A Theory of Fault-Tolerant Routing in Wormhole Networks

José Duato, *Member*, *IEEE*

Abstract— Fault-tolerant systems aim at providing continuous operation in the presence of faults. Multicomputers rely on an interconnection network between processors to support the message-passing mechanism. Therefore, the reliability of the interconnection network is very important for the reliability of the whole system.

This paper analyzes the effective redundancy available in a wormhole network by combining connectivity and deadlock freedom. Redundancy is defined at the channel level. We propose a sufficient condition for channel redundancy, also computing the set of redundant channels. The redundancy level of the network is also defined, proposing a theorem that supplies its value. This theory is developed on top of our necessary and sufficient condition for deadlock-free adaptive routing. The new theory also considers the failure of physical channels when virtual channels are used. Finally, we propose a methodology for the design of fault-tolerant routing algorithms, showing its application to *n*-dimensional meshes.

Index Terms—Adaptive routing, channel redundancy, fault-tolerant routing, interconnection networks, network redundancy, wormhole switching.

1 Introduction

MULTICOMPUTERS have experienced a rapid development during the last decade [4]. As the number of elements in a multicomputer increases, the likelihood of one or more elements failing increases too. Consequently, system reliability becomes a key issue in the design and implementation of large scale multicomputers. Several issues have been addressed in the field of fault-tolerant design and reliability analysis of multicomputers and distributed systems [39].

Fault-tolerant systems aim at providing continuous operation in the presence of faults. Multicomputers rely on an interconnection network between processors to support the message-passing mechanism. Therefore, the reliability of the interconnection network is very important for the reliability of the whole system. In fact, the loss of communication between any two nodes might prevent a distributed recovery, because the state of the computation in the isolated nodes is unreachable. Network fault tolerance has been defined as the maximum number of elements that can fail without inducing a possible disconnection in the network [42].

Designs 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.

Most multicomputers and distributed shared-memory multiprocessors are based on direct networks. These networks are easy to map on two dimensions and allow the exploitation of the communication locality [1]. Therefore, we will restrict our attention to direct networks. For these networks, a static design only requires the definition of a fault-tolerant routing algorithm.

2 Background and Motivation

Several fault-tolerant routing algorithms have been proposed for direct networks [10], [11], [30], [45], taking 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. These algorithms assume that the network uses store-and-forward switching. In most cases, the hypercube has been the preferred topology, due to the high degree and the high number of alternative paths. Additionally, several fault-tolerant algorithms have been proposed for broadcasting and gossiping [44], [2], [31]. Again, these algorithms have been designed for store-and-forward networks with hypercube topology.

Current multicomputers and multiprocessors use wormhole switching [16]. Each message is serialized into a sequence of data units or flits [15]. The flit at the head of a message governs the route and the remaining flits follow it in a pipeline fashion. If all the channels requested by the header are busy, it is blocked until one of those channels is freed; the flow control within the network blocks the trailing flits.

During the last few years, many fault-tolerant routing algorithms have been proposed for wormhole networks. These algorithms are based on different strategies. Some algorithms exploit the high number of alternative paths available in the hypercube topology [40], [43], [18], [13]. Lee and Hayes [40] propose marking certain fault-free nodes as

• *The author is with the Facultad de Informática, Universidad Politécnica de Valencia, P.O.B. 22012, 46071, Valencia, Spain. E-mail: jduato@gap.upv.es.*

Manuscript received 14 Oct. 1994.
For information on obtaining reprints of this article, please send e-mail to: transpds@computer.org, and reference IEEECS Log Number 101000.0.

unsafe. A node is in the unsafe state if it has at least two faulty or unsafe nearest neighbors. By not forwarding messages to unsafe nodes, routing and broadcasting can be simplified and their delay reduced. This mechanism requires an algorithm to mark unsafe nodes. Dally and Aoki proposed two adaptive routing algorithms based on the concept of *dimension reversal* (DR), or number of times that a message has been routed from a channel in one dimension to a channel in a lower dimension [18]. In the *dynamic* algorithm, a message can follow any nonminimal path. Whenever a message acquires a channel, it labels the channel with its current DR number. However, a message that reaches a node where all output channels are occupied by messages with equal or lower DRs must be routed in dimension order until delivery. This algorithm has very good fault-tolerant properties except when messages must be routed in dimension order.

When wormhole switching is used, low-dimensional meshes and k-ary n-cubes achieve a higher performance than hypercubes [1]. So, most recent proposals have been developed for those topologies. Linder and Harden [41] have proposed an adaptive and fault-tolerant routing algorithm for wormhole networks with a k-ary n-cube topology. It is based on the concept of *virtual network*. Virtual networks are mapped onto the physical network by splitting physical channels into virtual channels. The main drawback of this algorithm is that it requires up to 2^n virtual channels per physical channel, and additional virtual channels are required to support faulty channels. Glass and Ni have proposed a fault-tolerant routing algorithm for n-dimensional meshes that supports $n - 1$ dynamic faults. It is based on the turn model and does not require virtual channels [36]. However, it requires a synchronization between nonneighboring nodes and it is only partially adaptive. Additionally, it has poor performance in the absence of faults [6]. *Origin-based* routing is another partially adaptive fault-tolerant routing algorithm for meshes that does not require virtual channels [37]. Messages are routed in two phases. In the first phase, messages are routed adaptively towards the origin. In the second phase, messages are routed adaptively towards their destination. Depending upon its location, the placement of the origin can become a communication hot-spot.

An interesting set of proposals for meshes and k-ary n-cubes is based on the concept of *fault regions* [12], [7], [9], [8]. These regions have predefined shapes, usually rectangular or convex. When fault patterns have different shapes, an algorithm is required to mark fault-free nodes as faulty. Chien and Kim [12] propose a nonminimal partially adaptive routing algorithm for meshes, which can route around fault regions that do not include boundary nodes. However, this algorithm cannot efficiently handle faults on the boundaries of a mesh. Boppana and Chalasani [8] propose the use of *fault rings* and *fault chains*, which are sets of fault-free components around fault regions. By using additional virtual channels, messages can be routed along fault rings and fault chains. Several researchers are currently working on extending fault rings and fault chains to support nonrectangular fault regions.

An alternative way to guarantee that messages reach their destination in the presence of faults consists of aborting messages blocked by faults, sending them again across different paths [38].

Some of the limitations of wormhole switching to support faults have been highlighted by Gaughan and Yalamanchili, proposing a different approach to fault-tolerant routing [32], [34]. Instead of using wormhole, they propose the use of pipelined circuit switching (PCS) and header backtracking. The misroute backtracking protocol proposed by them supports a high number of static faults. However, PCS achieves lower performance than wormhole switching due to the overhead of path setup, especially for short messages. In order to overcome this limitation, a new hybrid switching technique, known as *scouting*, has been developed [27], [20]. In scouting, data flits follow the message header at a certain distance, therefore allowing limited backtracking. That distance can be dynamically adjusted, behaving like wormhole switching in the absence of faults and supporting misrouting and backtracking in the presence of faults.

PCS and scouting are very resilient to faults. However, they require more hardware support than wormhole switching. These techniques are mainly intended for systems where reliability is the primary design goal. On the other extreme of the design space, software-based fault tolerant routing allows routers to support faults without degrading the performance in the absence of faults [46]. These techniques are very simple, do not require complex hardware support, and can be implemented on top of oblivious routers as well as adaptive routers. The main drawback is that performance may degrade considerably when faults occur. These techniques are intended to provide some level of fault-tolerance at a minimum cost, and are suitable to commercial machines.

Most fault-tolerant routing algorithms focused on supporting static faults, i.e., faults that exist when the computer is powered on. However, in practice, faults may occur at any time (dynamic faults). In particular, a channel or a node may fail while a message is crossing it. In this case, the fragment of the message that has not crossed the faulty component yet remains blocked forever. In [33], [38], some hardware mechanisms are proposed to remove the fragments of an interrupted message pipeline, and to notify the source node so that the message is transmitted again. The *Reliable Router* [19] supports dynamic faults by using link-level retransmission in combination with a unique-token protocol. This mechanism does not require returning acknowledgments and keeping copies for possible retransmissions at the message source. Finally, it should be noted that fault-tolerant routing algorithms that rely on an algorithm to form fault regions may fail in the presence of dynamic faults. Effectively, after a node failure, a message may reach a node that will be included in a fault region before it has been included.

Finally, several fault-tolerant routers have been designed for a variety of switching techniques, including wormhole switching [19], pipelined circuit switching [3], and virtual cut-through switching [5]. In summary, there is a 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 proposals lie in the middle of the spectrum, aiming at increasing fault-tolerance in networks using wormhole switching by adding some hardware support.

