# Optimal Fully Adaptive Wormhole Routing for Meshes

Loren Schwiebert and D. N. Jayasimha
Department of Computer and Information Science
The Ohio State University
Columbus, Ohio 43210–1277
Email: {loren, jayasim}@cis.ohio-state.edu

## Abstract

*A deadlock-free fully adaptive routing algorithm for 2D meshes which is optimal in the number of virtual channels required and in the number of restrictions placed on the use of these virtual channels is presented. The routing algorithm imposes* less than half *as many routing restrictions as any previous fully adaptive routing algorithm. It is also proved that, ignoring symmetry, this routing algorithm is the* only *fully adaptive routing algorithm that achieves both of these goals. The implementation of the routing algorithm requires relatively simple router control logic. The new algorithm is extended, in a straightforward manner, to arbitrary dimension meshes. It needs only* $4n - 2$ *virtual channels, the minimum number for an* $n$*-dimensional mesh. All previous algorithms require an exponential number of virtual channels in the dimension of the mesh.*

**Keywords:** wormhole routing, mesh architectures, routing algorithms, deadlock freedom, optimality.

## 1 Introduction

Wormhole routing [6] has become the switching technique of choice in modern distributed-memory multiprocessors such as the Intel Paragon, the Symult 2010, the MIT J-machine, the Caltech MOSAIC and the nCUBE-2. Implementations of wormhole routing typically divide each message into packets, which are then divided into flits. The header flits of a packet contain the routing information and the data flits of the packet follow the header flits through the network in a pipelined fashion. All channels in the path are reserved from the time the header flits acquire the channel until the last data flit in the packet has traversed the channel. The pipelining of the flits makes the message latency largely insensitive to the distance between the source and destination, so there is lower message latency when there is little or no channel contention. Each channel has enough storage to buffer only a few flits, so wormhole routing requires significantly less hardware for buffering than packet switching does. These two factors make wormhole routing an attractive switching technique for distributed-memory multiprocessors.

The most significant problems with wormhole routing are message latency and contention that can occur with even moderate network traffic. Since *all* the channel queues in the path from the source to the destination are held from the time they are acquired until the entire message has been transmitted (which is after the entire path has been established except for very short messages which fit in the intermediate channel buffers), performance degradation due to contention can be severe and message latency can be unacceptably high. A message that requires several channels can block many messages while being transmitted. These blocked messages can in turn block other messages, which further increases the message latency. The problems of latency and contention could be partially overcome by providing additional *physical* channels between nodes in the network. This is an expensive solution, however. A more cost-effective method is to allow multiple *virtual* channels to share the same physical channel [4]. Each virtual channel has a separate buffer, with multiple messages multiplexed over the same physical channel. Both latency and contention can be further reduced by utilizing the multiple paths between the source and destination. However, because channels are held until the message has been transmitted, a routing algorithm with no restrictions on the use of virtual or physical channels can result in deadlock [6].

The simplest routing algorithms are *deterministic* (*oblivious*) and define a single path between the source and destination. A message must wait for each busy channel in the path to become available. On the other hand, adaptive routing algorithms support multiple paths between the source and destination. Adaptive routing algorithms are either minimal or non-minimal. Minimal routing algorithms allow only shortest paths to be chosen, while non-minimal routing algorithms also allow longer paths. Adaptive routing algorithms, whether minimal or non-minimal, can be further differentiated by the number of shortest paths allowed. Adaptive routing algorithms that do not allow *all*

Table 1: **Overview of Adaptive Routing Algorithms for Meshes**

| Author(s) | Fully Adaptive? | VCs for 2D Mesh | Comments |
|---|---|---|---|
| Chien & Kim [2] | Yes | 6 | Partially Adaptive for Higher Dimensions |
| Dally [3] | Yes | 6 | 2D Mesh Only |
| Dally & Aoki [5] | Yes | $n$ | 2D Mesh with $n \times n$ nodes |
| Glass & Ni [8] | Yes | 6 | 2D Mesh Only |
| Glass & Ni [9] | No | 4 | Roughly Half the Adaptiveness of Fully Adaptive |
| Jesshope, Miller & Yantchev [10] | Yes | 8 | Number of Virtual Channels is Exponential in Dimension of Mesh |
| Linder & Harden [11] | Yes | 6 | |

messages to use *any* shortest path are called *partially adaptive*. Adaptive routing algorithms that *do* allow all messages to use any shortest path are called *fully adaptive*. While all fully adaptive routing algorithms allow a message to use any *physical* channel that is part of a shortest path, different restrictions are placed on the choice of *virtual* channels on that physical channel, so not all fully adaptive routing algorithms are equivalent. Some fully adaptive routing algorithms allow more adaptiveness than others by placing fewer restrictions on the choice of *virtual* channels.

Separate buffers are needed for each virtual channel. Furthermore, routing algorithms that require more virtual channels need additional router control logic and are usually more complex. The multiplexing and scheduling of the virtual channels on the physical channel is also more complicated. In addition, router latency and cycle time increase with the number of virtual channels [1], so fewer virtual channels is better. Decreasing the number of virtual channels needed for a given adaptiveness is accomplished by using a less restrictive routing algorithm. Conversely, a less restrictive routing algorithm has better performance relative to other routing algorithms when the same number of virtual channels is used.

## 2 Previous Work

Adaptive routing algorithms have been developed for mesh, torus and hypercube topologies. Torus and hypercube topologies can be characterized as $k$-ary $n$-cubes, where $k$ is the radix and $n$ is the dimension. For example, an 8D hypercube is a 2-ary 8-cube and a $16 \times 16$ torus is a 16-ary 2-cube. An $n$-dimensional mesh is similar to a torus, except a mesh does not have any wrap-around channels. Routing algorithms for only mesh topologies are reviewed here, because wormhole routing has been used primarily on low-dimension meshes and the focus of this paper is on mesh topologies.

