A New Theory of Deadlock-Free Adaptive Multicast Routing in Wormhole Networks

J. Duato
Facultad de Informática
Universidad Politécnica de Valencia
P.O.B. 22012, 46071 - Valencia, SPAIN
E-mail: jduato@aii.upv.es

Abstract

A theory for the design of deadlock-free adaptive routing algorithms for wormhole networks has been proposed in [11, 14]. This theory supplies the sufficient conditions for an adaptive routing algorithm to be deadlock-free, even when there are cyclic dependencies between channels. Also, two design methodologies have been proposed.

Multicast communication refers to the delivery of the same message from one source node to an arbitrary number of destination nodes. Two multicast wormhole routing methods have been presented in [22] for multicomputers with 2D-mesh and hypercube topologies.

This paper develops the theoretical background for the design of deadlock-free adaptive multicast routing algorithms for wormhole networks. Some basic definitions and two theorems are proposed, developing conditions to verify that an adaptive multicast routing algorithm is deadlock-free, even when there are cyclic dependencies between channels. As an example, the multicast routing algorithms presented in [22] are extended, so that they can take advantage of the alternative paths offered by the network.

1 Introduction

Multicomputers [1] rely on an interconnection network between processors to support the message-passing mechanism. Second generation multicomputers use a routing mechanism known as *wormhole* routing [8]. As messages are typically at least a few words long, each message is serialized into a sequence of parallel data units, referred to as flow control units, or *flits* [7]. The flit at the head of a message governs the route. As the header flit advances along the specified route, the remaining flits follow it in a pipeline fashion. If the header encounters a channel already in use, it is blocked until the channel is freed; the flow control within the network blocks the trailing flits.

A more efficient flow control technique known as *mad postman* has been proposed in [18]. It forwards the message header even before the routing operation takes place. It is very well suited for meshes with narrow channels.

Deadlocks may appear if the routing algorithms are not carefully designed. A deadlock in the interconnection network of a multicomputer occurs when no message can advance toward its destination because the queues of the message system are full. The size of the queues strongly influences the probability of reaching a deadlocked configuration. First generation multicomputers buffer full messages or relatively large packets. On the contrary, second generation machines buffer flits, being more deadlock-prone. So, the only practical way to avoid deadlock is to design deadlock-free routing algorithms.

With wormhole routing or mad postman, buffer allocation cannot be restricted, because flits have no routing information. Once the header of a message has been accepted by a channel, the remaining flits must be accepted before the flits of any other message can be accepted. So, routing must be restricted to avoid deadlock. As mad postman has the same deadlock properties as wormhole routing, in what follows we will only refer to wormhole.

Dally [8] has proposed a methodology to design deterministic routing algorithms under general assumptions. He defines a channel dependency graph and establishes a total order among channels. Routing is restricted to visit channels in decreasing or increasing order to eliminate cycles in the channel dependency graph. This methodology has been applied to the design of routing chips for multicomputers [7] and multicomputer nodes with integrated communication support [3]. It has also been applied to systolic communication [19, 3].

Although Dally's theorem was proposed for deterministic routing, it has been extensively applied to adaptive routing [5, 9, 17, 18, 24]. An important study about deadlock-free adaptive routing is the turn model [17]. In this model, a message produces a turn when it changes direction, usually changing from one dimension to another. Turns are combined into cycles. Prohibiting just enough turns to break all the cycles prevents deadlock. However, the turn model is also based on Dally's theorem, requiring the absence of cycles in the channel dependency graph. This approach is too restrictive, limiting the number of alternative paths that can be used.

A different approach consists of allowing the existence of cyclic dependencies between channels [2, 6, 11, 29]. In this case, it is possible to provide fully adaptive routing with a modest number of virtual channels or buffers. For instance, the *-channels routing algorithm for the torus topology [2] prevents deadlock by dynamically disallowing paths based on occupancy of buffers in neighboring nodes. This means that routing decisions depend on the availability of accurate buffer status from neighboring nodes.

However, the theory proposed in [10, 11, 14] is not restricted to any particular topology or routing algorithm. Cyclic dependencies between channels are allowed, provided that there exists a connected channel subset free of cyclic dependencies. To simplify the application of the former theory, two design methodologies have been proposed. Some adaptive routing algorithms designed with the former methodologies have been evaluated by simulation [12, 13, 14, 16, 25, 26]. The results show an important improvement over previous routing algorithms.

The above mentioned routing schemes are intended for one-to-one communications. However, there are several parallel applications that require one-to-many communications, such as the simulation of computer networks, multiprocessors, circuits, etc. as well as many parallel algorithms. Multicast communication allows the delivery of the same message from one source node to an arbitrary number of destination nodes. nCUBE–2 supports broadcast and special multicast in which the destination nodes form a subcube [27]. For this purpose, nCUBE–2 supports tree-like wormhole routing. It also supports multiple node addresses in the address field of a message. In this case, the message transmission has to follow the order of addresses specified in the address field.