All of the previously mentioned routing algorithms for wormhole switching avoid deadlocks by preventing cyclic dependencies between channels, according to [16]. The only exceptions are the dynamic algorithm proposed in [18] and the abort-and-retry technique proposed in [38]. A higher routing flexibility can be obtained by allowing cyclic dependencies between channels [24], [28]. This additional flexibility can be used to route messages around faults, therefore avoiding the need for marking fault-free nodes as faulty, using algorithms to form fault regions or synchronizing nonneighboring nodes. Moreover, previous proposals focus on proposing fault-tolerant routing algorithms, without analyzing the redundancy available in the network.

This paper analyzes the effective redundancy available in a wormhole network by combining connectivity and deadlock freedom. This theory mainly considers channel faults. However, the results are extended by considering node faults. First, we state the problem in Section 3. Section 4 gives an informal description of the results proposed in this paper. Then, Section 5 summarizes some theoretical background that is required for the remaining sections. Section 6 analyzes the redundancy at the channel level. Section 7 deals with redundancy at the network level, defining it and computing its value. These results are extended by considering virtual channels in Section 8 and node faults in Section 9. Section 10 proposes a design methodology for fault-tolerant routing algorithms. Finally, Section 11 shows the application of the theoretical results, proposing an adaptive fault-tolerant routing algorithm. This theory is developed on top of our necessary and sufficient condition for deadlock-free adaptive routing [24], [28]. The new theory improves the results previously proposed by us in [22], [25]. Those results only supplied a lower bound for the redundancy level of the network. Additionally, we explicitly consider virtual channels in this paper, analyzing the effect of physical channel faults.

3 Statement of the Problem

In this paper, we answer a fundamental question regarding fault-tolerant routing: What is the maximum number of simultaneous faults tolerated by the routing function? If a routing function tolerates k faulty channels, it must remain connected and deadlock-free for any number of faulty channels less than or equal to k. Otherwise, the network can reach a deadlocked configuration when some channels fail.

In order to analyze network fault tolerance, connectivity cannot be defined without considering the routing algorithm. When some channels fail, a node may become isolated, even when there are some channels connecting it to other nodes, provided that those channels cannot be used by the routing algorithm. For example, consider a 2D mesh using XY routing. If an X channel fails, all the messages that request it will be blocked. However, there are some alternative healthy paths around the faulty channel. Those paths cannot be used because the routing algorithm does not allow misrouting. If the routing algorithm allowed misrouting, those paths could be used and messages would not block. Thus, connectivity must be defined as a property of the routing function [21], [23].

It may be thought that fault-tolerant routing algorithms must be designed in such a way that messages can use all the available channels in the network. The main limitation is that messages may deadlock. As stated above, if the routing function is supposed to tolerate k faulty channels, it must remain connected and deadlock-free for any number of faulty channels less than or equal to k.

4 Informal Description

There are some theoretical results that guarantee the absence of deadlock in the whole network by analyzing the behavior of the routing function R restricted to a subset of channels C_1 [21], [23]. Additionally, that behavior is not modified when some channels not belonging to C_1 are removed. Therefore, that theory can be used to guarantee the absence of deadlock when some channels fail. A more powerful theory has been proposed in [24], [28]. This theory supplies a necessary and sufficient condition for deadlock-free adaptive routing. It also guarantees the absence of deadlock by analyzing the behavior of a restricted routing function R_1. If R_1 is connected and free of cyclic dependencies between channels, then the routing function R is deadlock-free.

The above mentioned theory can be used to analyze channel and network redundancy. The basic idea is the following: Consider that there exists a restricted routing function R_1 that is connected and has no cyclic dependencies between channels. If all the channels that are not used by R_1 fail, R is still connected and deadlock-free. Therefore, all the channels that are not used by R_1 are *redundant*. However, if a channel supplied by R_1 for some destinations fails, we cannot guarantee that this fault will be tolerated.

In general, for a given routing function R, there exist several restricted routing functions $R_1, R_2, \ldots, R_k$ satisfying the above mentioned conditions. Each restricted routing function R_i only uses a subset of channels C_i. Let us consider the intersection of all the channel subsets C_i, $i = 1, 2, \ldots, k$. If the intersection set is not empty, removing a channel from it may produce deadlock. However, if the intersection set is empty, then we can remove any channel without disconnecting R and guaranteeing that there is no deadlock. Now consider the worst case: How many simultaneous faults are allowed? Provided that there are circuits to detect faults, an equivalent question is: How many channels can be simultaneously removed? The answer is easy: As long as at least one of the channel subsets C_i keeps all of its elements, we will be able to guarantee that the routing function R is connected and deadlock-free. The maximum number of simultaneous faults that are allowed in the worst case will be defined as the *redundancy level* of the network. The next sections formalize these ideas.

5 Deadlock-Free Adaptive Routing

The theory proposed in this paper is developed on top of the theory of deadlock-free adaptive routing proposed in

[24], [28]. That theory is described here in an informal way to make the paper self-contained. Please refer to [28] for a formal version as well as for examples of application and examples for the different kinds of channel dependency.

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. *An* adaptive routing function $R : N \times N \rightarrow \mathcal{P}(C)$, *where $\mathcal{P}(C)$ is the power set of C, supplies a set of alternative output channels to send a message from the current node n_c to the destination node n_d, $R(n_c, n_d) = \{c_1, c_2, \ldots, c_p\}$. A* selection function *selects a free output channel (if any) from the set supplied by the routing function.*

DEFINITION 3. *A routing function R for a given interconnection network I is* connected *iff, for any pair of nodes $x, y \in N$, it is possible to establish a* path $P(x, y) \in \mathcal{P}(C)$ *between them using channels belonging to the sets supplied by R.*

DEFINITION 4. *A routing function R for a given interconnection network I is* coherent *iff for every path P that can be established by R, every prefix of P is also a path of R. In other words, if a routing function R can establish a path $P(x, y)$ between x and y, it can also establish a path between x and any intermediate node crossed by $P(x, y)$ using a subset of the channels used by $P(x, y)$.*

DEFINITION 5. *A* routing subfunction R_1 *for a given routing function R is a restriction of R. For each destination, R_1 supplies a subset of the channels supplied by R. The set of all the channels supplied by R_1 is $C_1 = \bigcup_{\forall x,y \in N} R_1(x, y)$. As a particular case, the routing subfunction can be defined by giving the subset of channels C_1 used by it. In this case, $R_1(x, y) = R(x, y) \cap C_1 \; \forall x, y \in N$.*

DEFINITION 6. *Given an interconnection network I, a routing function R and a pair of adjacent channels $c_i, c_j \in C$, there is a* direct dependency *from c_i to c_j iff c_j can be requested immediately after using c_i by messages destined for some node x. Adjacency means that $d_i = s_j$.*

DEFINITION 7. *Given an interconnection network I, a routing function R, a routing subfunction R_1 and a pair of nonadjacent 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 for some node x, and c_i and c_j are the first and last channels in that path and the only ones in that path supplied by R_1 for the destination of the message. Therefore, c_j can be requested after using c_i by some messages. As c_i and c_j are not adjacent, some other channels not supplied by R_1 for the destination of the message are reserved while establishing the path between them. Those channels are supplied by R.*

DEFINITION 8. *Given an interconnection network I, a routing function R, a routing subfunction R_1, and a pair of adjacent channels $c_i, c_j \in C_1$, there is a* direct cross dependency *from c_i to c_j iff c_j can be requested immediately after using c_i by messages destined for some node x, c_j is supplied by R_1 for the destination of the message, and c_i cannot be supplied by R_1 for that destination. However, c_i is supplied by R_1 for some other destination(s).*

DEFINITION 9. *Given an interconnection network I, a routing function R, a routing subfunction R_1, and a pair of nonadjacent channels $c_i, c_j \in C_1$, there is an* indirect cross dependency *from c_i to c_j iff it is possible to establish a path from s_i to d_j for messages destined for some node x, c_i and c_j are the first and last channels in that path, c_j is the only channel in that path supplied by R_1 for the destination of the message, and c_i cannot be supplied by R_1 for that destination. However, c_i is supplied by R_1 for some other destination(s). Therefore, c_j can be requested after using c_i by some messages. As c_i and c_j are not adjacent, some other channels not supplied by R_1 for the destination of the message are reserved while establishing the path between them.*

DEFINITION 10. *A* channel dependency graph *D for a given interconnection network I and routing function R, is a directed graph, $D = G(C, E)$. The vertices of D are the channels of I. The arcs of D are the pairs of channels (c_i, c_j) such that there is a direct dependency from c_i to c_j.*

DEFINITION 11. *An* extended channel dependency graph D_E *for a given interconnection network I and routing subfunction R_1 of a routing function R, is a directed graph, $D_E = G(C_1, E_E)$. The vertices of D_E are the channels that can be supplied by the routing subfunction R_1 for some destinations. The arcs of D_E are the pairs of channels (c_i, c_j) such that there is either a direct, indirect, direct cross, or indirect cross dependency from c_i to c_j. It should be noted that the extended channel dependency graph has been redefined with respect to* [21], [23].

