The Case of Chaotic Routing Revisited

Cruz Izu[1], Ramon Beivide[2] and Jose Angel Gregorio[2]

[1]Computer Science Department
University of Adelaide, Australia
cruz@cs.adelaide.edu.au

[2]Computer Architecture Group
University of Cantabria, Spain
{mon, jagm}@atc.unican.es

Abstract.

This paper presents a new evaluation of the Chaos router, a cut-through non-minimal adaptive router, which was reported to reach 95% of its theoretical throughput limit, at the time where most router proposals only reached 60 to 80%. We will revisit the Chaos router design, provide a new vision of its strengths and relate them to the state-of-the-art in adaptive router design.

In particular, our analysis has identified a parameter of the router design that was not emphasized in the network evaluation presented by their authors, but that is the key to its outstanding performance. This parameter is the channel operation mode. By using the links in half-duplex mode, it allows adjacent network nodes to allocate their bandwidth to one or the other direction in response to the traffic needs. This channel operation mode reduces base latency and increases network throughput compared to full duplex mode for most synthetic traffic patterns.

1. Introduction

The performance of the interconnection network of a parallel computer has a great impact in the system's performance as a whole. K-ary n-cubes are the most common direct network topologies encompassing rings, meshes, and tori. A central element of this kind of network is the router that injects packets from (and delivers packets to) the computer node to which it is connected, and also routes incoming packets from neighbouring routers towards their destinations.

The information transmitted in a network cycle by the channel connecting two adjacent routers is denoted as a *phit*. In wormhole routers, the flow control unit (*flit*) is one or a few phits, thus requiring limited buffer space in the next node in order to advance. Routers using virtual cut-through (VCT) control the flow on a packet basis, thus increasing the buffer demands to at least an entire packet. Longer messages are broken into packets, sent independently and then reassembled at the destination's interface with the overhead this entails [14]. Due to its lower buffer requirement, wormhole was the choice on earlier designs [22] and consequently there is a large body of work on wormhole routers. Many systems have used wormhole but provided buffers with capacity for hundreds of phits. For example, each adaptive virtual channel in the Cray T3E [24] had a 110-phit input buffer.

The more recent BlueGene/L supercomputer uses VCT with variable packet size, ranging from 32 bytes to 256 bytes with a granularity of 32 bytes [2]. Note the choice of flow control not only defines the minimum buffer requirements but it also impacts on buffer management, deadlock avoidance and channel arbitration; in other words, it impacts on the entire router's organization. VCT is generally simpler to implement: as stalled packets are stored in a single node, we can view the network as a store-and-forward one when dealing with deadlock.

In respect to their routing algorithm, deterministic routers are simple to implement but they perform poorly under non-uniform traffic. As many parallel applications present specific non-uniform patterns, adaptive routing is preferable because it spreads the packets more evenly by exploiting the redundant paths provided by the network. However, this increases the risk of deadlock [9] and requires more resources such as complex arbitration and virtual lanes [3]. Although many adaptive routing mechanisms proved good on paper [19], only a few of them provided a good cost/performance ratio [8][21]. The implementation of an adaptive router should try to match the cycle time of an oblivious router, with a limited increase in its node latency. This is normally achieved through careful design and extensive pipelining [17][21][18].

Most adaptive routers choose minimal paths by selecting any of the output channels in the direction of travel, (i.e. +X, -Y) although non-minimal adaptive routers have been proposed to increase fault tolerance [6][12]. The Chaos router also uses non-minimal paths for two purposes: to allow packets that are close to their destinations to manoeuvre around congestion and to simplify the router organization as explained later.

The insights given in this paper are a by-product of using the Chaos simulator [7] to analyse the design of an oblivious VCT router that supported hybrid length traffic [13]. The simulator was interesting because it emulated a VCT router at the register level, down to their pipeline organization, at a time when most router evaluations did not take into account the routing complexity or its impact in node latency and clock cycle. By using the simulator we were able to learn about the Chaos router's low level design and its simulation environment to a level of detail not available from a journal paper. We reproduced its outstanding results and firstly attributed them to the carefully crafted pipelined implementation of the router that included most mechanisms

known to improve throughput such as adaptive routing, output and central buffering and congestion control. Thus, it took us a while to identify one of the key parameters that contributes to such high performance: the channel operation mode. As we will see in this work, the gains achieved by using channels in half-duplex mode are applicable not only to the Chaos design but to other VCT routers.

The rest of the paper is organized as follows: Section 2 describes and discusses the Chaos router implementation. Section 3 describes the simulation environment and provides a re-evaluation of the Chaos performance under full-duplex mode. Section 4 evaluates the impact that channel operation mode has on network performance for two VCT routers and section 5 summarizes the findings of this work.

2. The Chaos router

For completeness we will include a description of the router (with quotes from [4] in italic) and then discuss in detail the approach taken for each design issue: buffer organization, arbitration, congestion control and channel operation mode.

2.1 Chaos router description

Chaotic routing belongs to a queuing class of non-minimal adaptive routers. Therefore, the Chaos router has a central queue, which holds packets waiting for their outgoing links. If a packet wants to enter the central queue and the queue is full, then a packet from the queue has to be derouted to the next free output (this forces packets to use non-minimal paths).