Multicast communication can be implemented by multiple one-to-one or broadcast communications. However, these alternative methods create too much unnecessary traffic and are likely to cause message delay and network congestion, and even deadlock [28]. Multicast communication has been studied in [20, 21], in which several graph models and multicast routing algorithms have been proposed. The multicast deadlock problem has been studied in [4]. Three multicast protocols, multi-unicast, resumable and restricted branch multicast have been proposed to deal with the deadlock problem. However, these protocols are based on virtual cut-through.

The implementation of multicast communication using wormhole routing is more complex, because when a message header is blocked, the flits belonging to that message remain in the network. Then, a tree-like routing scheme is not suitable for wormhole networks because it produces many headers for each message, drastically increasing the probability that a message is blocked. In [22], a path-based multicast routing model has been proposed for multicast communications. In this model, flits are not replicated at intermediate nodes. Instead, the source node sends several copies of the message, each one destined for a subset of the destination nodes. When an intermediate destination is reached, flits are copied to local memory as they flow through the node. Also, two heuristic multicast routing algorithms have been described, showing the performance improvement over multiple one-to-one and broadcast communications.

The former algorithms are deterministic. They do not take advantage of the alternative paths offered by the network. Also, the absence of deadlocks has been guaranteed by eliminating the cyclic dependencies between channels. From the implementation point of view, it is desirable that the same routing algorithm is used for messages with one and with several destination nodes. However, as stated above, adaptive routing algorithms which allow cyclic dependencies between channels and take advantage of all the minimal paths offered by the network increase performance drastically [14].

Recently, Lin et al. [23] have proposed three adaptive multicast routing algorithms for 2D meshes, called PM, FM and LD. Although the authors state that the former algorithms are deadlock-free, no proof is given for the PM and FM routing algorithms. Additionally, those algorithms have cyclic dependencies between channels.

In this paper, we propose a theory for the design of deadlock-free adaptive multicast routing algorithms. Two theorems supply the sufficient conditions for an adaptive multicast routing algorithm to be deadlock-free, even when there are cyclic dependencies between channels. As an example, the algorithm proposed in [22] is extended, so that messages can use almost all the alternative minimal paths between successive destinations offered by the network.

Firstly, we introduce the new theory and the new kinds of channel dependency in section 2. Section 3 develops the theory for multicast wormhole routing. Section 4 proposes a new adaptive multicast routing algorithm, showing the application of the former theory. Finally, some conclusions are drawn.

2 Informal description

With deterministic routing, messages have a single routing option at each node. Then, it is necessary to remove all the cyclic dependencies between channels to prevent deadlocks [8]. When adaptive routing is considered, messages usually have several choices at each node. Then, it is not necessary to eliminate all the cyclic dependencies, provided that every message can always find a path toward its destination (connected subset) whose channels are not involved in cyclic dependencies.

For store-and-forward networks, it is sufficient with the existence of a connected channel subset C_1 whose channels are not involved in cyclic dependencies [11]. However, for wormhole networks, it is more complex. As each message can reserve several channels simultaneously, it is possible for a message to reserve a channel belonging to C_1 and then another channel not belonging to C_1, which may be involved in a cyclic dependency. In other words, a message involved in a cyclic dependency may occupy channels belonging to the escape paths, thus blocking other messages which try to use the channels of C_1 to reach their destination. We solved this problem by introducing indirect dependencies [11]. When a message uses a channel belonging to C_1, then another channel not belonging to C_1 and then try to use a channel belonging to C_1, there is an indirect dependency between both channels belonging to C_1. If cyclic dependencies are not allowed between channels belonging to C_1, even when indirect dependencies are considered, then we guarantee that the escape paths will not block forever.

In path-based multicast, after reaching an intermediate destination, a message header is routed again towards the next destination. This may produce additional channel dependencies as we will see with an example. Consider the XY routing algorithm for the 2D mesh. It prevents the use of a horizontal channel after using a vertical one. Consider a message with two destinations: the first one is below the current node and the second one is on the right from the first destination. If we use the XY routing algorithm for path-based multicast, the first destination is reached through a vertical channel. Then, a horizontal channel is used to reach the second destination, thus producing a dependency from vertical to horizontal channels. This dependency is due to the inclusion of multiple destinations on the message header and it is called direct multicast dependency. If we combine multicast dependencies with indirect dependencies, we obtain indirect multicast dependencies. By including all the new dependencies in the graph and prohibiting cycles in the escape channels, we can guarantee the absence of deadlock.