The following theorem proposes a necessary and sufficient condition for an adaptive routing function to be deadlock-free. It is an extension of the second theorem proposed in [21], [23].

THEOREM 1. *A coherent, connected, and adaptive routing function R for an interconnection network I is deadlock-free iff there exists a routing subfunction R_1 that is connected and has no cycles in its extended channel dependency graph D_E.*

6 CHANNEL REDUNDANCY

In this section, we analyze the redundancy at the channel level. We define redundant channels, also computing the set of redundant channels.

6.1 Assumptions

The assumptions given in [23] for adaptive routing are also valid here. These assumptions consider several aspects related to the use of wormhole switching. Some additional assumptions are required for fault-tolerant routing. These assumptions prevent messages from being routed across faulty channels. Also, they eliminate deadlocks induced by dynamic faults. The interested reader can refer to [33], [38] for hardware implementations of protocols supporting dynamic faults. This section only considers channel faults. The assumptions are the following:

1) Nodes are reliable. Only channel faults are considered. These faults can always be detected.
2) When a channel fails, it is marked as faulty at its source node. If there is a message in transit, the flits that have not crossed the channel yet are discarded to eliminate deadlocks induced by dynamic faults (faults that may appear at any time).
3) The routing function is not modified when one or more channels fail.
4) The selection function will not select any faulty channel. When all the channels supplied by the routing function are faulty, the message is discarded and removed from the network.

It should be noted that, when some flits or a whole message are discarded, the network relies on a hardware protocol to ensure that discarded messages will be transmitted again, and incomplete messages reaching their destination nodes will be eliminated [33], [38].

6.2 Definitions and Theorems

Now, we can give the following definition:

DEFINITION 12. *Given an interconnection network I and a routing function R, a channel $c_i \in C$ is* redundant *iff, after removing that channel, the resulting routing subfunction R_* is connected and deadlock-free.*

Before proposing the theorem, we need two simple lemmas:

LEMMA 1. *A routing function R for a given interconnection network I is connected if there exists a connected routing subfunction R_1.*

The proof is trivial.

LEMMA 2. *Given an interconnection network I, a routing function R, and a routing subfunction R_1, removing channels that are not supplied by R_1 for any destination does not add any dependency to the extended channel dependency graph of R_1.*

Taking into account the definitions of indirect dependency, direct cross dependency, and indirect cross dependency, the proof is also trivial. Moreover, removing channels that are not supplied by R_1 for any destination may remove indirect dependencies and indirect cross dependencies. Let C_1 be the set of all the channels supplied by R_1 for some destinations. When all the channels belonging to $C - C_1$ are removed, there is not any indirect dependency nor indirect cross dependency between the channels supplied by R_1.

Now we can propose the following theorem:

THEOREM 2. *Given an interconnection network I, a routing function R, and a channel subset $C_1 \subset C$, all the channels belonging to $C - C_1$ are redundant if there exists a routing subfunction R_1 that only supplies channels belonging to C_1 and R_1 is connected and it has no cycles in its extended channel dependency graph D_E.*

PROOF. $\Leftarrow$ Suppose that there exists a routing subfunction R_1 that only supplies channels belonging to C_1 and that R_1 is connected and it has no cycles in its extended channel dependency graph D_E. Let c_i be a channel belonging to $C - C_1$. Let us prove that it is redundant. Suppose that c_i is removed. Let

$$C_* = C - \{c_i\}$$

and

$$R_*(x, y) = R(x, y) \cap C_* \; \forall x, y \in N$$

As $c_i \in C - C_1$, then $C_1 \subseteq C_*$. Taking into account Lemma 2, removing c_i does not add any dependency between the channels supplied by R_1. As the extended channel dependency graph D_E is acyclic, the new extended channel dependency graph D_{E*}, defined with respect to R_*, is also acyclic. Thus, the routing subfunction R_1 is connected and it has no cycles in its extended channel dependency graph D_{E*}. Applying Theorem 1, R_* is deadlock-free. Also, as R_1 is connected, by Lemma 1, R_* is connected. So, c_i is redundant by Definition 12. □

It should be noted that Theorem 2 does not give a necessary condition. It should also be noted that, even if all the channels belonging to a given subset are redundant, it does not mean that all of them can be removed simultaneously without affecting connectivity or deadlock-freedom. It means that those channels can be removed one at a time. The question about how many simultaneous faulty channels are allowed will be addressed in Section 7.

The conditions imposed by Theorem 2 to guarantee that the channels belonging to $C - C_1$ are redundant, are the same as the ones imposed by Theorem 1 to guarantee that the routing function R is deadlock-free.

A single routing subfunction R_1 suffices for the application of Theorem 1. However, for a given routing function R, several routing subfunctions may exist, all of them being connected and having no cycles in their extended channel dependency graphs. Let $C_1, C_2, \ldots, C_k$ be the subsets containing the channels supplied by those routing subfunctions and let

$$C_s = (C - C_1) \cup (C - C_2) \cup \ldots \cup (C - C_k)$$
$$= C - (C_1 \cap C_2 \cap \ldots \cap C_k)$$

Thus, C_s is the set of redundant channels. In the case

$$C_1 \cap C_2 \cap \ldots \cap C_k = \varnothing$$

all the channels are redundant. The second design methodology proposed in [23] supplies a routing function in which all the channels are redundant, because it defines two disjoint channel subsets C_1 and C_2.

7 NETWORK REDUNDANCY

After dealing with channel redundancy, the next question is: How many simultaneous faulty channels are allowed in a given network? We already know that, after a channel fault, the routing function will perform correctly if the faulty channel was redundant. So, we can give the following definitions:

DEFINITION 13. *A given routing function R for an interconnection network I has a* redundancy level *equal to r iff*

$$\forall c_1, c_2, \ldots, c_r \in C, C_* = C - \{c_1, c_2, \ldots, c_r\}$$

defines a routing subfunction R_* that is connected and deadlock-free and

$\exists c_1, c_2, \ldots, c_{r+1} \in C$ such that $C_{**} = C - \{c_1, c_2, \ldots, c_{r+1}\}$

defines a routing subfunction R_{**} that is not connected or it is not deadlock-free. In other words, the routing function tolerates up to r simultaneous faulty channels in the worst case.

DEFINITION 14. *A given routing function R for an interconnection network I is* minimally connected *iff it is connected and, after removing any channel supplied by R for some destinations, R is no longer connected.*

As Theorem 1 gives a necessary and sufficient condition to prove that a given routing function is deadlock-free, Definition 13 will be useful to build a theory on that theorem. Now we can propose the following theorem:

THEOREM 3. *A coherent routing function R for an interconnection network I has a redundancy level equal to r iff there exist $k > r$ minimally connected routing subfunctions $R_1, R_2, \ldots, R_k$, and these are the only such routing subfunctions that have no cycles in their corresponding extended channel dependency graphs, and there exists a subset of $r + 1$ channels $C_m \subset C$ such that it is the smallest subset of C satisfying that*

$$\forall i \in \{1, 2, \ldots, k\}, C_m \cap C_i \neq \emptyset$$

where $C_i = \bigcup_{\forall x,y \in N} R_i(x, y)$ is the set of channels supplied by the routing subfunction R_i.

PROOF SKETCH. $\Leftarrow$ Suppose that there exist only k minimally connected routing subfunctions $R_1, R_2, \ldots, R_k$ that have no cycles in their corresponding extended channel dependency graphs. Let $C_1, C_2, \ldots, C_k$ be the sets of channels supplied by those routing subfunctions, as shown in Fig. 1. Also, suppose that there exists a subset of $r + 1$ channels $C_m = \{c_1, c_2, \ldots, c_{r+1}\}$ such that it is the smallest subset containing at least one channel in C_i, $\forall i \in \{1, 2, \ldots, k\}$.

In the example shown in Fig. 1, removing c_1 will prevent us from using R_1, R_2, and R_3 to prove that R is deadlock-free because those routing subfunctions are minimally connected. Similarly, removing c_2 will prevent us from using R_4, R_5, and R_6 to prove that R is deadlock-free. If all the channels in C_m are removed, then it is not possible to use any routing subfunction to prove that R is deadlock-free. Hence, in the worst case, R does not tolerate $r + 1$ faults. However, if only r channels are removed, at least one set C_i, $i \in \{1, 2, \ldots, k\}$ will keep all of its elements, allowing us to prove that R is deadlock-free. It should be noted that C_m is the smallest set containing at least one channel in C_i, $\forall i \in \{1, 2, \ldots, k\}$. Therefore, R tolerates up to r faults, and its redundancy level is equal to r.

