International Journal of Foundations of Computer Science
© World Scientific Publishing Company

On Achieving the Shortest-Path Routing in 2-D Meshes

Zhen Jiang
Computer Science Department, West Chester University
West Chester, Pennsylvania 19383, United States

and

Jie Wu
Computer Science and Engineering Department, Florida Atlantic University
Boca Raton, Florida 33431, United States

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

In this paper, we present a fully distributed process to collect and distribute the minimal connected component (MCC) fault information so that the shortest-path between a source and its destination can always be found in the corresponding information-based routing via routing decisions made at each intermediate node. Considering the communication cost in the above information distribution, a more practical implementation is provided with a low number of nodes involved in the information propagation. The experimental results show substantial improvement of our approach in terms of the success rate in finding the shortest-path and the average path length.

Keywords: 2-D meshes, fault information model, routing, shortest-path.

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. The routing time of packets is one of the key factors 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 routing path from the source to the destination is determined by the forwarding node selection at each intermediate node in a fully-distributed manner in order to make the entire system more scalable. It is necessary to present a routing scheme that always routes the package to the destination via the shortest-path so that the destination can be reached in the quickest way.

The *mesh-connected topology* [6, 12] is one of the most thoroughly investigated network topologies for multicomputer systems. 2-dimensional (2-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 2-D meshes [7, 12, 13, 15, 18, 20]. 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 [2, 8, 9, 14, 16, 21, 22, 23, 24, 26, 27].

The shortest-path is constructed among all of the non-faulty nodes under the existing network configuration after the failure. In the presence of node faults (link faults can be treated as node faults by disabling the corresponding adjacent nodes), appropriate fault information provided for routing decisions is the key to achieve the shortest-path routing. Most existing literature uses the simplest orthogonal convex region in the information model [2, 3, 4, 5, 10, 17, 24, 25]. To reduce the number of non-faulty nodes contained in rectangular faulty blocks, Wang [19] 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 for routing will definitely make the route non-shortest. 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 [11], the information of each MCC is propagated along an edge of its "forbidden region", also known as the boundary. For each routing case, a check of routing feasibility is conducted first at the source. It sends out two detection messages and waits for their responses in order to ensure the existence of a path with the Manhattan distance [1] between the source and the destination. Then, the routing process using any existing adaptive routing scheme starts if and only if such a path exists. The information saved along the boundary will be used in the routing decisions at those nodes to guarantee the success of the shortest-path routing. Such a routing that always routes the message along a Manhattan distance path is also called Manhattan routing.

However, without global information, no existing routing scheme using localized decisions can guarantee a shortest path when the Manhattan distance path does not exist between the source and the destination. When the number of faults increases, fewer routing cases will have the Manhattan distance path.

In this paper, we provide a new routing method to form a shortest path regardless of whether the Manhattan distance path exists between the source and the destination. The challenge lays in the development of an information model in which the shortest path for any source/destination pair can be described. Moreover, in order to make the whole system more scalable under such an information model, no global information is used. The contributions of this paper are threefold.

(i) We provide a method of collecting and distributing the MCC information via information exchanges among neighbors so that the shortest path for a given pair of a source and a destination can be determined via localized decisions. To reduce the cost of information distribution, a practical implementation is provided in which only a limited number of nodes along the boundaries are involved in the information exchanges and updates.

(ii) A new routing protocol which contains multiple phases of the Manhattan routing is provided. It can always form a shortest-path, even when the whole Manhattan distance path (between the source and the destination) does not exist. In this paper, we will prove that there is no path shorter than the one found under our model.

(iii) We develop a simulation to show substantial improvement of our new approach in the routing performance, compared with the best results known to date. The experimental results also show that our approach is cost-effective.

To summarize, in our new approach, the MCC information will propagate not only along the boundary presented in [11], but also into the forbidden region defined in [11]. Therefore, the information can be used to avoid a detour in cases where the Manhattan distance path does not exist. A routing method is proposed under this new information model to form the shortest-path. Due to the limited effect of each MCC, the whole routing path is partitioned into several Manhattan distance sub-paths. Obviously, the shortest-path in each phase does not imply that the entire path is the shortest. The success of using multiple shortest-paths distinguishes our approach from other schemes, while in each phase any existing adaptive routing can be applied under the guideline in [11]. Note that our new routing can always find a Manhattan distance path between the source and the destination if it exists. Considering the cost of broadcasting in forbidden regions, a practical implementation is proposed here. Two boundaries initialized from opposite corners of each MCC are used to bound each forbidden region, as opposed to only one as the case in [11] is. The information needs to be sent to a limited number of nodes along those boundaries. Our simulation results show that the shortest-path can be achieved in most cases, and that very few detours are needed in non-shortest-path routings.

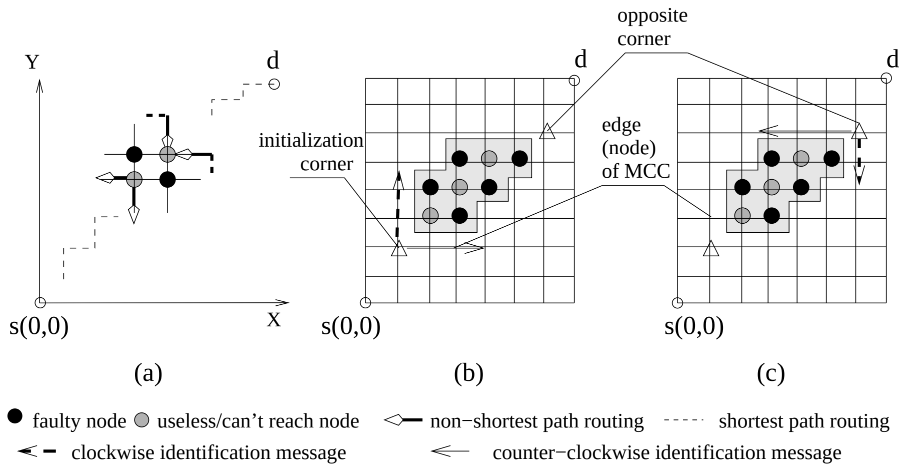
The remainder of this paper is organized as follows. Section 2 introduces some necessary notations and preliminaries, including the MCC model, the boundary model, and the existing method of Manhattan routing. In the same section, the detour problem in the case without a Manhattan distance path is discussed. Section 3 provides our new fault information model and the corresponding routing protocol. It is guaranteed that this new routing method will always find the shortest-path. The extended boundary model (which is more practical to implement) and the revised routing protocol are also presented in this section. In Section 4, the experimental results are provided to show the quality-improvement and the cost-effectiveness of our new shortest-path routing. Section 5 concludes the paper and provides directions for future research.