The main problem with the former approach is that obtaining the multicast dependencies is expensive. Effectively, for each pair of channels, all the possible source nodes must be considered. Moreover, for each source node, all the destination sets which require crossing the former channels must be considered. Then, we introduce the concept of compatibility between the routing function and the function which splits the destination sets. When those functions are compatible, the existence of a multicast dependency between two channels does not depend neither on the source node nor the destination set. In this way, we drastically simplify the application of this theory.

3 Deadlock-Free Adaptive Multicast Routing

This section develops the theoretical background for the design of deadlock-free adaptive multicast routing algorithms for networks using wormhole routing. An extended and more formal version of this theory can be found in [15].

3.1 Assumptions

The basic assumptions are an extension of the ones proposed in [14]. These assumptions are the following:

1. A node can generate messages destined for any set of nodes at any rate. The set of destination nodes for a message may be split into several ordered subsets at the source node, sending a copy of the message for each subset.

2. A node can generate messages of arbitrary length. Messages will generally be longer than a single flit. Also, message headers may require more than a single flit to store routing information.

3. Wormhole routing is used. So, once a queue accepts the first flit of a message, it must accept the remainder of the message before accepting any flits from another message. Also, a message is routed as soon as the routing information reaches a node. Then, a message may occupy several channels simultaneously.

4. A message arriving at any of its destination nodes is eventually consumed.

5. Path-based multicast is used. Then, flits are not replicated at any node, except when the message reaches an intermediate destination. In this case, flits are copied to local memory as they flow through the node. Flits are only removed from the network when they reach their last destination. Also, when several copies of a message must be forwarded, flow control is independent for each copy. In particular, flits are independently read from memory at the source node.

6. An available queue may arbitrate between messages that request that queue, but may not choose among waiting messages.

7. A queue cannot contain flits belonging to different messages. After accepting a tail flit, a queue must be emptied before accepting another header. Then, when a message is blocked, its header will always occupy the head of a queue.

8. The route taken by a message depends on its destination set and the status of output channels (free or busy). Destinations must be reached in the same order established at the source node. At a given node, the routing function supplies a set of output channels based on the current node and the next destination node. A selection from this set is made based on the status of output channels at the current node. So, *adaptive* routing will be considered.

3.2 Definitions

Before proposing the theorems, some definitions are needed:

Definition 1 An *interconnection network* I is a strongly connected directed multigraph, $I = G(N, C)$. The vertices of the multigraph N represent the set of processing nodes. The arcs of the multigraph C represent the set of communication channels. More than a single channel is allowed to connect a given pair of nodes. Each channel c_i has an associated queue. The source and destination nodes of channel c_i are denoted s_i and d_i, respectively.

Definition 2 A *destination set* $Q \subset N$ is an ordered set of nodes for which a message is destined. The lowest element in the set corresponds to the first destination to be reached. After reaching an intermediate destination n_d, the next destination to be reached is denoted $succ(n_d)$.

Definition 3 The *destination superset* $U \subset \mathcal{P}(N)$ for a given interconnection network I, where $\mathcal{P}(N)$ is the power set of N, is the set of all the destination sets for I.

Definition 4 A *split-and-sort function* $SS : N \times U \rightarrow \mathcal{P}(U)$ splits the destination set for a message generated at node n_s into one or more destination sets, possibly reordering the nodes. The destination sets supplied by this function are *valid*. Before sending a message to a given destination set, the split-and-sort function is applied to it.

Definition 5 Let F be the set of valid *channel status*, $F = \{free, busy\}$.

Definition 6 An *adaptive routing function* $R : N \times N \rightarrow \mathcal{P}(C)$, supplies a set of alternative output channels to send a message from the current node n_c to the next node n_d from its destination set Q, $R(n_c, n_d) = \{c_1, c_2, \ldots, c_p\}$. If the message has reached a node belonging to its destination set Q and it is not the last destination, the message header will be routed again in the same node toward the next destination $succ(n_d)$.

Definition 7 A *selection function* $S : \mathcal{P}(C \times F) \rightarrow C$ selects a free output channel (if any) from the set supplied by the routing function. From the definition, S takes into account the status of all the channels belonging to the set supplied by the routing function. The decomposition of the adaptive routing into two functions (routing and selection) will be critical while proving the theorems, because only the split-and-sort and routing functions determine whether a multicast routing algorithm is deadlock-free or not. Then, the selection function will only affect performance.

Definition 8 A routing function R for a given interconnection network I is *connected* iff, for any pair of nodes, it is possible to establish a *path* between them using channels belonging to the sets supplied by R.

Definition 9 A *routing subfunction* R_1 for a given routing function R and channel subset $C_1 \subseteq C$, is a routing function identical to R, except that it is restricted to use the channels belonging to C_1.

