Wormhole Routing in Parallel Computers

Ayse Yasemin Seydim
School of Engineering and Applied Sciences
Southern Methodist University
e-mail : yasemin@seas.smu.edu
May, 1998

Abstract -In order to offer low communication latency and reduced buffer requirements, wormhole routing has been used in almost all new generation parallel computers. It has been a powerful switching technique where its communication latency is distance insensitive. On the other hand, deadlock avoidance is the most critical issue in wormhole networks. If deadlock handling is not considered, wormhole routers can not be used in multiprocessors. Also, performance and fault-tolerancy are two dominant issues in the design of interconnection networks of large-scale multiprocessor architectures. In this paper, only a brief review of wormhole routing issues mentioned could be made amongst the extensive published works.

Introduction

A lot of research effort has been dedicated during the last decade to improve the performance of multicomputers. A key architectural issue is the interconnection networks. Since the number of nodes in the multicomputer network is increasing, the time required to move data between the nodes is important in total system performance. Whether a direct network system is used with message-passing or a shared-memory concepts, the transmission time become critical. Also, it will affect the possible granularity level of parallelism in executing an application program. One of the most powerful architectural scheme used in interconnection networks is wormhole routers and the related routing algorithms.

Among the other switching models, wormhole routing seems the most promising method for using less storage bandwidth in the nodes through which messages are routed. It is also making the message latency largely insensitive to the distance in the network. On the other hand, by its nature, it is subject to deadlock conditions as message blocking becomes so easy that contiguous flits cannot advance in any of the channels. Channels are the resources to catch the route to destination for the messages traversing throughout the wormhole network.

In order to reduce the impact of message blocking, physical channels may be split into virtual channels, and consequently, this can increase throughput as it provide dynamically sharing of the physical bandwidth among several messages. In any case, deadlock avoidance is the most critical issue in wormhole networks. Many deadlock-free deterministic and adaptive routing algorithms have been proposed for many years. Besides this important factor, fault-tolerancy and performance are two critical subjects in the design of interconnection networks of large-scale multiprocessor architectures. In this paper, only a brief review of wormhole routing issues could be made amongst the extensive published works.

Direct Networks

In a direct network architecture, each node has a *point-to-point*, or direct, connection to some number of other nodes, called neighboring nodes. Neighboring nodes may send packets to one another directly, while nodes that are not directly connected must rely on other nodes in the network to transfer packets from source to destination.

Although a router's function could be performed by the corresponding local processor, dedicated routers are used to allow overlapped computation and communication within each node. Each router supports some number of input and output channels. Internal channels connect the local processor memory to the router. External channels are used for communication between routers, and, therefore nodes. By connecting the input channels of one node to the output channels of other nodes, the *topology* of the direct network will be defined.

For topologies in which packets may have to traverse some intermediate nodes, the *routing algorithm* determines the path selected by a packet to reach its destination. At each node, the routing algorithm indicates the next channel to be used. Efficient routing is critical to the performance of interconnection network.

When a message or packet header reaches an intermediate node, a *switching* mechanism determines how and when the router switch is set, i.e. the input channel is connected to the output channel selected by the routing algorithm. The switching mechanism determines how network resources are allocated for message transmission.

Popular direct networks are:

- n-dimensional mesh
- *k*-ary *n*-cube or torus
- hypercube

n-dimensional mesh : formally has $k_0 \times k_1 \times ... \times k_{n-2} \times k_{n-1}$ nodes, k_i nodes along each dimension i, where $k_i >= 2$ (nodes have from n to 2n neighbors, depending on their location in the mesh)

k-ary n-cube : all nodes have the same number of neighbors (all k_i's are equal) There are wraparound channels in *k*-ary *n*-cube, which are not present in the n-dimensional mesh. A *k*-ary *n*-cube contains k^n nodes.

If k=2, every node n neighbors, one in each dimension.
If k > 2, every node has 2n neighbors, two in each dimension.
If n =1, k-ary n-cube collapses to a ring with k nodes.

hypercube : is a special case of both the *n*-dimensional mesh and the *k*-ary *n*-cube. A hypercube is an *n*-dimensional mesh in which $k_i = 2$ for all $0 \leq i \leq n-1$,i.e., a *2*-ary *n*-cube.

