8]. Each node generates 20-flit packets with exponentially distributed inter arrival times; the destinations are uniformly distributed over the network. The simulator collects performance data only after 4000 cycles, to allow the network to reach steady state and each simulation is halted after 20000 cycles.

4 Traffic patterns

In our model each node generates packets independently according to an exponential distribution and the destinations are chosen according to the following traffic patterns. To describe the patterns, let each node (x, y) also be labeled with a number in base k resulting from the concatenation of x and y. The binary representation of xy is $a_0 a_1 \ldots a_{(2\log_2 k)-1}$[3]. Also, let $\overline{0} = 1$ and $\overline{1} = 0$.

- *Uniform traffic.* Destinations are chosen at random with equal probability between the processing nodes.
- *Complement traffic.* Each node sends only to the destination given by $\overline{a_0 a_1 \ldots a_{(2\log_2 k)-1}}$.
- *Bit reversal.* Each node sends only to the destination given by $a_{(2\log_2 k)-1} a_{(2\log_2 k)-2} \ldots a_0$.
- *Transpose.* Each node sends only to the destination given by $a_{(\log_2 k)} a_{(\log_2 k)+1} \ldots a_{(2\log_2 k)-1} a_0 a_1 \ldots a_{(\log_2 k)-1}$.

These traffic patterns illustrate different features. The uniform one is a standard benchmark used in network routing studies. This generation pattern can be considered representative of well-balanced shared memory computations. The complement, bit reversal and transpose traffics are important because they occur in many numerical computations [17].

[2] Locations within the network are interpreted with respect to a logically flattened torus with origin $(0, 0)$.

[3] We will assume that k is a power of two.

5 Experimental results

The performance of an interconnection network is usually described by two main parameters: the *accepted bandwidth* or *throughput* and the *latency*. Throughput is defined as the sustained data delivery rate given some offered load. Two important throughput characteristics are the saturation point and the sustained delivery rate after saturation. Below saturation the throughput is simply equal to the offered load, because all packets are successfully injected and delivered by the network. The *network* latency is the time spent by a packet in the network from the insertion of the trailing flit till the reception of the tail flit at destination. It does not include the source queuing delay at the sender and it is often preferred to the *end-to-end* latency, which rises to infinity after saturation, to analyze network performance. Bandwidth and latency results for all the communication patterns and routing algorithms are summarized in Figure 2. Bandwidth is normalized with the maximum theoretical load for the uniform traffic, determined by the network bisection.

5.1 Uniform traffic

With a saturation point at 83% of the network capacity, the Chaos routing outperforms all three variants based on Duato's methodology, that approximately reach 40%, 60% and 70%, respectively with 2, 4 and 6 virtual channels. The last two variants suffer from post-saturation problems. These are caused in part by the non-uniformities introduced by the deadlock avoidance policy that leads to an unbalanced utilization of the virtual channels.

As can be seen in Figure 3 a) and b), while the surface plot of the Chaos routing is smooth, the Duato's algorithm induces a hill in the center of the torus and a narrow and deep valley corresponding to the nodes of the fifth row.

Also, the Chaos routing shows its superiority in terms of network latency below its saturation point. The better network latency obtained with 4 rather than 6 virtual channels in the minimal adaptive algorithm can be explained looking at the tail latency, which is proportional to the number of virtual channels. On the average, when the network is close to saturation, a packet encounters at least a link on its path where all virtual channels are in use.

5.2 Complement traffic

In the complement traffic all packets are reflected across the logical center of the network. Each packet traverses the network bisection[4], halving the theoretical upper bound. This communication pattern logically divides the network into four distinct 8-ary meshes, each executing the complement traffic on a smaller scale. One of this meshes is placed at the center of the torus, other two meshes are split into two symmetric parts on the torus' borders and the last mesh occupies the remaining four corners, as shown in Figure 3 c) and d). Looking at Figure 2 c), we can see that the throughput of the non minimal algorithm with 4 and 6 virtual channels is very close to optimality. On the other hand, the Chaos routing saturates at 30% of the capacity and after saturation the

[4]In the uniform traffic, on the average, only 50% of the packets crosses the network bisection.

throughput falls down to 17%. Also, Figure 2 d) shows that the network latency has a sharp increase, if compared to the other algorithms. This behavior can be explained noting that the average number of hops experienced by each packet is 32 rather than 8, the average distance on a 16-ary torus. The hot spots of this communication pattern are represented by the centers of the four meshes, while the less loaded regions are located on the borders of these meshes. Many packets are moved away from the hot spots and derouted towards these regions.