Many adaptive routing algorithms for meshes have been proposed [2, 3, 5, 8, 9, 10, 11]. Table 1 summarizes the main features of each algorithm. In the Table, VCs is used for number of bidirectional virtual channels per router.

Designing deadlock-free routing algorithms for wormhole routing was simplified by a proof that an acyclic channel dependency graph guarantees deadlock freedom [6]. Each node in the channel dependency graph is a virtual channel. A directed edge from one virtual channel to another means that a message could use the second virtual channel immediately after the first. Since the graph is acyclic, deadlock freedom can be proved by assigning a numbering to the edges of the graph, ensuring that virtual channels are used in always increasing or always decreasing order. A channel dependency graph is connected if there is a path from any source to any destination.

The proof in [6] is a necessary and sufficient condition for deterministic routing algorithms which can be characterized as functions of the form $R : C \times N \rightarrow C$, where the incoming channel, belonging to the set of channels $C$, and the destination, belonging to the set of nodes $N$, define an outgoing channel on which to route the message. Acyclicity of the channel dependency graph has also been used as a basis for developing adaptive routing algorithms defined by relations of the form $R : C \times N \rightarrow C^p$, where a set of channels, rather than a single channel, is defined on which the message can be routed. Requiring an acyclic channel dependency graph is overly restrictive for routing algorithms defined by relations of the form $R : N \times N \rightarrow C^p$, where the current node, rather than the incoming channel, and the destination define the set of channels on which the message can be routed [7]. Cycles may exist in the channel dependency graph if some subset of channels defines a

connected subgraph with an acyclic *extended* channel dependency graph. An extended channel dependency graph is a dependency graph which arises from direct and indirect dependencies. Each edge in the channel dependency subgraph defines a *direct* dependency. An *indirect* dependency is a dependency between two channels in the connected subgraph that exists only because of the intermediate use of channels not in the subgraph.

A method of analyzing routing algorithms based on the permitted and prohibited turns from one virtual channel to another is presented in [9]. The dependencies between virtual channels are characterized as turns. The set of possible turns are defined by the topology and are given here for meshes. The turns can be 90° turns (i.e., a traversal from a virtual channel in one direction to a virtual channel in any orthogonal direction), 0° turns (i.e., a traversal from one virtual channel to another virtual channel in the same direction) and 180° turns (i.e., a traversal from one virtual channel to a virtual channel in the opposite direction). Clearly, 180° turns are not used for minimal routing. The *turn model* groups the turns into cycles and breaks all cycles by prohibiting some turns. This is equivalent to removing edges from the channel dependency graph. It is then necessary to show that cycles cannot be created from the remaining turns. This is done by providing a numbering of the edges in the channel dependency graph.

For the 2D mesh, the algorithms proposed in [2, 3, 11] all produce an equivalent fully adaptive minimal algorithm called double-y. This routing algorithm requires two virtual channels in each Y direction and one virtual channel in each X direction. This set of virtual channels has a total of sixteen 90° turns and four 0° turns. Double-y allows eight of the sixteen 90° turns and prohibits all 0° turns.

Double-y was improved upon in [8] by use of the turn model. It was shown that double-y, while being fully adaptive, still places unneeded restrictions on the routing. The Maximally Fully Adaptive (mad-y) routing algorithm, which makes better use of the virtual channels and hence improves adaptiveness, was proposed. In addition, it was shown that a fully adaptive routing algorithm with fewer virtual channels could not be produced, based on the assumption that a cycle is a necessary and sufficient condition for deadlock with adaptive routing algorithms. Thus it was argued that mad-y is the best possible fully adaptive wormhole routing algorithm for 2D meshes. Any reduction in virtual channels or routing restrictions results in an algorithm that is either not fully adaptive or not deadlock-free. In the next section it is shown that the proposed algorithm is more adaptive than mad-y.

The following conventions are used throughout the paper. Channels are assumed to be bidirectional. All channels, whether physical or virtual, are referred to as virtual channels. N, S, E and W are used to indicate the appropriate direction, with N–W used to indicate a turn from North to West, for example. The symbol $VC_{nd}$ denotes virtual channel $n$ in the $d$ direction. For example, $VC_{1N}$ is virtual channel one in the North direction. The channel number is omitted when there is a single virtual channel in that direction. When the term routing algorithm is used in the rest of the paper, it refers to a fully adaptive deadlock-free wormhole routing algorithm for meshes.

## 3 Optimal Minimal Routing

The requirement that the channel dependency graph be acyclic forces unnecessary restrictions on the routing algorithm. Since mad-y has this requirement, it is not the least restrictive routing algorithm for 2D meshes. A new routing algorithm, the Optimal Fully Adaptive routing algorithm, which has substantially fewer restrictions, is proposed. This new algorithm is first presented for the 2D mesh and then generalized to $n$-dimensional meshes.

A block diagram of the router with the extra virtual channels placed in the North and South directions is shown in Figure 1(a). The virtual channels in the North and South directions are differentiated by marking $VC_{1N}$ and $VC_{1S}$ with a single dash and $VC_{2N}$ and $VC_{2S}$ with a double dash. This configuration has a total of sixteen 90° turns and four 0° turns. *Unrestricted* turns are indicated by solid lines, *restricted* turns by dashed lines and *prohibited* turns by dotted lines. The term *permitted* turns is used to describe the combination of restricted and unrestricted turns. The constraints imposed by the Optimal Fully Adaptive routing algorithm, referred to as opt-y, can be summarized succinctly: a message that needs to route further in the West direction cannot use $VC_{1N}$ or $VC_{1S}$. Opt-y has the following two sets of constraints (See Figures 1(b) and 1(c)):

- Two 90° turns, N–W using $VC_{1N}$ and S–W using $VC_{1S}$, are prohibited.
- The 0° turns from $VC_{2N}$ to $VC_{1N}$ and $VC_{2S}$ to $VC_{1S}$ are allowed only when the message does not need to route further West.