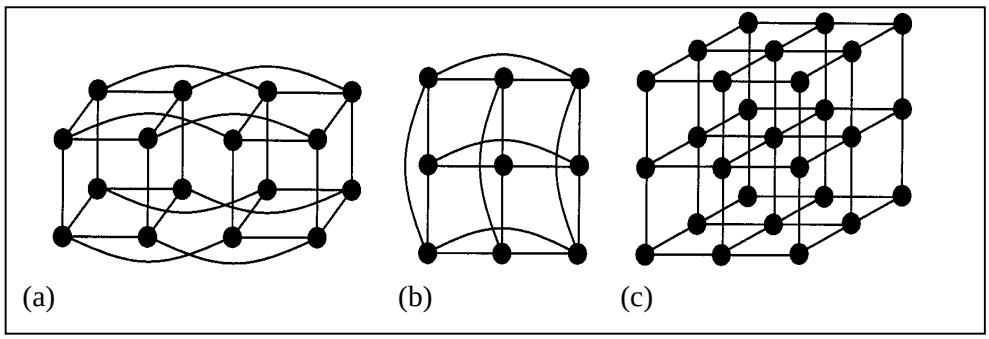


Figure 1. Direct Network Topologies
(a) 2-ary 4-cube (hypercube), (b) 3-ary 2-cube (torus), (c) 3-ary 3D mesh.

Switching Techniques

Switching techniques employed in multiprocessor networks initially followed those techniques employed in local and wide-area networks. However, as the application of multiprocessor systems spread into increasingly compute-intensive domains, the traditional layered communication designs borrowed from LANs became a limiting performance bottleneck. New switching techniques and implementations evolved that were better suited to the low latency demands of parallel programs[8].

In *circuit switching*, a physical path is reserved from source to destination before the transmission. A header of the message is injected to the network and when it reaches the destination, the complete path has been set up and an acknowledgment is sent back to source so that message contents may be transmitted at the full bandwidth of the hardware path(telephone system).

If the size of the message is not much greater than the size of the routing header, it become advantageous to transmit the message along with the header and buffer the message within the routers while waiting for a free link. This is referred to as *packet switching*. Also, a message can be partitioned and transmitted as fixed-length packets in which the first packet containing the routing and control information. Each packet is individually routed from source to destination. A packet is completely buffered (like mail service) at each intermediate node before it is forwarded to the next node. This is the reason why this switching technique is also called as *store-and-forward switching*.

Circuit switching is advantageous when messages are infrequent and long, i.e. when the transmission time long compared to the path setup time. On the other hand, the physical path is reserved for the duration of the message and may block other messages. In contrast, store-and-forward switching is advantageous when messages are short and more frequent, so that full utilization of the communication link can be realized. In this switching mechanism, each node must buffer every incoming packet, consuming memory space.

To decrease the amount of time spent transmitting data and full buffer requirement, *virtual cut-through* method is introduced, in which a packet is stored at an intermediate node only if the next required channel is busy. As soon as enough information available in the intermediate nodes,

forwarding begins even before the entire message is received. At high network loads virtual cut-through behaves like packet switching.

The Torus Routing Chip

The Torus routing chip[1] is a self-timed VLSI circuit that is designed to provide deadlock-free packet communication. Since it is self-timed, each processing node can operate at its own rate with no need for global synchronization. Synchronization is performed by arbiters in the chip when it is needed. Torus Routing Chip uses *cut-through* routing rather than store-and-forward routing to reduce the latency of communications. Instead of reading an entire packet into a processing node, the chip forwards each flit of the packet to the next node as soon as it arrives. There is no buffer usage and packets remain strictly within the TRC network until they reach their destination.

Wormhole Routing

Wormhole routing is a special case of cut-through switching. Instead of storing a packet completely in a node and then transmitting it to the next node, wormhole routing operates by advancing the head of a packet directly from incoming to outgoing channels of the routing chip. A packet is divided into a number of *flits* (flow control digits) for transmission. The size of a flit depends on system parameters, in particular, the channel width. The header flit(or flits) govern the route. As soon as a node examines the header flit(s) of a message, it selects the next channel on the route and begins forwarding flits down that channel. As the header advances along the specified route, the remaining flits follow in a pipeline fashion.