Definition 10 A split-and-sort function SS and a connected routing function R for an interconnection network I form a *compatible pair* (SS, R) iff, for all the source nodes and for every valid destination set Q, when a message destined for the set Q is being routed, the destination subset containing the destinations which have not been reached yet is a valid destination set for the node that contains the message header. It must be noticed that messages have no memory of the path they have followed, because the routing function only considers the current node and the next destination. This definition extends that basic idea (no memory) to the pair (SS, R).

Definition 11 Given an interconnection network I, a routing function R and a pair of channels $c_i, c_j \in C$, there is a *direct dependency* from c_i to c_j iff c_j can be used immediately after c_i by messages destined to some node n.

Definition 12 Given an interconnection network I, a compatible pair (SS, R) and a pair of channels $c_i, c_j \in C$, there is a *direct multicast dependency* from c_i to c_j iff c_j can be used immediately after c_i by messages destined to d_i and to another node, both belonging to a valid destination set for s_i.

Definition 13 Given an interconnection network I, a routing function R, a channel subset $C_1 \subset C$ which defines a routing subfunction R_1 and a pair of channels $c_i, c_j \in C_1$, there is an *indirect dependency* from c_i to c_j iff it is possible to establish a path from s_i to d_j for messages destined to some node n. c_i and c_j are the first and last channels in that path and the only ones belonging to C_1. Then, c_j can be used after c_i by some messages. As c_i and c_j are not adjacent, some other channels belonging to $C - C_1$ are used between them.

Definition 14 Given an interconnection network I, a compatible pair (SS, R), a channel subset $C_1 \subset C$ which defines a routing subfunction R_1 and a pair of channels $c_i, c_j \in C_1$, there is an *indirect multicast dependency* from c_i to c_j iff it is possible to establish a path from s_i to d_j for messages destined to some valid destination set Q. Again, c_i and c_j are the first and last channels in that path and the only ones belonging to C_1. As c_i and c_j are not adjacent, some other channels belonging to $C - C_1$ are used between them. Additionally, some intermediate destinations may be reached while establishing the path from s_i to d_j.

Definition 15 A *multicast channel dependency graph* D_M for a given interconnection network I and compatible pair (SS, R), is a directed graph, $D_M = G(C, E_M)$. The vertices of D_M are the channels of I. The arcs of D_M are the pairs of channels (c_i, c_j) such that there is either a direct or a direct multicast dependency from c_i to c_j.

Definition 16 An *extended multicast channel dependency graph* D_{EM} for a given interconnection network I and compatible pair (SS, R_1), where R_1 is a routing subfunction of a routing function R, is a directed graph, $D_{EM} = G(C_1, E_{EM})$. The vertices of D_{EM} are the channels that define the routing subfunction R_1. The arcs of D_{EM} are the pairs of channels (c_i, c_j) such that there is a direct, direct multicast, indirect or indirect multicast dependency from c_i to c_j.

Definition 17 A *configuration* is an assignment of a set of flits to each queue. All of the flits in any one queue belong to the same message (assumption 7). If the first flit is not a header and the next channel reserved by its header is c_j, then $next(c_i) = c_j$. A configuration is *legal* iff for each channel the queue capacity is not exceeded and all the flits stored in the queue (if any), which have the same destination set, can reach the channel from the previous node using the routing function. Also, the destination set for those flits must be valid.

Definition 18 A *deadlocked configuration* for a given interconnection network I and compatible pair (SS, R)

is a nonempty legal configuration verifying the following conditions:

1. There is not any flit one hop from its last destination.
2. Message headers cannot advance because the queues for all the alternative output channels supplied by the routing function are not empty (see assumption 7). As a particular case, when the header has reached an intermediate destination, the next destination is used for routing.
3. Data and tail flits cannot advance because the next channel reserved by their message header has a full queue.

Definition 19 A compatible pair (SS, R) for an interconnection network I is *deadlock-free* iff there is not any deadlocked configuration for that pair of functions.

Definition 20 A *sink* channel for a given interconnection network I and routing function R is a channel c_i such that all the flits that enter it reach their next destination in a single hop.

Definition 21 A *perfect sink* for a given interconnection network I and compatible pair (SS, R) is a channel c_i such that d_i is the last destination for messages reaching d_i through c_i. In other words, all the flits that enter a perfect sink reach their last destination in a single hop.

3.3 Theorems

Two theorems are proposed. The first one is an extension of Dally's theorem for adaptive multicast routing. The second one allows the design of adaptive multicast routing algorithms with cyclic dependencies in their multicast channel dependency graph.