The following restrictions arise solely from the previous constraints:

- The 90° turns W–N using $VC_{1N}$ and W–S using $VC_{1S}$ are allowed only when the message does not need to route further West. These turns are restricted only because the N–W turn from $VC_{1N}$ and S–W turn from $VC_{1S}$ are prohibited.
- The 0° turns from $VC_{1N}$ to $VC_{2N}$ and $VC_{1S}$ to $VC_{2S}$ are possible only when the message does not need to

route further West, because $VC_{1N}$ and $VC_{1S}$ are not used until the message has completed routing West.

- $VC_{1N}$ and $VC_{1S}$ cannot be used by the router at the source if the message needs to route West.

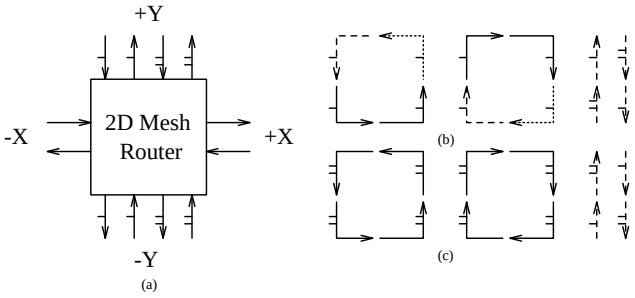


Figure 1: **Optimal Fully Adaptive Routing**

Several important properties are proved for opt-y:

- Opt-y is fully adaptive.
- Opt-y is deadlock-free.
- Opt-y requires only the minimum number of virtual channels.
- Opt-y places the fewest possible restrictions on the use of these virtual channels.
- Opt-y is the only algorithm to satisfy all four of the previous properties (except for algorithms symmetric to opt-y).

**Theorem 1** *Opt-y is fully adaptive.*

**Proof.** Recall that with minimal routing, each routing step takes the message closer to the destination. If the source and destination vary in only one dimension, there is a single minimal path. Opt-y can use this path – the single virtual channel in the East and West directions or either virtual channel in the North and South directions. For a destination that is Northeast (Southeast) of the source, the message can route adaptively along the North (South) and East virtual channels. For a destination that is Northwest (Southwest) of the source, the message can route adaptively along $VC_{2N}$ ($VC_{2S}$) and the West virtual channel. Once the message has routed completely West, it can use either of the North (South) virtual channels and switch between them. □

A message can use $VC_{1N}$ or $VC_{1S}$ only when the destination is not West of the current node. Otherwise, it must choose one of the other virtual channels. This restriction depends on the current node, not the incoming channel, so opt-y has the form $R : N \times N \rightarrow C^p$. The result proved in [7] can now be used to show that opt-y is deadlock-free.

Using the terminology in [7], the routing algorithm is denoted $R$ and the set of channels used by $R$ is denoted $C$. There is some subset of channels, $C_1 \subseteq C$, that defines a routing algorithm $R_1 \subseteq R$, i.e., $R_1$ is the routing algorithm $R$ restricted to using $C_1$. Since there are cycles in the channel dependency graph, it is not possible to provide a numbering of the channels that guarantees the channels are used in always increasing or always decreasing order. It is shown in [7] that any routing algorithm of the form $R : N \times N \rightarrow C^p$ is deadlock-free if it satisfies the following three conditions:

- The channel dependency graph defined by $C_1$ for $R_1$ is connected.
- The routing algorithm $R_1$ allows no cycles and is therefore deadlock-free.
- The additional channels ($C - C_1$) of routing algorithm $R$ do not introduce cycles in the extended channel dependency graph of $C_1$.

The idea behind the proof is that cycles are permitted if messages also have the possibility of switching to an acyclic path. The set of outgoing channels, $C^p$, always includes at least one channel in $C_1$, so a deadlock-free path is always available. In order for deadlock to be avoided, header flits cannot enter a virtual channel queue until the queue is empty. In addition, if all channels in $C^p$ are busy, the output selection policy must either defer selecting a channel or choose a busy channel in $C_1 \cap C^p$. For the channel dependency graph in Figure 1, $C_1$ consists of all the virtual channels except $VC_{2N}$ and $VC_{2S}$. Consequently, $R_1$ uses only the permitted 90° turns shown in Figure 1(b).

**Lemma 1** *The channel dependency graph defined by $C_1$ for $R_1$ is connected.*

**Proof.** If the source and destination differ in a single dimension, the message can use the virtual channel in that direction to reach the destination. For destinations that are Northeast (Southeast) of the source, the message can be routed adaptively North (South) and East. For destinations that are Northwest (Southwest) of the source, the message must route completely West and then North (South). □

**Lemma 2** *The routing algorithm $R_1$ allows no cycles and is therefore deadlock-free.*

**Proof.** The proof of this lemma can be found in [9], where $R_1$ is described as the West-First routing algorithm. □

**Lemma 3** *The additional channels of routing algorithm $R$ do not introduce cycles in the extended channel dependency graph of $C_1$.*