As Figure 1 shows, both the input and output ports have attached buffers with capacity for a single packet. The packet size is fixed to 20 phits. In normal operation packets enter into an input frame of the node, wait for an output frame of a profitable direction to become available, and move to that output frame in a VCT fashion. *The chaotic router minimises the queue overhead by eliminating it from the critical path of the routing decision*. Thus the core of a Chaos router looks like a minimal adaptive router without the need for multiple classes of queues to prevent deadlock.

At the output frame, packets wait for the bi-directionally shared channel to become available and advance to the next input frame when it becomes free. By *bi-directionally shared channel* they meant the two communication channels between adjacent routers are implemented on a single physical link, shared on a packet basis. As full duplex links are often described as bi-directional channels, and this is the only reference to the channel operation mode in [4], most readers would not have picked up they were using half-duplex links.

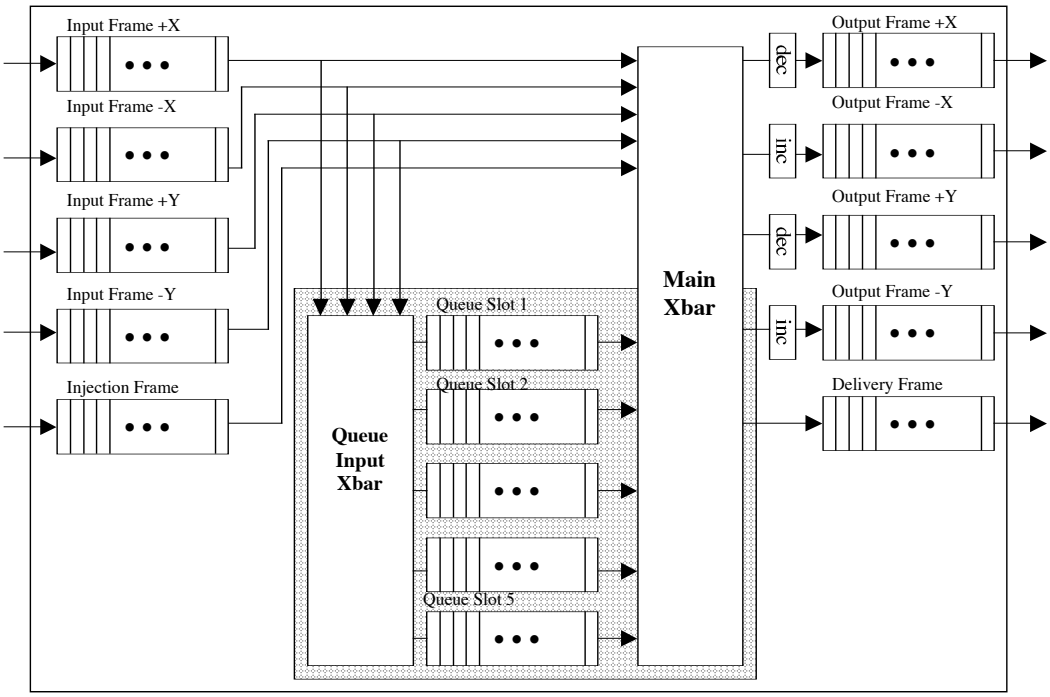

Figure 1. Two dimensional Chaos router diagram.

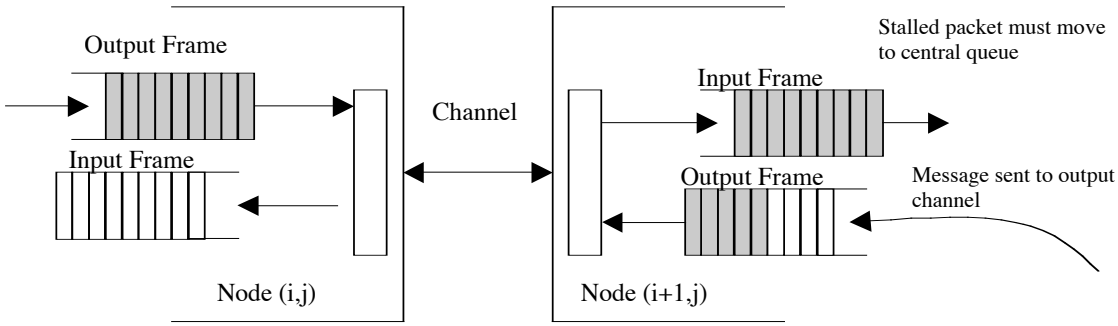

Figure 2. A message forced into the central queue to satisfy packet-exchange protocol.

To guarantee freedom from deadlock, every packet that arrives at an input frame must be *serviced* by the node in a bounded amount of time: if it cannot be routed towards its destination, it will either be stored in the central queue or derouted to the next available output.
Therefore, a packet is moved from its input buffer into the central queue in two cases:

1. The packet has stalled: both its head and tail are buffered in the same input frame, and there is room in the queue.
2. The routers on either side of the shared channel have packets to send to each other. The packet exchange protocol mandates both packets to be sent, regardless of input frame status. In the case there is a stalled packet in the input frame[1], it is moved to the central queue as illustrated in figure 2.

