Capacity Provisioning and Failure Recovery in Mesh-Torus Networks with Application to Satellite Constellations

Jun Sun and Eytan Modiano
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
{junsun, modiano}@mit.edu

Abstract—**This paper considers the link capacity requirement for a $N \times N$ mesh-torus network under a uniform all-to-all traffic model. Both primary capacity and spare capacity for recovering from link failures are examined. In both cases, we use a novel method of "cuts on a graph" to obtain lower bounds on capacity requirements and subsequently find algorithms for routing and failure recovery that meet these bounds. Finally, we quantify the benefits of path based restoration over that of link based restoration; specifically, we find that the spare capacity requirement for a link based restoration scheme is nearly N times that for a path based scheme.**

I. Introduction

The total capacity required by a satellite network to satisfy the demand and protect it from failures contributes significantly to its cost. To maximize the utilization of such a network, we explore the minimum amount of spare capacity needed on each satellite link, so as to sustain the original traffic flow during the time of a link failure. In general, for a link failure, restoration schemes can be classified as link based restoration, or path based restoration. In the former case, affected traffic (i.e. traffic that is supposed to go through the failed link) is rerouted over a set of replacement paths through the spare capacity of a network between the two nodes terminating the failed link. Path restoration reroutes the affected traffic over a set of replacement paths between their source and destination nodes [1, 2, 3, 5, 6]. The obvious advantages of using the link restoration strategy are simplicity and ability to repidly recover from failure events. However, as we will show later, the amount of spare capacity needed for the link based scheme is significantly greater than that of path based restoration since the latter has the freedom to reroute the complete source-destination using the most efficient backup path. On the other hand, the path restoration scheme is less flexible in handling failures [1, 2, 3].

This work was supported by DARPA under the Next Generation Internet initiative.

We investigate the optimal spare capacity placement problem based on mesh-torus topology which is essential for the multisatellite systems. An $n \times n$ mesh-torus is a two-dimensional (2-D) n-ary hypercube and differs from a binary hypercube in that each node has a constant number of neighbors (4), regardless of n. For the remainder of the paper, we will refer to this topology simply as a mesh. In particular, we are interested in the scenario where every node in the network is sending one unit of traffic to every other node (also known as *complete exchange* or *all-to-all communication*) [7]. This type of communication model is considered because the exact traffic pattern is often unknown and an all-to-all model is frequently used as the basis for network design. Even in the case of a predictable traffic pattern, links of a particular satellite will experience different traffic demand as the satellite flies over different location on earth. Thus, each link of that satellite must satisfy the maximum demand. Again, all-to-all traffic model helps capturing this effect. Hence we also assume that each satellite link has an equal capacity. Our results, while motivated by satellite networks [9, 10, 11], are equally applicable to other networks with a mesh topology such as multi-processor interconnect networks [12, 13, 14] and optical WDM mesh networks [2, 3]. Furthermore, while our results are discussed in the context of an $n \times n$ mesh for simplicity, they can be trivially extended to a more general $n \times m$ topology.

When using the path restoration schemes, the restoration can be performed at the global level by rerouting all the traffic (both those affected or unaffected by the link failure) in a network. However, this level of restoration requires recomputing a new path for each source-destination pair, thus it is impractical if a restoration time limit is imposed or when disruption of existing calls is unacceptable. We can also perform path restoration at the local level by rerouting only the traffic which is affected by the link failure. Obviously, the local level reconfiguration will require

	No restoration	Link based restoration	Path based restoration
Total Capacity (N odd)	$\frac{N^3-N}{4}$	$\frac{N^3-N}{3}$	$\frac{N^2(N^2-1)}{2(2N-1)}$
Total Capacity (N even)	$\frac{N^3}{4}$	$\frac{N^3}{3}$	$\frac{N^4}{2(2N-1)}$
Spare Capacity (N odd)	0	$\frac{N^3-N}{12}$	$\frac{N^3-N}{4(2N-1)}$
Spare Capacity (N even)	0	$\frac{N^3}{12}$	$\frac{N^3}{4(2N-1)}$

TABLE I
CAPACITY REQUIREMENTS UNDER LINK BASED AND PATH BASED RESTORATION.

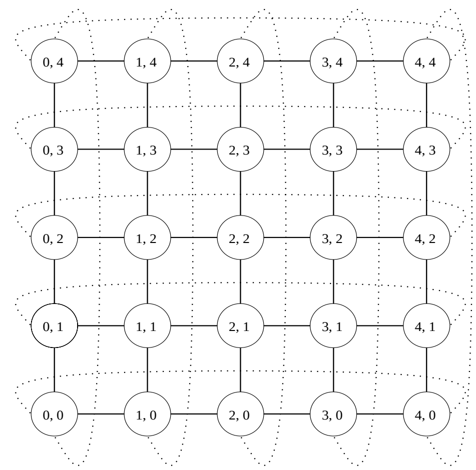

Fig. 1. A 2-dimensional 5-mesh.

at least as much spare capacity as the global level reconfiguration since the former is a subset of the latter. Nevertheless, as we show in section IV, the lower bound on the spare capacity needed, using global level reconfiguration, can be achieved by using local level reconfiguration.

To obtain the necessary minimum spare capacity, our approach is to first find the minimum capacity, say C_1, that each link must have in order to support the all-to-all traffic. We then obtain a lower bound, C_2, for the capacity needed on each link to satisfy the all-to-all traffic when one of the links fails. Consequently, the minimum spare capacity needed, C_{spare}, should be greater than the difference of C_2 and C_1. Since we do not restrict the reconfiguration (global level or local level) used to calculate C_2; $C_2 - C_1$ is a lower bound on C_{spare}, both at global level and local level. We will show that this lower bound on C_{spare} is achievable by using a path based restoration algorithm at a local level. Thus, the minimum spare capacity needed using path restoration strategy is C_{spare}. Table I summarizes capacity requirements under link based and path based restoration.

Communication on a mesh network has been studied in [4, 11, 14]. In [4], the authors consider processors communicating over a mesh network with the objective of broadcasting information. The work in [11] presents routing algorithm generating minimum propagation delay for satellite mesh networks. In [14], the authors propose new algorithms for all-to-all personalized communication in mesh-connected multiprocessors. These papers mentioned so far did not look into capacity provisioning and spare capacity requirement of the mesh network.

Path based and link based restoration schemes have been extensively researched [1, 2, 3, 5]. In [1], the authors study and compare spare capacity needed by using link based and path based schemes. The work of [5] provides a method for capacity optimization of path restorable networks and quantify the capacity benefits of path over link restoration. In [2, 3], the authors examines different approaches to restore mesh-based WDM optical networks from single link failures. In all the aforementioned papers, the spare capacity problem is formulated as an integer linear programming problem which is solved by standard methods. Our paper addresses the mesh structure for which we can get a closed form results for the spare capacity.

The structure of this paper is as follows: Section II gives necessary definitions and statement of the problem. In section III, a lower bound on C_1 is given along with a routing algorithm achieving this lower bound. The lower bound C_2 is presented also. We then show in section IV that the lower bound on C_{spare}, $C_2 - C_1$, can be achieved by a path based restoration algorithm. Section V concludes this paper.