**Proof.** The only possibility for a cycle in $C_1$ is for an *indirect* dependency to be introduced by using $VC_{2N}$ or $VC_{2S}$. $R_1$ is fully adaptive for all messages except messages whose destination is West of the source. Indirect dependencies that allow a use of the channels different from $R_1$ could arise in only two ways: (1) A message uses $VC_W$ after using $VC_{1N}$ or $VC_{1S}$. This cannot occur, since the routing algorithm, $R$, prohibits the use of $VC_{1N}$ or $VC_{1S}$ by any message that needs to route further West. (2) A message using some $VC_W$ later uses another $VC_W$ in a different row of the mesh. This is possible only because a message that is routed West can use $VC_{2N}$ or $VC_{2S}$ and later route West again. However, this indirect dependency does not cause a deadlock. It is sufficient to show that this indirect dependency does not create a cycle in the channel dependency graph for $C_1$. The channel dependency graph for $C_1$ has no dependencies from a channel in the East, North or South directions to a channel in the West direction, i.e., the West channels are always used before any other channel in $C_1$. The indirect dependencies create new dependencies only among the West virtual channels, however, these dependencies do not create a cycle. Since minimal routing is used, there are no cycles using only the West virtual channels and hence there are no cycles in the extended channel dependency graph of $C_1$. □

**Theorem 2** *Opt-y is deadlock-free.*

**Proof.** The proof follows immediately by lemmas 1 – 3. □

## 4 Proofs of Optimality

Opt-y is optimal in two ways: (1) It requires the minimum number of virtual channels per router. (2) It requires the fewest restrictions on the use of the virtual channels. Two reasonable and standard assumptions are made when proving the optimality of opt-y. First, optimality applies only to the number of virtual channels at the interior nodes of the mesh. Nodes on the edges of the mesh need to be connected to other nodes in only two or three directions, so fewer virtual channels could be used for these routers. Second, all the nodes use the same routing algorithm.

The optimality proofs are greatly simplified by proving that *all* configurations with fewer than six virtual channels and *most* configurations with six virtual channels are either not deadlock-free or not fully adaptive. A version of this result is proved in [8] under the assumption that only acyclic routing algorithms are deadlock-free. A stronger result is proved in this paper, since no assumptions are made about the fully adaptive routing algorithm.

**Note**: The following conventions are used for figures showing deadlock configurations. In each figure, the routers are represented as circles, with the virtual channels as edges connecting the routers. The source and destination routers of message $i$ are labeled S$i$ and D$i$, respectively. Virtual channels are labeled M$i$ to indicate that message $i$ has acquired that channel. Since bidirectional channels are assumed, arrows are used to indicate in which direction the message is proceeding. Messages on the left side of a vertical channel are using $VC_1$ and messages on the right side are using $VC_2$. Messages above a horizontal channel are using $VC_1$ and messages below are using $VC_2$.

**Theorem 3** *Deadlock-free fully adaptive wormhole routing on a 2D mesh requires a second virtual channel in two opposite directions: North and South or East and West.*

**Proof.** Consider the potential deadlock configuration shown in Figure 2(a). All four messages are waiting for a virtual channel in the North or West direction. If there is only one virtual channel in the North and West directions, then the routing algorithm is not deadlock-free. This applies to *any* fully adaptive routing algorithm. Similarly, Figure 2(b) shows a deadlock configuration if there is a single virtual channel in the North and East directions, Figure 2(c) shows a deadlock configuration if there is a single virtual channel in the South and West directions and Figure 2(d) shows a deadlock configuration if there is a single virtual channel in the South and East directions. At least two more virtual channels are needed to avoid all four deadlock cases and the virtual channels must be added in opposite directions. For example, the top two deadlock configurations can be avoided by adding a virtual channel in the North direction and the bottom two deadlock configurations can be avoided by adding a virtual channel in the South direction. □

**Lemma 4** *At least two* 90° *turns must be prohibited to make the routing algorithm deadlock-free.*

**Proof.** A knot is a set of nodes that forms a cycle and no node in the set has a path to a node outside the set. A knot in the channel dependency graph is a sufficient condition for deadlock. Knots can occur when a set of messages all route in the clockwise or counter-clockwise directions, so it is necessary to prevent both of these knots. At least one 90° turn must be prohibited to prevent each knot, so at least two 90° turns must be prohibited. □

By theorem 3, opt-y requires only the minimum number of virtual channels for deadlock-free fully adaptive routing. The remainder of this section is used to show that opt-y has the fewest restrictions of *any* fully adaptive routing algorithm with six virtual channels, and that, ignoring symmetry, *all* other fully adaptive routing algorithms have more restrictions than opt-y.

Since at least two 90° turns must be prohibited by lemma 4 and opt-y prohibits only two, the only other routing algorithms that need to be considered are those that