$\Rightarrow$ Let R be a coherent routing function with a redundancy level equal to r. So, R is connected and deadlock-free. By Theorem 1, there exists at least one routing subfunction that is connected and has no cycles in its extended channel dependency graph. Let $R_1, R_2, \ldots, R_k$ be all the minimally connected routing subfunctions satisfying those conditions. Let $C_1, C_2, \ldots, C_k$ be the sets of channels supplied by those routing subfunctions, as shown in Fig. 1. Let C_m be the smallest set containing at least one channel in C_i, $\forall i \in \{1, 2, \ldots, k\}$.

Note that $|C_m| \leq k$. Obviously, $k > r$. Now, we have to prove that $|C_m| = r + 1$. As indicated above, after removing all the channels in C_m, it is no longer possible to prove that R is deadlock-free. If $|C_m| \leq r$, removing r channels may produce deadlock, contrary to the assumption that R has a redundancy level equal to r. If $|C_m| > r + 1$, after removing any set of $r + 1$ channels, it will still be possible to find at least one set C_i, $i \in \{1, 2, \ldots, k\}$ that keeps all of its elements, allowing us to prove that R is deadlock-free. Therefore, the redundancy level is greater than r, contrary to the assumption that R has a redundancy level equal to r. □

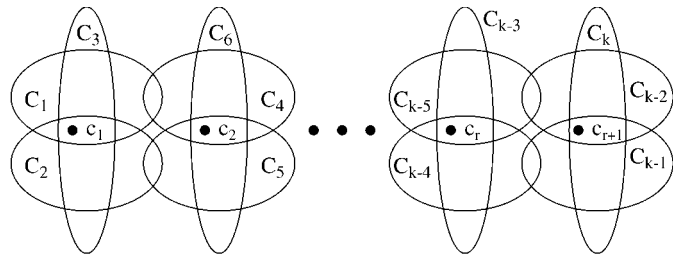

Fig. 1. Channel sets supplied by all the minimally connected routing subfunctions of R and the corresponding elements in C_m.

PROOF. $\Leftarrow$ Suppose that there exist k minimally connected routing subfunctions $R_1, R_2, \ldots, R_k$, and that these are the only such routing subfunctions that have no cycles in their corresponding extended channel dependency graphs. Let $C_1, C_2, \ldots, C_k$ be the sets of channels supplied by those routing subfunctions. Also, suppose that there exists a subset of $r + 1$ channels, $C_m \subset C$, such that it is the smallest subset of C satisfying that $\forall i \in \{1, 2, \ldots, k\}$, $C_m \cap C_i \neq \emptyset$.

Let $c_1, c_2, \ldots, c_r \in C$ be any set of r channels. Let us prove that $C_* = C - \{c_1, c_2, \ldots, c_r\}$ defines a routing subfunction R_* that is connected and deadlock-free.

As we have only removed r channels from C, $C - C_*$ cannot satisfy the condition given for C_m. Therefore:

$\exists i \in \{1, 2, \ldots, k\}$ such that $C_i \cap (C - C_*) = \emptyset \Rightarrow C_i \subseteq C_*$

Taking into account Lemma 2, removing $c_1, c_2, \ldots, c_r$ from C does not add any dependency between the channels belonging to C_i. As the extended channel dependency graph for R_i is acyclic, the new extended channel dependency graph for R_i, defined with respect to R_*, is also acyclic. Thus, there exists a routing subfunction R_i that is connected and has no cycles in its extended channel dependency graph. Applying Theorem 1, R_* is deadlock-free. Also, as R_i is connected, by Lemma 1, R_* is connected. Thus, R has a redundancy level greater than or equal to r.

Now, let us prove that the redundancy level is not greater than r. Suppose that we remove all the channels belonging to C_m. Let $C_{**} = C - C_m$ and let $R_{**}(x, y) = R(x, y) \cap C_{**}$ $\forall x, y \in N$. We are going to prove that R_{**} is not deadlock-free by contradiction. Suppose that there exists a routing subfunction R'_{**} of R_{**} that is connected and has no cycles in its extended channel

dependency graph. If R'_{**} is not minimally connected, then it will be possible to restrict it, generating another routing subfunction R''_{**} of R_{**} that is minimally connected. Note that R''_{**} is also a routing subfunction of R'_{**}. Hence, by Lemma 2, R''_{**} has no cycles in its extended channel dependency graph. Taking into account that

$$\forall i \in \{1, 2, \ldots, k\}, C_m \cap C_i \neq \emptyset$$

R''_{**} differs from R_j, $\forall j \in \{1, 2, \ldots, k\}$. Therefore, there exist more than k minimally connected routing subfunctions that have no cycles in their extended channel dependency graph, contrary to the statement of the theorem. So, there is not any routing subfunction of R_{**} that is connected and has no cycles in its extended channel dependency graph. Thus, taking into account Theorem 1, R_{**} is not deadlock-free. As C_m has $r + 1$ elements, the redundancy level of R is not greater than r.

$\Rightarrow$ Let R be a coherent routing function with a redundancy level equal to r. So, R is connected and deadlock-free. By Theorem 1, there exists at least one routing subfunction that is connected and has no cycles in its extended channel dependency graph. Let $R_1, R_2, \ldots, R_k$ be all the minimally connected routing subfunctions satisfying those conditions. Let $C_1, C_2, \ldots, C_k$ be the sets of channels supplied by those routing subfunctions. Let $C_m \subset C$ be the smallest subset of C satisfying that

$$\forall i \in \{1, 2, \ldots, k\}, C_m \cap C_i \neq \emptyset$$

Note that $|C_m| \leq k$. Now, we have to prove that $k > r$ and $|C_m| = r + 1$. Suppose that either $k \leq r$ or $|C_m| \leq r$. In both cases, $|C_m| \leq r$. If we remove all the channels belonging to C_m, then there is not any connected routing subfunction without cycles in its extended channel dependency graph. As a consequence, R is no longer deadlock-free and it does not tolerate r simultaneous faulty channels, contrary to the assumption that R has a redundancy level equal to r. So, $k > r$ and $|C_m| > r$.

Suppose that $|C_m| > r + 1$. In this case, there exist $k > r$ minimally connected routing subfunctions $R_1, R_2, \ldots, R_k$ that have no cycles in their corresponding extended channel dependency graphs, and there exists a subset with more than $r + 1$ channels, $C_m \subset C$, such that it is the smallest subset of C satisfying that

$$\forall i \in \{1, 2, \ldots, k\}, C_m \cap C_i \neq \emptyset$$

According to the proof for the sufficient condition of this theorem, R has a redundancy level greater than r, contrary to the initial assumption. Hence, $|C_m| = r + 1$ and the theorem holds. □

It should be noted that coherency is not required to prove the sufficient condition for deadlock-freedom [28]. Thus, for routing functions that are not coherent, Theorem 3 only supplies a lower bound for the redundancy level. Also, the theory proposed in [24], [28] can be easily extended for routing functions defined as $R : C \times N \rightarrow \mathcal{P}(C)$, thus considering the input channel to the current node instead of just the current node. However, from a theoretical point of view, the resulting theorems would only supply sufficient conditions. The reason being that a deadlocked configuration is not necessarily routable [14]. So, it may happen that some deadlocked configurations exist for a deadlock-free routing function, provided that those configurations cannot be reached starting from an empty network. As a consequence, the results proposed in this paper are also applicable to routing functions defined as $R : C \times N \rightarrow \mathcal{P}(C)$, but Theorem 3 will only supply a lower bound for the redundancy level.

Finally, it should be noted that the necessary and sufficient condition for deadlock-free routing in store-and-forward and virtual cut-through switching [29] is almost identical to the one for wormhole switching. The only differences are that coherency is not required, and the extended channel dependency graph only contains direct and direct cross dependencies. Therefore, the theoretical results presented in this paper are also valid for those switching techniques.

8 Networks with Virtual Channels

Several deterministic routing algorithms require virtual channels to prevent deadlock [16]. Virtual channels also allow several messages to share the bandwidth of physical channels, usually increasing performance [17]. Moreover, very efficient fully adaptive routing algorithms can be designed by using virtual channels [23]. Virtual channels are used in parallel computers like Cray T3D and T3E.

However, the theory proposed in previous sections is only applicable to physical channels if they are not split into virtual channels. If virtual channels are used, the theorems proposed in previous sections are only valid for virtual channels. Although a failure may affect a single virtual channel, the failure of physical channels is more frequent. In this case, all the virtual channels belonging to that physical channel will become faulty at the same time. In this section, we consider the failure of physical channels that are split into virtual channels. The definitions and theorems proposed in previous sections must be modified to consider physical channels instead of virtual channels. We assume that the number of virtual channels per physical channel is the same for all the physical channels.