Because most flits contain no routing information, the flits in a message must remain in contiguous channels of the network and cannot be interleaved with the flits of other messages. When the header flit of a message is blocked, all of the flits of a message stop advancing and block the progress of any other message requiring the channels they occupy.

Wormhole routing avoids memory bandwidth in the nodes through which messages are routed. Only a small FIFO flit buffer can be used. It also makes the network latency largely insensitive to path length. On the other hand, in order to reduce the effect of message blocking, physical channels may be split into virtual channels and these will be used to increase the total throughput of the physical channel[2]. Virtual channels are logical entities associated with a physical link used to distinguish multiple data streams traversing the same physical channel. They are multiplexed over a physical channel in a demand-driven manner, with bandwidth allocated to each virtual channel as needed.

Some of the direct networks that use wormhole routing are Ncube-2 (hypercube), Intel Touchstone Delta (2D mesh), Paragon (2D mesh), MIT J-Machine (3Dmesh) and Cray T3D (3D torus).

Switch-based systems, which are using special switches for the communication between any two nodes can also use wormhole routers and some of these are TMC CM-5, Meiko CS-2 and IBM SP1. The switch-based systems are not studied in this paper.

Routing Algorithms

In an intercommunication network, routing algorithms that are used for determining the path to the destination node can be classified [8] according to their:

- number of destinations- Unicast: packets may have a single destination, Multicast: packets may hay multiple destinations
- place where routing decisions are taken- Centralized: by centralized controller, Source: by the source node, Distributed: determined in a distributed manner while the packet travels, Multiphase: hybrid, source node computes some destinations, path established in a distributed manner
- way of implementation- Table-Lookup: looking at a routing table, Finite-State Machine: executing a routing algorithm in software or hardware according to a finite-state machine
- adaptivity- Deterministic: always supply the same path between a source/destination pair, Adaptive: use information about network traffic and/or channel status to avoid congested or faulty regions of the network
- progressiveness- Progressive: move the header forward, reserving a new channel at each routing operation, Backtracking: allow header to backtrack, releasing previously reserved channels (used for fault-tolerant routing)
- minimality- Profitable(minimal): supply channels that bring the packet closer to its destination, Misrouting(non-minimal): may also supply channels that send the packet away from its destination
- number of alternative paths- Fully Adaptive, Partially Adaptive

Deadlocks and Wormhole Routing

In wormhole routing, contiguous flits in a packet are always contained in the same or adjacent nodes of the network. This can cause difficulties, as possibility of deadlock arises. Deadlock in the interconnection network occurs when no message can advance towards its destination because the queues of the message system are full. No communication can occur over the deadlocked channels until exceptional action is taken to break the deadlock.

The technique of virtual channels allow deadlock-free routing to be performed in any tightly connected interconnection network. This technique involves splitting physical channels on cycles into multiple virtual channels and then restricting the routing so the dependence between the virtual channels is acyclic. When it was first implemented, the Torus routing chip was considered to provide a deadlock-free packet communication in *k*-ary *n*-cube(torus) networks with up to k=256 processors in each dimension. The design methodology based on this chip become an important milestone of the routing techniques developed ever[1].

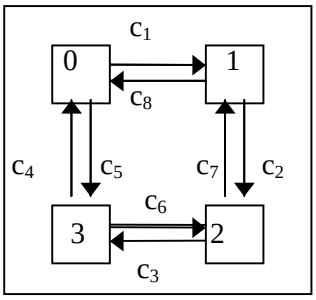
When wormhole switching was proposed, deadlock avoidance techniques were immediately considered. The probability of deadlock in wormhole networks is high enough to block the whole network in a few second, if the routing algorithm is not deadlock-free. When a packet is holding a channel, then it requests the use of another channel, there is a dependency between those channels. Both channels are in one of the paths that may be followed by the packet. When wormhole switching is used, those channels are not necessarily adjacent because a packet may hold several channels simultaneously. Also, at a given node, packet may request the use of several channels, then selecting one of them (when the algorithm is adaptive). All the requested

channels are candidates for selection. When all alternative output channels are busy, the packet will get the first requested channel that becomes available. So, all the requested channels produce dependencies, even if they are not selected in a given routing operation.