2. Preliminary

A 2-dimensional (2-D) $n \times n$ mesh with n^2 nodes has an interior node degree of 4. Nodes along each dimension are connected as a linear array. Each node u has an address (x_u, y_u), where $0 \leq x_u, y_u < n$. Node (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. The Manhattan distance between any two nodes u and v is the geographic distance $\mid x_u - x_v \mid + \mid y_u - y_v \mid$, denoted by $M(u, v)$. Note that node s is the source node, u is the current node, and d is the destination node. We consider the positions of the source and destination when the new faulty components are constructed. Without loss of generality, assume $x_s = y_s = 0$ and $x_d, y_d \geq 0$. Due to the effect of faulty nodes, a path with the length $M(s, d)$ may not exist. $D(s, d)$ denotes the length of a shortest-path between s and d and $D(s, d) \geq M(s, d)$. In general, $[x : x',\ y : y']$ represents a rectangular region with four vertexes: (x, y), (x, y'), (x', y'), and (x', y). Specifically, $[x : x,\ y : y']/[x : x',\ y : y]$ represents a line segment along the Y/X dimension.

The formation of MCC in 2-D meshes [19] is based on the notions of *useless* and *can't-reach* nodes. A useless node is such a node that, once a routing enters it, the next move must occur in a $-X/-Y$ direction, which in turn makes the routing non-shortest. A can't-reach node is such a node that, for a routing to enter it, a move in the $-X/-Y$ direction must be taken, making the routing non-shortest. The status of faulty, useless, and can't-reach nodes can be determined through a labeling procedure. Initially, label all faulty nodes as *faulty*, and all non-faulty nodes as *safe*. If a node is safe, but its $+X$ neighbor and $+Y$ neighbor are faulty or useless, it is labeled *useless*. If the $-X$ neighbor and $-Y$ neighbor are faulty or can't-reach, such a safe node is labeled *can't-reach*. The nodes are iteratively labeled until there is no new useless or can't-reach node. All faulty, useless, and can't-reach nodes are also called *unsafe* nodes. The other nodes are called *safe nodes*. Neighboring unsafe nodes form an MCC. The labeling procedure can quickly identify the non-faulty nodes in MCCs. Each active node collects its neighbors' statuses, and updates its own status. Only those affected nodes update their statuses. Figure 1 (a) shows the idea of the definition of useless and can't-reach nodes. Figure 1 (b) shows a sample of MCC under the assumption that $x_s = y_s = 0$ and $x_d, y_d \geq 0$. In this paper, we focus on the case when the shortest-path exists. Therefore, we have the following assumption: (a) the entire network is connected, and (b) the source and the destination are safe nodes.

After the labeling process, a distributed process that is presented in [11] is conducted to collect the shape information of each MCC and distribute it to a limited number of nodes, also called boundaries. This process starts from an *initialization corner*. An MCC with initialization-corner c is denoted by $F(c)$. The initialization corner is a safe node with two edge neighbors of the same MCC in the $+X$ and $+Y$ dimensions. A safe node with an unsafe neighbor is called an *edge* node of the corresponding MCC. A safe node with two edge neighbors of the same MCC in the $-X$ and $-Y$ dimensions is called the *opposite corner*. From that initialization corner, two identification messages, one clockwise and one counter-clockwise each carrying partial region information, will propagate along the edges of an MCC and reach its opposite corner. By collecting the location information of each node that these two messages passed through, the shape of this MCC can be identified at the opposite



Y
X
d
s(0,0)
(a)
(b)
(c)
opposite corner
initialization corner
edge (node) of MCC
● faulty node
● useless/can't reach node
non−shortest path routing
shortest path routing
clockwise identification message
counter−clockwise identification message

Figure 1: (a) Definition of useless and can't-reach nodes in [19]. (b) A sample of MCC and its propagation activated at the initialization corner. (c) Identified information re-sending.

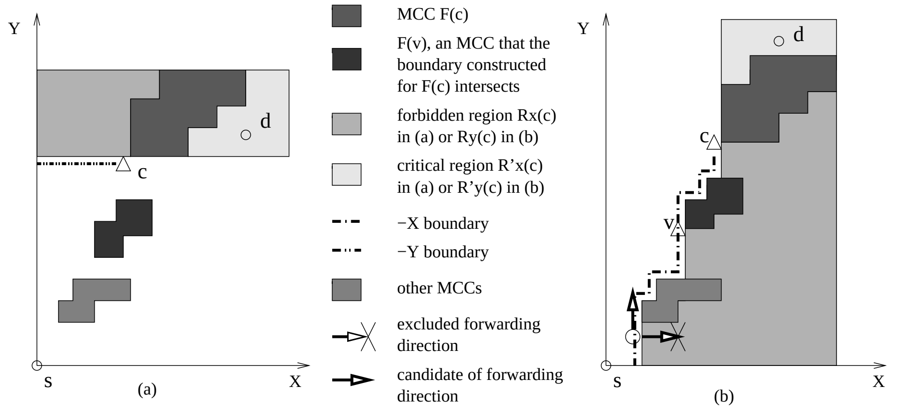
corner (see Figure 1 (b)). After that, the propagation will continue and bring the identified shape information $F(c)$ back to the initialization corner (see Figure 1 (c)). The region right below $F(c)$ is called the *forbidden region*, noted by $R_Y(c)$. The region right above $F(c)$ is called the *critical region*, noted by $R'_Y(c)$. To guide the routing process, one boundary message (also called $-X$ boundary [11]) will carry the information $F(c)$, $R_Y(c)$, and $R'_Y(c)$ and propagate to all the nodes along the line $x = x_c$ until it reaches the edge of the entire mesh. When this boundary intersects with another MCC $F(v)$, a right turn is made. After that, it will go along the edges of $F(v)$ to join the same boundary constructed for $F(v)$. From the corner v, $R_Y(v)$ merges into $R_Y(c)$ ($R_Y(c) = R_Y(c) \cup R_Y(v)$) (see Figure 2 (b)). Similarly, another boundary propagation (construction of $-Y$ boundary [11]) carrying $F(c)$, $R_X(c)$, and $R'_X(c)$ goes along $y = y_c$ (see Figure 2 (a)) and will make a left turn if necessary. The whole procedure is shown in Algorithm 1.

Algorithm 1 [11]: Boundary construction for an MCC $F(c)$ ($B1$).

