A New Fault Information Model for Fault-Tolerant Adaptive and Minimal Routing in 3-D Meshes *

Zhen Jiang
Department of Computer Science
Information Assurance Center
West Chester University
West Chester, PA 19383
zjiang@wcupa.edu

Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
jie@cse.fau.edu

Dajin Wang
Department of Computer Science
Montclair State University
Upper Montclair, NJ 07043
wang@pegasus.montclair.edu

*The work was supported in part by NSF grants ANI 0083836, CCR 9900646, CNS 0422762, CNS 0434533, and EIA 0130806.
Abstract

In this paper we extend Wang’s Minimal-Connected-Component (MCC) model [15] in 2-D meshes to 3-D meshes. It is based on our early work on fault tolerant adaptive and minimal routing [18] and boundary information model [10] in 3-D meshes. We study the fault tolerant adaptive and minimal routing from $(0,0,0)$ (source) to $(x,y,z)$ (destination, $x,y,z \geq 0$) and consider the positions of source and destination when the new faulty components in 3-D meshes are constructed. Specifically, all faulty nodes will be contained in some disjoint faulty components and a healthy node will be included in a faulty component only if using it in the routing will definitely cause a non-minimal routing path. A sufficient and necessary condition is proposed for the existence of the minimal routing path in presence of our faulty components. Our simulation results show substantial improvement of new faulty block in terms of higher percentage of minimal routing assured at the source in 3-D meshes.

Index Terms: Adaptive routing, fault information models, fault tolerance, minimal routing, 3-D meshes.

1 Introduction

In a multicomputer system, a collection of processors (or nodes) work together to solve large application problems. These nodes communicate data and coordinate their efforts by sending and receiving packets through the underlying communication network. Thus, the performance of such a multicomputer system depends on the end-to-end cost of communication mechanisms. Routing time of packets is one of the key factors that are critical to the performance of multicomputers. Basically, routing is the process of transmitting data from one node called the source node to another node called the destination node in a given system. A minimal routing always routes the packet to the destination through a shortest path. The mesh-connected topology [6, 11] is one of the most thoroughly investigated network topologies for multicomputer systems. Like 2-dimensional (2-D) meshes, 3-D meshes are lower dimensional meshes that have been commonly discussed due to structural regularity for easy construction and high potential of legibility of various algorithms. Some multicomputers were built based on the 3-D meshes [1, 11].

As the number of nodes in a mesh-connected multicomputer system increases, the chance of failure also increases. The complex nature of networks also makes them vulnerable to disturbances. Therefore, the ability to tolerate failure is becoming increasingly important for obtaining high assurance systems, especially in the communication subsystem. Several studies have been conducted which achieve fault tolerance by adding (or deleting) extra components of the system [7, 14]. However, adding and deleting nodes and/or links require modifications of network topologies which may be expensive and difficult. We focus here on achieving fault tolerance using the inherent redundancy present in 3-D meshes, without adding spare nodes and/or links.

Most existing literatures [2, 3, 4, 5, 8, 12, 13, 20] use the simplest orthogonal convex region to model node faults (link faults can be treated as node faults by disabling the corresponding adjacent nodes). Wu
provided a node labelling scheme in [16] that identifies nodes (faulty and non-faulty) that cause routing detours in 2-D meshes and such nodes are called unsafe nodes. Connected unsafe nodes form a rectangular region, also called rectangular faulty block. In [9, 16], the information of rectangular faulty blocks is distributed to a limited number of nodes at the boundary lines to avoid a routing message entering a detour area. By using this so called limited global fault information at boundary lines in 2-D meshes, Wu’s fully adaptive routing proposed in [16] can easily find a minimal path. To reduce the number of non-faulty nodes contained in rectangular faulty blocks, Wang [15] proposed the minimal connected component (MCC) model as a refinement of the rectangular faulty block model by considering the relative locations of source and destination nodes. The original idea is that a node will be included in an MCC only if using it in a routing will definitely make the route non-minimal. It turns out that each MCC is of the rectilinear monotone polygonal shape and is the absolutely minimal fault region in 2-D meshes.

In this paper, we provide a boundary construction for MCCs in 2-D meshes through messages exchanges among neighboring nodes. With the information of MCCs at the boundary lines, Wang’s sufficient and necessary condition of the existence of the minimal routing can be re-written so that not only a minimal routing can be ensured at the source node but also a minimal path can be formed by routing decisions at intermediate nodes along the path. After that, MCC model and its boundary construction will be extended to 3-D meshes. A sufficient and necessary condition is proposed for the existence of the minimal routing. Based on this condition, a new (fully) adaptive routing is provided to build a minimal path between source and destination nodes as long as there exists such a minimal path. Compared with other fault information such as a routing table associated with each node, the update of our information converges quickly. When a disturbance occurs, only those affected nodes need to update fault information. It reduces oscillation caused by unstable information. It also reduces the memory requirement to store fault information in the whole network. Extensive simulation has been done to determine the number of non-faulty nodes included in MCCs in 3-D meshes and the rate of success minimal routing under MCC model. The result obtained is compared with the best existing known result.