II. PRELIMINARIES

We start out with a description of the network topology and traffic model, and follow it with a sequence of formal definitions and terminology that will be used in subsequent sections.

Definition *1:* The 2-dimensional N-mesh is an undirected graph $G = (V, E)$, with vertex set

$$V = \{\vec{a} \mid \vec{a} = (a_1, a_2) \text{ and } a_1, a_2 \in \mathcal{Z}_N\},$$

where $\mathcal{Z}_N$ denotes the integers modulo N, and edge set

$$E = \{(\vec{a}, \vec{b}) \mid \exists\, j \text{ such that } a_j \equiv (b_j \pm 1) \; mod \; N \text{ and } a_i = b_i \text{ for } i \neq j, i, j \in \{1, 2\}\}.$$

The above definition is from [7]. A 2-dimensional N-mesh has a total of N^2 nodes. Each node has two neighbors in the vertical and horizontal dimension, for a total of four neighbors. We associate each satellite with a fixed node, (a_1, a_2), in the mesh. Undirected edges of the mesh are also referred to as links. Fig. 1 shows a 2-dimensional 5-mesh. The notion 2-dimensional ∞-mesh is used to denote the case where N is arbitrarily large, and it is the same as an infinity grid.

***Definition** 2:* A cut $(S, V - S)$ in a graph $G = (V, E)$ is partition of the node set V into two nonempty subsets, a set S and its complement $V - S$.

Here the notation Cut-Set$(S, V - S) = \{(\vec{a}, \vec{b}) \in E \mid \vec{a} \in S, \vec{b} \in V - S\}$ denotes the set of edges of the cut (i.e. the set of edges with one end node in one side of the cut and the other on the other side of the cut).

***Definition** 3:* The size of a Cut-Set$(S, V - S)$ is defined as $C(S, V - S) = \mid$ Cut-Set$(S, V - S) \mid$.

For $G = (V, E)$ and $\mathcal{P}(V)$ denote the power set of the set V (i.e. the set of all subsets of V). Let $\mathcal{P}_n(V)$ denote the set of all n-elements subsets of V.

***Definition** 4:* Let $G = (V, E)$ be a 2-dimensional N-mesh, the function $\varepsilon_N : \mathcal{Z}^+ \rightarrow \mathcal{Z}^+$ is defined as

$$\varepsilon_N(n) = \min_{S \in \mathcal{P}_n(V)} C(S, V - S).$$

The function $\varepsilon_N(n)$ returns the minimum number of edges that must be removed in order to split the 2-dimensional N-mesh into two parts, one with n nodes and the other with $N^2 - n$ nodes. Similarly, $\varepsilon_\infty(n)$ is defined to be the minimum number of edges that must be removed in order to split the ∞-mesh into two disjoint parts, one of which containing n nodes.

To achieve the minimum spare capacity, we consider the shortest path algorithm. Shortest paths on 2-dimensional N-mesh are associated with the notion of *cyclic distance* which we will define next [8].

***Definition** 5:* Given three integers, i, j, N, the cyclic distance between i and j modulo N is given by

$$D_N(i, j) = \min\{(i - j) \; mod \, N), (j - i) \; mod \, N)\}.$$

III. Capacity Requirement without Link Failures

To obtain the necessary capacity, C_1, that each link must have in order to support the all-to-all traffic without link failure, we first provide a lower bound on C_1. An algorithm achieving the lower bound will also be presented. For the proof of the lower bound on C_1, we are aware of the existance of a simpler proof (using Proposition 1 in [4]) than the one we described below. However, the cut method we used here will help us find the lower bound, C_2, on the minimum capacity needed on each link in the event of a link failure. Therefore, we decide to use the same cut method consistently in proving the lower bound on C_1 and the lower bound C_2.

A. A Lower Bound on the Primary Capacity

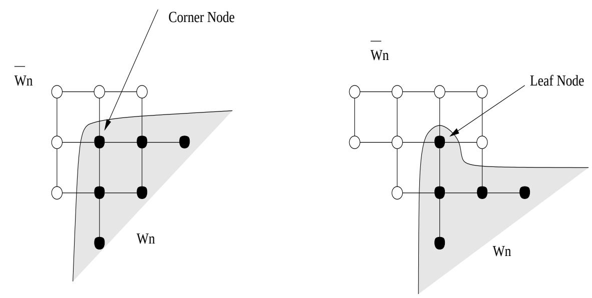

Fig. 2. Representation of corner node and leaf node.

To find a lower bound on C_1, we state the following lemmas which will prove to be useful tools in the subsequent sections. First, we give a brief explanation of the terminology and notation used in the lemmas and their proofs. For $G = (V, E)$ defined as an infinite mesh, an *inner edge (i, j)* of a set $W \subset V$ is $(i, j) \in E$ such that $i \in W$ and $j \in W$. A *corner node x* of the set W is defined to be a node $x \in W$ such that two of its four neighboring nodes are also in the set W while the other two are in $\overline{W}$. And of those two neighboring nodes in W, they form a $90°$ angle with respect to node x (as shown in Fig. 2). Similarly, a *leaf node x* of set W is defined to be a node $x \in W$ such that three of its four neighboring nodes are in $\overline{W}$, and the last one is in W. When all nodes in W are connected, we use the term *shape of the set W* to refer to the collective shape of nodes in W. For example, we say that the shape of the set shown in Fig. 3(a) is square and the shape of the set in Fig. 3(b) is rectangular. Lastly, we use the term *minimum set* W_n to refer any set such that $C(W_n, \overline{W_n}) = \varepsilon_\infty(n)$.

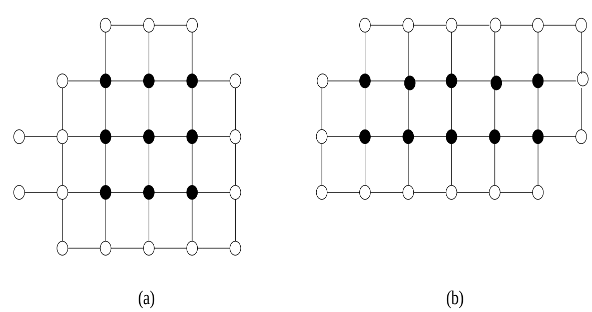

Fig. 3. An illustration of the square shape and the rectangular shape.

***Lemma** 1:* Let $G = (V, E)$ be an infinite mesh. An arbitrary set $W_n \in V$ such that $\varepsilon_\infty(n) = C(W_n, \overline{W_n})$ must satisfy the following properties:

1. $\forall x \in W_n, \exists \, y \in W_n$ such that $(x, y) \in E$. In other words, nodes in W_n should be connected.

2. Nodes in W_n should be clustered together to form a rectangular shape (including square) if possible.
3. $\varepsilon_\infty(n)$ is an even number for all $n \in \mathcal{Z}^+$.
4. $\varepsilon_\infty(n)$ is a monotonically nondecreasing function of n.