1. From the initialization corner c, two identification messages (one clockwise and one counter-clockwise) are sent along the edges of MCC until they reach the opposite corner c'. The locations of all intermediate nodes are collected in order to form the shape $F(c)$ at node c'. Then, the forbidden and critical regions ($R_X(c)$, $R_Y(c)$, $R'_X(c)$, $R'_Y(c)$) are identified.

2. From node c', the propagation will continue until the identified information reaches back to node c.

3. From node c, the triple ($F(c)$, $R_X(c)$, $R'_X(c)$)/($F(c)$, $R_Y(c)$, $R'_Y(c)$) is propagated along line $y = y_c$ / $x = x_c$. When the propagation intersects with an-



Y
X
s
c
d
(a)
MCC F(c)
F(v), an MCC that the boundary constructed for F(c) intersects
forbidden region Rx(c) in (a) or Ry(c) in (b)
critical region R'x(c) in (a) or R'y(c) in (b)
−X boundary
−Y boundary
other MCCs
excluded forwarding direction
candidate of forwarding direction
Y
X
s
c
v
d
(b)

Figure 2: (a) Sample of MCC information propagation. (b) Sample of information propagation with intersection and the routing decision using the corresponding propagation information.

other MCC, say $F(v)$, it will make a left/right turn to join the same boundary constructed for $F(v)$. Then, $R_X(v)$ / $R_Y(v)$ will merge into $R_X(c)$ / $R_Y(c)$.

Before the routing starts, a detection is activated at the source node s to ensure the existence of a Manhattan distance path. After that, the routing decision at each node along the path, including the source node s, basically has two forwarding directions: $+X$ and $+Y$. The information saved along the boundaries is used to prevent the routing from entering the forbidden region by excluding the corresponding direction from the candidates of the forwarding direction. Then, any fully-adaptive routing could be applied to forward the message. The procedure of the routing decision using boundary information is listed in Algorithm 2 and can be seen in the sample in Figure 2 (b).

Algorithm 2 [11]: Manhattan routing at the current node u with the information of $d(x_d, y_d)$

1. Add the $+X/+Y$ direction into the set of candidates of forwarding directions P, if $x_d > 0/y_d > 0$ and the $+X/+Y$ neighbor is not fault.
2. For each triple (shape information of an MCC $F(c)$, its corresponding forbidden region $R(c)$, and its corresponding critical region $R'(c)$) found at u, remove the direction from P if using it will cause the routing to enter region $R(c)$ while $d \in R'(c)$.
3. Apply any fully-adaptive routing to select a forwarding direction from set P.
4. Forward the routing message along the selected direction.

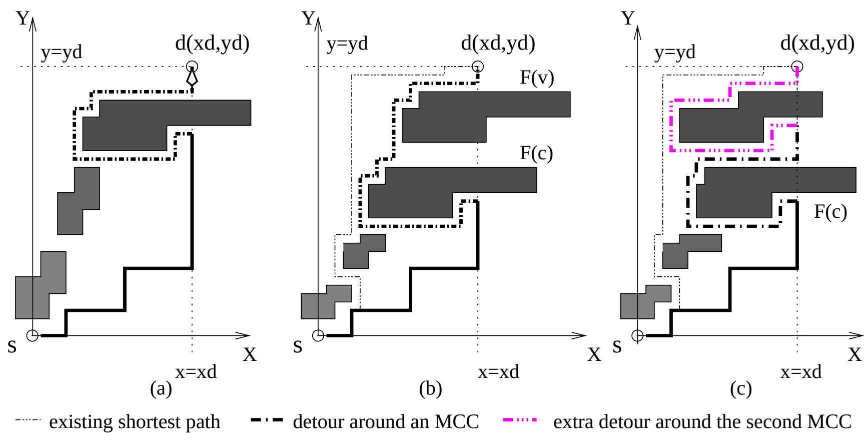


Figure 3: (a) Detour in E-cube routing [2]. (b) Detour of routing in Algorithm 3 when it is applied to the case $D(s,d) > M(s,d)$. (c) Extra detour.

As more faults occur in the networks, more (and bigger) MCCs will form. As a result, we have more routing cases that are blocked by MCCs and do not have the Manhattan distance path, i.e., $D(s,d) > M(s,d)$. The above routing cannot find the existing connected path due to the failure in the check of routing feasibility. Therefore, the problem lays in finding the shortest-path when $D(s,d) > M(s,d)$.

In E-cube routing [2], when the forwarding direction is blocked, it will detour around the fault region to reach the other side (see in Figure 3 (a)). With this kind of detour, the routing in Algorithm 2 can find a path from s to d when $D(s,d) > M(s,d)$. Therefore, the check of routing feasibility is unnecessary. The routing process is rewritten in Algorithm 3. When such a routing intersects with an MCC, say $F(c)$, it will route around the fault region in the clockwise direction. In this way, the joint boundary from the second MCC $F(v)$ can be used to save detours of $F(v)$ (see Figure 3 (b)). However, this routing is not shortest-path routing. Moreover, when the whole detour path around $F(c)$ is inside the forbidden region of another MCC, the routing can be blocked again and require additional detours (see Figure 3 (c)). In the following section, we consider all the cases and propose a new information model to help the routing achieve the shortest-path.

Algorithm 3: Routing in Algorithm 2 with E-cube routing detour ($RB1$), at the current node u with the information of $d(x_d, y_d)$

1. Add $+X/+Y$ direction into the set of candidates of forwarding directions P, if $x_d > 0/y_d > 0$ and the $+X/+Y$ neighbor is neither fault nor the preceding node.

2. Same as step 2 in Algorithm 2.

3. If $P = \phi$ (the routing is blocked by an MCC), select $-X$ or $-Y$ direction to route around the MCC in clockwise direction and go to step 5.

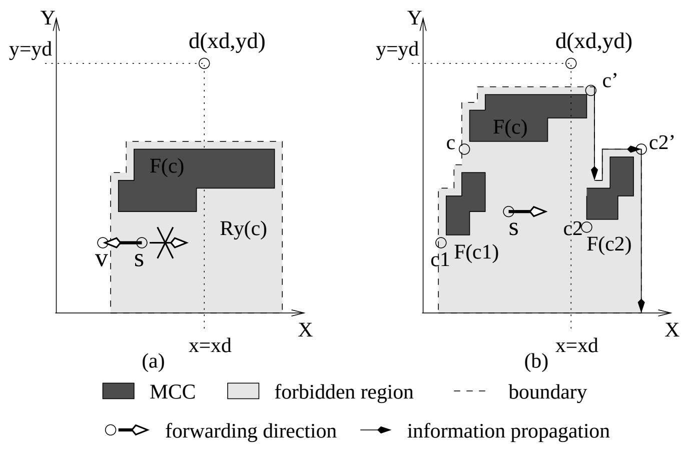


Figure 4: (a) A "must-take" detour. (b) A sample of complete information broadcasting and a correct routing decision based on such information.