5.3 Transpose traffic

In the transpose traffic the destination of each packet is a reflection of the source along the diagonal. This causes a continuous area of congestion along this diagonal and on the opposite corners of the flattened torus. Though this communication pattern is fairly different from the complement traffic, from Figure 2 e) and f) we can see that the performance results are very similar. The minimal adaptive algorithm with 4 and 6 virtual channels saturates at 45% and 47% of the capacity while the Chaos routing reaches 40% and then fall to 22% after saturation. The rise in the network latency and the average path length of 23 hops after saturation witness the strong presence of misrouting in this case too. Packets bounce back and stay for a while in the two regions adjacent the main diagonal before reaching their destination.

5.4 Bit reversal traffic

Unlike the previous traffic patterns, bit reversal has no easy geometric interpretation. There are 16 nodes that have a palindrome bit string and do not inject any packet into the network. They generate some underloaded areas that are located along or near the two main diagonals according to a symmetric layout. While the Chaos routing saturates at 62% of the capacity, the minimal adaptive algorithm with 4 and 6 virtual channels suffer from early saturation, due to a limited number of nodes that can not keep up with an injection rate bigger than 40%. Nevertheless, the global throughput reaches 60% and 70% when the applied load is equal to the network capacity. The non minimality of the Chaos routing is very effective in avoiding small congested areas that prove "difficult" for the minimal routing but, as in the complement and transpose traffic, packets tend to swamp the less loaded areas and after saturation the network latency peaks at 600 cycles.

6 Conclusion

This paper has provided new insight over existing work on adaptive routing present in the literature [19]. While our experimental results confirm that the Chaos routing provides a better performance under uniform traffic, we have shown the potential problems caused by an unlimited degree of adaptivity. In the communication patterns that have underloaded regions, packets tend to move, after saturation, towards these less loaded areas. This causes a performance drop both on the throughput and on the network latency.