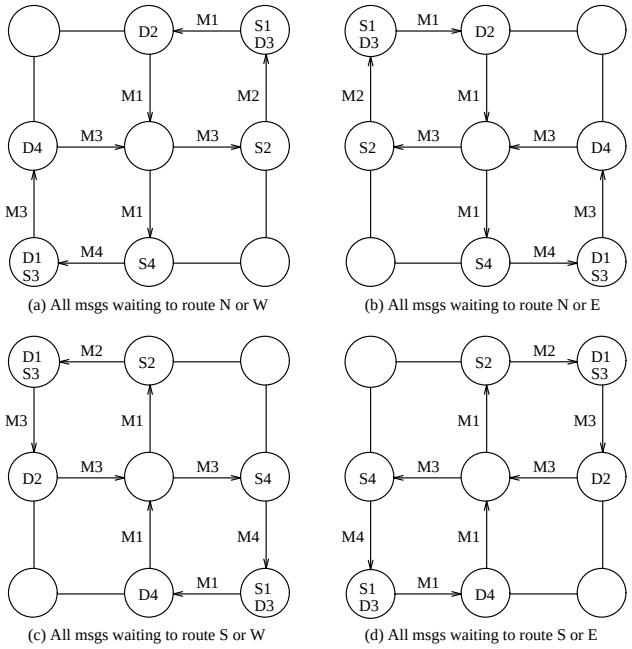


Figure 2: **Potential Deadlock Configurations**

prohibit only two 90° turns. Ignoring symmetry, there are only three such routing algorithms: North-Last, Negative-First and West-First [9]. In addition, by theorem 3, only configurations with a second virtual channel in the North and South or East and West directions need to be considered. Therefore, there are six different routing algorithms to consider. Opt-y, which uses West-First, is deadlock-free. A deadlock configuration is shown for each of the five remaining possibilities. Deadlocks arise regardless of whether or not 0° turns are permitted. **Note**: In some of the examples, deadlock can be avoided by prohibiting additional 90° turns. Such modifications are irrelevant, since the goal is to minimize the number of restrictions.

**Lemma 5** *Opt-y is not deadlock-free if the restrictions shown in Figure 1 are applied to $C_1$, but the virtual channels are added in the East and West directions.*

**Proof.** The routing algorithm is fully adaptive only if the 90° turns N–W and S–W using $VC_{2W}$ are unrestricted. A deadlock configuration is shown in Figure 3. □

**Lemma 6** *The North-Last routing algorithm is not deadlock-free with only six virtual channels and two prohibited* 90° *turns.*

**Proof.** The turn model representation of the North-Last routing algorithm with the second virtual channel in the East and West directions is shown in Figure 4(a). Only the 90° turns are shown, since the 0° turns are not used

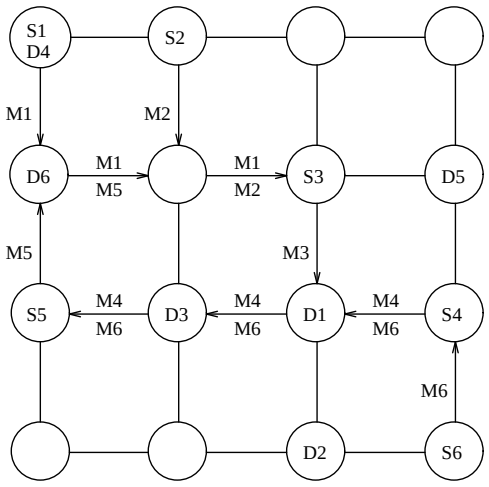


Figure 3: **Deadlock for Opt-y with E and W virtual channels**

for showing deadlock. The deadlock configuration for this routing algorithm is shown in Figure 4(b). The turn model representation with the second virtual channel in the North and South directions is shown in Figure 5(a). The deadlock configuration for this routing algorithm is shown in Figure 5(b). □

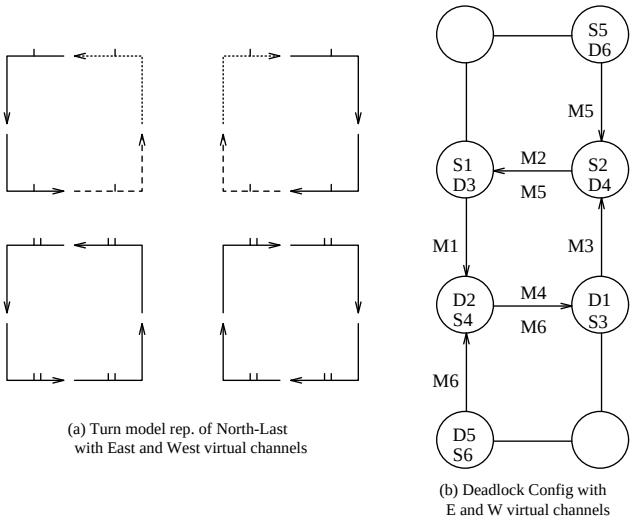


Figure 4: **North-Last with E and W virtual channels**

**Lemma 7** *The Negative-First routing algorithm is not deadlock-free with only six virtual channels and two prohibited* 90° *turns.*

**Proof.** The turn model representation of the Negative-First routing algorithm with the second virtual channel in the

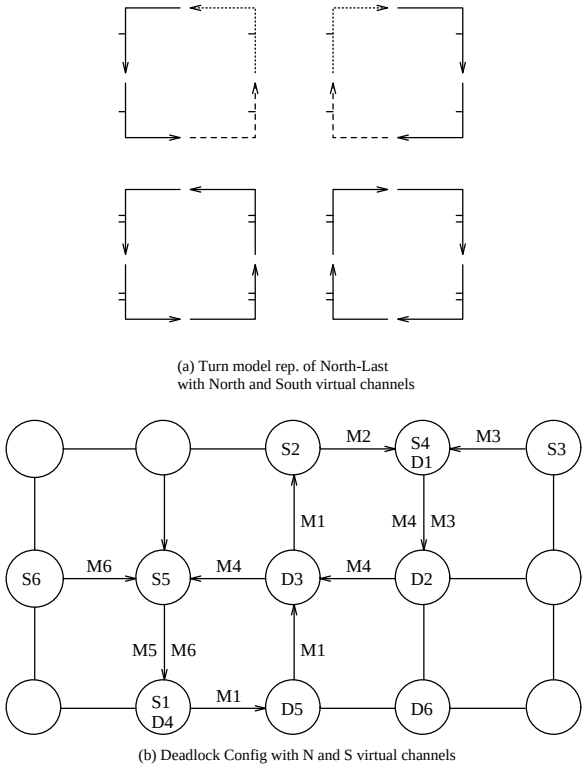


Figure 5: **North-Last with N and S virtual channels**

East and West directions is shown in Figure 6(a). Only the 90° turns are shown, since the 0° turns are not used for showing deadlock. The deadlock configuration for this routing algorithm is shown in Figure 6(b). The turn model representation with the second virtual channel in the North and South directions is shown in Figure 7(a). The deadlock configuration for this routing algorithm is shown in Figure 7(b). □

The only remaining possibility is to divide the restrictions between the virtual channels so that some are applied to $VC_1$ and the remaining to $VC_2$. However, a deadlock configuration is always possible, regardless of which two 90° turns are prohibited. This leads to theorem 4, which is proved for *symmetric* routing algorithms. Symmetry is preserved if the virtual channels are added in the X dimension with a corresponding set of restrictions and/or the restrictions are applied to $VC_2$ instead of $VC_1$.