Theorem 1 *A compatible pair (SS, R) for an interconnection network I, where R is connected and adaptive, is deadlock-free if there are no cycles in its multicast channel dependency graph D_M.*

Proof sketch:
$\Leftarrow$ As the multicast channel dependency graph for (SS, R) is acyclic, it is possible to establish an order between the channels of C. As R is connected and the pair (SS, R) is compatible, the minimals of that order are also perfect sinks. Suppose that there is a deadlocked configuration for (SS, R). Let c_i be a channel of C with a nonempty queue such that there are no channels less than c_i with a nonempty queue. If c_i is a minimal (that is, a perfect sink) then the flit at the queue head can reach its last destination in a single hop and there is no deadlock. Otherwise, using the channels less than c_i, the flit at the queue head of c_i can advance and there is not a deadlock.□

Theorem 2 *A compatible pair (SS, R) for an interconnection network I, where R is connected and adaptive, is deadlock-free if there exists a subset of channels $C_1 \subseteq C$ that defines a connected routing subfunction R_1 and the pair (SS, R_1) has no cycles in its extended multicast channel dependency graph D_{EM}.*

Proof sketch:
$\Leftarrow$ The case $C_1 = C$ is trivial. Otherwise $C_1 \subset C$. As the extended multicast channel dependency graph for (SS, R_1) is acyclic, it is possible to establish an order between the channels of C_1. As R_1 is connected and the pair (SS, R_1) is compatible, the minimals of that order are also perfect sinks. Suppose that there is a deadlocked configuration for (SS, R). There are two possible cases:

a) The queues for channels belonging to C_1 are empty. As R_1 is connected and the header flits are at queue heads, each header can be routed using channels belonging to C_1 and there is no deadlock.

b) The queues for channels belonging to C_1 are not empty. Let c_i be a channel of C_1 with a nonempty queue such that there are no channels less than c_i with a nonempty queue. Again, there are two possible cases:

b1) If c_i is a minimal (perfect sink) then the flit at the queue head is not blocked and there is no deadlock.

b2) If c_i is not a minimal, all the channels of C_1 less than c_i will have empty queues, existing three possible cases:

b2.1) If c_i has a header at the queue head, it can be routed because R_1 is connected and there is no deadlock.

b2.2) If there is a data flit at the queue head of c_i and $next(c_i)$ belongs to C_1, that flit can also advance.

b2.3) If $next(c_i)$ belongs to $C - C_1$, we have to use the indirect and indirect multicast dependencies in the extended multicast channel dependency graph. Let c_k be the channel containing the header of the data flits contained in c_i. Then, it is possible to find a channel c_j belonging to C_1 to route that header, because R_1 is connected. In that case, there is an indirect or indirect multicast dependency from c_i to c_j $(c_i > c_j)$, implying that c_j is empty and there is no deadlock.□

There are some interesting considerations:

1. The basic idea behind theorem 2 is that one can have a split-and-sort function and an adaptive routing function with cyclic dependencies between channels, provided that there are alternative paths without cyclic dependencies to send a given flit towards its destination set. As messages are several flits long, the extended multicast channel dependency graph must be used to take into account indirect and indirect multicast dependencies.

2. This theory is valid for both, routing and multicast routing. It is also valid when messages with a single destination and with multiple destinations are mixed together. This property is very important for real machines. The use of the same routing function for routing and multicast routing drastically simplifies the implementation of the router.

3. Usually, routing functions verify that $\forall c_i \in C, c_i \in R(s_i, d_i)$. In this case, the only way to avoid direct multicast and indirect multicast dependencies consists of restricting the destination sets. This is the reason why we have defined the split-and-sort function, restricting the set of valid destination sets for each node.

4. This theory could have been developed without defining the compatibility between the split-and-sort function and the routing function. In that case, multicast dependencies should have been defined in a different way, making the proofs simpler. However, we think that compatibility is an interesting property for real implementations. Moreover, the current definitions for multicast dependencies are easier to handle. Then, programs that build the channel dependency graphs from the definitions for SS and R run faster.

4 Design Example

In this section we describe an adaptive multicast routing algorithm. It will serve as an example of the application of the theory proposed in section 3. In this example, we will extend the multicast routing algorithms described in [22] by allowing messages to use the alternative minimal paths between successive destinations offered by the interconnection network.

The algorithms described in [22] have two parts: message preparation and routing. The first part splits the destination set for a message into two or more subsets, then it sorts each subset. This part has been defined as split-and-sort function in section 3.2. The second part determines the path followed by the message until all the destination nodes are reached. It corresponds to the routing function defined in section 3.2. There is no counterpart for the selection function because the routing algorithms are deterministic. Two algorithms have been described in [22], dual-path and multi-path. They only differ in the message preparation part, splitting the destination set either into two subsets (dual-path) or into several subsets (multi-path). As we will only modify the routing function, we will focus on the dual-path algorithm.