The challenge here is not only to conduct a theoretical study on the MCC model in 3-D meshes and its corresponding sufficient and necessary condition of existence of minimal routing, but also to search for a practical and efficient implementation in a system where each node knows only the status of its neighbors. First, after each non-faulty node in an MCC is labelled, the whole 3-D fault region should be identified. Then, the identified information of this MCC will be propagated in two dimensions along some 2-D surfaces (also called boundaries) to avoid the routing entering the detour area (in the direction of the other dimension). Finally, a new routing with two phases is proposed. In phase one, the boundary information of any MCC that may block the routing message is collected and used to build the assurance of minimal routing at the source node. The routing process at the source will be activated only if a minimal path exists. In phase two, the routing process at each intermediate node between source and destination nodes will forward the message to the next node along the path. It uses the boundary information to avoid the message entering the detour area and keep the routing path minimal. It is noted that all these are implemented through the message
transmission (including information messages and routing messages) between two neighboring nodes along one of those three dimensions \(X, Y\) and \(Z\).

The remainder of the paper is organized as follows: Section 2 introduces some necessary notations and preliminaries. Section 3 provides our boundary construction for MCCs in 2-D meshes and our boundary-information-based minimal and adaptive routing. In the same section, Wang’s sufficient and necessary condition for the existence of minimal path is re-written without using global information. Section 4 extends the above MCC model in 2-D meshes to 3-D meshes. A node labelling scheme for non-faulty nodes in each MCC and the construction of boundaries for an MCC are presented. Section 5 provides a sufficient and necessary condition for the existence of minimal routing in 3-D meshes. In Section 6, the minimal and adaptive routing in 2-D meshes is extended to 3-D meshes. Its improvement comparing with the best existing routing in terms of the rate of success minimal routing is shown in experimental results in Section 7. The improvement of our MCC model comparing with the best existing fault information model in 3-D meshes in terms of the number of unsafe nodes contained in fault regions which cannot be used to communicate is also shown in section 7. Section 8 concludes this paper and provides ideas for future research.

2 Preliminary

A \(k\)-ary \(n\)-dimensional mesh with \(k^n\) nodes has an interior node degree of \(2n\) and the network diameter is \((k - 1)n\). Each node \(u\) has an address \((u_1, u_2, \ldots, u_n)\), where \(0 \leq u_i \leq k - 1\). Two nodes \((v_1, v_2, \ldots, v_n)\) and \((u_1, u_2, \ldots, u_n)\) are neighbors if their addresses differ in one and only one dimension, say dimension \(i\); moreover, \(|v_i - u_i| = 1\). Basically, nodes along each dimension are connected as a linear array. In a 2-D mesh, each node \(u\) is labelled as \((x_u, y_u)\) and the distance between two nodes \(u\) and \(v\), \(D(u, v)\), is equal to \(|x_v - x_u| + |y_v - y_u|\). Assume node \(u\) is the current node and \(d\) is the destination node. Simply, for a node \(u(x_u, y_u)\), node \(v(x_u + 1, y_u)\) is called the \(+X\) neighbor of \(u\). Respectively, \((x_u - 1, y_u)\), \((x_u, y_u - 1)\), and \((x_u, y_u + 1)\) are \(-X\), \(-Y\) and \(+Y\) neighbors of node \(u\) in 2-D meshes. When node \(v\) is a neighbor of node \(u\), \(v\) is called a preferred neighbor if \(D(v, d) < D(u, d)\); otherwise, it is called a spare neighbor. Respectively, the corresponding connecting directions are called preferred direction and spare direction. Without loss of generality, assume node \(s(0, 0)\) is source node and node \(d(x_d, y_d)\) \((x_d, y_d \geq 0)\) is the destination node.

A routing process is minimal if the length of the routing path from source node \(s\) to destination node \(d\) is equal to \(D(s, d)\). Similarly, in a 3-D mesh, \((0, 0, 0)\) is the source node, \(u(x_u, y_u, z_u)\) is the current node, \(d(x_d, y_d, z_d)\) \((x_d, y_d, z_d \geq 0)\) is the destination node, and the distance between two nodes \(u\) and \(v\), \(D(u, v)\), is equal to \(|x_v - x_u| + |y_v - y_u| + |z_v - z_u|\). \((x_u + 1, y_u, z_u)\), \((x_u - 1, y_u, z_u)\), \((x_u, y_u + 1, z_u)\), \((x_u, y_u - 1, z_u)\), \((x_u, y_u, z_u + 1)\) and \((x_u, y_u, z_u - 1)\) are \(+X\), \(-X\), \(+Y\), \(-Y\), \(+Z\), and \(-Z\) neighbors of node \(u\).

The formation of MCC in 2-D meshes [15] is based on the notions of useless and can’t-reach nodes (see in Figure 1 (a)): A node labelled useless is such a node that once a routing from \((0, 0)\) to \((x_d, y_d)\) \((x_d, y_d \geq 0)\) enters it, the next move must take either \(-X\) or \(-Y\) direction, making the routing non-minimal. A node

---

4
Algorithm 1: Labelling procedure of MCC for the routing from $(0, 0)$ to $(x_d, y_d)$ ($x_d, y_d \geq 0$)

1. Initially, label all faulty nodes as *faulty* and all non-faulty nodes as *safe*.
2. If node $u$ is safe, but its $+X$ neighbor and $+Y$ neighbor are faulty or useless, $u$ is labelled *useless*.
3. If node $u$ is safe, but its $-X$ neighbor and $-Y$ neighbor are faulty or can’t-reach, $u$ is labelled *can’t-reach*.
4. The nodes are recursively labelled until there is no new useless or can’t-reach node.
5. All faulty, useless, and can’t-reach nodes (other than safe nodes) are also called *unsafe* nodes.

labelled can’t-reach is such a node that for a routing to enter it, a $-X$ or $-Y$ direction move must be taken, making the routing non-minimal. The node status (faulty, useless, and can’t-reach) can be determined through a labelling procedure and all faulty, useless, and can’t-reach nodes are also called unsafe nodes. The labelling procedure is given in Algorithm 1 and it can quickly identify those non-faulty nodes in MCCs. Each active node collects its neighbors’ status and updates its status. Only those affected nodes update their status. Eventually, neighboring unsafe nodes form an MCC. Figure 1 (a) shows the idea of the definition of useless and can’t-reach nodes. Figure 1 (b) shows some samples of MCCs for the routing from $(0, 0)$ to $(x_d, y_d)$ ($x_d, y_d \geq 0)$.