**Theorem 4** *Ignoring symmetry, opt-y prohibits fewer* 90° *turns than any other fully adaptive deadlock-free routing algorithm which uses only six virtual channels.*

**Proof.** By lemma 4, at least two 90° turns must be prohibited. Ignoring symmetry, there are only three such routing

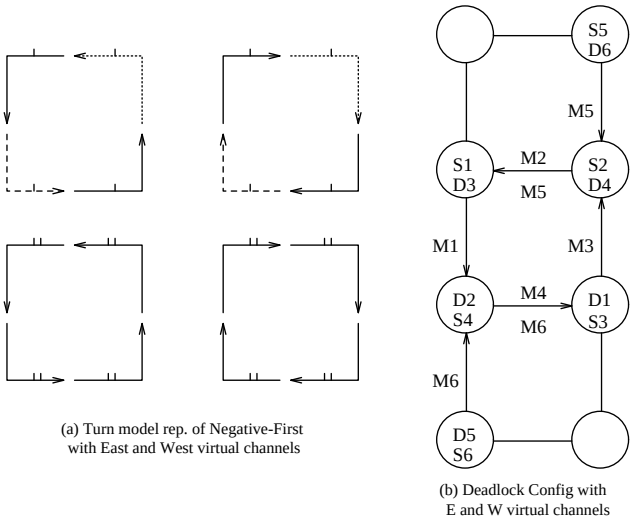


Figure 6: **Negative-First with E and W virtual channels**

algorithms [9]. Opt-y uses one of these. However, by lemma 5, it is not deadlock-free if the virtual channels are added in the East and West directions instead of the North and South directions. The other possibilities, North-Last and Negative-First, are both not deadlock-free (by lemma 6 and lemma 7, respectively). Since there are no other possibilities to consider, opt-y is the only deadlock-free fully adaptive routing algorithm that requires only six virtual channels and prohibits only two 90° turns. □

**Theorem 5** *No routing algorithm using only six virtual channels per router can have fewer restrictions than those imposed by opt-y.*

**Proof.** By lemma 4 and theorem 4, it is clear that only the 0° turns need to be considered. Opt-y restricts the use of the 0° turns from $VC_{2N}$ to $VC_{1N}$ and from $VC_{2S}$ to $VC_{1S}$ to messages that do not need to route further West. Figure 8 gives an example where deadlock occurs when the restriction on the 0° turn from $VC_{2N}$ to $VC_{1N}$ is removed. A similar deadlock configuration can be constructed if the restriction on the 0° turn from $VC_{2S}$ to $VC_{1S}$ is removed. Since the restrictions on the other two 0° turns are a direct consequence of the prohibited 90° turns, opt-y cannot be made more adaptive. □

## 5 Comparison

Opt-y is compared with mad-y, the least restrictive of any previously proposed routing algorithm. Since mad-y also requires a second virtual channel in both the North and South directions, the virtual channel requirements are the same for both routing algorithms.

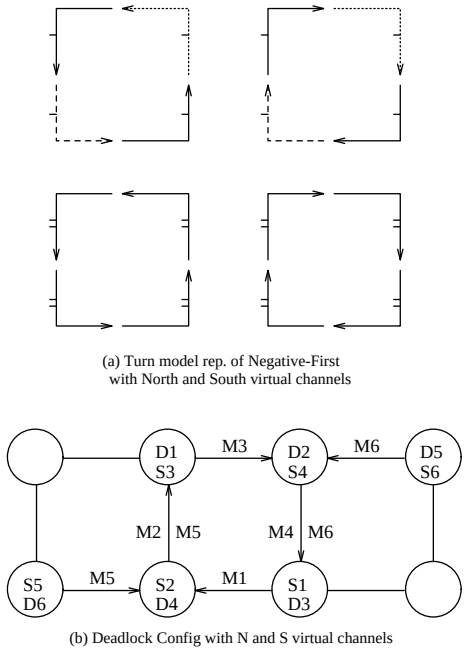


Figure 7: **Negative-First with N and S virtual channels**

Mad-y is shown in Figure 9. Mad-y has the following two sets of constraints:

- Four of the sixteen 90° turns are prohibited.
- 0° turns from $VC_{2N}$ to $VC_{1N}$ and $VC_{2S}$ to $VC_{1S}$ are prohibited.

The following three restrictions arise solely from the previous constraints:

- 0° turns from $VC_{1N}$ to $VC_{2N}$ and $VC_{1S}$ to $VC_{2S}$ are allowed only when a message does not need to route further West and has not already routed East. The West-bound restriction is because a message cannot route West from $VC_{2N}$ or $VC_{2S}$.The East-bound restriction is because a message never uses $VC_{1N}$ or $VC_{1S}$ after routing East.
- The two 90° turns N–E using $VC_{1N}$ and S–E using $VC_{1S}$ are allowed only when the message has not routed East. These turns are restricted only because a message cannot use $VC_{1N}$ or $VC_{1S}$ after $VC_E$.
- The two 90° turns W–N using $VC_{2N}$ and W–S using $VC_{2S}$ are allowed only when the message is not routed further West. These turns are restricted only because the turns from $VC_{2N}$ or $VC_{2S}$ to $VC_W$ are prohibited.