4. Same as step 3 in Algorithm 2.

5. Same as step 4 in Algorithm 2.

---

3. Fault Information Model for the Shortest-Path Routing

In this section, we introduce our method of achieving the shortest-path routing in 2-D meshes in the presence of faults. We will prove that no path is shorter than the one found in our routing. Considering the communication cost of the information distribution in the above method, a practical implementation is also provided.

In [11], it has been proved that the routing is blocked under an MCC model if and only if the source is inside the forbidden region of one MCC while the destination is inside its critical region (see Figure 4 (a)). To simplify the discussion, we focus on the situation when the Manhattan routing is blocked in the $+Y$ direction, i.e., $s \in R_Y \wedge d \in R'_Y$. For the remaining situation, the results can be obtained by simply rotating the mesh. As shown in the sample in Figure 4 (a), had the detours along the $-X$ direction been made to node v, a shortest-path would be found in Algorithm 3. The problem with routing that does not make such detours is the lack of MCC information in the routing decision at the nodes inside of the forbidden region. In our new information model, the MCC information will broadcast to all of the nodes inside of the forbidden region so that the shortest-path can always be found.

Simply, for a certain MCC - $F(c)$ besides the $-X$ boundary initialized at the corner c - the $+X$ boundary is initialized at the opposite corner c' and propagated along line $y = y_{c'}$. If necessary, it will make a left turn to join the same $+X$ boundary for another MCC $F(v)$ at the corresponding opposite corner v'. After the joint point, the forbidden region of $F(v)$ will merge into that of $F(c)$. Figure 4 (b) shows an example of how the $+X$ boundary for $F(c)$ joins that for $F(c_2)$ $(= F(v))$ at node c'_2 $(= v)$. The area between these two boundaries is defined as the forbidden region of $F(c)$, denoted by $R_Y(c)$. Obviously, it expands the original forbidden region in [11]. When each node along the $-X$ boundary forms its triple information, it has only the stable information of the left bound of $R_Y(c)$, which is also the path of $-X$ boundary construction, but does not have the information of that expanded part. The triple is denoted by $(F(c), R_{Y_{-X}}(c), R'_Y(c))$ where $R_{Y_{-X}}(c)$ refers to the left part of the forbidden region bounded by the $-X$ boundary and line $x = x_{c'}$. Similarly, each node along the $+X$ boundary will form a triple $(F(c), R_{Y_{+X}}(c), R'_Y(c))$, where $R_{Y_{+X}}(c)$ means the right part of the forbidden region bounded by line $x = x_c$ and the $+X$ boundary. Obviously, we have $R_Y(c) = R_{Y_{-X}}(c) \cup R_{Y_{+X}}(c)$. To obtain the information of the other boundary and to identify $R_Y(c)$, the triple formed at a node along one boundary will be sent along the X dimension to reach the other boundary. At each intermediate node it passes, such information will also be sent in the $+Y$ direction. It is noted that the determined information is used in such broadcasting and each node will not accept duplicates from its neighbors. Eventually, at each node inside of the forbidden region $R_Y(c)$, the identified information $(F(c), R_Y(c), R'_Y(c))$ forms. The whole process of information propagation is shown in Algorithm 4.

Algorithm 4: Complete boundary construction and information propagation for the forbidden region $R_Y(c)$ of an MCC $F(c)$ (proposed information model $B2$, replacing Algorithm 1)

1. Apply step 1 in Algorithm 1.
2. After the triple information is identified at the opposite corner c', it will propagate as $(F(c), R_{Y_{+X}}(c), R'_Y(c))$ to build the $+X$ boundary with the construction process of $-X$ boundary in step 3 in Algorithm 1, but will always make a left turn.
3. Apply step 2 in Algorithm 1.
4. Apply step 3 in Algorithm 1 to build $-X$ boundary and form the triple $(F(c), R_{Y_{-X}}(c), R'_Y(c))$ at each boundary node.
5. Simultaneously, the triple received at each boundary node will broadcast along the X dimension and then in the $+Y$ direction, until it reaches the other boundary. For each node, we receive triples from both boundaries, and identified information $(F(c),\ R_Y(c),\ R'_Y(c))$ is formed.

Therefore, the routing decision at node s knows the existence of the blocking MCC. According to the location and the shape of such an MCC, the detour direction can be easily determined at node s: If $M(s,c) + M(c,d) \leq M(s,c') + M(c',d)$, the

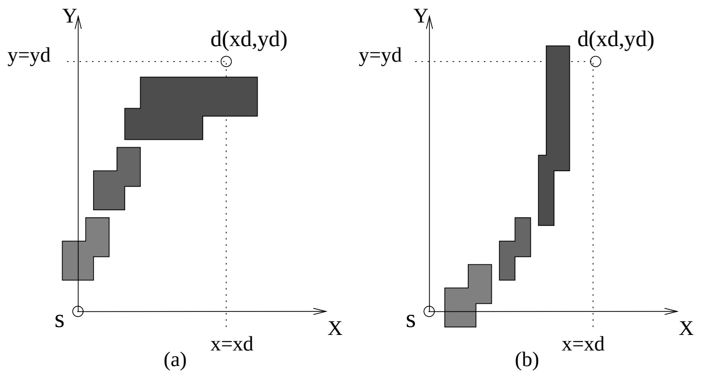


Figure 5: (a) Type-I sequence of MCC. (b) Type-II sequence of MCC.

routing detours along the $-X$ direction. Otherwise, the $+X$ direction detour is taken (see Figure 4 (b)).