Let C_p be the set of physical channels of the interconnection network I. Let v be the number of virtual channels per physical channel. Let C be the set of virtual channels of I. Each physical channel, $p_i \in C_p$, is split into v virtual channels $c_{i,1}, c_{i,2}, \ldots, c_{i,v} \in C$. If $C_{p1} \subset C_p$ is a subset of physical channels, the set containing the corresponding virtual channels will be denoted as C_1. In other words, if $C_{p1} = \{p_1, p_2, \ldots, p_k\}$ then $C_1 = \{c_{1,1}, \ldots, c_{1,v}, c_{2,1}, \ldots, c_{2,v}, \ldots, c_{k,1}, \ldots, c_{k,v}\}$.

First, we can update some definitions:

Definition 15. *Given an interconnection network I and a routing function R, a physical channel $p_i \in C_p$ is* redundant *iff, after removing the channels $c_{i,1}, c_{i,2}, \ldots, c_{i,v}$, the resulting routing subfunction R_* is connected and deadlock-free.*

It should be noted that a physical channel is not necessarily redundant if all of its constituent virtual channels are redundant. The reason is that a virtual channel may rely on another virtual channel from the same physical channel in order to be redundant.

DEFINITION 16. *A given routing function R for an interconnection network I has a* redundancy level *equal to r iff*

$$\forall p_1, p_2, \ldots, p_r \in C_p,$$

$$C_* = C - \{c_{1,1}, \ldots, c_{1,v}, c_{2,1}, \ldots, c_{2,v}, \ldots, c_{r,1}, \ldots, c_{r,v}\}$$

defines a routing subfunction R_* that is connected and deadlock-free and

$$\exists p_1, p_2, \ldots, p_{r+1} \in C_p \text{ such that}$$

$$C_{**} = C - \{c_{1,1}, \ldots, c_{1,v}, c_{2,1}, \ldots, c_{2,v}, \ldots, c_{r+1,1}, \ldots, c_{r+1,v}\}$$

defines a routing subfunction R_{**} that is not connected or it is not deadlock-free.

Now, we can propose the following theorems:

THEOREM 4. *Given an interconnection network I, a routing function R, and a channel subset $C_{p1} \subset C_p$, all the channels belonging to $C_p - C_{p1}$ are redundant if there exists a routing subfunction R_1 that only supplies channels belonging to C_1 and R_1 is connected and it has no cycles in its extended channel dependency graph D_E.*

THEOREM 5. *A coherent routing function R for an interconnection network I has a redundancy level equal to r iff there exist $k > r$ minimally connected routing subfunctions, $R_1, R_2, \ldots, R_k$, and these are the only such routing subfunctions that have no cycles in their corresponding extended channel dependency graphs, and there exists a subset of $r + 1$ physical channels, $C_{pm} \subset C_p$, such that it is the smallest subset of C_p satisfying that*

$$\forall i \in \{1, 2, \ldots, k\}, C_m \cap C_i \neq \emptyset$$

where $C_i = \bigcup_{\forall x,y \in N} R_i(x, y)$ is the set of channels supplied by the routing subfunction R_i.

The proof for these theorems is similar to the proof for Theorems 2 and 3. For routing functions that are not coherent, as well as for routing functions defined as $R : C \times N \rightarrow \mathcal{P}(C)$, Theorem 5 only supplies a lower bound for the redundancy level.

9 NODE FAULTS

In previous sections, we only considered channel faults. In this section, we are going to extend those results, so that they can be applied when one or more nodes fail. The assumptions given in Section 6.1 are also valid here, except that nodes may also fail. Some additional assumptions are required:

1) Nodes and / or channels may fail. Faults can always be detected.
2) When a node fails, all its input channels are marked as faulty at their source nodes. Thus, taking into account assumption 4 for channel faults, messages destined for faulty nodes are always discarded if the routing function is livelock-free.

When a node $n_i \in N$ fails, the new set of processing nodes is $N_* = N - \{n_i\}$. Messages destined for n_i will be discarded and cannot produce deadlock. Thus, the definitions given in Section 5 can be rewritten by using N_* instead of N. In particular, a routing function is now connected when it is able to establish a path between any pair of nodes, excluding the faulty one. Taking into account the new definitions, all the results obtained in Sections 6 and 7 are applicable when one or more nodes fail.

10 DESIGN METHODOLOGY

The application of the theory proposed in previous sections is not straightforward. Analyzing the redundancy level of a fault-tolerant routing algorithm requires finding all the routing subfunctions that satisfy some properties. This problem is conjectured to be NP-complete. Additionally, the theory proposed in previous sections is not easy to apply to the design of fault-tolerant routing algorithms. So, some heuristics are required for the analysis and design of fault-tolerant algorithms. These heuristics should exploit the regularity of the network topology.

In [23], we proposed a methodology for the design of fully adaptive routing algorithms. That methodology starts from a deterministic or partially adaptive routing algorithm, adding channels in a regular way, and extending the routing function so that it uses the new channels. Similar to that theory, here we propose a methodology for the design of fault-tolerant routing algorithms. The steps are the following:

1) Given an interconnection network, I, define a connected nonminimal fully-adaptive routing function R for it. This routing function should be deadlock-free.
2) Find a connected routing subfunction R_1 of R such that its extended channel dependency graph is acyclic.
3) Set the desired value for the redundancy level of the network. This value must be lower than the minimum node degree. Let r be the desired redundancy level.
4) Consider a node x in the network. For each output channel c_i of this node in the extended channel dependency graph of R_1, remove c_i and any combination of $r - 1$ output channels from x. If the routing subfunction R_1 is no longer connected after removing c_i, find an output channel c_j of node x such that, when added to R_1, it connects it again, and the corresponding extended channel dependency graph is acyclic. Channel c_j cannot be one of the channels that were removed. This step should be repeated for all the combinations of $r - 1$ output channels from x. The symmetry of the routing function can be taken into account to reduce the number of combinations under study.
5) If the routing function does not behave in a regular way, repeat the previous step for a selection of nodes that represent all the alternative behaviors of the routing function. If the previous step was not successful, try to extend R by adding a layer of virtual channels that route messages through alternative physical paths, and go to step 1.

This methodology is not as simple and accurate as the one proposed in [23]. It is only intended to serve as a guideline in the development of fault-tolerant routing algorithms. The first two steps can be performed in reverse order, similar to the methodology proposed in [23]. In other words, we can start by defining a connected routing function that has no cycles in its channel dependency graph, then adding more channels to make it fully-adaptive and nonminimal. Step 4 tries to guarantee the redundancy level at a single node in the network by making sure that any combination of r faulty output channels at that node does not prevent us from finding a suitable routing subfunction that is connected and has no cycles in its extended channel dependency graph. Note that the worst case usually happens when all the faulty channels are adjacent to the same node. Finally, step 5 takes advantage of the regularity of the network and the routing function, therefore reducing the analysis of the whole network to those cases for which the routing function behaves in a different way.

11 Design Example

In this section, we present a fault-tolerant routing algorithm for n-dimensional meshes. We will build it step by step to illustrate the application of the design methodology proposed in Section 10.

Step 1. A well-known set of nonminimal routing algorithms for 2D meshes can be derived from the turn model [35]. In particular, the *west-last* routing algorithm allows fully adaptive routing when the destination node is at the east of the current node. Similarly, the *east-last* routing algorithm allows fully adaptive routing when the destination node is at the west of the current node. Both algorithms can be combined to achieve nonminimal fully adaptive routing by using two virtual networks. Each physical channel is split into two virtual channels, each one belonging to a different virtual network. One virtual network is used to send messages towards the east (X-positive). It will be referred to as *eastward* virtual network. The second virtual network is used to send messages towards the west (X-negative) and it will be referred to as *westward* virtual network. When the X coordinates of the source and destination nodes are equal, messages can be introduced in the eastward (westward) virtual network, except when the destination is located on the east (west) border of the network. As mentioned above, the routing algorithms for those virtual networks are west-last and east-last, respectively. Note that if the resulting routing algorithm is restricted to use only minimal paths, it is equivalent to the algorithms proposed in [41] and [12].

Although the combination of west-last and east-last allows fully adaptive routing in 2D meshes, it does not allow 180-degree turns. These turns are required to tolerate channel faults in the border of the network, as shown in Figs. 2b, 2c, and 2f. Also, the extension of west-last and east-last for n-dimensional meshes does not allow fully adaptive routing because that extension would produce cyclic dependencies between channels. Fortunately, Theorem 1 allows the existence of such cyclic dependencies.