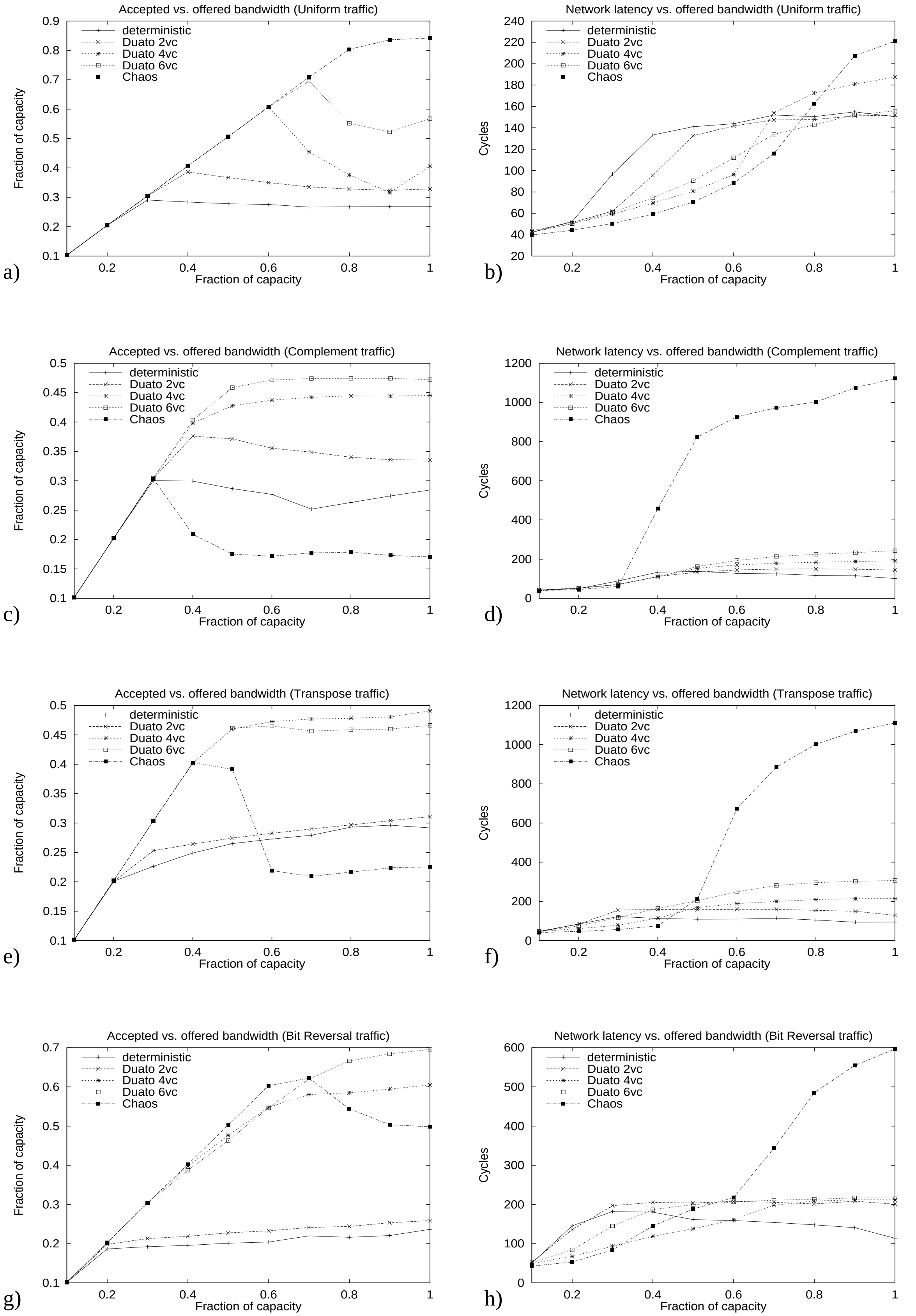

Figure 2: Network performance under the uniform, complement, transpose and bit reversal communication patterns

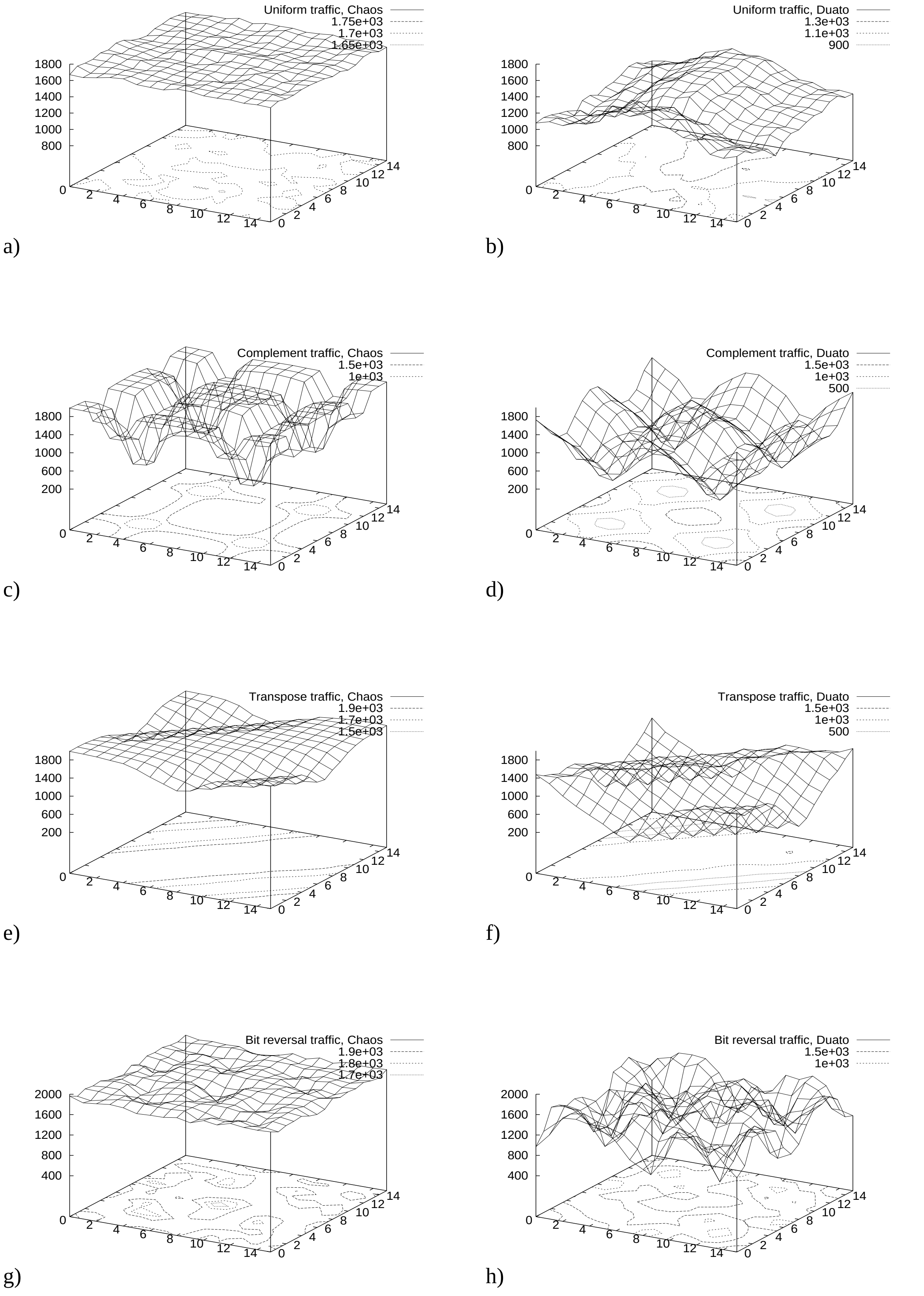

Figure 3: Number of packets passing through each router at saturation, for the Chaos and Duato routing algorithms. The torus is represented as a flattened grid.