As indicated in [19], the Manhattan routing is usually blocked by a sequence of MCCs, not just a single one. If the routing is blocked in the $+Y$ direction, the corresponding sequence ($F(q_1)$, $F(q_2)$, ..., $F(q_n)$) is called a type-I sequence [19] (see Figure 5). Respectively, we have a type-II sequence blocking the routing in the $+X$ direction. Let q_i and q'_i denote the initialization corner and the opposite corner of each MCC $F(q_i)$, $1 \le i \le n$, respectively. A type-I sequence can be identified by the following properties:

$$\begin{cases} (0, y_{q_1}) \in F(q_1) \wedge 0 < y_{q'_1} < y_d \\ (x_d, y_{q_n}) \in F(q_n) \wedge 0 < y_{q_n} < y_d \\ x_{q_i} \le x_{q_{i+1}} \le x_{q'_i} \\ y_{q'_i} < y_{q'_{i+1}} \\ \nexists F(v), R_Y(q_i) \subset R_Y(v) \subset R_Y(q_{i+1}) \end{cases} \tag{1}$$

In Equation 1, the first four properties guarantee $R_Y(q_i) \subset R_Y(q_{i+1})$ for any $1 \le i < n$. The last property guarantees that every shortest-path will be considered for the detour around the blocking MCCs. Under our new information model with Algorithm 4, the routing at the source s ($u = s$) knows the information of each MCC of its blocking sequence when $D(s,d) > M(s,d)$. Furthermore, it knows the information of all MCCs that block the Manhattan distance path to the destination d. Note that $F(q_1)$, $F(q_2)$, $\cdots$, $F(q_n)$ is the closest sequence, with the initialization corner q_i and the opposite corner q'_i for each each $F(q_i)$ ($1 \le i \le n$). The routing has three options to detour around the closest sequence in the shortest-path: (a) through node q_1, denoted by P_0, (b) through node q'_n, denoted by P_n, and (c) through two consecutive MCCs, $F(q_i)$ and $F(q_{i+1})$ ($1 \le i < n$), denoted by P_i. The length of its shortest-path to the destination can be easily determined in a recursive function:

$$D(u,d) = \begin{cases} \min_{0\le i\le n}\{P_i\} & \textit{the closest blocking} \\ & \textit{sequence } F(q_1), F(q_2), \cdots, \\ & F(q_n) \textit{ is found.} \\ & \\ M(u,d) & \textit{otherwise} \end{cases} \tag{2}$$

where

$$P_i = \begin{cases} M(u,q_1) & +D(q_1,d) & i=0 \\ M(u,q_i') & +M(q_i',q_{i+1}) & 1\le i<n \\ & +D(q_{i+1},d) & \\ M(u,q_n') & +D(q_n',d) & i=n \end{cases} \tag{3}$$

After that, for the shortest-path P_i, the routing message will be forwarded to the corresponding corner, q_{i+1} $(0 \le i < n)$ or q_n'. From that intermediate destination, the routing will continue until the destination d is reached. The detailed multi-phase routing is shown in Algorithm 5.

Algorithm 5: Routing at the current node u with the information of $d(x_d, y_d)$ $(RB2)$

1. If $u = d$, stop.

2. If no blocking sequence is found (with Equation 1), apply routing decision in Algorithm 2 to forward the routing message. Otherwise, apply the following to detour.

3. Among all the sequences found in step 1, identify the closest one ($F(q_1)$, $F(q_2)$, $\cdots$, $F(q_n)$).

4. Use Equation 2 to calculate the distance of the shortest-path $D(s,d)$ among the following paths: (a) through node q_1, denoted by P_0, (b) through node q_n', denoted by P_n, and (c) through two consecutive MCCs ($F(q_i)$ and $F(q_{i+1})$, $1 \le i < n$), denoted by P_i.

5. If P_0 / P_n is the selected shortest-path, apply routing decision in Algorithm 2 to reach the intermediate destination $d' = q_1$ / q_n'. Otherwise, for the selected shortest-path P_i $(1 \le i < n)$, apply routing decision in Algorithm 2 to form the Manhattan distance paths $M(s, q_i')$ and $M(q_i', q_{i+1})$ to reach the intermediate destination $d' = q_{i+1}$.

An example is shown in Figure 6. At the source s $(u = s)$, $F(c_2)$, $F(c_3)$, $F(c_4)$, and $F(c_5)$ are found in the closest sequence of MCCs ($F(q_1)$, $F(q_2)$, $F(q_3)$, and $F(q_4)$ respectively) with Equation 1. Meanwhile, the information of all MCCs ($F(c_2)$, $F(c_3)$, $F(c_4)$, $F(c_5)$, $F(c_6)$, $F(c_7)$, $F(c_8)$, $F(c_9)$) and their initialization /

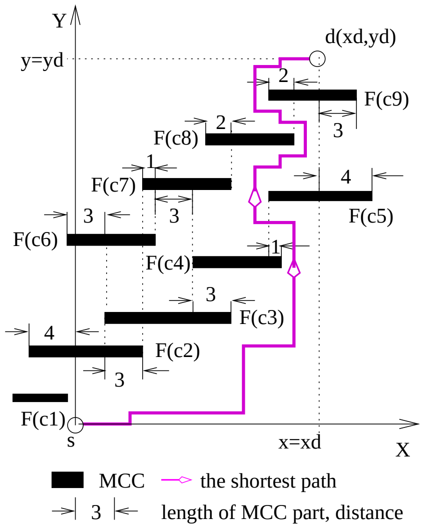


Figure 6: Multi-phase routing through the corner q_4 of $F(q_4)$, i.e., c_5 of $F(c_5)$ in figure.

opposite corners (c_2/c_2', c_3/c_3', c_4/c_4', c_5/c_5', c_6/c_6', c_7/c_7', c_8/c_8', and c_9/c_9') can be obtained because s is inside their forbidden regions. Initially, we have

$$\begin{cases} D(c_2,d) = M(c_2,d) \\ D(c_5',d) = M(c_5',d) \\ D(c_6,d) = M(c_6,d) \\ D(c_7,d) = M(c_7,d) \\ D(c_8,d) = M(c_8,d) \\ D(c_9,d) = M(c_9,d) \\ D(c_9',d) = M(c_9',d) \end{cases}$$

Then, we have

$$\begin{cases} D(c_3,d) & = M(c_3,c_6') + M(c_6',c_7) + D(c_7,d) = M(c_3,d) + 1 * 2 \\ D(c_4,d) & = M(c_4,c_7') + M(c_7',c_8) + D(c_8,d) = M(c_4,c_8') + M(c_8',c_9) + D(c_9,d) \\ & = M(c_4,d) + 2 * 2 \\ D(c_5,d) & = M(c_5,c_8') + M(c_8',c_9) + D(c_9,d) = M(c_5,d) + 2 * 2 \end{cases}$$

By calling the recursive function in Equation 2, the shortest-path P_3 to detour around that sequence of MCCs is determined because $M(s,c_4')+M(c_4',c_5)+D(c_5,d) = M(s,d) + 6$ is the minimum. That is, the routing will form the paths $M(s,c_4')$ and $M(c_4',c_5)$ to reach the intermediate destination c_5. After that, by repeating the above process at node c_5, the routing will find a path to d passing nodes c_8' and c_9. The recursive function guarantees that the entire path is the shortest-path in such a multi-phase routing. The following theorem ensures there is no path shorter than the one found in $RB2$ routing.