The central queue has capacity for 5 packets. Since packets only enter the central queue when there is congestion in the router, most packets bypass the queue altogether, reducing the queue management overhead.
Whenever an output frame becomes available (it does not contain a packet header), the router selects a packet to advance to that output frame as follows:

1. If the queue is full, a randomly selected packet is routed to the output frame (most likely to be misrouted). Note that the randomised choice is the key to avoid starvation with minimal cost.
2. If the queue is not full, it will select a packet that is requesting that channel (if any).
3. If no packets in the queue are to be sent to this output, but a packet in the input frame can be profitably routed out, it is sent to the output frame.

The Chaos pipeline has four primary stages: receive the header across the network channel into the input frame, decode the header and identify profitable output channels, select a single output frame to route the packet to, and move the header across the crossbar to the output frame where the header is updated.

[1] Packets from the injection frame do not enter the queue due to deadlock prevention constrains

2.2 Buffer organization

Earlier work on buffer organization has been focused on the *input versus output* buffering dilemma for FIFO queues [15]. The former provides a simple implementation but introduces head-of-line blocking (HLB). The latter eliminates this problem but requires multi-port output queues to accommodate the simultaneous arrival of packets from different inputs that may select the same output channel. Since then, a considerable body of work has gone into finding alternatives such as [16],[26],[25] to combine the benefits of each approach.

The Chaos router did simply that by providing a single packet queue both at the input and output ports. Therefore, output queues didn't need to be multi-ported since when many packets arrived for the same output, they could be buffered at their input queues while one of them moved in VCT fashion to the selected output. As most packets are queued at the output frames, or moved into the central queue if the output is full, HLB is practically eliminated. Besides, as the central queue is not in the critical path, the only downside of having the central queue is the additional silicon area required.

2.3 Arbitration

Arbitration in any adaptive router is normally the critical stage of the router pipeline. Each input packet may request more than one output so that the allocation of outputs to inputs cannot be done in parallel as in the oblivious counterpart.

In order to reduce this complexity in the Chaos router, only one new crossbar connection may be set up per cycle. In addition, the Chaos router uses an output driven design [11]. Each cycle, a single arbitration occurs to select the packet (from input or queue) to move into the next free output frame. This greatly simplifies the arbitration phase, allowing for a reasonable pipeline design.

The simultaneous arrival of multiple packets will result in a serialized allocation of inputs to outputs. This has a negligible impact on network performance when the packet length is large enough in relation to the network degree. In other words, a d-degree router will receive d phits per cycle and provided packets are larger that d phits, it will be able to

keep all outputs busy. The longer the packet and the lower the network load, the less likely for two headers to arrive in the same cycle and delay one another. At heavy loads, the arbitration delay will range from *1* to *d-1* cycles, which is low compared to the blocking delay due to network contention.

2.4 Congestion control

As described in subsection 2.1, the injection frame is treated as an input frame, except that packets are never moved into the central queue. As packets in the central queue have priority over input frame packets, packet injection is throttled by the cental queue's population. This reduces throughput degradation at saturated loads by preventing the nodes from overflowing the central and output buffers. This strategy though, may lead to starvation as a node can be prevented from injecting a packet indefinitely if the incoming traffic from its neighbours does not by-pass the central queue and thus is given higher priority to progress.

Note that as stalled packets are moved into the central queue and the channel is shared on a packet basis, a packet will be derouted when the congestion is high or the packet is involved in a deadlock. As the occurrence of deadlock in a fully adaptive network is low [20], the majority of misrouting actions will be caused by network congestion.

In short, although most routers benefit from some kind of congestion control at high loads, this mechanism is critical for the Chaos router to limit misrouting and make a better use of the channel bandwidth.

2.5 Channel operation mode

The router default configuration has ***bi-directionally shared channels***; in other words, the two network channels that link adjacent nodes are implemented using a half-duplex link. The link is multiplexed amongst the two network channels at each side on a packet basis. In [4] there was no explanation for this design choice or its impact on network performance. In their chip implementation though, they indicate the decision to use half-duplex was based on pin limitation [5], and their final implementation required a *dead* cycle to reverse the channel direction. Thus, for a 20 phit packet the effective channel utilization is limited to 95%. However, their network evaluation did not take into account this arbitration cost.

To the best of our knowledge, all other routers are designed using full duplex links [2,12,18,21,23,24,25,26], and there is no study for direct networks that consider the impact that channel operation mode has on router performance. Thus, a fair evaluation of the Chaos router should cover this point.

3. Chaos Router re-evaluation

This section provides a re-evaluation of the Chaos router under full-duplex configuration and compares the results with those provided in [4]. We have used the Chaos simulator as provided by their authors [7] and only alter the channel operation mode so that the router description and pipelined organization remains unchanged. Hence, packets are 20 phits long, and the buffer capacity is of one packet per input or output frame, plus 5 packets in the central queue, as in the original evaluation.

Although the Chaos architecture specifies half-duplex channels, the Chaos simulator can also be configured to have full duplex links - which is the standard for all other network evaluations. Appendix A shows the configuration file for a 256-node 2D torus under these two scenarios.

For a fixed phit size, the half-duplex configuration will obviously have half the bisection bandwidth of its full-duplex counterpart, and its theoretical maximum throughput [1] for a 16x16 torus will be 64 phits/cycle compared to 128 phits/cycle for the full-duplex case. Comparing these two networks with a fixed phit size is not fair but we are doing it in order to reflect the fact that the original paper provides one set of network responses that corresponded to the half-duplex case but that were compared by the research community to other works, which correspond to full-duplex network configurations.