References

[1] J. D. Allen, P. T. Gaughan, D. E. Schimmel, and S. Yalamanchili. Ariadne - An Adaptive Router for Fault-tolerant Multicomputers. *Computer Architecture News (Special Issue ISCA'21 Proceedings)*, 22(2):278–288, April 1994.

[2] P. E. Berman, L. Gravano, and G. D. Pifarre. Adaptive Deadlock- and Livelock-Free Routing with all Minimal Paths in Torus Networks. In *Proceedings of the Fourth Symposium on Parallel Architectures and Algorithms*, pages 3–12, 1992.

[3] Kevin Bolding. *Chaotic Routing: Design and Implementation of an Adaptive Multicomputer Network Router.* PhD thesis, University of Washington, Department of Computer Science and Engineering, Seattle, WA, July 1993.

[4] Kevin Bolding, Melanie L. Fulgham, and Lawrence Snyder. The Case for Chaotic Adaptive Routing. Technical Report UW-CSE 94-02-04, University of Washington, Department of Computer Science and Engineering, Seattle WA, February 1994.

[5] Andrew A. Chien. A Cost and Speed Model for k-ary n-cube Wormhole Routers. In *Hot Inteconnects '93*, Palo Alto, California, August 1993.

[6] Andrew A. Chien and Jae K. Kim. Planar Adaptive Routing: Low-Cost Adaptive Networks for Multiprocessors. In *Proceedings of the 19th Annual Symposium on Computer Architecture*, pages 268–277, May 1992.

[7] William J. Dally and Hiromichi Aoki. Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels. *IEEE Transactions on Parallel and Distributed Systems*, 4(4):466–475, April 1993.

[8] William J. Dally, Larry R. Dennison, David Harris, Kinhong Kan, and Thucydides Xanthopoulos. The Reliable Router: A Reliable and High-Performance Communication Substrate for Parallel Computers. In Kevin Bolding and Lawrence Snyder, editors, *First International Workshop, PCRCW'94*, volume 853 of *LNCS*, pages 241–255, Seattle, Washington, USA, May 1994.

[9] William J. Dally and Charles L. Seitz. The Torus Routing Chip. *Distributed Computing*, 1:187–196, 1986.

[10] William J. Dally and Charles L. Seitz. Deadlock-Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Transactions on Computers*, C-36(5):547–553, May 1987.

[11] A. DeHon, F. Chong, M. Becker, E. Egozy, H. Minsky, S. Peretz, and T. F. Knight Jr. Metro: A Router Architecture for High-Performance, Short-Haul Routing Networks. *Computer Architecture News (Special Issue ISCA'21 Proceedings)*, 22(2):266–277, April 1994.

[12] José Duato. Deadlock-Free Adaptive Routing Algorithms Multicomputers: Evaluation of a New Algorithm. In *Proceedings of the Third IEEE Symposium in Parallel and Distributed Processing*, pages 840–847, December 1991.

[13] José Duato. A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks. *IEEE Transactions on Parallel and Distributed Systems*, 6(10):1055–1067, October 1995.

[14] José Duato and Pedro López. Performance Evaluation of Adaptive Routing Algorithms for k-ary n-cubes. In Kevin Bolding and Lawrence Snyder, editors, *First International Workshop, PCRCW'94*, volume 853 of *LNCS*, pages 45–59, Seattle, Washington, USA, May 1994.

[15] C. J. Glass and L. M. Ni. The Turn Model for Adaptive Routing. In *Proceedings of the 19th Annual Symposium on Computer Architecture*, pages 278–287, May 1992.

[16] J. H. Kim, Ziqiang Liu, and Andrew A. Chien. Compressionless Routing: A Framework for Adaptive Fault-tolerant Routing. *Computer Architecture News (Special Issue ISCA'21 Proceedings)*, 22(2):289–300, April 1994.

[17] F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann Publishers, San Mateo, CA, USA, 1992.

[18] D. H. Linder and J. C. Harden. An adaptive and Fault-Tolerant Wormhole Routing Strategy for k-ary n-cubes. *IEEE Transactions on Computers*, 40(1):2–12, January 1991.

[19] Thu Duc Nguyen and Lawrence Snyder. Performance Analysis of a Minimal Adaptive Router. In Kevin Bolding and Lawrence Snyder, editors, *First International Workshop, PCRCW'94*, volume 853 of *LNCS*, pages 31–44, Seattle, Washington, USA, May 1994.

[20] Fabrizio Petrini. SMART: a Simulator of Massive ARchitectures and Topologies. internal note, March 1995.