Theorem 1: *For a given pair of the safe source and the safe destination under*

MCC model, the $RB2$ routing in Algorithm 5 will find a path between them if such a path exists and there is no path shorter than this one.

Proof. Assume that s and d are not disconnected by faults. If a Manhattan distance path between s and d exists, such a path can be found easily via step 2 of $RB2$ routing by applying routing decision in Algorithm 2. Otherwise, at least one sequence of either type-I or type-II can be found between s and d. Because s and d are safe, we cannot find both kinds of sequences. By applying the recursive function D in Equation 2, the path that detours around such a sequence can be found via step 5. In this way, the routing can advance in the blocking direction. Because the distance from s and d in such a blocking direction is fixed, the multi-phase routing will reach the node having Manhattan distance path to d and will eventually arrive at node d.

The routing path found in Algorithm 5 only has to detour around the MCCs in the blocking sequence(s). If a shorter path from s to d exists, such a path must pass an active but MCC-unsafe node. Because s and d are all safe, using any node inside an MCC will definitely cause a detour. In other words, we must have a shorter path only using safe nodes. This contradicts with the results of Equation 2. □

The above shortest-path routing method requires information broadcasting. To reduce the broadcast overhead, a more practical information model is proposed for our shortest-path routing. In this extended model, the information of each MCC $F(c)$ is only propagated to the boundary nodes. The propagation will be split into two when it intersects with another MCC $F(v)$, instead of just making a turn. These two split propagations will route around $F(v)$, one in the clockwise direction and the other in the counter-clockwise direction. After that, the first one will merge to the $+X$ boundary for $F(v)$ at its opposite corner v'. The second one will merge to the $-X$ boundary for $F(v)$ at its initialization corner v. When this is the first intersection of the $-X$ boundary and $x_c > x_{v'}$, at that split point, $F(c)$ will be identified as a candidate of the succeeding MCC of $F(v)$ in its type-I sequence. The relation $F(v) \rightarrow F(c)$ will also be sent by each split propagation. It is noted that the region information, $R_Y(c)$ and $R'_Y(c)$, is only needed and forwarded by those nodes along the $-X$ boundary for $F(c)$. The details of our extended information model are shown in Algorithm 6.

Algorithm 6: Information propagation for the forbidden region $R_Y(c)$ of an MCC $F(c)$ (extended information model $B3$)

1. Apply steps 1 and 2 in Algorithm 1.
2. Apply step 3 in Algorithm 1 to construct the $-X$ boundary.
3. When the propagation intersects with another MCC, say $F(v)$, the shape information $F(c)$ will be propagated in the way of the $+X$ boundary through its opposite corner v' (as shown in step 2 in Algorithm 4).
4. If the above intersection is the first one of $-X$ boundary and $x_c > x_{v'}$, $F(c)$ might be the succeeding MCC of $F(v)$ in a type-I sequence. Thus, the relation

$F(v) \rightarrow F(c)$ is sent in both the above propagations (steps 2 and 3) from that intersection.

At the initialization corner c of an MCC $F(c)$, among all relation records received $I(c) = \{F(c) \rightarrow F(v)\}$, the succeeding MCC of $F(c)$ in a type-I sequence, $F(w)$, can be determined by:

$$\begin{cases} F(c) \rightarrow F(w) \in I(c) \\ \forall F(\tau) \in \{F(\zeta) \mid F(c) \rightarrow F(\zeta) \in I(c)\}, y_w \leq y_\tau \end{cases} \tag{4}$$

Based on the propagation process in Algorithm 6, a boundary node for $F(c)$ will obtain the same relation information as its initialization corner c does. That is, this $F(w)$ can also be determined at any boundary node for $F(c)$. Therefore, at any boundary node δ, with the information of the destination node d, a type-I blocking sequence $(F(q_1), F(q_2), \cdots, F(q_n))$ can be determined by:

$$F(q_i) = \begin{cases} F(\alpha),\ x_\alpha < x_\delta < x_{\alpha'},\ y_\delta < y_{\alpha'} & i = 1 \\ \textit{the succeeding MCC of } F(q_{i-1}) & i > 1 \\ \textit{determined in Equation } 4 & \\ F(\beta), x_\beta < x_d < x_{\beta'},\ y_\beta < y_d & i = n \end{cases} \tag{5}$$

Therefore, the routing decision at boundary node δ can have the knowledge of all the blocking sequences and the corresponding MCC shape information. By using the same strategy in $RB2$ routing in Algorithm 5, the routing will find a path to detour the blocking sequence and then reach the destination. The following theorem shows that such a path from δ to d can be guaranteed as well as the one obtained with Algorithm 5. The detailed routing based on our extended information model, $RB3$, is shown in Algorithm 7.

Theorem 2: *When the source is a boundary node, the path found in $RB3$ routing in Algorithm 7 will not be longer than that found in $RB2$ routing in Algorithm 5.*

Proof. If the Manhattan distance path exists between s and d, both Algorithm 5 and Algorithm 7 will apply the routing decision in Algorithm 2 to find it. Otherwise, the information propagation in Algorithm 6 ensures the same blocking sequence can be identified in Equation 1 and Equation 5. By applying Equation 2, these two routings will find the same shortest-path. □

Algorithm 7: Routing at the current node u with the information of $d(x_d, y_d)$ ($RB3$)

1. Same as step 1 in Algorithm 5.

2. With Equation 5, find the closest sequence blocking the Manhattan distance path from u to d.

3. If such a sequence is not found, apply the routing decision in Algorithm 2 to forward the routing message. Otherwise, apply the following detour.

4. For each MCC $F(\tau)$ in the sequence found in the above step, with the initialization corner τ and the opposite corner τ', find the sequence blocking the Manhattan distance path from τ to d or the Manhattan distance path from τ' to d (with Equation 5).