Proof: Property (1) is easy to show. If there exists a node $s \in W_n$ such that s is not connected to any other nodes in W_n, simply discarding s and adding a new node which is connected to nodes of W_n will result in a smaller $C(W_n, \overline{W_n})$, a contradiction to the definition of $\varepsilon_\infty(n)$.

To show (2), suppose the set W_n is not clustered together to form a rectangular shape, then by grouping nodes into rectangle will decrease $C(W_n, \overline{W_n})$. Again, we have a contradiction.

Property (3) is true because we have $C(W_n, \overline{W_n}) = 4n - 2$(number of inner edge in W_n), for any set of W_n. Therefore, $\varepsilon_\infty(n)$ will always be an even number.

To show that $\varepsilon_\infty(n)$ is a nondecreasing function, suppose there exists $k \in \mathcal{Z}^+$ such that $m_1 = \varepsilon_\infty(k+1) < \varepsilon_\infty(k) = m_2$ where $\varepsilon_\infty(k+1) = C(W_{k+1}, \overline{W_{k+1}})$. The set W_{k+1} must contain a corner node, say a; or a leaf node, say b. If node a or node b is removed from W_{k+1}, the resulting set, say W'_k, will have k nodes remaining. We get $C(W'_k, \overline{W'_k}) \leq m_1$ which contradicts the fact that $\varepsilon_\infty(k) = m_2 > m_1$. Thus, property (4) is true. ∎

***Lemma** 2:* Let $G = (V, E)$ be an infinite mesh, then

$$\varepsilon_\infty(n^2) = 4n$$

and

$$\varepsilon_\infty(n^2 + k) = \begin{cases} 4n + 2 & \text{for} \quad 1 \leq k \leq n \\ 4n + 4 & \text{for} \quad n + 1 \leq k \leq 2n + 1 \end{cases}$$

for $n, k \in \mathcal{Z}^+$ where $\mathcal{Z}^+$ denotes the set of positive integer.

The above lemma gives the minimum number of edges that must be removed from E in order to split a specified number of nodes from the mesh. Intuitively, the set of n nodes to be removed from the mesh must be clustered together.

Proof: We will show $\varepsilon_\infty(n^2) = 4n$, $\forall n \in \mathcal{Z}^+$, and the set of n^2 nodes must be arranged in a square shape in order to achieve the minimum size of the cut. From the properties of the minimum set in the previous lemma, we know the minimum set has to be clustered in a rectangular shape. Suppose we have a set of n^2 nodes arranged in the rectangular form shown in Fig. 4. We know that $ab = n^2$ for some $a, b \in Z$ and size of the cut is $2(a+b)$. Minimizing the size of the cut results in $a = b = n$. The uniqueness of a square configuration can be shown by inspection. To show that $\varepsilon_\infty(n^2 + k) = 4n + 2$ for $1 \leq k \leq n$, we prove

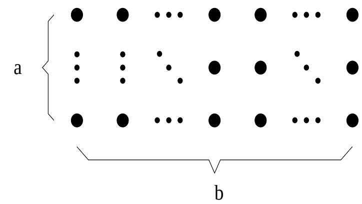

Fig. 4. An arrangement of n^2 nodes in rectangular shape.

that $\varepsilon_\infty(n^2 + k) \geq 4n + 2$ for $1 \leq k \leq n$. Then, by construction, $\varepsilon_\infty(n^2 + k) = 4n + 2$ for $1 \leq k \leq n$. From property (4) and the uniqueness of the square configuration, we see that $\varepsilon_\infty(n^2 + 1) > \varepsilon_\infty(n^2) = 4n$. From property (3), $\varepsilon_\infty(n^2 + 1) \neq 4n + 1$. Therefore, $\varepsilon_\infty(n^2 + 1) \geq 4n + 2$. By the monotonicity of $\varepsilon_\infty(\cdot)$, $\varepsilon_\infty(n^2 + k) \geq 4n + 2$ for $1 \leq k \leq n$. To show achievability, we first arrange the n^2 nodes in square. Then, connecting the extra k nodes around the square will yield $\varepsilon_\infty(n^2 + k) = 4n + 2$ for $1 \leq k \leq n$.

Showing that $\varepsilon_\infty(n^2 + k) = 4n + 4$ for $n + 1 \leq k \leq 2n + 1$ can be done similarly. ∎

***Corollary** 1:* For $\varepsilon_\infty(n)$ defined in above lemma, $\varepsilon_\infty(n) \geq 4\sqrt{n} \quad \text{for} \quad n \in \mathcal{Z}^+$.

Proof: The statement is obviously true for n such that $n = k^2$ for some $k \in \mathcal{Z}^+$. Now consider the case where $n \neq k^2$ for $\forall k \in \mathcal{Z}^+$. Let m be the largest integer such that $m^2 < n$. From Lemma 1, we then have

$$\begin{aligned} n - m^2 > m \quad &\Rightarrow \quad \varepsilon_\infty(n) = 4m + 4 \\ n - m^2 < m \quad &\Rightarrow \quad \varepsilon_\infty(n) = 4m + 2 \end{aligned}$$

So for n such that $(m+1)^2 > n > m^2 + m$, we have $4m + 4 = 4\sqrt{(m+1)^2} > 4\sqrt{n}$. Similarly, for n such that $m^2 + m > n > m^2$, we have $4m + 2 = 4\sqrt{(m + \frac{1}{2})^2} > 4\sqrt{m^2 + m} > 4\sqrt{n}$. Thus, $\varepsilon_\infty(n) \geq 4\sqrt{n} \quad \text{for} \quad n \in \mathcal{Z}^+$. ∎

***Corollary** 2:* Let $G = (V, E)$ be an infinite mesh with an arbitrary link failure, then

$$\varepsilon_\infty(n^2) = 4n - 1$$

and

$$\varepsilon_\infty(n^2 + k) = \begin{cases} 4n + 1 & \text{for} \quad 1 \leq k \leq n \\ 4n + 3 & \text{for} \quad n + 1 \leq k \leq 2n + 1 \end{cases}$$

for $n, k \in \mathcal{Z}^+$ where $\mathcal{Z}^+$ denotes the set of positive integer.

Proof: The proof of this corrollary follows similar steps to those used in the proof of the lemma. By including the failed link in the cut set, the number of edges needed to be removed for this new topology is one less than that of regular infinite mesh (without link failure). ■

So far the function $\varepsilon_\infty(n)$ has been the focus of our discussion. Since the satellite network that we model is a 2-dimensional N-mesh, it is essential to know $\varepsilon_N(n)$. In a 2-dimensional N-mesh, a horizontal row of nodes (a vertical column of nodes) forms a horizontal (vertical) ring. When n is very small compared to N, splitting a set of n nodes from the N-mesh is similar to cutting the set of n nodes from ∞-mesh; more precisely, $\varepsilon_\infty(n) = \varepsilon_N(n)$. The ring structure of the 2-dimensional N-mesh does not affect the minimum size of a cut when n is relatively small. Nevertheless, when n is large, taking advantage of the ring structure of the 2-dimensional N-mesh will result in $\varepsilon_N(n) < \varepsilon_\infty(n)$.