Figure 3 shows throughput and latency for random uniform and hot spot traffic patterns. In the latter, the traffic sent to 10 nodes (randomly selected) is four times that sent to the other nodes; this models cases in which references to program data such as synchronization locks, bias packets destinations toward a few nodes. Figure 4 shows the network response under well-known traffic permutations such as bit reversal, bit complement and transpose. It is clear from both figures that network performance depends heavily on the channel configuration chosen. When traffic in both directions is balanced, such as in random traffic, the differences are limited, as both channels are used most of the time anyway. When the traffic is unbalanced, the half-duplex configuration makes a better use of the network links. This is significant for most traffic permutations such as bit reversal and bit complement.

Remember the dashed lines correspond to the results reported in [4] while the continuous lines are those obtained under the standard full-duplex mode. The clear gap between them explains why the initial Chaos results did not match the reader's intuition when seen as a full duplex adaptive network.

Figure 5 shows network latency as a function of the offered load expressed in bits/cycle/node. (in the chaos router, a phit was equal to 16 bits). This figure exemplifies the limitations of using normalized loads to estimate network performance, and it also reminds us that in this section we are comparing two networks with different bisection bandwidths. We can only do that in terms of how well each network configuration uses their network links, as discussed above.

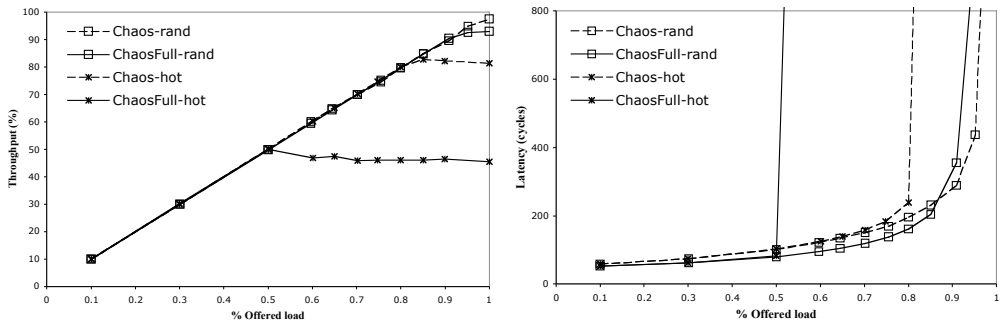

Figure 3. Normalized throughput and latency for a 256-node torus under random and hot spot traffic.

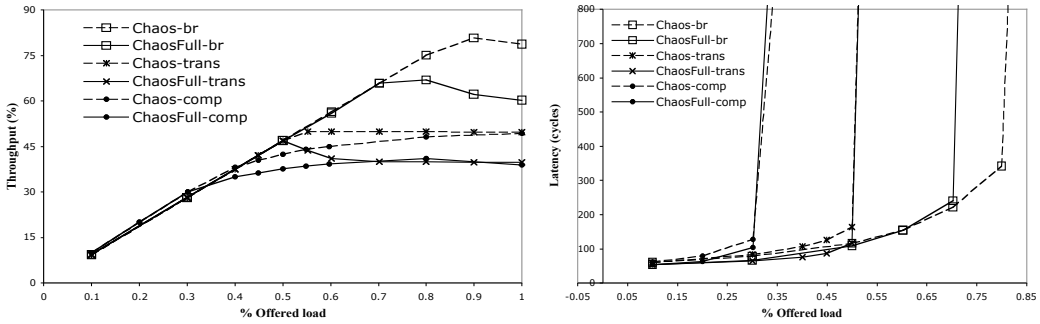

Figure 4. Normalized throughput and latency for a 256-node torus under bit reversal, transpose and complement permutations.

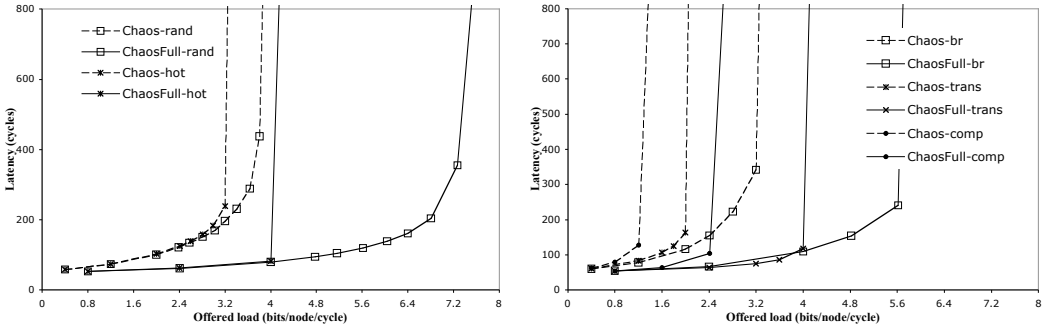

Figure 5. Network latency versus offered load for a 256-node torus under a range of traffic patterns.

Traffic	**Average number deroutes**		**Max number deroutes**	
Pattern	***Full-duplex***	***Half-duplex***	***Full-duplex***	***Half-duplex***
Random	0.038	0.028	3	3
Hot Spot	2.089	0.386	160	49
Bit reversal	0.585	0.344	11	9
Transpose	0.009	0.012	4	3
Complement	0.377	0.358	11	10

Table 1. Level of misrouting for the full-duplex and half-duplex chaos networks at full load.