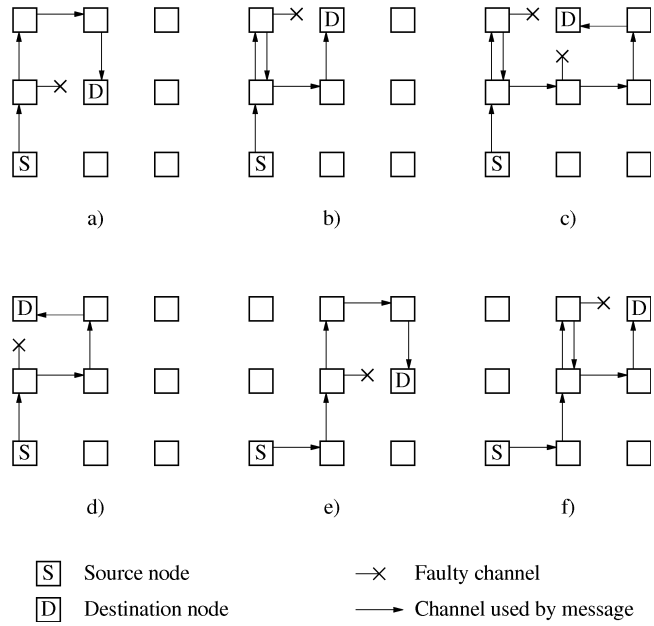

Fig. 2. Routing examples with faulty channels on a 2D mesh.

Suppose that west-last is extended for n-dimensional meshes in such a way that nonminimal fully adaptive routing and 180-degree turns are allowed in all the dimensions with two exceptions: dimension-ordered routing is used in the east border of the mesh, and messages traveling west cannot turn again. Also, suppose that east-last is extended in a similar way. The resulting routing algorithm is very flexible. Fig. 2 shows the paths followed when some channels fail. Also, Fig. 3 shows the paths followed when some nodes fail. Note that, in order to maximize performance, nonminimal paths are only used when there are faulty channels or nodes. Also, when nonminimal paths must be followed, channels crossing dimensions other than X have a higher priority. This is required to avoid routing to the east or west borders of the mesh for as long as possible because only dimension-ordered routing is allowed in those borders.

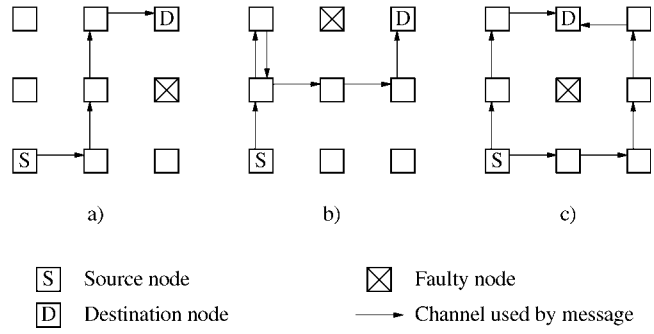

Fig. 3. Routing examples with faulty nodes on a 2D mesh. Case c) shows two alternative paths.

Step 2. Despite its flexibility, the described routing algorithm is deadlock-free. The application of Theorem 1 to prove that the routing algorithm is deadlock-free requires the definition of a suitable routing subfunction. We will

define it by defining the subset of channels C_1 that can be supplied by the routing subfunction (see Definition 5). In the eastward (westward) virtual network, we have chosen the east and west channels from each node plus the channels located on the east (west) border. The routing subfunction R_1, defined by C_1, is obviously connected. Also, messages introduced in the eastward (westward) virtual network cannot use channels from the other virtual network. So, there is no channel dependency between channels of C_1 that belong to different virtual networks. Therefore, we only need to prove that the extended channel dependency graph for R_1 in one virtual network has no cycles.

Fig. 4 shows part of the extended channel dependency graph for R_1 on a 16-node 2D mesh. Black circles represent the unidirectional channels belonging to C_1 and are labeled as *ci, j*, where *i* and *j* are the source and destination nodes, respectively. As a reference, channels are also represented by thin lines. Thick lines represent channel dependencies, dashed arrows corresponding to indirect dependencies. For the sake of clarity, we have only drawn two indirect dependencies. All the indirect dependencies are shown in Fig. 5. It can be seen that the graph is acyclic. When the mesh has more than two dimensions, the graph is identical, except on the east border. As channels on the east border can only be used crossing dimensions in ascending order, there is not any cyclic dependency between them. So, *R* is deadlock-free.

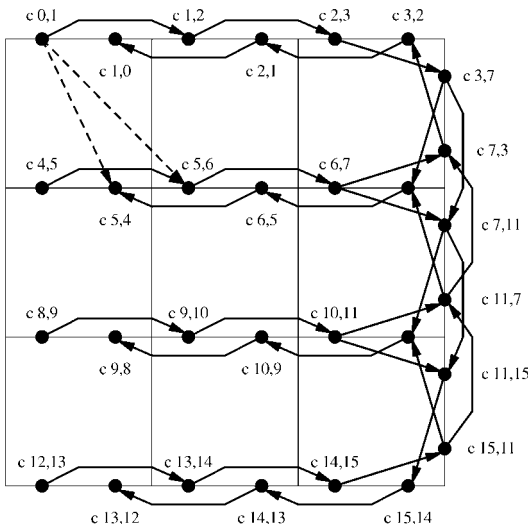

Fig. 4. Extended channel dependency graph for R_1 (only two indirect dependencies are shown).

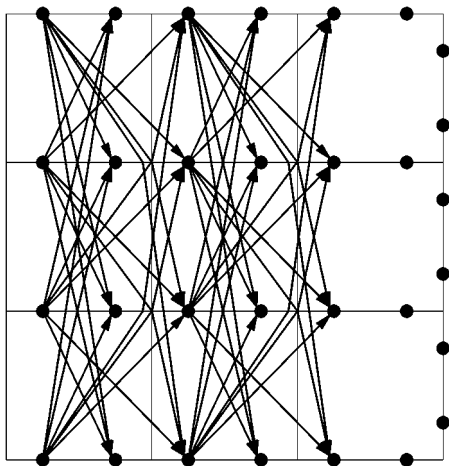

Fig. 5. Indirect dependencies of the extended channel dependency graph for R_1.

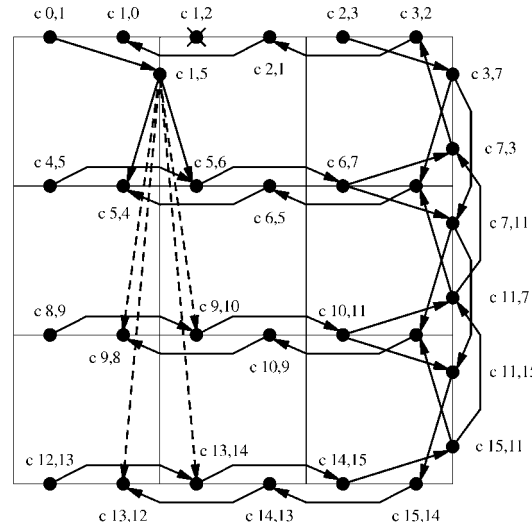

Fig. 6. Extended channel dependency graph for R_2 (only a few indirect dependencies are shown).

Step 3. In an *n*-dimensional mesh, some nodes have degree equal to *n*. So, network redundancy level cannot be greater than $n - 1$.

Step 4. According to Theorem 4, all the channels that do not belong to the X dimension (except the ones on the east and west borders) are redundant. Also, all the east (west) virtual channels in the eastward (westward) virtual network are redundant. Effectively, suppose that we remove from C_1 an east channel c_i belonging to the eastward virtual network. As a consequence, R_1 is no longer connected. Now, we add to the subset C_1, a channel with the same source node as c_i that crosses any other dimension, obtaining the subset C_2. Note that this can be done even if all the output channels but one fail. The new routing subfunction R_2 is connected. The channel dependencies introduced by the new channel do not produce any cycle, because they replace indirect dependencies. Fig. 6 shows part of the extended channel dependency graph for R_2 on a 16-node 2D mesh. In this example, channel *c*1,2 has been replaced with channel *c*1,5. We have only included indirect dependencies starting from channel *c*1,5.

Step 5. The only channels in the eastward (westward) virtual network that are not redundant are the west (east) channels, as well as the channels located on the east (west) border. This limitation can be easily removed by using two extra virtual networks. The extra eastward (westward) virtual network routes messages in the same way as the eastward (westward) virtual network. When a message traveling on the eastward (westward) virtual network reaches the east (west) border and its only path to the destination is a nonredundant faulty channel, then the message is transferred to the extra westward (eastward) virtual network until it reaches its destination. The resulting routing algorithm will

be referred to as Double East-Last West-Last (DELWL). It should be noted that the extra virtual networks have the same properties as the basic virtual networks. Additionally, messages can be transferred from a virtual network to the corresponding extra virtual network, but cannot return. Therefore, the use of the extra virtual networks cannot produce deadlock.