Now, let's define the following sets:

$$
\begin{aligned}
\mathcal{A}_1 \equiv& \{1, 2, \ldots, \frac{N^2}{4}\},\\
\mathcal{A}_2 \equiv& \{x \mid x \in \{\frac{N^2}{4}+1, \ldots, \frac{N^2}{2}\} \text{ and } (x \bmod N) \neq 0\},\\
\mathcal{A}_3 \equiv& \{x \mid x \in \{\frac{N^2}{4}+1, \ldots, \frac{N^2}{2}\} \text{ and } (x \bmod N) = 0\},\\
\mathcal{O}_1 \equiv& \{1, 2, \ldots, \frac{N^2-1}{4}\},\\
\mathcal{O}_2 \equiv& \{x \mid x \in \{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\}\\
& \text{ and } (x \bmod N) \neq 0\}, \text{ and}\\
\mathcal{O}_3 \equiv& \{x \mid x \in \{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\}\\
& \text{ and } (x \bmod N) = 0\}.
\end{aligned}
$$

***Lemma** 3:* Let $G = (V, E)$ be a 2-dimensional N-mesh, for N even,

$$\varepsilon_N(n) = \begin{cases} \varepsilon_\infty(n) & \text{for} \quad n \in \mathcal{A}_1 \\ 2N+2 & \text{for} \quad n \in \mathcal{A}_2 \\ 2N & \text{for} \quad n \in \mathcal{A}_3 \end{cases}$$

for N odd,

$$\varepsilon_N(n) = \begin{cases} \varepsilon_\infty(n) & \text{for} \quad n \in \mathcal{O}_1 \\ 2N+2 & \text{for} \quad n \in \mathcal{O}_2 \\ 2N & \text{for} \quad n \in \mathcal{O}_3 \end{cases}$$

Proof: From Fig. 5, we see that $\varepsilon_N(n) \leq 2N\ \forall n$ such that $(n \bmod N) = 0$ and $\varepsilon_N(n) \leq 2N+2$ if $(n \bmod N) \neq 0$. For n small, $\varepsilon_N(n) = \varepsilon_\infty(n)$. When $n = \frac{N^2}{4} + k$ for $k \geq 1$, we have $\varepsilon_\infty(\frac{N^2}{4}+k) \geq 2N+2$. Therefore, we can use the splitting method in Fig. 5, which will result in a cut size of $2N+2$, to separate the two sets. For N odd, $\varepsilon_\infty(\frac{N^2-1}{4}+1) = \varepsilon_\infty((\frac{N-1}{2})^2 + \frac{N-1}{2} + 1) = 4(\frac{N-1}{2}) + 4 = 2N+2$. Again, we can use the method in Fig. 5 to separate the sets. ■

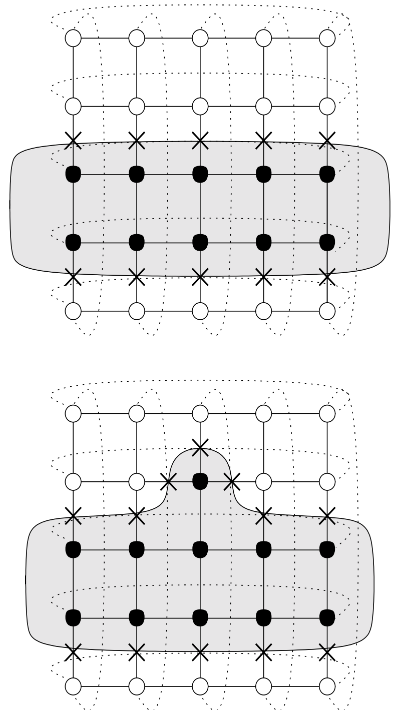

Fig. 5. Ways of splitting the N-mesh into two disjoint parts.

***Theorem** 1:* On a 2-dimensional N-mesh, the minimum capacity, C_1, that each link must have in order to support all-to-all traffic is at least $\frac{N^3}{4}$ for N even, and $\frac{N^3-N}{4}$ for N odd.

Proof: Consider a fixed n between 1 and N^2-1. The idea is to use a cut to separate the network (N-mesh) into two disjoint parts, with one part containing n nodes and the other containing N^2-n nodes. Based on the all-to-all traffic model, we know the exact amount of traffic, $C_{cross} = 2n(N^2-n)$, that must go through the cut. Therefore, from max-flow min-cut theorem [15] we know that simply dividing C_{cross} by the minimum size of cutset $\varepsilon_N(n)$ will give us *a* lower bound on C_1, and let's call this bound B_n. It implies that each link in the network must have capacity of at least B_n in order to satisfy the all-to-all traffic demand. This prompts us to find $B^{C_1}_{max}$ which is the maximum of B_n over all $n \in \{1, \ldots, N^2-1\}$. We say that $B^{C_1}_{max}$ is the best lower bound for C_1 in the sense that it is greater or equal to any other lower bound for C_1.

For N even, let

$$
\begin{aligned}
B^{C_1}_{max} &= \max_{n \in \{1,\ldots,N^2-1\}} \left\lceil \frac{2(N^2-n)n}{\varepsilon_N(n)} \right\rceil \qquad (1)\\
&= \max \left\{ \max_{n \in \mathcal{A}_1} \left\lceil \frac{2(N^2-n)n}{\varepsilon_\infty(n)} \right\rceil, \right.
\end{aligned}
$$

$$\max_{n\in\mathcal{A}_2}\left[\frac{2(N^2-n)n}{2N+2}\right],$$
$$\max_{n\in\mathcal{A}_3}\left[\frac{2(N^2-n)n}{2N}\right]\bigg\}. \qquad (2)$$

The case for N odd is the same except that $\mathcal{A}_1, \mathcal{A}_2,$ and $\mathcal{A}_3$ in (2) are replaced by $\mathcal{O}_1, \mathcal{O}_2,$ and $\mathcal{O}_3$. Solving the maximization problem, we get

$$B^{C_1}_{max} = \begin{cases} \max\left\{\alpha_e, \frac{N^4}{2(2N+1)}, \frac{N^3}{4}\right\} & \text{for } N \text{ even} \\ \max\left\{\alpha_o, \frac{N^4-1}{2(2N+1)}, \frac{N^3-N}{4}\right\} & \text{for } N \text{ odd} \end{cases}$$

where α_e (α_o) in the above equation is the result of the first term of equation (2) for N even (odd). Here, explicit evaluation of α_e and α_o is unnecessary. Instead, by using Corollary 1, an upper bound on α_e and α_o will be sufficient for us to solve the maximization problem. Since $\varepsilon_\infty(n) \geq 4\sqrt{n}$ for $n \in \mathcal{Z}^+$, the following equation holds:

$$\begin{aligned}\alpha_e &= \max_{n\in\mathcal{A}_1}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)}\right] \leq \max_{n\in\mathcal{Z}^+}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)}\right] \\ &\leq \max_{n\in\mathcal{Z}^+}\left[\frac{2(N^2-n)n}{4\sqrt{n}}\right] = \frac{3N^3}{16} < \frac{N^3}{4}\end{aligned}$$

$\alpha_o < \frac{N^3-N}{4}$ can be shown similarly. Thus, we have

$$B^{C_1}_{max} = \begin{cases} \frac{N^3}{4} & \text{for } N \text{ even} \\ \frac{N^3-N}{4} & \text{for } N \text{ odd} \end{cases}$$

■

***Corollary** 3:* On a 2-dimensional N-mesh with an arbitrary link failed, the lower bound, C_2, on the minimum capacity that each link must have in order to support all-to-all traffic is $\frac{N^4}{2(2N-1)}$ for N even, and $\frac{N^2(N^2-1)}{2(2N-1)}$ for N odd.