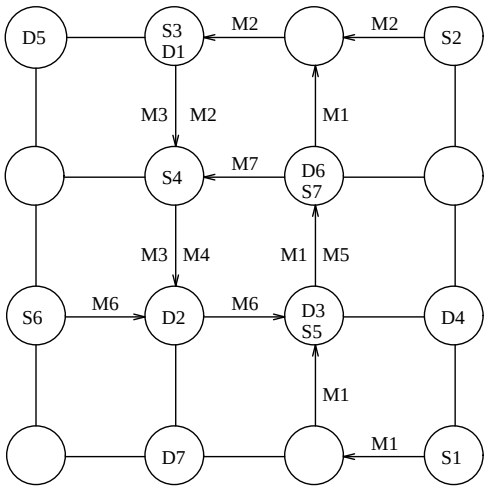


Figure 8: **Deadlock due to Fewer Restrictions**

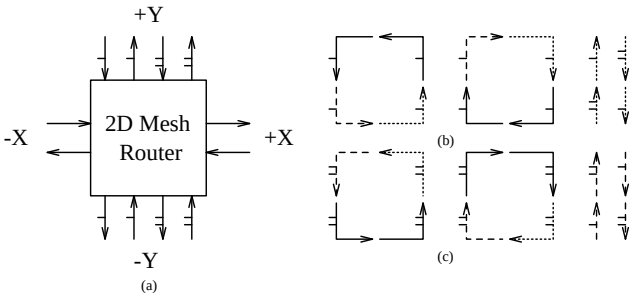


Figure 9: **Maximally Fully Adaptive Routing**

Mad-y allows four more 90° turns than double-y, but these turns are restricted so that a message uses them at most once. Additionally, a message can make at most one 0° turn, since a message cannot use $VC_{1N}$ or $VC_{1S}$ after switching to $VC_{2N}$ or $VC_{2S}$. Finally, a message cannot make both a restricted 90° turn and a 0° turn. A message that makes a restricted W–N or W–S turn is no longer in $VC_{1N}$ or $VC_{1S}$, so it cannot make a 0° turn. Similarly, a message that makes a restricted N–E or S–E turn never uses $VC_{1N}$ or $VC_{1S}$ again, so it cannot make a 0° turn. A message that makes a 0° turn cannot make a restricted turn to the East, because the message never uses $VC_{1N}$ or $VC_{1S}$ again. Similarly, a message can make a 0° turn only after routing completely West, so it cannot later make a restricted turn from the West.

For opt-y, only two 90° turns are prohibited and only two 90° turns are restricted. For mad-y, four 90° turns are prohibited and four 90° turns are restricted. Mad-y also prohibits half the 0° turns and restricts the other half. For opt-y, all the 0° turns are restricted, so none of the 0° turns

are prohibited. Therefore, opt-y imposes *less than half* as many restrictions as imposed by mad-y.

Opt-y also makes much better use of the restricted turns. The 0° turns are restricted to messages that no longer need to route West, but messages can switch from either virtual channel to the other. This allows a message to make a 0° turn *more than once*. A message also has the possibility of making one of the restricted 90° turns *and* some 0° turns.

Not only does opt-y have fewer restrictions, the router control logic is simpler as well. This is because the routing relation of mad-y is of the form $R : C \times N \rightarrow C^p$, so the set of outgoing channels from which a message can choose depends not only on the destination but also on the incoming channel. Opt-y does not differentiate based on the incoming channel because the routing relation is of the form $R : N \times N \rightarrow C^p$. This simplifies the hardware required to choose the outgoing channel on which to route the message, since the choice is independent of the incoming channel.

## 6 Generalization

The Optimal Fully Adaptive routing algorithm can be extended to $n$-dimensional meshes in a straightforward manner. Minimal routing is still assumed. The routing algorithm needs only the minimum number of virtual channels. It is not proved that the routing algorithm imposes the fewest restrictions. The routing algorithm is generalized using the following steps:

- Assign a channel to each direction of each dimension.
- Number the dimensions in some order and add a second virtual channel to both directions of all dimensions except the first dimension.
- Allow a message to route along the second virtual channel at any time.
- For each dimension except the last, choose one of the two directions. Prohibit a message from routing on the first virtual channel until it has completed routing in the chosen direction of *all* lower dimensions.
- Allow a message to make a 0° turn between the virtual channels only after the message has completed routing in the chosen direction of *all* lower dimensions.

For example, consider a 4D mesh with dimensions XYZW and number the dimensions 1, 2, 3 and 4 respectively. The positive and negative directions in the Z dimension are called Up and Down respectively. The positive and negative directions in the W dimension are called In and Out respectively. Let the negative directions in the X, Y and Z dimensions be the chosen directions. A second virtual channel is added in both directions of dimensions Y, Z and W. A message can route adaptively in any dimension using virtual channel one once it has completed routing in the chosen direction of all lower dimensions. For example, a message can use $VC_{1N}$ and $VC_{1S}$ only after it has completed routing West. Similarly, a message can use $VC_{1U}$ and $VC_{1D}$ only after it has completed routing West and South. A message can use $VC_{1I}$ and $VC_{1O}$ only after it has completed routing West, South and Down. A message can route, without any restrictions, in the X dimension and in any direction using virtual channel two.