5. Repeat step 3 until there is no new sequence identified.

6. Same as step 4 in Algorithm 5.

7. Same as step 5 in Algorithm 5.

As in the sample in Figure 6, s is at a boundary for MCC $F(c_1)$. With the information collected, the closest blocking MCC $F(c_2)$ and its succeeding MCCs $F(c_3)$, $F(c_4)$, and $F(c_5)$ can be determined as the closest blocking sequence. After that, it is identified that no sequence blocks the path from the initialization corner c_2 to d. Meanwhile, the sequence $(F(c_6), F(c_7), F(c_8), F(c_9))$ blocking the path from the initialization corner c_3 to d is identified. This process will continue until the Manhattan distance paths $M(c_6, d)$, $M(c_7, d)$, $M(c_8, d)$, $M(c_9, d)$ and $M(c_9', d)$ can be identified. That is, no new MCC blocks the routing path. Then, by applying Equation 2, the distance of shortest-path $D(s, d)$ can be determined. Furthermore, the corresponding intermediate destination can be found and the routing message will be forwarded along the shortest-path in multiple phases.

If node s is not on any boundary line and is inside the forbidden region of an MCC, the above routing will miss the shortest-path in the case P_0; that is, the corresponding Algorithm 7 cannot always find the shortest path from a source to its destination. However, if each MCC can be controlled within a certain size, the routing will quickly reach the boundary line. That is, the number of detours is limited and the length of routing path is very close to the minimum. Moreover, the more MCCs that appear in the mesh, the greater our chances will be that our routing finds the shortest-path among the remaining n cases from P_1 to P_n. Considering the communication cost saved under the extended information model, such a sub-optimal routing using only the boundary information is preferred since the performance is still acceptable. In the next section, we use the experimental results to illustrate the substantial improvement of the extended information model on communication cost in terms of the number of nodes involved in the information propagation. Our experimental results also show the acceptable performance of routing under the extended information model in terms of (a) the success rate of the shortest-path routing, and (b) the relative error of the average length of routing path to the optimal result. These results will be compared with the best results known to date in [2] and [11].

4. Simulation

In this section, we verify the improvement of our information-based routing on the ability of achieving the shortest-path from a simulator, compared with the best results known to date. Such experimental results prove the effectiveness of our

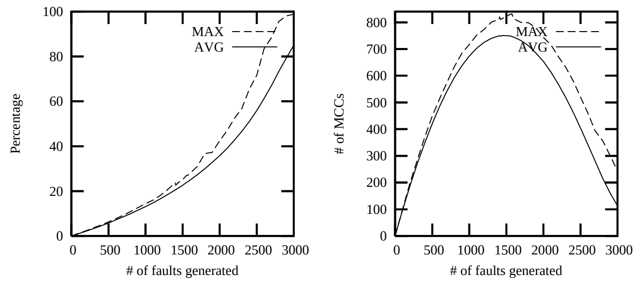


Figure 7: Network configuration under MCC model. (a) Percentage of disabled area to the total area of the meshes, and (b) number of MCCs.

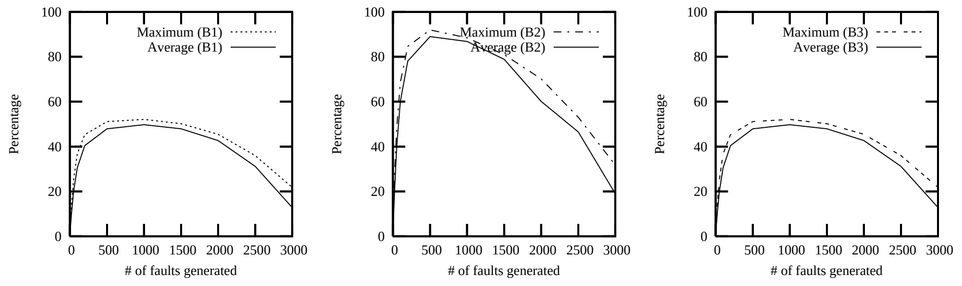


Figure 8: Percentage of nodes involved in information propagation to the total safe nodes in the meshes.

information models in terms of the success rate of finding a shortest-path, and the average length of routing paths. The simulator also compares the implementation of our information model and its extension on the number of nodes involved in the information distribution (i.e., the cost of information model). The results show that the routing using only boundary information is cost-effective.

This simulation is conducted on a 100×100 mesh with the number of faulty nodes randomly generated. It is noted that when more than 3000 faults occur in the mesh, the entire network will be disabled under the MCC model. To have a fair comparison, we only show the results when the number of faults is no more than 3000. The simulator starts from building MCCs with a given number of random faulty nodes. We take as many as 10,000 independent samples to determine the average MCC configuration, which is shown in Figure 7. Then, in each configuration case, we implement the boundary information model $B1$[11], the proposed information model $B2$, and its extension model $B3$. After that, we test the corresponding routings $RB1$, $RB2$ and $RB3$ respectively, by randomly picking the source and destination and conducting the routings 1,000 times. We assume that the source has the path to the destination. Thus, we only conduct the test in the configurations when the entire mesh is not disconnected by faults.

Figure 8 shows the percentage of the number of nodes involved in the information

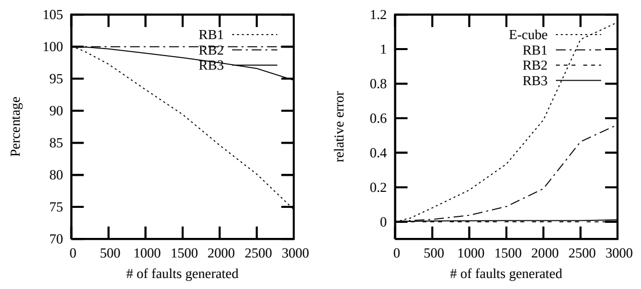


Figure 9: (a) Percentage of success in finding the shortest-path. (b) Relative error of routing path achieved to the shortest-path.

propagation to the total safe nodes in the meshes in the information models $B1$, $B2$, and $B3$ (i.e., cost of information models). The results show that $B2$ has the highest communication cost. However, it is still not as expensive as using global information. It is noted that when the entire mesh has up to 100 MCCs, the information only needs to broadcast to 20% of the safe nodes. $B1$ has the lowest communication cost. The results of $B3$ are very close to those of $B1$ because in most cases, the $+X$ boundary for one MCC shares the nodes with the $-X$ boundaries for other MCCs. Figure 9 (a) shows the percentage of successful shortest-path conducted in different routings $RB1$, $RB2$ and $RB3$. The results show that with the information broadcasting, the routing $RB2$ always achieves the shortest-path (= 100%). With the help of only $-X$ boundaries, the routing $RB1$ can successfully find the shortest-path in more than 75% of all cases. With the information model $B3$ proposed in this paper, the corresponding routing $RB3$ can find the shortest-path in more than 95% of all cases. However, under the $B3$ model, there is no need for information broadcasting. Figure 9 (b) shows the comparison of length of routing path achieved in each routing with the optimal result, i.e., the length of the shortest-path. It shows that the routing $RB1$ will experience many detours in cases where the Manhattan distance path does not exist. The average length of the routing path is very close to that of E-cube routing in [2], which only requires the information of neighbors. Under the proposed information model $B2$, the corresponding routing $RB2$ will guarantee the shortest-path (relative error = 0). Under the proposed extension model $B3$, the routing $RB3$ will find the shortest-path in most cases, and for those non-shortest-path cases, the number of detours is limited. The results are very close to the optimal ones. This figure supports our statement on the significance of our new information model and its practical implementation on achieving the shortest-path routing.