Proof: The proof of this corollary is similar to the proof of Theorem 1. We still use the max-flow min-cut theorem to compute the best lower bound C_2. In this case, we have

$$B^{C_2}_{max} = \max_{n\in\{1,\ldots,N^2-1\}}\left[\frac{2(N^2-n)n}{\varepsilon_N(n)-1}\right] \qquad (3)$$

$$= \max\left\{\max_{n\in\mathcal{A}_1}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}\right],\right.$$
$$\max_{n\in\mathcal{A}_2}\left[\frac{2(N^2-n)n}{2N+2-1}\right],$$
$$\left.\max_{n\in\mathcal{A}_3}\left[\frac{2(N^2-n)n}{2N-1}\right]\right\} \qquad (4)$$

Notice the difference between the above equations and equations (1) and (2) in the proof of theorem 1. Because of the failed link, the denominator of (3) is changed to $\varepsilon_N(n) - 1$ by Corollary 2.

Solving the maximization problem, we get

$$B^{C_2}_{max} = \begin{cases} \max\left\{\alpha_e, \frac{N^4}{2(2N+1)}, \frac{N^4}{2(2N-1)}\right\} & \text{for } N \text{ even} \\ \max\left\{\alpha_o, \frac{N^4-1}{2(2N+1)}, \frac{N^2(N^2-1)}{2(2N-1)}\right\} & \text{for } N \text{ odd} \end{cases}$$

where α_e (α_o) in the above equation is the result of the first term of equation (4) for N even (odd). Again, explicit evaluation of α_e and α_o is unnecessary. Instead, by using $4\sqrt{n} - 1 \geq 3.5\sqrt{n}$ $\forall n \geq 5$, an upbound on α_e and α_o will provide us the essential information to solve the maximization problem. Since $\varepsilon_\infty(n) \geq 4\sqrt{n}$ for $n \in \mathcal{Z}^+$, the following equation holds

$$\begin{aligned}\alpha_e &= \max_{n\in\mathcal{A}_1}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}\right] \leq \max_{n\in\mathcal{Z}^+}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}\right] \\ &\leq \max\left[\max_{n\in\{1,\cdots,4\}}\frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}, \max_{n\geq 5}\frac{2(N^2-n)n}{3.5\sqrt{n}}\right] \\ &< \frac{N^4}{2(2N-1)}\end{aligned}$$

$\alpha_o < \frac{N^2(N^2-1)}{2(2N-1)}$ can be shown similarly. Thus, we have

$$B^{C_2}_{max} = \begin{cases} \frac{N^4}{2(2N-1)} & \text{for } N \text{ even} \\ \frac{N^2(N^2-1)}{2(2N-1)} & \text{for } N \text{ odd} \end{cases}$$

■

B. Algorithm Achieving the Lower Bound on C_1

In this section, we show that the lower bound on C_1 can be achieved by using a simple routing algorithm called the *Dimensional Routing Algorithm*. As we have mentioned earlier, the routing algorithm will use the shortest path between source and destination nodes. Below is a description of the *Dimensional Routing Algorithm*:

1. From the source node $\vec{p} = (p_1, p_2)$, move horizontally in the direction of shortest cyclic distance to the destination node $\vec{q} = (q_1, q_2)$; if there is more than one way to route the traffic, pick the one that moves in the (+) direction (mod N), i.e. $(p_1, p_2) \rightarrow ((p_1+1)\ mod\ N, p_2) \rightarrow ((p_1+2)\ mod\ N, p_2) \rightarrow \cdots \rightarrow (q_1, p_2)$. Route the traffic for $D_N(p_1, q_1)$ hops where $D_N(p_1, q_1)$ denotes the shortest cyclic distance (hops) between $\vec{p}$ and $\vec{q}$ in horizontal direction.
2. Move vertically in the direction of shortest cyclic distance to the destination node; if there is more than one way to route the traffic, pick the one that moves in the (+) direction (mod N). Route the traffic for $D_N(p_2, q_2)$ hops where $D_N(p_2, q_2)$ denotes the shortest cyclic distance (hops) between $\vec{p}$ and $\vec{q}$ in vertical direction.

That is, the routing path will include the following nodes, $\vec{p} = (p_1, p_2) \rightarrow (q_1, p_2) \rightarrow (q_1, q_2) = \vec{q}$. The above algorithm ensures the existence of a unique shortest path between every node $\vec{p}$ and $\vec{q}$ regardless of whether N is even or odd, and consequently, facilitates the analysis of link load.

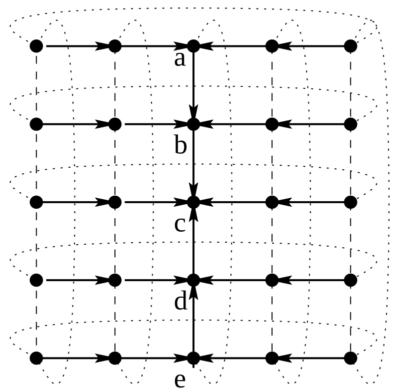

Fig. 6. An illustration of traffic flow into node c by using Dimensional Routing Algorithm.

***Theorem** 2:* Let $G = (V, E)$ be a 2-dimensional N-mesh, by using the *Dimensional Routing Algorithm* above, to satisfy the all-to-all traffic, the maximum load on each link is $\frac{N^3}{4}$ for N even and $\frac{N^3-N}{4}$ for N odd.

Proof: The *Dimensional Routing Algorithm* ensures one unique path between a source and destination pair. Thus, in order to compute the maximum load on a link, we need only count the (maximum) number of pairs of nodes that communicate through a specific link. Without loss of generality, consider the link $l_{\vec{b}\vec{c}}$ in Fig. 6. We see that ten units of traffic heading for node $\vec{c}$ must go through $l_{\vec{b}\vec{c}}$. By the symmetry of the mesh topology and *Dimensional Routing Algorithm*, five units of traffic heading for node $\vec{d}$ must go through $l_{\vec{b}\vec{c}}$ since five units of traffic heading for node $\vec{c}$ go through $l_{\vec{a}\vec{b}}$. Extending this argument, we see from Fig. 6 that an additional ten units of traffic destined for node $\vec{b}$ and five units of traffic headed to node $\vec{a}$ must communicate through $l_{\vec{b}\vec{c}}$. Again, by symmetry, the total load on any link of the graph (denoted by T_l), in the case of $N = 5$, is $T_l = 5 + 10 + 10 + 5 = 30$. In general, for N odd, we have the following formula:

$$T_l = 2N \sum_{i=1}^{\frac{N-1}{2}} i = \frac{N^3 - N}{4}.$$

For N even, using the same routing algorithm, we get $T_l = \frac{N^3}{4}$. ■

Clearly, using the *Dimensional Routing Algorithm*, we see that the lower bound of link capacity in the Theorem 1 is achieved. Now, with the minimum link capacity needed (C_1) and the lower bound of link capacity for mesh with a failed link (C_2) computed, we are able to derive the minimum spare capacity that each link must have in order to sustain the all-to-all traffic during the time of a link failure.