3 Boundary information in MCC model in 2-D meshes

In this section, we provide a distributed process to collect the information of each MCC and distribute it along the boundaries so that not only the existence of a minimal path can be ensured at the source node but also a minimal routing can be constructed by routing decisions at intermediate nodes along the path. The
new routing process provided in this section can find a minimal path from the source to destination nodes whenever this path exists.

3.1 Corner and boundary of MCC in 2-D meshes

To collect the information of all MCCs for routing process, each MCC needs to identify its fault region. Any node inside fault region MCC is called an unsafe node. Otherwise, it is called a safe node. Any safe node with an unsafe neighbor in an MCC is called an edge node of that MCC. A corner is a safe node with two edge nodes of the same MCC in different dimensions or a safe node with two unsafe node of the same MCC in different dimensions. After the labelling procedure, the identification process starts from an initialization corner. The initialization corner is a corner with two edge nodes of the same MCC in the $+X$ and $+Y$ dimensions. A safe node with two edge nodes of the same MCC in the $-X$ and $-Y$ dimensions is called the opposite corner of that initialization corner (of the same MCC).

From that initialization corner, two identification messages, one clockwise and one counter-clockwise, are initiated. Each message carries partial region information: First, they will be sent to these two edge neighbors. Such propagation will continue along the edges until the messages reach the opposite corner. When the clockwise message passes through an intermediate corner $u(x_u, y_u)$, node information $(x_u, y_u)$ will be attached to the message. This information will be used at the opposite corner to form the shape of this MCC. Similarly, the counter-clockwise message will also bring the node information of every intermediate corner it passed through to the opposite corner. After these two messages meet at the opposite corner, the
propagation will continue and bring the identified information back to the initialization corner. This time, no new intermediate corner needs to be identified and no new information will be added into each message. Figure 2 shows a sample of the identification process.

An MCC has only one initialization corner \( c(x_c, y_c) \) and one opposite corner \( c'(x_{c'}, y_{c'}) \). If two identification messages cannot meet at that opposite corner, or if any of them finds the change of shape when it is sent back to \( c \), it suggests that this MCC is not stable. The message is discarded to avoid generating incorrect MCC boundary information. If only one message is received at the initialization corner, the other has been discarded in the propagation procedure and this message should also be discarded. Normally, a TTL (time-to-live) is associated with each identification message and the corresponding message will be discarded once the time expires.

In 2-D meshes, an MCC with the initialization corner \( c \) is noted by \( M(c) \). The region right below it should be forbidden for the routing message to enter in the \(+X\) direction if the destination is right above it. This region is called the forbidden region, noted by \( Q_Y(c) \). The corresponding region right above \( M(c) \) is called critical region, noted by \( Q'_Y(c) \). Similarly, the routing message should avoid to enter the forbidden region \( Q_X(c) \) on the left side of \( M(c) \) in the \(+Y\) direction if the destination is in the critical region \( Q_X(c) \) on the right side of \( M(c) \). To guide the routing process, two boundary messages will be initiated when two identification messages are both received at node \( c \). One boundary message (also called \( X \) boundary) will carry the information \( M(c) \), \( Q_X(c) \), and \( Q'_X(c) \) and propagate to all the nodes along the boundary line \( y = y_c \) in the \(-X\) direction until it reaches the edge of this 2-D meshes. When this boundary line intersects with another MCC \( (M(v)) \), a turn in the \(-Y\) direction is made. After that, it will go along the edges of \( M(v) \) to join the same boundary of \( M(v) \) at the initialization corner \( v \). At that corner \( v \), \( Q_X(v) \)

**Figure 3. Samples of boundary construction under MCC model in 2-D meshes.**
Algorithm 2: Identification process and boundary construction of an MCC

1. Identification of edge nodes, the initialization corner \( c(x_c, y_c) \), intermediate corners, and the opposite corner \( c'(x_c', y_c') \).

2. Identification process of MCC \( M(c) \): (a) From node \( c \), two identification messages (one clockwise and one counter-clockwise) are sent along the edge nodes of \( M(c) \) until they reach node \( c' \). (b) Partial region information (including the node information of all intermediate corners and corner \( c \)) is transferred to form the shape of \( M(c) \) at node \( c' \). (c) After they meet at node \( c' \), the propagation will continue until the identification information reaches back to node \( c \). (d) The stable shape of \( M(c) \) can be ensured at node \( c \) once two identification messages are both received. Meanwhile, the critical and forbidden regions \( \{Q_X(c), Q_Y(c), Q'_X(c), Q'_Y(c)\} \) are identified.