.

Table 1 shows the level of misrouting at saturation for each traffic pattern under the two network configurations. Note that each time a packet is derouted, its path increases by 2 hops. As half-duplex reduces congestion, it results in a lower number of misroutes under any traffic pattern.

The hot-spot pattern performance is interesting because congestion builds much faster around the hot spots, particularly if they are not evenly distributed. We can see that the hot-spot pattern exhibits the highest level of misrouting, increasing each packet average path by 5 and 1.2 hops for full-duplex and half-duplex respectively. The maximum number of deroutes per packet is significant, 160 and 49 respectively. This is not surprising, as the chaos router deals with congestion by misrouting packets. It also means that for this pattern, the local throttle of packet injection is not enough to keep network congestion at a reasonable level.

The half-duplex configuration helps to reduce congestion by allocating more bandwidth to the hot-spot direction. Its links reached 95% utilization of which 15% corresponded to misrouted packets. The full duplex networks reached 77% link utilization but 31 % was used to misroute packets. We may speculate that the Chaos router is able to handle a considerable level of network congestion, after which misrouting becomes ineffective as the use of the output channels by the misrouted packets triggers more misrouting actions. We should note misrouting decreased to more reasonable levels (12% of a total 89% link utilization) when the packet length increased to 40 phits. The full duplex network under hot spot traffic reaches congestion levels close to that threshold, hence its variable performance.

4. Impact of the channel operation mode in VCT routers

The results from the previous section indicate a half-duplex implementation can make better use of the network bandwidth for non-uniform loads.

Thus, it is of interest to compare the two channel configurations under fairer conditions by assuming constant node bandwidth and taking into account the added cost of reversing direction in the half-duplex case. Given that the full duplex channels are "w" bits wide, their half-duplex counterparts will be "2w" bits wide. Consequently, their view of a packet having 40w bits will be a 40-phit and a 20-phit packet respectively. In both cases the input and output frames have capacity for a single packet2. As both networks have the same bisection bandwidth, their normalized loads are comparable. Their maximum load will be 128*w bits per cycle (or 0.5*w bits/node/cycle).

To account for the cost of reversing direction, we have modified the simulator to include a dead cycle when the channel direction is reversed. Its impact in latency is negligible as half-duplex mode reduces base latency by 10 cycles, but it will reduce effective channel utilization when both directions are heavily used.

4.1 Chaos Router : Full duplex vs Half-duplex

Figures 6 and 7 show the network performance under a range of traffic patterns. As a packet in a nearly empty network will halve its transmission time, all patterns exhibit lower latencies for the half-duplex case.

Half-duplex achieves a higher throughput for all traffic patterns but random. The channel arbitration uses 2 to 4% of the link capacity, so that throughput is slightly reduced when compared with the results from section 3.

Again, the more unbalanced the use of the network links, the higher the gains exhibited by the half-duplex configuration. This is not surprising, as this model reflects the bi-directional highway lane model, which exploits the unbalance in commuters' traffic by allocating more lanes to the most popular direction at each time of the day.

As we mentioned before, the performance for hot-spot traffic in the full duplex case is significantly better that in the previous experiment, packet length being the only change. In extensive tests under hot-spot traffic, most loads (which differ in the location of the 10 hot-spot nodes) reached similar peak throughput, around 75-80%. One of them, though, exhibited high levels of misrouting for both channel modes, reaching 27% and 42% for half-duplex and full-duplex respectively. This load also exhibited the highest network population, another indicator of network congestion. This seems to confirm our theory that misrouting may be ineffective when congestion levels reach a high threshold. On the other hand, misrouting combine with throttled injection deals successfully with most types of network loads as seen under typical permutation traffic patterns.

Finally, we have also considered the impact of using half-duplex in a chaos router with pipelined channels [22], The cost of reversing direction will increase from 1 to p+1 cycles being p the number of *phits* on the fly. Table 2 summarizes the results obtained when considering pipelined channels with p being 2 or 3, As expected, the half-duplex configuration exhibited lower performance for random traffic for which peak throughput decreased by 6% and 10% respectively in relation to its full-duplex counterpart. On the other hand, the benefits of half-duplex configuration out-weigh its cost for all other non-random patterns.

Remember that the half-duplex configuration provides lower network latency for all patterns at low and medium loads. Thus, half-duplex mode remains a better choice for the Chaos router, regardless of the physical link's length or delay.

[2] Note the buffer capacity in bits is still the same for both routers

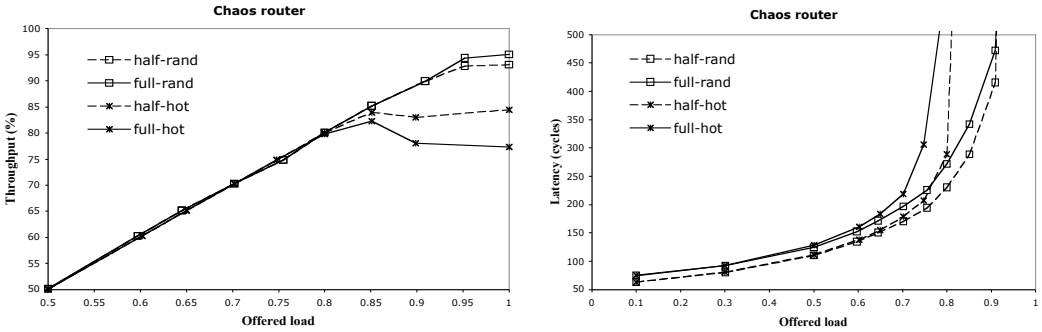

Figure 6. Normalized throughput and latency for a 256-node torus under random and hot spot traffic.