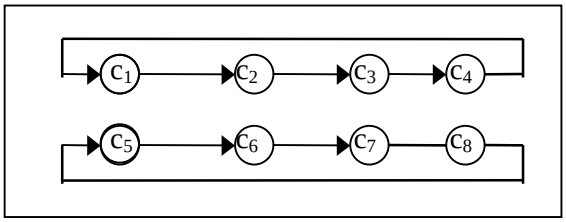
The behavior of packets regarding deadlock is different depending on whether there is a single or several routing choices at each node. With deterministic routing, packets will have a single routing choice at each node. If a set of packets such that every packet in the set has reserved a channel and it requires a channel held by another packet in the set, it is obvious that, channels cannot be granted to any of the packets and there will be a deadlock. So, it is necessary to remove all the cyclic dependencies between channels to prevent deadlocks[2].

A *channel dependence graph* can be used to see the dependencies between nodes to develop a deadlock-free routing algorithm. The channel dependence graph is defined as a directed graph, D=G(C,E) , where C is the set of vertices which are the channels of the network and E is the pairs of channels connected by the routing algorithm. In other words, if a pair (c_i, c_j) $\in$ E(D), then c_i and c_j are, an input channel and an output channel of a node respectively, and the routing algorithm may route packets from c_i to c_j. A routing algorithm for a direct network is deadlock-free if and only if there is no cycle in the channel dependence graph.

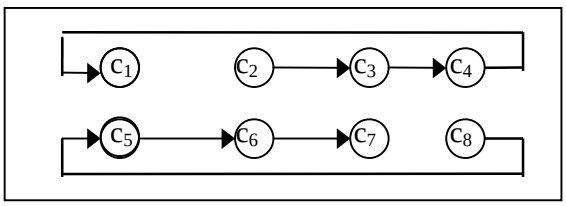
Figure 2 shows the channel dependence graph method. Since there are two cycles in the dependence graph deadlock is possible. The broken cycles are shown in Figure 2(c) part.



(a) direct network of 2x2 mesh, or a 2-cube, or a 4-ary 1-cub, or a 2x2 torus.



(b) channel dependence graph



(c) channel dependence graph based on restricted routing

Figure 2. A four-node network and corresponding channel dependence graphs

Deadlock-Free Routing Algorithms

Deadlocks are mainly avoided by using a proper routing algorithm within the network. Many routing algorithms that provide deadlock-free communication have been proposed. Some of the basic algorithms are briefly explained below.

Dimension-Ordered Routing

One approach to designing a deadlock-free algorithm for a wormhole-routed network is to ensure that cycles are avoided in the channel dependence graph. This can be achieved by assigning each channel a unique number and allocating channels to packet strictly ascending or descending order[2][3]. Routing is restricted to visit channels in order (ascending or descending) to eliminate cycles in the graph. If the routing restriction disconnect the network, physical channels are split into virtual channels to connect the network again.

A channel numbering scheme often used *n*-dimensional meshes is based on the dimension of channels. In dimension-ordered routing, each packet is routed in one dimension at a time, arriving at the proper coordinate in each dimension before proceeding to the next dimension. By enforcing a strictly monotonic order on the dimensions traversed, deadlock-free routing is guaranteed. In an *n*-cube, each node is represented using an *n*-bit binary number. Each node has *n* outgoing channels, and the i^{th} channel corresponds to the i^{th} dimension.

The virtual networks are then realized on the physical network. This is done to ensure that cyclic dependencies cannot occur and deadlock freedom is assured. Duato [5] provided a powerful extension to this paradigm and eliminated the necessity to restrict routing to acyclic virtual networks. This opened the doors to new classes of protocols that are known deductively to be deadlock-free by virtue of satisfying the relaxed constraints that are specified.

Duato states a general theorem defining a criterion for deadlock freedom and then uses the theorem to propose a fully-adaptive, profitable, progressive protocol. It is stated that by separating virtual channels on a link into restricted and unrestricted partitions, fully adaptive routing can be performed and yet be deadlock-free.

Adaptive Routing

The main disadvantage of deterministic routing is that it cannot respond to dynamic network conditions, such as congestion. Although Dally's [2] proposal was for deterministic routing, it has been applied to adaptive routing. An adaptive routing algorithm for a wormhole-routed network, however, must address the deadlock issue. To do so often requires the use of additional channels; in particular, some adjacent nodes must be connected by multiple pairs of opposite unidirectional channels. These pairs of channels may share one or more physical channels[3].