3. \( X / Y \) boundary construction of \( M(c) \): A boundary construction is activated at node \( c \) after it receives two identification messages back. The information of \( M(c) \), \( Q_X(c) \), \( Q_Y(c) \), and \( Q'_X(c), Q'_Y(c) \) is propagated along the boundary line \( y = y_c \) or \( x = x_c \). When the propagation intersects another MCC \( M(v) \), it will make a turn in the \(-Y / -X\) direction and go along the edges of \( M(v) \). Eventually, it will join the same boundary of \( M(v) \). Since then, the area information of forbidden region of \( M(v) \) \( (Q_X(v) \cup Q_Y(v)) \) will merge into that of \( M(c) \) \( Q_X(c) \cup Q_Y(c) \).

mergers into \( Q_X(c) = Q_X(c) \cup Q_X(v) \). Similarly, another boundary propagation (construction of \( Y \) boundary) carrying \( M(c) \), \( Q_Y(c) \), and \( Q'_Y(c) \) will go along \( x = x_c \) in the \(-Y\) direction and make a turn in the \(-X\) direction if necessary. Figure 3 shows some samples of boundary construction. The whole procedure is shown in Algorithm 2.

3.2 Sufficient and necessary condition for the existence of minimal routing in 2-D meshes

MCC model includes much fewer non-faulty nodes in its fault region than the conventional rectangular model in 2-D meshes. Many non-faulty nodes that would have been included in rectangular faulty blocks now can become candidate routing nodes. As a matter of fact, MCC is the “ultimate” minimal fault region: that is, no non-faulty node contained in an MCC will be useful in a minimal routing. A routing enters a non-faulty node in the MCC would force a step that violates the requirement for a minimal routing. In other words, the MCC is a fault information model that provides the maximum possibility to find minimal routing path in the presence of faults. If there exists no minimal routing under MCC model, there will be absolutely no minimal routing. In [15], a sufficient and necessary condition was provided for the existence of minimal routing. This can be re-written as the following:

**Lemma 1:** A routing does not have a minimal path iff there exists an MCC \( M(c) \) that its forbidden region \( Q_X(c) \) contains the source \( s \) and its critical region \( Q'_X(c) \) contains the destination \( d \), or its forbidden region \( Q_Y(c) \) contains \( s \) and its critical region \( Q'_Y(c) \) contains \( d \).

**Proof:** It’s obvious that there is a sequence of MCCs \( (M_1, M_2, ..., M_n) \), such that (a) \( M_1 \) contains a node
Figure 4. (a) Type-I MCC sequence. (b) Type-II MCC sequence.

(0, y₁) and 0 < y₁ < y_d, (b) M_n contains a node (x_d, y_d) and 0 < y_n < y_d, and (c) For all M_i and M_{i+1} (1 ≤ i ≤ n - 1), min{a | (a, b) ∈ M_{i+1}} - 1 ≤ max{u | (u, v) ∈ M_i} ≤ max{y | (x, y) ∈ M_{i+1}} - 1 and max{v | (u, v) ∈ M_i} < max{y | (x, y) ∈ M_{i+1}}, if and only if there exists an MCC M(c) that s ∈ Q_X(c) and d ∈ Q_X(c).

That sequence is also called Type-I sequence in [15]. Similarly, there is a sequence of MCCs (M_1, M_2, ..., M_n), such that (a) M_1 contains a node (x_1, 0) and 0 < x_1 < x_d, (b) M_n contains a node (x_n, y_d) and 0 < x_n < x_d, and (c) For all M_i and M_{i+1} (1 ≤ i ≤ n - 1), min{b | (a, b) ∈ M_{i+1}} - 1 ≤ max{v | (u, v) ∈ M_i} ≤ max{y | (x, y) ∈ M_{i+1}} - 1 and max{u | (u, v) ∈ M_i} < max{y | (x, y) ∈ M_{i+1}}, if and only if there exists an MCC M(c) that s ∈ Q_Y(c) and d ∈ Q_Y(c). That sequence is also called Type-II sequence in [15]. Based on the proof of the theorem in [15] that a minimal routing can be found if and only if neither Type-I sequence nor Type-II sequence exists, the statement is proved to be true. For the details, refer to [15].

**Theorem 1:** A routing does not have a minimal path iff there exists an MCC that its critical region Q'_X(c) contains the destination d(x_d, y_d) and its X boundary does not intersect with the segment [0 : 0, 0 : y_d], or its critical region Q'_Y contains d and its Y boundary does not intersect with the segment [0 : x_d, 0 : 0].

**Proof:** Based on our boundary construction for X boundary in Algorithm 2, when d ∈ Q'_X(c), s ∈ Q_X(c) if and only if the X boundary does not intersect with the segment [0 : 0, 0 : y_d]. Similarly, when d ∈ Q'_Y(c), s ∈ Q_Y(c) if and only if the Y boundary does not intersect with the segment [0 : x_d, 0 : 0]. With Lemma 1, the statement is easy to prove.

### 3.3 Boundary-information-based routing under MCC model in 2-D meshes

Wu proposed a minimal and adaptive routing in n-D meshes in [17]. It can easily be extended to a routing in 2-D meshes under MCC model (see in Algorithm 3). In this routing, at the source node s, a detection is
Algorithm 3: Routing from $s(0, 0)$ to $d(x_d, y_d)$ ($x_d, y_d \geq 0$)

1. Feasibility check: At source $s$, send two detection messages (the first along $+Y$ direction and the second along $+X$ direction) until they reach the line $x = x_d$ or line $y = y_d$. If the first/second message intersects with another MCC, make a turn to $+X$/$+Y$ direction first. And then turn back to the $+Y$/$+X$ direction at the first intermediate corner encountered. If it intersects with the segment $[0 : x_d, y_d : y_d] / [x_d : x_d, 0 : y_d]$, return YES to node $s$; otherwise, return No.