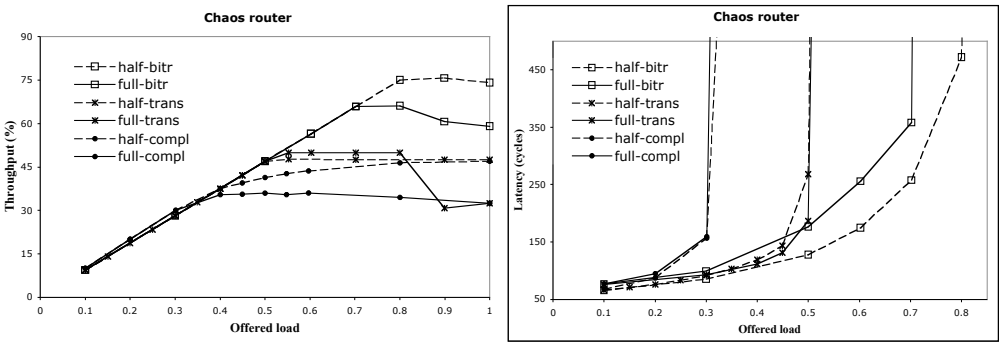

Figure 7. Normalized throughput and latency for a 256-node torus under bit reversal, transpose and bit complement permutations**.**

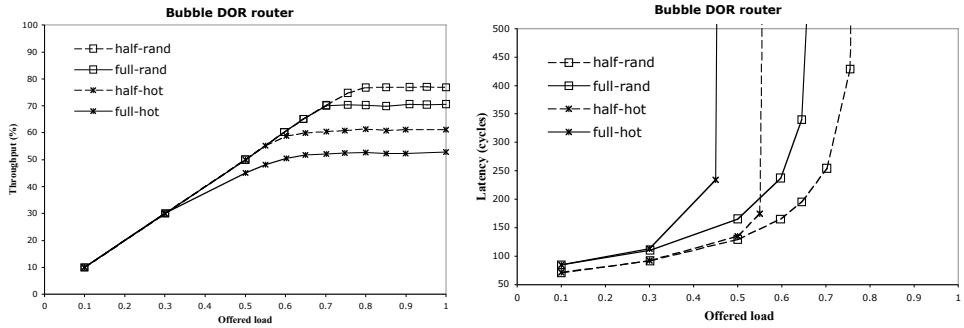

Figure 8. Normalized throughput and latency for a 256-node static network under random and hot spot traffic.

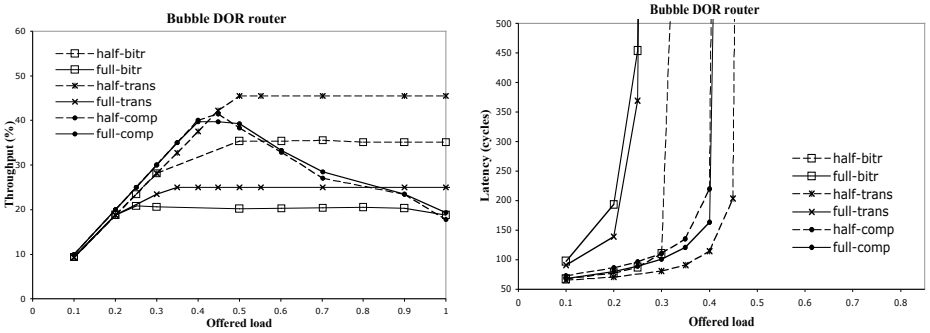

Figure 9. Normalized throughput and latency for a 256-node static network under bit reversal, transpose and bit complement permutations**.**

Traffic Pattern	2 phits		3 phits	
	Full	*Half*	*Full*	*Half*
Random	94.7	88.7	94.7	84.9
Hot Spot	82.1	83.9	81.1	81.7
Bit reversal	59.4	70.1	58.9	67.7
Tranpose	30.9	43.2	31.3	43.1
Complement	33.1	44.9	34.2	43

Table 2. Network throughput at full load for a 256 torus network with pipelined channels (2 or 3 *phits* on the fly) for various synthetic traffic patterns.

4.2 DOR router: Half-duplex vs Full duplex

To complete this study, we have used the Chaos simulator to evaluate the impact that the channel configuration has in a simpler VCT router based on bubble flow control [10]. This will confirm that the findings from this work are applicable to a wider range of designs.

The Bubble DOR router is similar to the Chaos one except that there is no central queue and the output frame is selected using dimensional order routing (DOR). Deadlock is avoided by preventing any node from exhausting the buffer capacity in the direction of travel as in [10], thus no virtual channels are required. Again packets have $40w$ bits, being w and $2w$ the width of the full duplex and half-duplex channels. In this router evaluation the input and output frames have capacity for two packets each.

DOR is known to exhibit low throughput for most permutation patterns, due to its unbalanced use of network channels. Thus, it is not surprising to see in Figures 8 and 9 that the gains achieved by the half-duplex configuration are even greater that in the Chaos counterpart. In particular, for the transpose permutation, its throughput increases from 25% to 45%, matching that of the Chaos router. This is because the traffic is very un-evenly distributed in each direction, the best scenario for the half-duplex configuration.