One general adaptive routing technique works by partitioning the channels into disjoint subsets. Each subset constitutes a corresponding sub-network. Packets are routed through different sub-networks, depending on the location of destination nodes. Figure 3 shows the application of the method to a 2D mesh. As in Figure 3(a) shows, the mesh contains an additional pair of channels added to the Y dimension. The network can be partitioned into two sub-networks called the +X

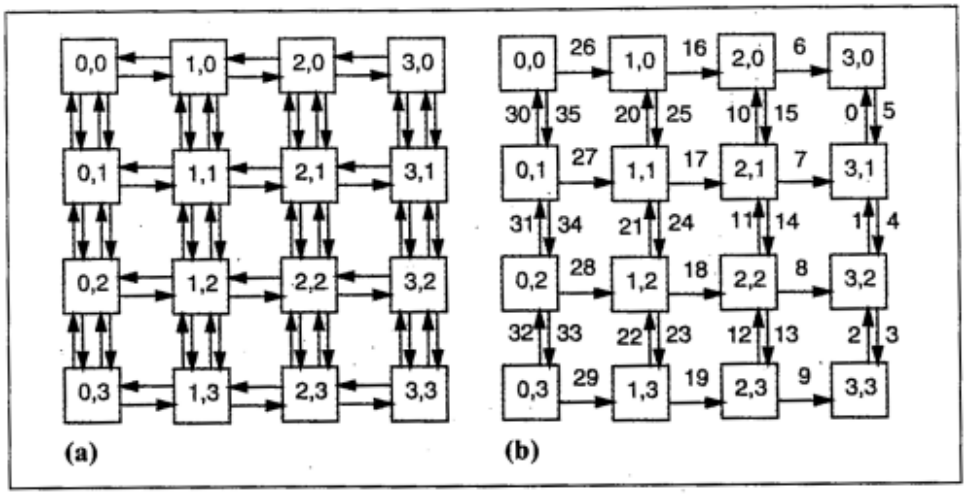


Figure 3. Adaptive double Y-channel routing for a 2D mesh:
(a) double Y-channel 2D mesh, (b) +X sub-network and labeling

and -X sub-network, each having a pair of channels in the Y dimension and a unidirectional channel in the X dimension. The +X sub-network is shown in Figure 3(b). If the destination node is to the right of the source, that is if d_x (x-coordinate of destination node's address) is greater than s_x (x-coordinate of source node's address), the packet will be routed through the +X sub-network. If $d_x < s_x$, the -X sub-network is used. If $d_x = s_x$, the packet can be routed using either sub-network.

This double Y-channel routing algorithm is minimal and fully adaptive; that is a packet can be delivered through any of the shortest paths. The algorithm can be proved to be deadlock-free by ordering the channels appropriately. Such an ordering of the channels in the +X sub-network is shown in Figure 3(b). For any pair of source and destination nodes, the channels will be traversed in descending order, no matter which shortest paths are taken. Hence, deadlock cannot occur. For example; in Figure 3(b), any of the minimal paths from node (1,0) to node (2,2) - specifically, (25,24,18), (25,17,14), and (16,15,14) - are valid.

Providing deadlock-free minimal fully adaptive routing algorithms for the hypercube, 2D torus, or more general *k*-ary *n*-cube topologies may require additional channels. It was shown that a k-ary *n*-cube can be partitioned into 2^{n-1} sub-networks, n+1 levels per sub-network, and k^n channels per level. The number of additional channels increases rapidly with n. While this approach does provide minimal fully adaptive routing, the cost associated with the additional channels makes it impractical when *n* is large.

Turn Model

An innovative view of the conditions for deadlock-free adaptive routing is the *turn model*. However, the turn model is also based on Dally's [2] theorem, requiring the absence of cycles in the channel dependency graph. The turn model provides a systematic approach to the development of maximally adaptive routing algorithms, both minimal and non-minimal, for a given network without adding channels. Deadlock occurs because the packet routes contain turns

that form a cycle. The following six steps can be used to develop maximally adaptive routing algorithms for n-dimensional meshes and *k*-ary *n*-cubes :

1. Classify channels according to the direction in which they route packets.
2. Identify the turns that occur between one direction and another, omitting 0-degree and 180-degree turns.
3. Identify the simple cycles these turns can form.
4. Prohibit one turn in each cycle.
5. In the case of *k*-ary *n*-cubes, incorporate as many turns as possible that involve wraparound channels.
6. Add 180-degree and 0-degree turns if there are multiple channels in the same direction.