IV. Capacity Requirement for Recovering from a Link Failure

Under the condition of an arbitrary link failure, we investigate the spare capacity needed to fully restore the original traffic, using the link based restoration method and path based restoration method.

A. Link Based Restoration Strategy

Consider that an arbitrary link, $l_{\vec{u}\vec{v}}$ (connecting nodes $\vec{u}$ and $\vec{v}$), failed in the 2-dimensional N-mesh. We know from the previous section that there are $\frac{N^3-N}{4}$($\frac{N^3}{4}$) units of traffic on $l_{\vec{u}\vec{v}}$ have to be rerouted for N odd (even). Since the link based restoration strategy is used here, these $\frac{N^3-N}{4}$ units of traffic in and out of node $\vec{u}$ have to be rerouted through the remaining three links connecting to node $\vec{u}$ ($l_{\vec{u}\vec{v}}$ is already broken). We then have the following theorem:

***Theorem** 3:* Using link based restoration strategy in the event of a link failure, the minimum spare capacity that each link must have in order to support the all-to-all traffic is $\frac{N^3-N}{12}$ for N odd and $\frac{N^3}{12}$ for N even.

Proof: By using link based restoration scheme, a lower bound on spare capacity is $\frac{N^3-N}{12}$ for N odd and $\frac{N^3}{12}$ for N even from the argument stated in the previous paragraph. To show achievability, we refer to Fig. 7. Since the restoration paths are disjoint, we can reroute $\frac{1}{3}$ of the affected traffic through each of the three disjoint paths. Hence, the lower bound is achieved. ■

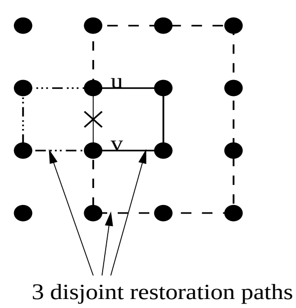

Fig. 7. Restoration paths using link based recovery scheme.

B. Path Based Restoration Strategy

B.1 Lower Bound on the Minimum Spare Capacity

***Theorem** 4:* On a 2-dimensional N-mesh with an arbitrary failed link, the minimum spare capacity, C_{spare}, that each link must have in order to support all-to-all traffic is at least $\frac{N^3}{4(2N-1)}$ for N even, and $\frac{N^3-N}{4(2N-1)}$ for N odd.

Proof: From Theorem 2, for a regular 2-dimensional N-mesh, we know that the capacity that each link must have in order to satisfy all-to-all traffic is $\frac{N^3}{4}$ for N even, and $\frac{N^3-N}{4}$ for N odd. In case of an arbitrary link failure, from Corollary 3, at least a capacity of $\frac{N^4}{2(2N-1)}$ ($\frac{N^2(N^2-1)}{2(2N-1)}$) is needed on each link to sustain the original traffic flow for N even (odd). We need to have an extra capacity of $C_{spare} \geq C_2 - C_1$ on each link. Thus, we have

$$C_{spare} \geq \begin{cases} \frac{N^4}{2(2N-1)} - \frac{N^3}{4} = \frac{N^3}{4(2N-1)} & \text{for } N \text{ even} \\ \frac{N^2(N^2-1)}{2(2N-1)} - \frac{N^3-N}{4} = \frac{N^3-N}{4(2N-1)} & \text{for } N \text{ odd} \end{cases}$$

■

B.2 Algorithm Using Minimum Spare Capacity

In this section, we will show that the minimum spare capacity needed on each link is $\frac{N^3}{4(2N-1)}$ for N even and $\frac{N^3-N}{4(2N-1)}$ for N odd. In other words, the lower bound in Theorem 4 is tight. We show the achievability by presenting a primary routing algorithm, and subsequently, a path-based recovery algorithm which fully restores the original traffic by using the minimum spare capacity in case of a link failure. We focus on the case of N odd for simplicity. To show the achievability for N even, a different set of primary routing algorithm and recovery algorithm is needed (not presented in this paper).

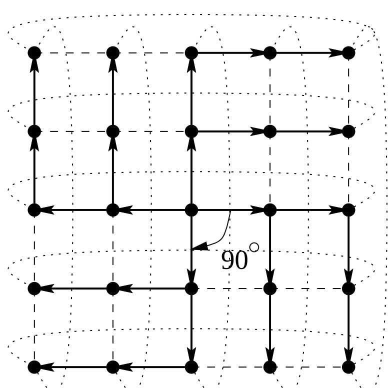

Fig. 8. Routing path of the Rotational Symmetric routing algorithm. Rotating the graph by 90° does not change the configuration.

First, we describe the primary routing algorithm that we call *Rotational Symmetric Routing Algorithm*, or *RS Routing Algorithm*, used to route the all-to-all traffic. We use the *RS Routing Algorithm* instead of the *Dimensional Routing Algorithm* as our primary routing algorithm because the former simplify the construction and analysis of the restoration algorithm. Specifically, with the *Dimensional Routing Algorithm*, the traffic routes on horizontal and vertical links are not symmetric; hence a different restoration algorithm would be required for vertical and horizontal link failure. In contrast, the *RS Routing Algorithm* is symmetric and vertical or horizontal link failure can be treated using the same recovery algorithm. The case of a horizontal link failure is the same as the vertical link failure if we rotate the topology by 90° (shown in Fig. 8).

RS routing algorithm

Each node $\vec{a}$ in a 2-dimensional N-mesh has a pair of integers (a_1, a_2) associated with it. To route one unit of traffic from the source node $\vec{p}$ to the destination node $\vec{q}$, do the following:

1. Change coordinate and compute the relative position of the destination node with respect to the source node. Specifically, shift the source node to $(0,0)$ by applying the transformation $T_{\vec{p}}$. Here, the transformation $T_{\vec{p}} : \mathcal{Z}_N \times \mathcal{Z}_N \rightarrow \mathcal{Z}_N \times \mathcal{Z}_N$ is defined as $\vec{d} = T_{\vec{p}}(\vec{q}) = T_{\vec{p}}(q_1, q_2) = (d_1, d_2)$, where for $i = 1, 2$

$$d_i = \begin{cases} q_i - p_i, \\ \quad \text{if } -\frac{N-1}{2} \leq q_i - p_i \leq \frac{N-1}{2} \\ (q_i - p_i) \ mod \ N, \\ \quad \text{if } -(N-1) \leq q_i - p_i < -\frac{N-1}{2} \\ -([-(q_i - p_i)] \ mod \ N), \\ \quad \text{if } \frac{N-1}{2} < q_i - p_i \leq N-1 \end{cases}$$

Here, $(-n) \ mod \ p$ is defined as $p - n \ mod \ p$ if $0 < n \ mod \ p < p$. Thus, we will have $T_{\vec{p}}(\vec{p}) = (0,0)$. Fig. 9 illustrates this transformation.