First of all, we are going to describe the dual-path algorithm. We will restrict our discussion to the 2D-mesh. Suppose that the address of a node is represented by its integer coordinate (x, y), where the lower left node is $(0, 0)$. The label assignment function L for a $m \times n$ mesh is defined as

$$L(x, y) = \begin{cases} y * n + x & \text{if } y \text{ is even} \\ y * n + n - x - 1 & \text{if } y \text{ is odd} \end{cases}$$

This function defines a Hamiltonian path in the network. See Fig. 1 for an example of label assignment on a 8×8 mesh. The message preparation algorithm (split-and-sort function SS) divides the destination set Q for a message generated at node n_s into two subsets Q_H and Q_L, such that Q_H contains all the destination nodes with L value higher than $L(n_s)$ and Q_L the nodes with L value lower than $L(n_s)$. Then, the destination nodes in Q_H are sorted in ascending order using L as the key. The same is done for Q_L, but in descending order. The routing function is defined in [22] as $R : N \times N \rightarrow N$. This function supplies the destination node of the output channel instead of the channel itself. If the current and destination nodes for the message are denoted n_c and n_d, respectively, then $R_1(n_c, n_d) = n_r$, such that n_r is a neighboring node of n_c and

$$L(n_r) = \begin{cases} max\{L(x)|L(x) \leq L(n_d)\} & \text{if } L(n_c) < L(n_d) \\ min\{L(x)|L(x) \geq L(n_d)\} & \text{if } L(n_c) > L(n_d) \end{cases}$$

where x is a neighboring node of n_c.

Then, this function routes a message by sending it to the neighboring node with the highest (lowest) label not higher (lower) than the label of the destination node. As proved in [22], this routing function is deadlock-free and it forwards messages following a minimal path between successive destinations.

Now, we are going to extend the former routing algorithm. It defines a minimal path connected deterministic routing function R_1. Now, each physical channel is split into a set of additional virtual channels. Let C_1 be the set of original channels and let C be the set of all the (virtual) channels in the network. Let C_{xyH} be the set of output channels from node x such that each channel belongs to a minimal path from x to y and the label of its destination node is not higher than the label of y. Let C_{xyL} be the set of output channels from node x such that each channel belongs to a minimal path from x to y and the label of its destination node is not lower than the label of y. The new routing function R is defined as follows:

$$\forall x, y \in N, \;\; R(x, y) =$$

$$\begin{cases} R_1(x, y) \cup (C_{xyH} \cap (C - C_1)) & \text{if } L(x) < L(y) \\ R_1(x, y) \cup (C_{xyL} \cap (C - C_1)) & \text{if } L(x) > L(y) \end{cases}$$

In other words, the new routing function can use any of the new channels belonging to a minimal path between successive destinations, except when a node with a label higher (for $L(x) < L(y)$) or lower (for $L(x) > L(y)$) than the label of the destination node is reached. Alternatively, the channels supplied by R_1 can also be used. This algorithm is quite similar to the LD algorithm [23], except that our algorithm follows a minimal path between successive destinations and has less routing restrictions, as we will see. Of course, our algorithm can be easily modified so that it follows non-minimal paths.

Fig. 1 shows the label assignment and a routing example for a 8×8 mesh. In what follows, we will refer to nodes using their labels instead of their integer coordinates. The destination set for the example has been split into two subsets, $Q_H = \{41, 52, 53, 62\}$ and $Q_L = \{1\}$. Solid arrows show the path supplied by the deterministic routing function R_1. Dashed arrows show the additional paths offered by the adaptive routing function R. Inside each path, all the new virtual channels can be used by R.

It can be seen that R does not use all the minimal paths between successive destinations. For instance,