The fundamental concept behind the turn model is to prohibit the smallest number of turns such that cycles are prevented. However, prohibiting fewer than four turns can still prevent cycles. So, a west-first routing algorithm is suggested such that; route a packet first west, if necessary, and then adaptively south, east, and north. Two turns to the west is prohibited and to travel west, a packet must begin in that direction. Figure 4 shows three example paths for the west-first algorithm. The channels marked as unavailable are either faulty or being used by other packets. One of the paths shown is minimal, while the other two paths are non-minimal, resulting from routing around unavailable channels. Because cycles are avoided, west-first routing is deadlock-free.

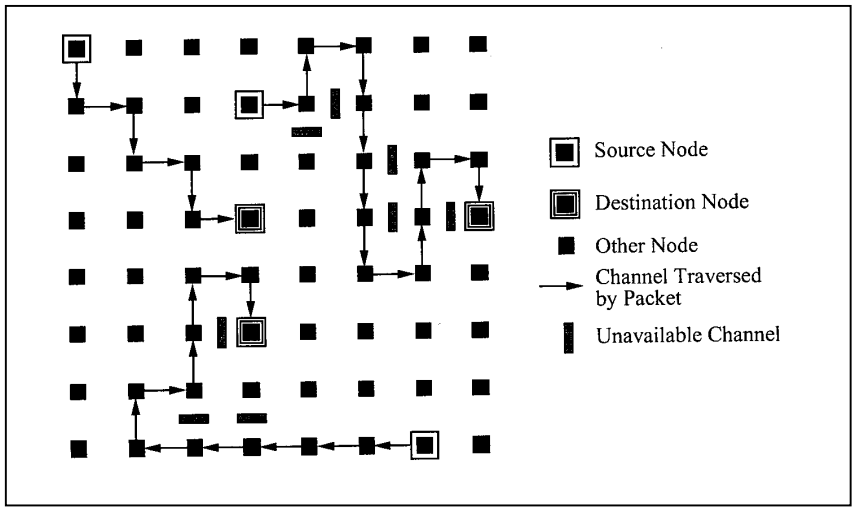


Figure 4. Examples of west-first routing in an 8x8 2D mesh.

Fault-Tolerant Routing in Wormhole Networks

Fault-tolerant systems aim at providing continuous operation in the presence of faults. In the interconnection networks reliability is very important for the reliability of the whole system. In fact, the loss of communication between any two nodes could prevent a distributed recovery, because computation states in the faulty nodes is unreachable. *Network fault tolerance* has been defined as, the maximum number of elements that can fail without causing a possible disconnection in the network.

Design of fault-tolerant interconnection networks can be divided into two classes: *dynamic* and *static*. A dynamic design has some spare channels and switches, allowing the reconfiguration of the network and preserving the original topology. A static design provides a fault-tolerant routing algorithm that will bypass any faulty node or channel. In both cases, there is an upper bound for the number of successive faults tolerated by the network before repairing. For direct networks, a static design only requires the definition of a fault-tolerant algorithm[6].

Several fault-tolerant algorithms have been proposed for direct networks taking the advantage of the alternative paths offered by the network topology. They focus on guaranteeing that any node can be reached in the presence of a given number of faults. Fault tolerant algorithms for wormhole networks are based on different strategies. Some of them exploit the high number of alternative paths available in the hypercube topology. Marking certain fault-free nodes as unsafe, such that if the node has at least two faulty or unsafe neighbors, is another strategy used. By not forwarding messages to unsafe nodes, routing and broadcasting can be simplified and their delay reduced in this way. Some of the algorithms introduced fault regions, in which fault patterns are in predefined shapes. Sets of fault-free components around faulty regions are proposed. An algorithm based on the turn model and does not require virtual channels was proposed for n-dimensional meshes that supports n-1 dynamic faults[7].