2. Routing decision and message sending:
   (a) At source $s$, if both return are YES in the feasibility check, go to next step; otherwise stop the routing since there is no minimal path.
   (b) At the current node $u(x_u, y_u)$, including node $s$, (i) add all the preferred directions into the set of candidates of forwarding directions $F$ and find all the records MCCs, (ii) for each $M(c)$ found in the above step, exclude $+Y$ direction from $F$ if $d \in Q_X(c)$ and exclude $+X$ direction from $F$ if $d \in Q_Y(c)$, (iii) apply any fully adaptive and minimal routing process to pick up a forwarding direction from set $F$, and (iv) forward the routing message along the selected forwarding direction to the next node.

first activated to make sure if there exists a minimal path. Otherwise, the routing will stop. First, the source node $s$ will play the role as the destination in Theorem 1. It sends two detection messages, one along $+Y$ direction and one along $+X$ direction. The first one is to check if there exists a type-I sequence [15]. If it can reach the segment $[0 : x_d, y_d : y_d]$, there is no type-I sequence and a minimal path is not impossible (return “YES”). If it intersects with another MCC, make a turn to $+X$ direction first. And then turn back to the $+Y$ direction at the first intermediate corner encountered. Similarly, the second one is to check if there exists a type-II sequence. If the source $s$ can know both segments can be reached, based on the sufficient and necessary condition for the existence of minimal path in Theorem 1, a minimal routing is feasible from $d$ to $s$. Then, it is derived that a minimal routing is feasible from the source to destination.

At each node along the routing path, including the source node $s$, the routing process basically has two preferred directions: $+X$ and $+Y$ directions. The boundary information of an MCC with the destination in the critical region will help the routing process avoid entering the forbidden region by excluding the corresponding preferred direction from candidates of forwarding direction. After that, any fully adaptive and minimal routing process could be applied to pick up the forwarding direction and forward the routing message along this direction to the corresponding neighbor. The procedure of feasibility check and routing decision using boundary information can also be seen in the samples in Figure 5.

4 MCC model in 3-D meshes

In this section, we present our distributed solution for constructing MCCs and propagating the region information in 3-D meshes. First, the status of each node inside fault region is identified in a labelling
process. Then, each 2-D section of a 3-D fault region and its neighboring sections are identified in an identification process. After that, the information of 2-D sections of a fault region are collected along the edges of this fault region in the edge construction. With this information, the region is identified as an MCC and the information of its shape, forbidden region, and critical region is formed. Finally, in boundary construction, this information will be propagated along the boundary of this MCC to avoid the routing entering its forbidden region.

4.1 Labelling process

A useless node \( u \) in an MCC in 2-D meshes has two useless or faulty neighbors in \( +X \) and \( +Y \) directions. Based on the label scheme in Algorithm 1, any \( +X/+Y \) routing will be blocked by the faulty nodes if it enters node \( u \). For a non-faulty node \( u \) in 3-D meshes, if it has only two useless or faulty neighbors in \( +X \) and \( +Y \) directions, the routing message can take the \( +Z \) direction and route around the fault region. Therefore, a non-faulty node is useless in 3-D meshes iff it has three useless or faulty neighbors in \( +X \), \( +Y \), and \( +Z \) directions. Similarly, a non-faulty node is can’t-reach iff it has three can’t-reach or faulty neighbors in \( -X \), \( -Y \), and \( -Z \) directions. The corresponding labelling scheme is shown in Algorithm 4.

**Algorithm 4**: Labelling procedure of MCC in 3-D meshes

1. Initially, label all faulty nodes as *faulty* and all non-faulty nodes as *safe*.
2. If node \( u \) is safe, but its \( +X \) neighbor, \( +Y \) neighbor, and \( +Z \) neighbor are faulty or useless, \( u \) is labelled *useless*. 

---

**Figure 5.** (a) Feasibility check for a case without minimal routing path. (b) Feasibility check to ensure the existence of minimal routing path. (c) Routing decisions in routing process to construct a minimal routing path.
Figure 6. (a) Sample MCCs in 3-D meshes, and (b) the section of MCC on the plane $z = 5$.

3. If node $u$ is safe, but its $-X$ neighbor, $-Y$ neighbor, and $-Z$ neighbor are faulty or can’t-reach, $u$ is labelled can’t-reach.

4. The nodes are recursively labelled until there is no new useless or can’t-reach node.

5. All faulty, useless, and can’t-reach nodes are also called unsafe nodes.

Figure 6 shows two sample MCCs in 3-D meshes. First, $(5, 5, 6)$, $(6, 5, 5)$, $(5, 6, 5)$, $(6, 7, 5)$, $(7, 6, 5)$, $(5, 4, 7)$, $(4, 5, 7)$, and $(7, 8, 4)$ are faulty nodes. Then, $(5, 5, 5)$ becomes useless and $(5, 5, 7)$ becomes can’t-reach according to our labelling process in Algorithm 4. One MCC contains only one faulty node $(7, 8, 4)$ and the other MCC contains all the other faulty, useless, and can’t-reach nodes. Usually, a 2-D section of the MCC parallel to plane $x = 0$, plane $y = 0$, or plane $z = 0$ is not a convex polygon. A convex polygon has been defined in [19] as a polygon $P$ for which a line segment connecting any two points in $P$ lies entirely within $P$. A section of the second MCC on the plane $z = 5$ (see in Figure 6) shows a hole at $(6, 6, 5)$ in the MCC region.