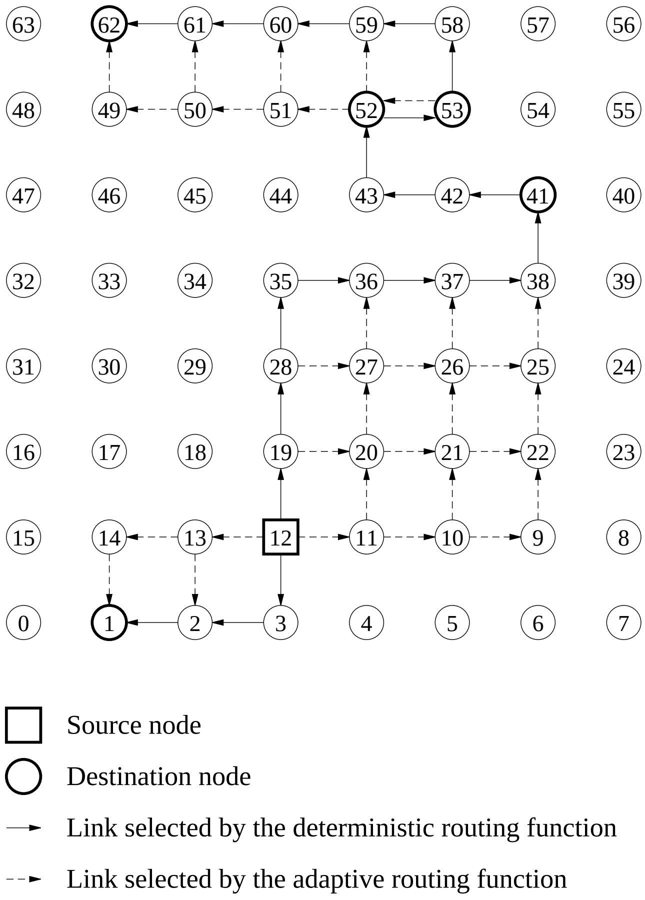

Figure 1: Label assignment and routing example for a 8×8 mesh.

when the next destination is node 41, the message is not allowed to reach nodes 44, 43 or 42, because they have a higher label value than the next destination node. However, R may route a message destined for a node with a higher label value through nodes with decreasing label values. This is the case when the next destination is node 41, existing dashed arrows between the following pairs of nodes: $(12, 11)$, $(11, 10)$, $(10, 9)$, $(28, 27)$, $(27, 26)$ and $(26, 25)$. In these cases, R always uses channels belonging to $C - C_1$. A similar situation appears when the next destination node has a lower label value than the source node. This is the case when the next destination is node 1. The LD algorithm [23] does not support this kind of routing, because it requires the absence of cyclic dependencies between channels.

Let us analyze the indirect dependencies which appear in the example shown in Fig. 1 by considering some paths supplied by R. For instance, if the header of the message destined for node 41 is at node 19, then it is routed to node 28 using a channel of C_1, then to node 27 using a channel of $C - C_1$ and then to node 36 using a channel of C_1, there is an indirect dependency between the channels of C_1. If node 28 were a destination node, the former example would correspond to an indirect multicast dependency.

Also, some indirect multicast dependencies appear between horizontal and vertical channels, due to messages which come back over their way after reaching an intermediate destination node. An example can be seen in Fig. 1, when the message header is at node 52, and for instance, it is routed to node 53 using a channel of C_1, then to nodes 52 and 51 using channels of $C - C_1$, and finally, to node 60 using a channel of C_1.

The subset of channels C_1 defines a connected routing subfunction R_1. It is easy to see that the pair (SS, R_1) has no cycles in its extended multicast channel dependency graph. Then, by theorem 2, the pair (SS, R) is deadlock-free.

As virtual channels share a single physical channel, the above adaptive routing function allows messages to cross the physical channels following almost any minimal path between successive destinations, increasing the number of alternative paths and decreasing network contention. The higher the distance between successive destinations, the higher the number of alternative paths offered by the adaptive routing function. Then, it is very well suited for large networks.

5 Conclusions

A theory for the design of deadlock-free adaptive multicast routing algorithms has been proposed for wormhole networks. Firstly, the new kinds of channel dependencies produced by multicast routing have been identified, defining two channel dependency graphs. Some new concepts, like the compatibility between the routing and split-and-sort functions, have been introduced. Then, an extension of Dally's theorem has been presented, allowing the design of adaptive multicast routing algorithms. However, the absence of cycles in the multicast channel dependency graph is too restrictive.

Theorem 2 gives a more flexible condition for the development of deadlock-free adaptive algorithms, by allowing the existence of cycles in the multicast channel dependency graph. The only requirement is the existence of a channel subset which defines a connected routing subfunction with no cycles in the extended multicast channel dependency graph.

Finally, an example showing the application of the former theory has been described. This example extends the multicast routing algorithm proposed in [22] by allowing the use of almost all the minimal paths between successive destination nodes.

References

[1] W.C. Athas and C.L. Seitz, "Multicomputers: Message-passing concurrent computers," *IEEE Comput. Mag.*, vol. 21, no. 8, pp. 9–24, Aug. 1988.

[2] P.E. Berman, L. Gravano, G.D. Pifarré and J.L.C. Sanz, "Adaptive deadlock- and livelock-free routing with all minimal paths in torus net-

works," in *Proc. 4th ACM Symp. Parallel Algorithms Architectures*, June 1992.

[3] S. Borkar et al., "iWarp: An integrated solution to high-speed parallel computing," in *Proc. Supercomputing'88*, Nov. 1988.