2. Divide the nodes of the 2-dimensional N-mesh into four quadrants with the source node as the origin (shown in Fig. 9). Specfically, let

$$\begin{aligned} \mathcal{Q}_1 = \ & \{(a,b) \mid a, b \in \mathcal{Z}_N \\ & \text{and } 0 \leq a \leq \frac{N-1}{2}, 0 < b \leq \frac{N-1}{2}\}, \\ \mathcal{Q}_2 = \ & \{(a,b) \mid a, b \in \mathcal{Z}_N \\ & \text{and } -\frac{N-1}{2} \leq a < 0, -\frac{N-1}{2} \leq b \leq 0\}, \\ \mathcal{Q}_3 = \ & \{(a,b) \mid a, b \in \mathcal{Z}_N \\ & \text{and } -\frac{N-1}{2} \leq a \leq 0, -\frac{N-1}{2} \leq b < 0\}, \text{ and} \\ \mathcal{Q}_4 = \ & \{(a,b) \mid a, b \in \mathcal{Z}_N \\ & \text{and } 0 < a \leq \frac{N-1}{2}, -\frac{N-1}{2} \leq b \leq 0\}. \end{aligned}$$

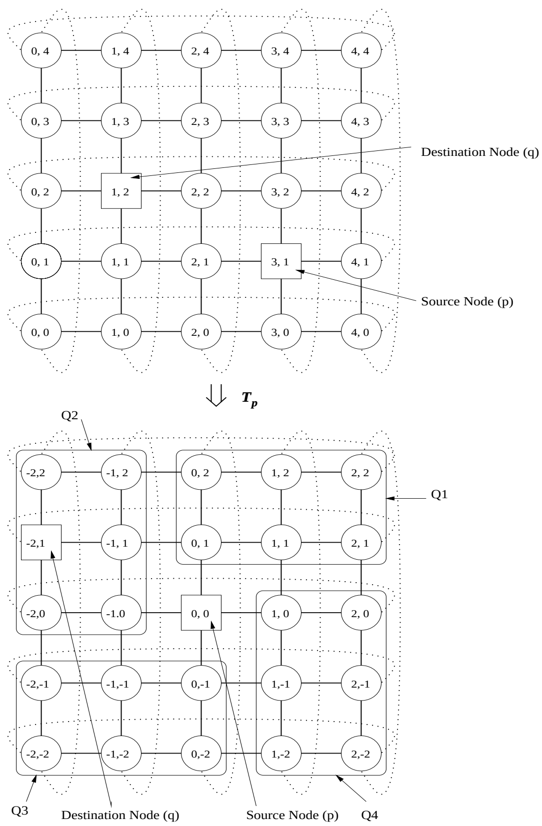

Fig. 9. Change of coordinate by using transformation $T_{\vec{p}}$.

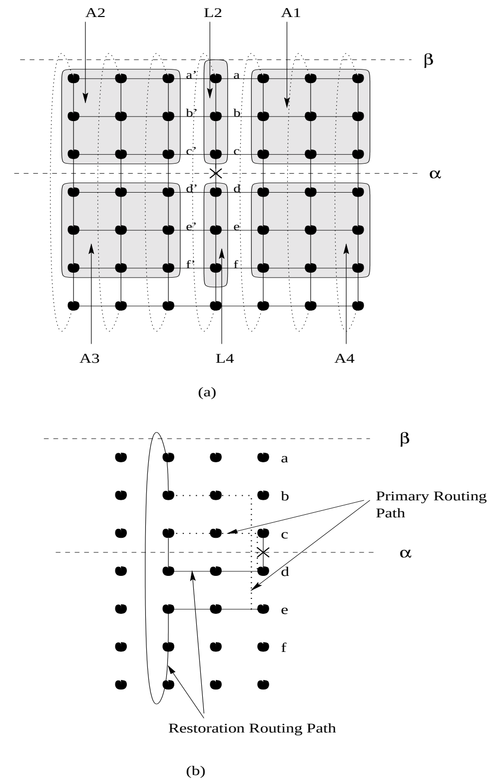

Fig. 10. Routing path of the restoration algorithm

3. If $\vec{d} = T_{\vec{p}}(\vec{q}) \in (\mathcal{Q}_1 \cup \mathcal{Q}_3)$, route the traffic vertically in the direction of shortest cyclic distance to the destination node by $D_N(p_2, q_2)$ hops. Then, route the traffic horizontally in the direction of shortest cyclic distance to the destination node by $D_N(p_1, q_1)$ hops.
If $\vec{d} = T_{\vec{p}}(\vec{q}) \in (\mathcal{Q}_2 \cup \mathcal{Q}_4)$, route the traffic horizontally in the direction of shortest cyclic distance to the destination node by $D_N(p_1, q_1)$ hops. Then, route the traffic vertically in the direction of shortest cyclic distance to the destination node by $D_N(p_2, q_2)$ hops.

Now, considering all traffic that has a particular node $\vec{c}$ as their destination, their routing paths are rotational symmetric by the above algorithm. That is, rotating all of the routing paths by an integer multiple of 90° will result in having the same original routing configuration. This idea is best illustrated by Fig. 8. *RS routing algorithm* also achieves the lower bound on C_1. The proof is straightforward and thus omitted here.

Our goal here is to recover the original traffic flow by adding an extra amount of capacity, which is equal to the lower bound calculated in Theorem 4, on each link. Now, we present an example to illustrate the key ideas of the recovery algorithm. Without loss of generality, suppose that link $l_{\vec{c}\vec{d}}$ failed in the 2-dimensional 7-mesh shown in Fig. 10(a). We need to find all possible source destination pairs (S-D pairs) that are affected by the failed link first. From the *RS routing algorithm*, these S-D pairs can be determined exactly. Specifically, let the source node be $\vec{s}$ and destination node be $\vec{t}$. The set of failed traffic F is defined as $F = F_1 \cup F_2 \cup F_3 \cup F_4 \cup F_5 \cup F_6$ where

$$F_1 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in A_2 \text{ and } \vec{t} \in L_4; D_N(s_1, t_1) \le \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \le \frac{N-1}{2}\},$$

$$F_2 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in L_2 \text{ and } \vec{t} \in A_3; D_N(s_1, t_1) \le \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \le \frac{N-1}{2}\},$$

$$F_3 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in A_4 \text{ and } \vec{t} \in L_2; D_N(s_1, t_1) \le \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \le \frac{N-1}{2}\},$$

$$F_4 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in L_4 \text{ and } \vec{t} \in A_1; D_N(s_1, t_1) \le \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \le \frac{N-1}{2}\},$$

$$F_5 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in L_4 \text{ and } \vec{t} \in L_2; D_N(s_1, t_1) \le \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \le \frac{N-1}{2}\}, \text{ and}$$

$$F_6 = \{(\vec{s}, \vec{t}) \mid \vec{s} \in L_2 \text{ and } \vec{t} \in L_4; D_N(s_1, t_1) \le \frac{N-1}{2}$$

and $D_N(s_2,t_2) \leq \frac{N-1}{2}\}$.