5. Conclusion

In this paper, we have proposed a fully-distributed process to collect and distribute MCC block information in 2-D meshes for the corresponding information-based routing to achieve the shortest-path via multiple Manhattan routing phases.

The use of an MCC model guarantees that there is no existing path shorter than the one we found. A practical implementation of such an information model with a low number of nodes involved in the information propagation along the boundary lines is also presented. The simulation results have shown the substantial improvement of our methods on achieving the shortest-path routing in terms of the success rate and the average path length. In our future work, we will also extend our results to higher dimension networks and networks with irregular topologies.

Acknowledgements

This research was supported in part by NSF grants CCR 0329741, CNS 0422762, CNS 0434533, CNS 0531410, and CNS 0626240.

References

1. Definition of the Manhattan distance, document is available at *http://www.nist.gov/dads/HTML/manhattanDistance.html* .
2. R. V. Boppana and S. Chalasani, "Fault tolerant wormhole routing algorithms for mesh networks," *IEEE Transactions on Computers* **C-44** (1995) 848–864.
3. Y. M. Boura and C. R. Das, "Fault-tolerant routing in mesh networks," *Proc. of 1995 International Conference on Parallel Processing*, August 1995, pp. 106–109.
4. S. Chalasani and R. V. Boppana, "Communication in multicomputers with nonconvex faults," *IEEE Transactions on Computers* **C-46** (1997) 616–622.
5. A. A. Chien and J. H. Kim, "Planar-adaptive routing: low-cost adaptive networks for multiprocessors," *Journal of ACM* **C-42** (1995) 91–123.
6. W. J. Dally, "The J-machine: System support for Actors," in *Actors: Knowledge-Based Concurrent Computing*, eds. Hewitt and Agha (MIT Press, 1989).
7. A. Gara and et al, "Overview of the Blue Gene/L system architecture," *IBM Journal of Research and Development* **C-49** (2005) 195–212.
8. G. Glass and L. Ni, "Maximally fully adaptive routing in 2D meshes," *Proc. of ICPP'92*, 1992, pp. 101–104.
9. M. Gomez, N. Nordbotten, J. Flich, P. Lopez, A. Robles, J Duato, T. Skeie and O. Lysne, "A routing methodology for achieving fault tolerance in direct networks," *IEEE Transactions on Computers* **C-55** (2006) 400–415.
10. R. L. Hadas and E. Brandt, "Origin-based fault-tolerant routing in the mesh," *Future Generation Computer Systems* **C-11** (1995) 603–615.
11. Z. Jiang, J. Wu and D. Wang, "A new fault information model for fault-tolerant adaptive and minimal routing in 3-D meshes," *Proc. of ICPP'05*, 2005, pp. 500–507.
12. R. K. Koeninger, M. Furtney and M. Walker, "A shared memory MPP from Cray research," *Digital Technical Journal* **C-6** (1994) 8–21.
13. S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä and A. Hemani, "A network on chip architecture and design methodology," *Proc. of VLSI Annual Symposium (ISVLSI'02)*, 2002, pp. 105–112.
14. V. Puente, J. Gregorio, R. Beivide and F. Vallejo, "A low cost fault tolerant packet routing for parallel computers," *Proc. of the Int'l Parallel and Distributed Processing Symposium (IPDPS'03)*, 2003, CD-ROM.
15. M. Schäfer, T. Hollstein, H. Zimmer and M. Glesner, "Deadlock-free routing and

component placement for irregular mesh-based networks-on-chip," *Proc. of 2005 IEEE/ACM Int'l Conf. on Computer-aided Design*, 2005, pp. 238–245.

16. L. Sheng and J. Wu, "Maximum-shortest-path (MSP) is not optimal for a general N $\times$ N torus," *IEEE Transactions on Reliability* **C-52** (2003) 22–25.
17. C. C. Su and K. G. Shin, "Adpative fault-tolerant deadlock-free routing in meshes and hypercubes," *IEEE Transactions on Computers* **C-45** (1996) 672–683.
18. M. Taylor and et al, "The raw microprocessor: a computational fabric for software circuits and general-purpose programs," *IEEE Micro* **C-22** (2002) 25–35.
19. D. Wang, "A rectilinear-monotone polygonal fault block model for fault-tolerant minimal routing in meshes," *IEEE Transactions on Computer* **C-52** (2003).
20. D. Wiklund and D. Liu, "SOCBUS: switched network on chip for hard real time embedded systems," *Proc. of the Int'l Parallel and Distributed Processing Symposium (IPDPS'03)*, 2003, CD-ROM.
21. J. Wu, "Fault-tolerant adaptive and minimal routing in mesh-connected multicomputers using extended safety levels," *IEEE transactions on Parallel and Distributed Systems* **C-11** (2000) 149–159.
22. J. Wu, "Maximum-shortest-path (MSP): an optimal routing policy for mesh-connected multicomputers," *IEEE Transactions on Reliability* **C-48** (1999) 247–255.
23. J. Wu and Z. Jiang, "Extended minimal routing in 2-D meshes with faulty blocks," *Proc. ICDCSW'02*, 2002, pp. 49–56.
24. D. Xiang, "Fault-tolerant routing in hpercube multicomputers using local safety information," *IEEE Transactions on Parallel and Distributed Systems* **C-12** (2001) 942–951.
25. D. Xiang, A. Chen and J. Wu, "Reliable broadcasting in wormhole-routed hypercube-connected networks using local safety information," *IEEE Transactions on Reliability* **C-52** (2003) 245–256.
26. J. Zhou and F. Lau, "Fault-tolerant wormhole routing in 2D meshes," *Proc. of ISPAN'00*, 2000, pp. 94–101.
27. J. Zhou and F. Lau, "Multi-phase minimal fault-tolerant wormhole routing in meshes," *Parallel Computing* **C-30** (2004) 423–442.