5. Conclusions

This work has provided an insight into the performance of the Chaos router as reported in [4]. We have identified that the channel operation mode has a significant impact on network performance. We should note that the half-duplex mode is only applicable to VCT routers in which channel allocation is done on a packet basis.

We measured the network response of the Chaos router for both full-duplex and half-duplex modes. This re-evaluation showed the half-duplex configuration increases link utilization for all synthetic traffic patterns, more so when the traffic load is unbalanced. This can be easily explained by the fact that network bandwidth is allocated to each direction as required by traffic needs.

A fairer comparison of the two channel operation modes was presented under fixed node bandwidth and taking into account the cost of reversing the channel direction, which in the chaos implementation was of a dead cycle between the transmission of two packets. The evaluation of two VCT routers showed that half-duplex configuration improves network latency by reducing the transmission time at low and medium loads. It also increased their peak throughput for all non-uniform traffic patterns by overlapping when possible the idle cycles in each network direction.

Further study is required to assess the impact that channel operation mode has on network performance under real application loads and the cost of implementing half-duplex channels on VCT routers under current technological constrains.

Acknowledgments

This work has been partially supported by Ministerio de Ciencia y Tecnologia, Spain, under grant TIC2001-0591-C02-01.

We would like to thank the people involved in the Chaotic Routing Project at the Department of Computer Science and Engineering, University of Washington for providing public access to their simulator source code.

References

[1] A. Agarwal, “Limits on Interconnection Network Performance”, IEEE Trans. On Comp., Vol 2, nº4, pp:398-412, October 1991

[2] NR Adiga, GS Almasi, Y Aridor, M Bae, Rajkishore Barik, et al., “An Overview of the BlueGene/L Supercomputer”, Proc. of SuperComputing 2002, Baltimore, Nov. 16-22, 2002

[3] K. Aoyama, A. Chien: The Cost of Adaptivity and Virtual Lanes”, Journal of VLSI Design, 2(4), 1995, pp.315-333.

[4] K. Bolding, M. L. Fulgham, L. Snyder, “The Case for Chaotic Adaptive Routing. IEEE Trans. Computers 46(12): 1281-1291 (1997)

[5] K. Bolding, S. Cheung, S. Choi, C. Ebeling, S. Hassoun, T. Ngo, R. Wille. "The Chaos Router Chip: Design and Implementation of an Adaptive Router", Proceedings of IFIP Conf. on VLSI. Sept. 1993 pp. 311-320

[6] R. Boppana and S. Chalasani, "Fault-tolerant routing with non-adaptive wormhole algorithms in mesh networks, " Proc. of Supercomputing, pp. 693-702, 1994

[7] The Chaos simulator code is available at http://www.cs.washington.edu/research/projects/lis/Chaos/www/simulator.html

[8] J. Duato. “A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks” IEEE Trans. On Parallel and Distributed Systems, vol.6, no.10, pp.1055-1067, October 1995,

[9] W. J. Dally and C. L. Seitz, “Deadlock-Free Message Routing in Multiprocessor Interconnection Networks”, IEEE Trans. on Comp., Vol. C-36, 5, pp. 547-553, 1987.

[10] C. Carrion, R. Beivide, J.A. Gregorio, and F. Vallejo. “A Flow Control Mechanism to Prevent Message

Deadlock in k-ary n-cube Networks" HiPC'97, December, 1997.

[11] M. L Fulgham and L. Snyder,"A Comparison of Input and Output Driven Routers". Lectures Notes in computer Science, vol. 1123 Proc EuroPar 1996, 195-203

[12] P. T. Gaughan, S. Yalamanchili, "A Family of Fault-Tolerant Routing Protocols for Direct Multiprocessor Networks". IEEE Trans. Parallel Distrib. Syst. 6(5): 482-497 (1995)

[13] C. Izu and A. Arruabarrena, "Applying Segment Routing to k-ary n-cube networks", Proc. Int. Conference on Supercomputing, 1998, pp 409-416.

[14] V. Karamcheti and A. A. Chien, "Do Faster Routers Imply Faster Comunication? Proc. Parallel Computer Routing and Communications Wrokshop, PCRCW'94, pp 1-15.

[15] M. J. Karol, M. G. Hluchyj, S. P. Morgan, "Input Versus Output Queuing on Space Division Packet Switch", IEEE Transactions On Communications, vol. COM-35, no. 12, pp. 1347-1356, December 1987.

[16] M. Katevenis, P. Vatsolaki, A. Efthymiou, and M. Stratakis "VC-level Flow Control and Shared Buffering in the Telegraphos Switch", IEEE Hot Interconnects III, August 1995.

[17] S. Konstantinidou and L. Snyder: "The Chaos router: A practical application of randomization in network routing". Proc. 2nd Ann. Symp. on Parallel Algorithms and Architectures SPAA'90, pp. 21-30.

[18] S S Mukherjee, P. Bannon, S. Lang, A. Spink and D. Webb "The Alpha 21364: Network architecture ". IEEE Micro, 22(1):26-35, January/February 2002

[19] L.M. Ni and P.K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks" IEEE Computer Magazine, vol. 26, no.2, pp. 62-76, Feb. 1993.