[4] G.T. Byrd, N.P. Saraiya and B.A. Delagi, "Multicast communication in multiprocessor systems," in *Proc. 1989 Int. Conf. Parallel Processing,* pp. I–196–I–200, Aug. 1989.

[5] A.A. Chien and J.H. Kim, "Planar-adaptive routing: Low-cost adaptive networks for multiprocessors," in *Proc. 19th Annu. Int. Symp. Comput. Architecture*, May 1992.

[6] R. Cypher and L. Gravano, "Requirements for deadlock-free adaptive packet routing," in *Proc. 11th ACM Symp. Principles Distributed Computing*, 1992.

[7] W.J. Dally and C.L. Seitz, "The torus routing chip," *Distributed Comput.*, vol. 1, no. 3, pp. 187–196, Oct. 1986.

[8] W.J. Dally and C.L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Comput.*, vol. C-36, no. 5, pp. 547–553, May 1987.

[9] W.J. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Trans. Parallel Distributed Syst.*, vol. 4, no. 4, pp. 466–475, April 1993.

[10] J. Duato, "On the design of deadlock-free adaptive routing algorithms for multicomputers: theoretical aspects," in *Proc. 2nd European Distributed Memory Computing Conf.*, April 1991.

[11] J. Duato, "On the design of deadlock-free adaptive routing algorithms for multicomputers: design methodologies," in *Proc. Parallel Architectures Languages Europe 91*, June 1991.

[12] J. Duato, "Deadlock-free adaptive routing algorithms for multicomputers: evaluation of a new algorithm," in *Proc. 3rd IEEE Int. Symp. Parallel Distributed Processing*, Dec. 1991.

[13] J. Duato, "Improving the efficiency of virtual channels with time-dependent selection functions," in *Proc. Parallel Architectures Languages Europe 92*, June 1992.

[14] J. Duato, "A new theory of deadlock-free adaptive routing in wormhole networks," *IEEE Trans. Parallel Distributed Syst.* To appear.

[15] J. Duato, "A new theory of deadlock-free adaptive multicast routing in wormhole networks," submitted to *IEEE Trans. Parallel Distributed Syst.*

[16] P.T. Gaughan and S. Yalamanchili, "Adaptive routing protocols for hypercube interconnection networks," *IEEE Comput. Mag.*, vol. 26, no. 5, pp. 12–23, May 1993.

[17] C.J. Glass and L.M. Ni, "The turn model for adaptive routing," in *Proc. 19th Annu. Int. Symp. Comput. Architecture*, May 1992.

[18] C.R. Jesshope, P.R. Miller, and J.T. Yantchev, "High performance communications in processor networks," in *Proc. 16th Annu. Int. Symp. Comput. Architecture*, May–June 1989.

[19] H.T. Kung, "Deadlock avoidance for systolic communication," in *Proc. 15th Annu. Int. Symp. Comput. Architecture*, May–June 1988.

[20] Y. Lan, A.H. Esfahanian and L.M. Ni, "Multicast in hypercube multiprocessors," *J. Parallel Distributed Comput.,* pp. 30–41, Jan. 1990.

[21] X. Lin and L.M. Ni, "Multicast communication in multicomputers networks," in *Proc. 1990 Int. Conf. Parallel Processing,* pp. III–114–III–118, Aug. 1990.

[22] X. Lin and L.M. Ni, "Deadlock-free multicast wormhole routing in multicomputer networks," in *Proc. 18th Annu. Int. Symp. Comput. Architecture*, May 1991.

[23] X. Lin, P.K. McKinley and A.H. Esfahanian, "Adaptive multicast wormhole routing in 2D mesh multicomputers," in *Proc. Parallel Architectures Languages Europe 93*, June 1993.

[24] D.H. Linder and J.C. Harden, "An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes," *IEEE Trans. Comput.*, vol. C-40, no. 1, pp. 2–12, Jan. 1991.

[25] Z. Liu and H. Wu, "Performance evaluation of adaptive wormhole routing in 3D mesh networks," in *Proc. 26th Annu. Simulation Symp.*, 1993.

[26] P. López and J. Duato, "Deadlock-free adaptive routing algorithms for the 3D-torus: limitations and solutions," in *Proc. Parallel Architectures Languages Europe 93*, June 1993.

[27] NCUBE Company, *NCUBE 6400 Processor Manual,* 1990.

[28] L.M. Ni, "Communication issues in multicomputers," in *Proc. First Workshop Parallel Processing,* Taiwan, pp. 52–64, Dec. 1990.

[29] G.D. Pifarré, L. Gravano, S.A. Felperin and J.L.C. Sanz, "Fully adaptive minimal deadlock-free packet routing in hypercubes, meshes, and other networks," in *Proc. 3rd ACM Symp. Parallel Algorithms Architectures*, 1991.