There is wide spectrum of proposals to tolerate faults in the interconnection network, ranging from simple software techniques for commercial machines to sophisticated hardware support for systems where reliability is the primary design goal. Most of the proposals lie in the middle of the spectrum, aiming at increasing fault-tolerance in networks using wormhole switching by adding some hardware support[6].

The flexibility of the avoiding deadlocks by preventing cyclic dependencies between channels can be used to route messages around faults. Most of the proposals focused on fault-tolerant algorithms, without analyzing the redundancy available in the network. Duato [6] analyzed the effective redundancy available in the wormhole network by combining connectivity and deadlock freedom. A methodology to guide the design of fault-tolerant algorithm is proposed theoretically.

Conclusion

When designing an interconnection network for a parallel computer, the designer has to consider several important issues. Software messaging layer of the intercommunication network can be the largest overhead in the communication latency. Reducing or hiding this latency will have an important impact on the performance of the network. In wormhole networks, when virtual circuits are established between nodes, caching and reusing these circuits would improve the software overhead[8].

Designers should consider a switching technique that will satisfy the basic needs, the message length and buffer size that can be used. Wormhole routing should be preferred, if messages are longer than the preferred buffer size.

Current deadlock avoidance techniques allow fully adaptive routing across physical channels. However, some buffer resources(virtual channels) must be dedicated to avoid deadlock by providing escape paths to messages blocking cyclically. On the other hand, progressive deadlock recovery techniques require a minimum amount of dedicated hardware to deliver deadlocked packets. Deadlock recovery techniques do not restrict routing at all, therefore allowing the use of

all the virtual channels to increase routing freedom, achieving the highest performance when packets are short. However, when packets are long or have very different lengths and the network approaches the saturation point, the small bandwidth offered by the recovery hardware may saturate. In this case, some deadlocked packets may have to wait for long, thus degrading performance and making latency less predictable. Also, recovery techniques require efficient deadlock detection mechanisms. Currently available detection techniques only work efficiently when all packets are short and have a similar length. Otherwise, many false deadlocks are detected, quickly saturating the bandwidth of the recovery hardware. The poor behavior of current deadlock detection mechanism considerably limits the practical applicability of deadlock recovery techniques unless all the packets are short. This may change when more accurate distributed deadlock detection mechanisms are developed.

Routing algorithm, number of virtual channels, buffer size are the other key issues to be considered when the performance of the system is concerned. From the reliability point of view, an interconnection network must be reliable, such that when faults are more frequent a more complex hardware support can be considered. The fault tolerance properties of the routing algorithm are constrained by the underlying switching technique. If the performance is more important than reliability, fault tolerance should be achieved without modifying the switching technique. If reliability is more important than performance, appropriate switching technique must be used to increase fault tolerance and additional support for reliable transmission should be included in the network. Consequently, designers must carefully balance the benefits of the network against the significant costs.

REFERENCES

[1] W.J. Dally, and C.L. Seitz, “The torus routing chip”, Distributed Computing, vol.1 , no.3, pp. 187-196, 1986.

[2] W.J. Dally, and C.L. Seitz, “Deadlock-Free Message Routing in Multiprocessor Interconnection Networks”, IEEE Trans. Computers, vol. C-36, no.5,pp.547-553, May 1987.

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

[4] P.T. Gaughan, and S. Yalamanchili, “Adaptive Routing Protocols for Hypercube Interconnection Networks”, IEEE Computer, vol.26, no.5, pp.12-23, May 1993.

[5] J. Duato, “A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks”, IEEE Trans. Parallel and Distributed Systems, vol.6, pp. 1055-1067, Oct. 1995.

[6] J. Duato, “A Theory of Fault-Tolerant Routing in Wormhole Networks”, IEEE Trans. Parallel and Distributed Systems, vol.8, no.8, pp. 790-802, Aug. 1997.

[7] C.J. Glass, and L.M. Ni, “Fault-Tolerant Wormhole Routing in Meshes without Virtual Channels”, IEEE Trans. Parallel and Distributed Systems, vol.7, no.6, pp. 620-635, June 1996.

[8] J. Duato, S. Yalamanchili, and L. Ni, “Interconnection Networks”, IEEE Computer Society Press, Los Alamitos, CA., 1997.