4.2 Identification process

The identification process for an MCC in 3-D meshes is based on the one for the MCC in 2-D meshes. It starts from the identification of each 2-D section on $XY$ plane, $YZ$ plane, and $XZ$ plane simultaneously. Simply, we call these sections $XY$ sections, $YZ$ sections, and $XZ$ sections respectively. For each 2-D section, for example, a $XY$ section, a two-head-on message identification process in algorithm 2 is activated at its corner $c$, say one with the minimum coordinate along $X$ dimension of those which have the maximum coordinate along $Y$ dimension. Such a corner is also called a $(+Y−X)$-corner of this $XY$ section. The corner of a section uses the same definition in section 3 for an orthogonal fault region in 2-D meshes. This $XY$ section may have several corners with the maximum coordinate along $X$ dimension and the one with the minimum coordinate along $Y$ dimension is called $(+X−Y)$-corner of this section. Respectively, we
have (+X – Z)- and (+Z – X)-corners of a XZ section, and (+Y – Z)- and (+Z – Y)-corners of a YZ section. Each YZ / XZ section will be identified by a similar process initialize from its (+Z – Y)- / (+X – Z)-corner. It is noted that these two identification messages may meet at any edge node of the 2-D section, not necessary a corner node.

After section identification, six kinds of edges of each MCC: (+Y – X)-edge, (+Y – Z)-edge, (+X – Y)-edge, (+X – Z)-edge, (+Z – Y)-edge and (+Z – X)-edge are identified for the boundary construction. Any of these edges is defined by all its edge nodes and each edge node is the corresponding corner in its 2-D section. The identification process for any edge starts from each of these corners and has three phases. It is to find a path to link each pair of the preceding node and its succeeding node along the edge so that the whole edge can be formed. In phase one, a message will be initiated at the start corner and route around its 2-D section to find a path to the neighboring section (in a certain direction). In phase two, this message will be propagated along that path to the neighboring section. In phase three, it will route around that neighboring section to reach its corresponding corner. Once this message reaches that corner, those two corners in neighboring sections are identified as preceding and succeeding edge nodes and the path between them will be used for future edge construction.

For example, the (+Y – X)-edge of an MCC is defined by the (+Y – X)-corners of all XY sections of this MCC. The identification process for this (+Y – X)-edge starts from each of these (+Y – X)-corners. In phase one, from a (+Y – X)-corner c(x_c, y_c, z_c) in the XY section z = z_c in Figure 7 (a), a message will be sent to route around this section. When such a message passes through an edge node u(x_u, y_u, z_c) with an unsafe neighbor in the −Y dimension, the identified information of YZ section on the plane x = x_u is used to find a neighboring section on plane z = z_c + 1. A neighboring section exists if z_c is not the minimum coordinate in the +Z dimension in that YZ section. In phase two, the neighboring section is found and the

Figure 7. (a) Identification for XY sections, (b) identification for the +Y – X edge, (c) MCC information at the edge node, and (d) boundary construction.
message will go around the corresponding YZ section to the neighboring XY section (seen in Figure 7 (a)). In phase three, once the message arrives at an edge node of the neighboring XY section, a two-head-on message propagation will be initiated to go around that section (one clockwise and one counter-clockwise) to reach its corresponding (+Y − X) corner u′ (seen in Figure 7 (b)). At node u′, c is identified as its succeeding node along the edge and the information of the path to node c (see in Figure 7 (c)) is saved for future information propagation.

4.3 Edge construction

If the neighboring section cannot be found in phase one in the above identification process, that start corner is identified as one end of this edge and the corresponding section is identified as the surface of this MCC (see corner c′ in Figure 7 (a)). In phase three of the above identification process, a concave region of MCC containing the start corner can be identified if there is another 2-D section in the same plane with the origin section by checking the information of the YZ section at each node the message passes. For example, in Figure 7 (b), at node u′′ the message passes, such a concave region is found. Once such a concave region is found, the start corner (corner c′′ in the example in Figure 7 (b)) will be identified as one end of another edge (an inner edge of this MCC towards a concave area).

Starting from an end node u identified, a collection process is activated to collect all the links between preceding and succeeding nodes along this edge and form the whole edge. A message is sent along the path to the succeeding node of u and its propagation will continue until it reaches the other end (an edge node without any succeeding node). At each edge node v it passes through, the section information is collected and the previously saved information is used to form a part of this MCC M(v). This M(v) includes the area of this MCC from the current node M(v) to the section with end node u. With the information of M(v), the information of forbidden region QY(v) and the critical region Q0Y(v) can also be formed at node v (see in the example in Figure 7 (c)).

4.4 Boundary construction

At each edge node u, say along a (+Y − X)-edge, after M(u) is formed, the information of M(u), QY(u), and Q0Y(u) will be propagated along the boundary (also called (+Y − X)-boundary) to block the routing from entering the detour area QY(u) in the +X dimension if the destination is inside the critical region Q0Y(u). Initially, a message carrying the information is sent from node u along the −Y dimension. Once this message intersects with another MCC at node u′(xu′, yu′, zu′), the boundary of node u will make a −X turn and route around the XY section of the latter MCC (in the counter-clockwise direction) until it joins the boundary of the latter MCC which has started from its (+Y − X)-edge node v on plane z = zu′. When the message propagates along the boundary of node v, the forbidden region of node v (QY(v)) will be merged into QY(u) (QY(u) = QY(v) ∪ QY(u)). If node u is the (−Z)-most edge node (the edge node without any succeeding node) of the origin MCC, at node u′, a copy of the message will be sent to the
Algorithm 5: Identification and boundary construction of an MCC in 3-D meshes

1. Identification of each 2-D section by using the identification process in Algorithm 2.

2. Identification of each edge: six kinds of edge nodes \((+Y - X)\)- and \((+X - Y)\)-corners in the \(XY\) section, \((+X - Z)\)- and \((+Z - X)\)-corners in the \(XZ\) section, and \((+Y - Z)\)- and \((+Z - Y)\)-corners in the \(YZ\) section) are identified for each kind of boundary. For each edge node, (a) a message is sent along its 2-D section to find the path to its neighboring section in a certain direction; (b) the message reaches the neighboring section along this path; (c) an identification process in algorithm 2 is applied to reach the corresponding corner in this neighboring section, the preceding node of the start corner.