The Optimal Fully Adaptive routing algorithm is fully adaptive and deadlock-free for arbitrary dimension meshes. Furthermore, this algorithm uses only the minimum number of virtual channels. The proofs, given in [12], are extensions of the proofs for the 2D mesh.

Limited research has been done on wormhole routing for higher dimension meshes. For example, mad-y has been defined only for 2D meshes [8]. The partially adaptive routing algorithm proposed by Glass and Ni in [9], while requiring no additional virtual channels, allows only about $1/2^{n-1}$ of the paths provided by a fully adaptive routing algorithm for an $n$-dimensional mesh. Three virtual channels per physical channel are used by Chien and Kim in [2] to support partially adaptive routing for arbitrary dimension meshes. Jesshope, Miller and Yantchev propose a routing algorithm that is fully adaptive by dividing an $n$-dimensional mesh into $2^n$ regions and using separate virtual channels for each region [10]. Fully adaptive routing on arbitrary dimension meshes requires $2^{n-1}$ subnetworks with $n+1$ levels per subnetwork and one virtual channel per level for each router using the routing algorithm proposed by Linder and Harden in [11]. Table 2 summarizes the number of virtual channels per router required by each routing algorithm.

It can be seen from Table 2 that there is a *dramatic* reduction in the number of virtual channels required by the Optimal Fully Adaptive routing algorithm compared to the algorithms proposed in [10, 11]. In addition, fully adaptive routing is possible using about *two-thirds* the virtual channels needed for partially adaptive routing using the routing algorithm proposed in [2]. The only adaptive algorithm that requires fewer virtual channels is the partially adaptive algorithm proposed in [9]. With less than twice as many virtual channels, the Optimal Fully Adaptive routing algorithm allows an average of $2^{n-1}$ times as many paths.

## 7 Conclusion

A deadlock-free fully adaptive routing algorithm has been proposed for wormhole routing on $n$-dimensional meshes. It has been proved for the 2D mesh that the routing algorithm requires the minimum number of virtual channels and places the fewest restrictions on the use of those

Table 2: **Virtual Channels per Router**

| Topology | Deterministic | Partially Adaptive | | Fully Adaptive | | |
|---|---|---|---|---|---|---|
| | Dimension Order | Glass & Ni | Chien & Kim | Jesshope, Miller & Yantchev | Linder & Harden | Optimal Fully Adaptive |
| 2D Mesh | 4 | 4 | 6 | 8 | 6 | 6 |
| 3D Mesh | 6 | 6 | 12 | 24 | 16 | 10 |
| 4D Mesh | 8 | 8 | 18 | 64 | 40 | 14 |
| $n$D Mesh | $2n$ | $2n$ | $6n-6$ | $n2^n$ | $(n+1)2^{n-1}$ | $4n-2$ |

virtual channels. Ignoring symmetry, opt-y is the *only* fully adaptive routing algorithm for 2D meshes that satisfies both of these properties.

An important open problem currently under investigation is what constitutes a necessary and sufficient condition for deadlock freedom in adaptive routing algorithms. Designing optimal routing algorithms for arbitrary topologies would be greatly simplified by a necessary and sufficient condition for proving routing algorithms are deadlock-free. The necessary and sufficient condition proved in [6] applies only to deterministic routing. There is no known necessary and sufficient condition for adaptive routing relations of the form $R : C \times N \rightarrow C^p$ or $R : N \times N \rightarrow C^p$.

The routing algorithm requires only the minimum number of virtual channels for an $n$-dimensional mesh. It has not been shown that the proposed routing algorithm has the minimum number of restrictions for arbitrary dimension meshes. Proving optimality would require the examination of many possible routing algorithms. A necessary and sufficient condition for deadlock freedom would greatly facilitate the proof of this open problem by reducing the cases that need to be considered.

## Acknowledgments

The authors would like to thank Dave Lutz, D. K. Panda, Kant Patel, and Shobana Balakrishnan for valuable discussions which have improved the quality of the paper.

## References

[1] A. A. Chien. A Cost and Speed Model for k-ary n-cube Wormhole Routers. In *Hot Interconnects '93*, August 1993.

[2] A. A. Chien and J. H. Kim. Planar-Adaptive Routing: Low-cost Adaptive Networks for Multiprocessors. In *Proceedings of the* $19^{th}$ *Annual International Symposium on Computer Architecture*, pages 268–277, 1992.

[3] W. J. Dally. Fine-Grain Message-Passing Concurrent Computers. In *Proceedings of the Third Conference on Hypercube Concurrent Computers*, volume 1, pages 2–12, 1988.

[4] W. J. Dally. Virtual-Channel Flow Control. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):194–205, March 1992.

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

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

[7] J. Duato. On the Design of Deadlock-Free Adaptive Routing Algorithms for Multicomputers: Design Methodologies. In *Parallel Architectures and Languages Europe 91*, volume I, pages 390–405, 1991.

[8] C. Glass and L. M. Ni. Maximally Fully Adaptive Routing in 2D Meshes. In *International Conference on Parallel Processing*, volume I, pages 101–104, 1992.

[9] C. Glass and L. M. Ni. The Turn Model for Adaptive Routing. In *Proceedings of the* $19^{th}$ *Annual International Symposium on Computer Architecture*, pages 278–287, 1992.

[10] C. R. Jesshope, P. R. Miller, and J. T. Yantchev. High Performance Communications in Processor Networks. In *Proceedings of the* $16^{th}$ *Annual International Symposium on Computer Architecture*, pages 150–157, 1989.

[11] 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.

[12] L. Schwiebert and D. N. Jayasimha. Optimal Fully Adaptive Wormhole Routing for Meshes. Technical Report OSU-CISRC-4/93-TR16, The Ohio State University, 1993.