In summary, east (west) virtual channels belonging to the same physical channel rely on other physical channels to be redundant. Thus, all the east (west) physical channels are redundant. This is also the case for channels located on the east (west) border. So, each physical channel can be replaced by another physical channel with the same source node. In most cases, the routing algorithm offers $2(n-1)$ alternative paths in case of fault, where n is the number of dimensions of the mesh. When the faulty channel is adjacent to a node in a corner of the n-dimensional mesh (the worst case), only $n-1$ alternative paths exist. All these paths can be used by the routing algorithm. Effectively, if a faulty channel is found while the message is traveling across the east or west border, it is transferred to the corresponding extra virtual network. After that, all the $n-1$ minimal and nonminimal paths available on that border will be tried before leaving it. If one of them is fault-free, the message will be delivered. Otherwise, the message tries the last path leaving the border. As shown above, for each of those paths, it is possible to replace one X channel in C_1 by another channel, thus defining another routing subfunction without cyclic dependencies between channels. The following theorem computes the redundancy level more formally.

THEOREM 6. *The DELWL routing algorithm for n-dimensional meshes has a redundancy level equal to* $n-1$.

PROOF SKETCH. Let us consider the eastward virtual network. The application of Theorem 3 requires finding all the minimally connected routing subfunctions, as well as the smallest channel subset satisfying some properties. It should be noted that the smallest channel subset is associated with the worst case. The worst case occurs when all the faulty channels are adjacent to the same node. Let us consider a node x. Depending on the location of x, there are several cases. We are going to study four extreme cases. The remaining cases lie in between.

1) Node x has $2n$ output channels. This case corresponds to a node inside the mesh. Starting from the routing subfunction R_1 defined above, we can obtain the routing subfunctions $R_2, \ldots, R_{2n-1}$ by replacing the output channel from node x in the X-positive direction by one of the $2(n-1)$ output channels in the remaining dimensions. These routing subfunctions are minimally connected and have no cycles in their extended channel dependency graphs (see Fig. 6). The smallest subset C_m is formed by all the output channels from node x except the one in the X-negative direction. So, $|C_m| = 2n-1$.
2) Node x has $n+1$ output channels, two of them in dimension X and one in every other dimension. This case corresponds to a node in an edge of the mesh. Starting from the routing subfunction R_1 defined above, we can obtain the routing subfunctions $R_2, \ldots, R_n$ by replacing the output channel from node x in the X-positive direction by one of the $n-1$ output channels in the remaining dimensions. These routing subfunctions are minimally connected and have no cycles in their extended channel dependency graphs (see Fig. 6). The smallest subset C_m is formed by all the output channels from node x except the one in the X-negative direction. So, $|C_m| = n$.
3) Node x has $2n-1$ output channels, one of them in the X-negative direction and two in every other dimension. This case corresponds to a node in the east border of the mesh. Note that messages can only be routed in dimension-order. As messages meeting a faulty channel in the east border of the mesh are transferred to the extra westward virtual network, this case is equivalent to case 1 once the transfer has been made.
4) Node x has n output channels, one of them in the X-negative direction. This case corresponds to a node in a corner of the east border of the mesh. Note that messages can only be routed in dimension-order. As messages meeting a faulty channel in the east border of the mesh are transferred to the extra westward virtual network, this case is equivalent to case 2 once the transfer has been made.

Other intermediate cases are possible. When node x has two output channels in some dimensions and a single channel in some other dimensions, $|C_m|$ ranges from $n+1$ to $2n-2$. Similar considerations can be made for the westward virtual network.

For the extra virtual networks, it should be noted that messages are only transferred to those networks when the destination lies in one of the borders of the mesh and a faulty channel is met. Once a message has been transferred to an extra virtual network, it will only cross a channel in the X dimension if all the channels in the remaining dimensions are faulty. Therefore, messages will only be routed along a channel in the X dimension to form a nonminimal path toward their destination. This path only requires a pair of channels in the X dimension, one in each direction. As a consequence, a message transferred to the extra eastward (westward) virtual network is destined for a node in the west (east) border of the mesh, and cannot reach the east (west) border. So, only cases 1 and 2 can occur in the extra virtual networks.

Therefore, in the worst case, $|C_m| = n$. Taking into account Theorem 3, the redundancy level is equal to $n-1$. □

So, four virtual networks are required to implement fully adaptive routing and to obtain a redundancy level equal to $n-1$. These networks can be implemented by using four virtual channels per physical channel. For a 2D mesh, the redundancy level is equal to one. This does not mean that a

single fault is supported. Indeed, most configurations with many simultaneous faults are supported. In particular, any combination of rectangular fault regions is supported. The algorithm routes messages around those regions in the same way that it routes messages around single faults (see Figs. 2 and 3).

The DELWL routing algorithm has some interesting characteristics. Comparing it with previously proposed fault-tolerant routing algorithms for wormhole switching:

1) It supports channel faults without having to mark one of the adjacent nodes as faulty.
2) It does not require any synchronization between non-neighboring nodes.
3) It can be combined with the hardware mechanisms proposed in [33] to support dynamic faults.
4) Although it does not require any algorithm to mark fault-free nodes as faulty, the algorithm proposed in [8] can be used to increase the number of fault patterns supported by the DELWL routing algorithm.
5) It does not require changing the routing function in the presence of faults.
6) It does not require any special care, like draining channels in the presence of faults.

These interesting characteristics are due to the routing flexibility provided by the DELWL routing algorithm. Anyway, this is only an example of application of the theory proposed in previous sections. Our main emphasis has been in providing a theoretical background that can be used for the design of very powerful fault-tolerant routing algorithms.

12 Conclusions

This paper has analyzed the effective redundancy available in an interconnection network using wormhole switching. Besides considering the connectivity, the effective redundancy also considers that a routing function must remain deadlock-free, even in the presence of faults. This theory has been developed on top of our necessary and sufficient condition for deadlock-free adaptive routing, in order to achieve the maximum routing flexibility in the presence of faults.

First of all, we have analyzed the redundancy at the channel level, defining channel redundancy and giving a sufficient condition for a channel to be redundant. Also, the set of redundant channels has been computed. Then, we have analyzed the redundancy at the network level. After defining the redundancy level of a routing function, a theorem is proposed. This theorem establishes the necessary and sufficient conditions for a routing function to achieve a given redundancy level. We have considered virtual channels explicitly, analyzing the effect of failures in physical channels when they are split into virtual channels. Node faults have also been considered.

The necessary and sufficient condition for deadlock-free routing in store-and-forward and virtual cut-through switching is almost identical to the one for wormhole switching. As a consequence, the theoretical results presented in this paper are also valid for those switching techniques.

Finally, we have proposed a methodology to guide the design of fault-tolerant routing algorithms. As an example of application, we have developed a fault-tolerant routing algorithm for n-dimensional meshes that has a redundancy level equal to $n - 1$, therefore allowing up to $n - 1$ simultaneous faults in the worst case.

Acknowledgments

We would like to thank Prof. Sudhakar Yalamanchili and the anonymous referees for their helpful comments and suggestions. This work was supported by the Spanish CICYT under Grant TIC94–0510–C02–01. A short version of this paper was presented at the 1994 International Conference on Parallel and Distributed Systems.

References

[1] A. Agarwal, "Limits on Interconnection Network Performance," *IEEE Trans. Parallel and Distributed Systems*, vol. 2, no. 4, pp. 398-412, Oct. 1991.

[2] M.S. Alam and R.G. Melhem, "How to Use an Incomplete Binary Hypercube for Fault Tolerance," *Hypercube and Distributed Computers*, F. André and J.P. Verjus, eds., pp. 329-341. North-Holland, 1989.

[3] J.D. Allen, P.T. Gaughan, D.E. Schimmel and S. Yalamanchili, "Ariadne—An Adaptive Router for Fault-Tolerant Multicomputers," *Proc. 21st Ann. Int'l Symp. Computer Architecture*, Apr. 1994.

[4] W.C. Athas and C.L. Seitz, "Multicomputers: Message-Passing Concurrent Computers," *Computer*, vol. 21, no. 8, pp. 9-24, Aug. 1988.

[5] K.W. Bolding and W. Yost, "Design of a Router for Fault Tolerant Networks," *Proc. Parallel Computer Routing and Comm. Workshop*, pp. 226-240, May 1994.

[6] R.V. Boppana and S. Chalasani, "A Comparison of Adaptive Wormhole Routing Algorithms," *Proc. 20th Ann. Int'l Symp. Computer Architecture*, May 1993.

[7] R.V. Boppana and S. Chalasani, "Fault-Tolerant Routing with Non-Adaptive Wormhole Algorithms in Mesh Networks," *Proc. Supercomputing '94*, pp. 693-702, 1994.

[8] R.V. Boppana and S. Chalasani, "Fault-Tolerant Wormhole Routing Algorithms for Mesh Networks," *IEEE Trans. Computers*, vol. 44, no. 7, pp. 848-864, July 1995.