3. Edge construction: If no neighboring section is found or there is another section in the plane with the origin one, the start corner will be identified as an end node without preceding node. From this end node, a message is sent to the succeeding node along the edge and the propagation will continue until it reaches the other end edge node. It is to collect the section information at each edge node \(c\) and the previously saved information will be used to form the concerning MCC part \(M(c)\), the forbidden region \(Q(c)\), and the critical region \(Q'(c)\) for the boundary construction starting from \(c\).

4. Boundary construction: After \(M(u)\), \(Q(u)\), and \(Q'(u)\) is formed at an edge node \(u\), say along the \((+Y - X)\)-edge, the information \(M(u)\), \(Q_Y(u)\), \(Q'_Y(u)\) will be propagated along its \((+Y - X)\)-boundary in the \(-Y\) direction. If it intersects with another MCC \(M(v)\), it will join the boundary of the “nose” part of \(M(v)\) and merge the forbidden region \(Q_Y(v)\) into \(Q_Y(u)\).

corresponding edge node \(v\). From node \(v\), it will go along that \((+Y - X)\)-edge and reach the \((-Z)\)-most edge node. At each edge node \(v'\) it passes through, a boundary is constructed for node \(u\) with the information of \(M(u)\), \(Q'_Y(u)\), and \(Q_Y(u) \cup Q_Y(v')\). An sample of boundary construction is shown in Figure 7 (d). The whole procedure to collect and distribute MCC information is shown in Algorithm 5.

5 Sufficient and necessary condition for the existence of minimal routing in 3-D meshes

In this section, a sufficient and necessary condition with boundary information is presented to ensure the existence of minimal routing in 3-D meshes.

After the boundary construction, a boundary node will have the region information \((M(c))\), the forbidden region information \((Q(c) = Q_X(c), Q_Y(c), \text{or} Q_Z(c))\), and the critical region information \((Q'(c) = Q'_X(c), Q'_Y(c), \text{or} Q'_Z(c))\). When a routing enters this node and its destination \(d\) is inside the critical region, the boundary line can be used as a part of routing path to route around the \(M(c)\) and avoid detours. Thus, we have the following sufficient and necessary condition for the existence of minimal routing in 3-D meshes:

Theorem 2: A routing does not have a minimal path iff there exists an MCC that (a) its critical region \(Q'_X\) contains the destination \(d(x_d, y_d, z_d)\) and neither its \((+X - Y)\)-boundary nor its \((+X - Z)\)-boundary intersects with the surface \(\{0 : 0, 0 : y_d, 0 : z_d\}\), (b) its critical region \(Q'_Y\) contains the destination \(d\) and
neither its \((+Y - X)\)-boundary nor its \((+Y - Z)\)-boundary intersects with the surface \(0 : x_d, 0 : 0, 0 : z_d\), or (c) its critical region \(Q'_Z\) contains the destination \(d\) and neither its \((+Z - Y)\)-boundary nor its \((+Z - Y)\)-boundary intersects with the surface \(0 : x_d, 0 : y_d, 0 : 0\).

Proof: The necessity is obvious. We prove the sufficiency here. Assume there is no minimal path from \(s\) to \(d\). Considering the region of minimal paths (RMP) that includes each node that can be reached from \(s\) in \(+X\), \(+Y\), or \(+Z\) routing. Once any boundary intersects with this RMP, it will also intersects with one of those three surfaces: \((-Y)\)-surface on the \(-Y\) side, \((-Z)\)-surface on the \(-Z\) side, and \((-X)\)-surface on the \(-X\) side. It is obvious that the boundary of the closest MCC which has \(d\) inside its \(Q'_Y\) does not intersect with this RMP. Otherwise, the routing process can find a minimal path along the boundary to the area \(Q'_Y\), and then find a minimal path to \(d\). That means, the RMP is blocked by other MCCs in the \(+X\) and \(+Z\) dimension. Based on the boundary construction, there is an MCC has \(d\) inside its \(Q_Z\) and neither its \(+Z - X\) boundary nor \(+Z - Y\) boundary intersects the RMP and the surface \(0 : x_d, 0 : y_d, 0 : 0\). Otherwise, there is an MCC has \(d\) inside its \(Q_X\) and neither its \(+X - Y\) boundary nor \(+X - Z\) boundary intersects the RMP and the surface \(0 : 0, 0 : y_d, 0 : z_d\).

6 Boundary-information-based routing under MCC model in 3-D meshes

Based on Theorem 2, Wu’s routing in [18] in 3-D meshes is extended to a routing under MCC model (see in Algorithm 6). Such a routing can find a minimal path from the source and destination nodes whenever this path exists.

Similar to the routing in 2-D meshes, at the source node \(s\), a detection is first activated to make sure if there exists a minimal path. Otherwise, the routing will stop. The source node \(s\) will play the role as the
Algorithm 6: Routing from \(s(0, 0, 0)\) to \(d(x_d, y_d, z_d)\) \((x_d, y_d, z_d \geq 0)\)

1. Feasibility check: At source \(s\), send detection messages along three surfaces of RMP, which is the region including each node that can be reached from \(s\) in \(+X\), \(+Y\), or \(+Z\) routing: \((-X)\)-surface on the \(-X\) side, \((-Y)\)-surface on the \(-Y\) side, and \((-Z)\)-surface on the \(-Z\) side. Once any of these messages on \((-X)\)-surface reaches the surface \([0 : x_d, y_d : y_d, 0 : z_d]\), it will return the result back to \(s\). Similarly, the propagation on \((-Y)\)- or \((-Z)\)-surface is to see if the surface \([0 : x_d, 0 : y_d, z_d : z_d]\) or \([x_d : x_d, 0 : y_d, 0 : z_d]\) can be reached. 