[20] T.M. Pinkston and S. Warnakulasuriya: On Deadlocks in Interconnection Networks. Proc. 24th International Symposium on Computer Architecture ISCA 1997 pp. 38-49

[21] V. Puente, C. Izu, J.A. Gregorio, R. Beivide, and F. Vallejo, "The Adaptive Bubble router", Journal on Parallel and Distributed Computing, vol 61, no. 9, pp.1180-1208 September 2001.

[22] C.L. Seitz, W-K Su, "A family of routing and communication chips based on the Mosaic". Proc. of the 1993 Symp, on Research on Integrated Systems, The MIT Press, 1993, pp. 320-337.

[23] S. L. Scott and J.R. Goodman, "The Impact of Pipelined Channels on k-ary n-cube Networks", IEEE Transaction on Parallel and Distributed Systems. vol. 5, no 1 pp. 2-16 January 1994..

[24] S. L. Scott and G. Thorson, "The Cray T3E networks: adaptive routing in a high performance 3D torus," in Proc. of Hot Interconnects IV, Aug. 1996.

[25] R. Sivaram, C.B. Stunkel, and D.K. Panda, "HIPQS: a High-Performance Switch Architecture Using Input Queuing", Proc. IPPS/SPDP'98, March 1998.

[26] Y. Tamir, and G.L. Frazier, "Dynamically-allocated Multiqueue buffers for VLSI Communication Switches", IEEE Transactions on Computers, vol. 41, no. 2, pp. 725-737, June 1992

Appendix A. Configuration files for Chaos

```
/*** Chaos routing algorithm ***/
#define CHAOS 1
#define CYGRA 0
#define OBLIVIOUS 0
#define WORMHOLE 0

/*** latency (in cycles) across a node ***/
#define NODE_LATENCY 4
/*** cycles to stall on a queue send ***/
#define Q_SEND_STALL 3
/*** Multiqueue size ***/
#define Q_CAP 2*D+1

/*** torus topology ***/
#define WRAP 1
#define OPEN 0
/*** 256 node network ***/
#define N 256
/*** 2 dimensions ***/
#define D 2
/*** 16 nodes per dimension ***/
#define K 16
/*** maximum distance between any two nodes ***/
#define MAX_DIST (((K-1)/2 + 1)*D + 1)

/*** uni-directional channels ***/
#define UNI 1
#define BI 0
#define XBAR_RATE 2
#define Q_BUS_RATE 2
/*** total number of channels ***/
#define NUM_CHAN (N*D*2)
/*** number of virtual channels per physical channel ***/
#define NUM_VC 1
/*** number of outframes which can own any channel
***/
#define CHAN_OWNERS NUM_VC

/*** message length distribution ***/
#define RANDOM_LENGTH 0
#define LONG_SHORT 0
#define LENGTH 20
#define AVE_LENGTH LENGTH
/*** number of cycles to route a message out of a fifo
***/
#define ROUTE_WINDOW 20
/*** minimum injection period ***/
#define MIN_INJ_PERIOD (((double) K)/8.0 * ((double)
AVE_LENGTH))
/*** maximum buffer size in flits ***/
#define FIFO_MAX_SIZE 20
/*** inframe buffers size in flits ***/
#define INF_FIFO_SIZE 20
/*** outframe buffers size in flits ***/
#define OUTF_FIFO_SIZE 20
/*** internal buffers size in flits ***/
#define Q_FIFO_SIZE 20
```

```
/*** Chaos routing algorithm ***/
#define CHAOS 1
#define CYGRA 0
#define OBLIVIOUS 0
#define WORMHOLE 0

/*** latency (in cycles) across a node ***/
#define NODE_LATENCY 4
/*** cycles to stall on a queue send ***/
#define Q_SEND_STALL 3
/*** Multiqueue size ***/
#define Q_CAP 2*D+1

/*** torus topology ***/
#define WRAP 1
#define OPEN 0
/*** 256 node network ***/
#define N 256
/*** 2 dimensions ***/
#define D 2
/*** 16 nodes per dimension ***/
#define K 16
/*** maximum distance between any two nodes ***/
#define MAX_DIST (((K-1)/2 + 1)*D + 1)

/*** bi-directional channels ***/
#define UNI 0
#define BI 1
#define XBAR_RATE 1
#define Q_BUS_RATE 1
/*** total number of channels ***/
#define NUM_CHAN (N*D)
/*** number of virtual channels per physical channel ***/
#define NUM_VC 1
/*** number of outframes which can own any channel
***/
#define CHAN_OWNERS (2*NUM_VC)

/*** message length distribution ***/
#define RANDOM_LENGTH 0
#define LONG_SHORT 0
#define LENGTH 20
#define AVE_LENGTH LENGTH
/*** number of cycles to route a message out of a fifo
***/
#define ROUTE_WINDOW 20
/*** minimum injection period ***/
#define MIN_INJ_PERIOD (((double) K)/4.0 * ((double)
AVE_LENGTH))
/*** maximum buffer size in flits ***/
#define FIFO_MAX_SIZE 20
/*** inframe buffers size in flits ***/
#define INF_FIFO_SIZE 20
/*** outframe buffers size in flits ***/
#define OUTF_FIFO_SIZE 20
/*** internal buffers size in flits ***/
#define Q_FIFO_SIZE 20
```