[9] S. Chalasani and R.V. Boppana, "Fault-Tolerant Wormhole Routing in Tori," *Proc. Eighth Int'l Conf. Supercomputing*, July 1994.

[10] M.-S. Chen and K.G. Shin, "Depth-First Search Approach for Fault-Tolerant Routing in Hypercube Multicomputers," *IEEE Trans. Parallel and Distributed Systems*, vol. 1, no. 2, pp. 152-159, Apr. 1990.

[11] M.-S. Chen and K.G. Shin, "Adaptive Fault-Tolerant Routing in Hypercube Multicomputers," *IEEE Trans. Computers*, vol. 39, no. 12, pp. 1,406-1,416, Dec. 1990.

[12] A.A. Chien and J.H. Kim, "Planar-Adaptive Routing: Low-Cost Adaptive Networks for Multiprocessors," *J. ACM*, vol. 42, no. 1, pp. 91-123, 1995.

[13] G. Chiu and S. Wu, "Fault-Tolerant Routing Strategy in Hypercube Systems," *Proc. 24th Int'l Symp. Fault-Tolerant Computing*, pp. 382-391, 1994.

[14] R. Cypher and L. Gravano, "Requirements for Deadlock-Free Adaptive Packet Routing," *Proc. 11th ACM Symp. Principles of Distributed Computing*, 1992.

[15] W.J. Dally and C.L. Seitz, "The Torus Routing Chip," *Distributed Computing*, vol. 1, no. 3, pp. 187-196, Oct. 1986.

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

[17] W.J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194-205, Mar. 1992.

[18] W.J. Dally and H. Aoki, "Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels," *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 4, pp. 466-475, Apr. 1993.

[19] W.J. Dally, L. Dennison, D. Harris, K. Kan, and T. Xanthopoulus, "The Reliable Router: A Reliable and High-Performance Communication Substrate for Parallel Computers," *Proc. Parallel Computer Routing and Comm. Workshop*, pp. 241-255, May 1994.

[20] B.V. Dao, J. Duato, and S. Yalamanchili, "Configurable Flow Control Mechanisms for Fault-Tolerant Routing," *Proc. 22nd Ann. Int'l Symp. Computer Architecture*, June 1995.
[21] J. Duato, "On the Design of Deadlock-Free Adaptive Routing Algorithms for Multicomputers: Design Methodologies," *Proc. Parallel Architectures Languages Europe 91*, June 1991.
[22] J. Duato, "A Theory to Increase the Effective Redundancy in Wormhole Networks," *Proc. Int'l Conf. Decentralized Distributed Systems*, Sept. 1993.
[23] J. Duato, "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 12, pp. 1,320-1,331, Dec. 1993.
[24] J. Duato, "A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks," *Proc. 1994 Int'l Conf. Parallel Processing*, Aug. 1994.
[25] J. Duato, "A Theory to Increase the Effective Redundancy in Wormhole Networks," *Parallel Processing Letters*, vol. 4, nos. 1/2, pp. 125-138, 1994.
[26] J. Duato, "A Theory of Fault-Tolerant Routing in Wormhole Networks," *Proc. Int'l Conf. Parallel and Distributed Systems*, Dec. 1994.
[27] J. Duato, B.V. Dao, P.T. Gaughan, and S. Yalamanchili, "Scouting: Fully Adaptive, Deadlock-Free Routing in Faulty Pipelined Networks," *Proc. Int'l Conf. Parallel and Distributed Systems*, Dec. 1994.
[28] J. Duato, "A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 6, no. 10, pp. 1,055-1,067, Oct. 1995.
[29] J. Duato, "A Necessary and Sufficient Condition for Deadlock-Free Routing in Cut-Through and Store-and-Forward Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 7, no. 8, pp. 841-854, Aug. 1996.
[30] A.-H. Esfahanian and S.L. Hakimi, "Fault-Tolerant Routing in de Bruijn Communication Networks," *IEEE Trans. Computers*, vol. 34, no. 9, pp. 777-788, Sept. 1985.
[31] P. Fraigniaud, "Fault-Tolerant Gossiping on Hypercube Multicomputers," *Proc. Second European Distributed Memory Computing Conf.*, Apr. 1991.
[32] P.T. Gaughan and S. Yalamanchili, "Pipelined Circuit-Switching: A Fault-Tolerant Variant of Wormhole Routing," *Proc. Fourth IEEE Int'l Symp. Parallel and Distributed Processing*, Dec. 1992.
[33] P.T. Gaughan, B.V. Dao, S. Yalamanchili, and D.E. Schimmel, "Distributed, Deadlock-Free Routing in Faulty, Pipelined *k*-ary *n*-cubes," Technical Report GIT-CSRL/93-11, Georgia Inst. of Technology, Nov. 1993.
[34] P.T. Gaughan and S. Yalamanchili, "A Family of Fault-Tolerant Routing Protocols for Direct Multiprocessor Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 6, no. 5, pp. 482-497, May 1995.
[35] C.J. Glass and L.M. Ni, "The Turn Model for Adaptive Routing," *Proc. 19th Ann. Int'l Symp. Computer Architecture*, May 1992.
[36] C.J. Glass and L.M. Ni, "Fault-Tolerant Wormhole Routing in Meshes," *Proc. 23rd Int'l Symp. Fault-Tolerant Computing*, June 1993.
[37] R.L. Hadas and E. Brandt, "Origin-Based Fault-Tolerant Routing in the Mesh," *Proc. First IEEE Symp. High Performance Computer Architecture*, 1995.
[38] J.H. Kim, Z. Liu, and A.A. Chien, "Compressionless Routing: A Framework for Adaptive and Fault-Tolerant Routing," *Proc. 21st Ann. Int'l Symp. Computer Architecture*, Apr. 1994.
[39] J. Kuhl and S. Reddy, "Fault-Tolerance Considerations in Large Multiple-Processor Systems," *Computer*, vol. 19, no. 3, pp. 56-67, Mar. 1986.
[40] T. Lee and J.P. Hayes, "A Fault-Tolerant Communication Scheme for Hypercube Computers," *IEEE Trans. Computers*, vol. 41, no. 10, pp. 1,242-1,256, Oct. 1992.
[41] D.H. Linder and J.C. Harden, "An Adaptive and Fault Tolerant Wormhole Routing Strategy for *k*-ary *n*-cubes," *IEEE Trans. Computers* vol. 40, no. 1, pp. 2-12, Jan. 1991.
[42] D. Pradhan, "Dynamically Restructurable Fault-Tolerant Processor Network Architectures," *IEEE Trans. Computers*, vol. 34, no. 5, pp. 434-447, May 1985.
[43] C.S. Raghavendra, P.-J. Yang, and S.-B. Tien, "Free Dimensions, an Effective Approach to Achieving Fault Tolerance in Hypercubes," *Proc. 22nd Int'l Symp. Fault-Tolerant Computing*, pp. 170-177, 1992.
[44] P. Ramanathan and K.G. Shin, "Reliable Broadcast in Hypercube Multicomputers," *IEEE Trans. Computers*, vol. 37, no. 12, pp. 1,654-1,657, Dec. 1988.
[45] M.A. Sridhar and C.S. Raghavendra, "Fault-Tolerant Networks Based on the de Bruijn Graph," *IEEE Trans. Computers*, vol. 40, no. 10, pp. 1,167-1,174, Oct. 1991.
[46] Y.-J. Suh, B.V. Dao, J. Duato, and S. Yalamanchili, "Software Based Fault-Tolerant Oblivious Routing in Pipelined Networks," *Proc. 1995 Int'l Conf. Parallel Processing*, Aug. 1995.

José Duato received the electrical engineering and PhD degrees from the Polytechnical University of Valencia, Valencia, Spain, in 1981 and 1985, respectively.

In 1981, he joined the Department of Systems Engineering, Computers, and Automation at the Polytechnical University of Valencia, where he is currently a professor of computer architecture and technology. He has developed several courses on computer organization, peripheral devices, computer architecture, and multicomputer design. His current research interests are in the analysis and design of interconnection networks for multicomputer and multiprocessor systems, and networks of workstations. His theory of deadlock-free adaptive routing for wormhole networks has been highly cited and used by other researchers. In particular, it has been used in the development of the routing algorithms for the MIT Reliable Router and the Cray T3E. He is the coauthor with S. Yalamanchili and L.M. Ni of the text *Interconnection Networks: An Engineering Approach*, to be published by the IEEE Computer Society Press.

Professor Duato is serving as a member of the editorial board of *IEEE Transactions on Parallel and Distributed Systems*. He also serves on the program committees for several major conferences. Dr. Duato received the Best Curriculum in Electrical Engineering Nation-Wide award in 1982.