2. Routing decision and message sending:

   (a) At source \(s\), if all three surfaces are reached in the feasibility check, go to next step; otherwise stop the routing since there is no minimal path.

   (b) At the current node \(u(x_u, y_u, z_u)\), including node \(s\), (i) add all the preferred directions into the set of candidates of forwarding directions \(F\) and find all the records of MCCs, (ii) for each MCC found in the above step, exclude direction from \(F\) if the destination is in the critical region and the neighbor of \(u\) along this direction is inside forbidden region, (iii) apply any fully adaptive and minimal routing process to pick up a forwarding direction from set \(F\), and (iv) forward the routing message along the selected forwarding direction to the next node.

destination in Theorem 2 and three detection messages will be sent from \(s\) along each surface of the region of minimal path (RMP): \((-Y)\)-surface on the \(-Y\) side, \((-Z)\)-surface on the \(-Z\) side, and \((-X)\)-surface on the \(-X\) side. RMP is a region from \(s\) including each node that can be reached from \(s\) in \(+X\), \(+Y\), or \(+Z\) routing. For each surface, say \((-X)\)-surface, a message will first propagated to the neighbors of \(s\) in the \(+Y\) and \(+Z\) direction. After that, at each node \(u\) that receives it, this message will continue to propagate to \(u\)'s \(+Y\) neighbor and \(+Z\) neighbor. If any propagation intersects with another MCC, it will make a \(+X\) turn and go back to the original direction as it is possible. If the message can reach the surfaces \([0 : x_d, y_d : y_d, 0 : z_d]\), it will return this information back to \(s\). The propagation of detection messages on other surfaces are similar. If the source \(s\) can know all those surfaces \([x_d : x_d, 0 : y_d, 0 : z_d]\), \([0 : x_d, y_d : y_d, 0 : z_d]\), and \([0 : x_d, 0 : y_d, z_d : z_d]\) can be reached, based on the sufficient and necessary condition for the existence of minimal path in Theorem 2, a minimal routing is feasible from \(d\) to \(s\). Then, it is derived that a minimal routing is feasible from the source to destination. Figure 8 shows some samples of this feasibility check.

At each node, including the source node \(s\), the routing process basically has three preferred directions: \(+X\), \(+Y\), and \(+Z\) directions. The boundary information of an MCC with the destination in the critical region will help the routing process avoid entering the forbidden region by excluding the corresponding preferred direction from candidates of forwarding direction. After that, any fully adaptive and minimal routing process could be applied to pick up the forwarding direction and forward the routing message along this direction to the corresponding neighbor. Figure 9 shows some samples of routing under our MCC model in 3-D meshes.
7 Simulation

Our new routing process can find a minimal path from $s$ to $d$ whenever it exists. However, it needs a broadcasting along three surfaces. Wu presented a sufficient detection process for minimal routing in [18]; that is, if a minimal path is ensured in the detection process, then a minimal path can be found, but not the other way around. To reduce the cost of detection, we replace our detection process in Algorithm 6 by that in [18] and extend the routing in [18] under our MCC model. We compare the performance of the routing using rectangle fault block (RFB), the extension of routing in [18] under MCC model (MCC(wu)), and our new minimal routing (MCC) in terms of the number of unsafe nodes which cannot be used to communicate, the number of rounds of information exchanges and updates in labelling process, and the percentage rate of success minimal routing. In our simulation, we randomly generate numbers of faults (from 1 to 500) and a pair of the source and the destination in a $30 \times 30 \times 30$ 3-D mesh.

We show the results in Figure 10. Comparing with the rectangle fault block, MCC model has much fewer number of unsafe nodes when the number of faults is bigger than 400 and can enable much more end-to-end communication in the whole network. Also, the speed of labelling process under our MCC model is very fast. Moreover, even when the number of faults reaches to 500, our new routing can still find a minimal path for most of cases (>$ \ 99.9\%$). When the routing using rectangle fault block in [18] is extended under our MCC model, the increased rate of success minimal routing shows the improvement of our MCC model in 3-D meshes.

8 Conclusion

In summary, the contribution of this paper is listed as the following:

1. We have rewritten the MCC model in 2-D meshes without using global information so that the shape information at our boundaries can be used not only to ensure the existence of a minimal path, but also,
Figure 10. (a) Number of unsafe nodes (log), (b) number of rounds of information exchanges and updates in labelling process, (c) percentage rate of success minimal routing.

to form a minimal routing by routing decisions at intermediate nodes along the path. This routing will find a minimal path from source $s$ to destination $d$ whenever it exists.

2. We have extended the MCC model in 2-D meshes to 3-D meshes. The labelling process, edge identification process, edge construction, and boundary construction are presented to collect and distribute the MCC information for the minimal routing.

3. Based on the information collected in the above limited-global-information model, a new minimal routing in 3-D meshes has been provided. It will find a minimal path from $s$ to $d$ whenever it exists.

4. Our experimental results have shown the improvement of our MCC model by comparing with the best existing known result.

In our future work, we will extend our results to dynamic networks in which all the faulty components can occur during the routing process. Also, our results will be extended to higher dimension networks.

References