In the 2-dimensional 7-mesh with a link failure, the sets A_1, A_2, A_3, A_4, L_2 and L_4 are shown in Fig. 10(a). More generally, with a failed vertical link connecting nodes $\vec{v} = (v_1, v_2)$ and $\vec{u} = (v_1, (v_2+1) mod N)$, after taking the transformation $T_{\vec{v}}$, we can define these sets as the following:

$$A_1 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } 1 \leq a \leq \frac{N-1}{2}, 1 \leq b \leq \frac{N-1}{2}\},$$
$$A_2 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } -\frac{N-1}{2} \leq a \leq -1, 1 \leq b \leq \frac{N-1}{2}\},$$
$$A_3 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } -\frac{N-1}{2} \leq a \leq -1, -[\frac{N-1}{2}-1] \leq b \leq 0\},$$
$$A_4 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } 1 \leq a < \frac{N-1}{2}, -[\frac{N-1}{2}-1] \leq b \leq 0\},$$
$$L_2 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } a = 0, 1 \leq b \leq \frac{N-1}{2}\}, \text{ and}$$
$$L_4 = \{(a,b) \mid a,b \in \mathcal{Z}_N \text{ and } a = 0, -[\frac{N-1}{2}-1] \leq b \leq 0\}.$$

A simple way for recovering a failed traffic is to reverse its routing order. That is, if the primary routing scheme is to route the traffic horizontally in the direction of shortest cyclic distance first, the recovery algorithm will route the traffic vertically first (shown in Fig. 10(b)). Thus, traffic that is supposed to go through the failed link will circumvent the failed link. Consider now the vertical links crossing line α in Fig. 10(a) and the affected traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$. Rerouting (i.e. reversing the routing order) all of the affected traffic in $F_1 \cup F_2 \cup F_3 \cup F_4$ through the vertical links crossing line α will add an additional 12 units of traffic on each of these six vertical links. Fig. 11(a) illustrates the recovering paths of the traffic (originating from nodes a', b', and c') in the set F_1, which are being rerouted through the link $l_{\vec{c'}\vec{d'}}$. Recovering paths for the traffic in F_2, although not shown here, is just a flip of Fig. 11(a) with respect to the line α. The total amount of rerouted traffic in $F_1 \cup F_2$ added on link $l_{\vec{c'}\vec{d'}}$, which is 12, exceeds the lower bound of spare capacity, $C_2 - C_1 = \lceil \frac{N^3-N}{4(2N-1)} \rceil = 7$. However, utilizing the ring structure of the mesh topology, we can reroute half of the affected traffic through links crossing line β (illustrated in Fig. 11(b)). This way, we have a total of six units traffic through the link $l_{\vec{c'}\vec{d'}}$ (three from F_1 and three from F_2). For the traffic in the set $F_5 \cup F_6$, we can reroute half of them (six units) through the link $l_{\vec{g}\vec{a}}$. The remaining six units of traffic can be routed evenly through the six vertical links crossing line α. Thus, we can restore the original traffic flow by using only an additional $C_2 - C_1$ amount of capacity on each vertical link.

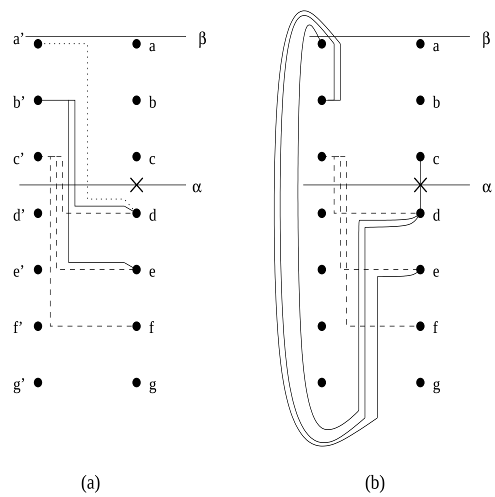

Fig. 11. Restoration path for the 2-dimensional 7-mesh

So far we have only discussed the load on a vertical link. Now, we will address the question of whether the additional traffic on each horizontal link will exceed $C_2 - C_1$. For example, on the link $l_{\vec{d'}\vec{d}}$ in Fig. 10(a), one may find that the number of rerouted traffic from the set $F_1 \cup F_2$, nine, exceeds $C_2 - C_1 = 7$ after reversing the routing order of the affected traffic. However, as we reroute the affected traffic circumventing the failed link, we not only put an additional nine units of traffic ($\vec{s} \in A_2, \vec{t} = \vec{d}$) on link $l_{\vec{d'}\vec{d}}$ but also take nine units of traffic ($\vec{s} \in L_2, \vec{t} \in L_3$) away from link $l_{\vec{d'}\vec{d}}$. Overall, we have zero additional rerouted traffic from the set $F_1 \cup F_2$ go through link $l_{\vec{d'}\vec{d}}$. Nevertheless, traffic in the set $F_5 \cup F_6$ does add extra units of traffic on the link $l_{\vec{d'}\vec{d}}$. By rerouting half of the traffic in $F_5 \cup F_6$ (six) through the link $l_{\vec{g}\vec{a}}$ (without using any horizontal link), we can then distribute the rest of the traffic in $F_5 \cup F_6$ (six) evenly, so as to satisfy the spare capacity constraint.

As we have mentioned earlier, only the traffic in the set $\bigcup_{i=1}^{6} F_i$ are being rerouted in our path based recovery algorithm. Traffic which is unaffected by the failed link remains intact in the recovery algorithm.

Lastly, we cannot include the full details of the path based restoration algorithm in this paper due to space limitation. For the same reason, we state the following theorem, which shows that the lower bound on the spare capacit $(C_2 - C_1)$ is indeed achievable, without proof.

***Theorem** 5:* On a 2-dimensional N-mesh, to restore the original all-to-all traffic in the event of a link failure, we need a spare capacity of $\frac{N^3-N}{4(2N-1)}$ on each link for N odd and $\frac{N^3}{4(2N-1)}$ for N even by using the restoration algorithm.

V. Conclusion

This paper examines the capacity requirements for mesh networks with all-to-all traffic. This study is particularly useful for the purpose of design and capacity provisioning in satellite networks. A novel technique of cuts on a graph is used to obtain a tight lower bound on the capacity requirements. This cut technique provides an efficient and simple way of obtaining lower bounds on spare capacity requirements for more general failure scenarios such as node failures or multiple link failures.

Another contribution of this work is in the efficient restoration algorithm that meets the lower bound on capacity requirement. Our restoration algorithm is relatively fast in that only those traffic streams affected by the link failure must be rerouted. Yet, our algorithm utilizes much less spare capacity than link based restoration (factor of N improvement). Furthermore, in order to achieve high capacity utilization, our algorithm makes use of capacity that is relinquished by traffic that is rerouted due to the link failure (i.e. stub release [5]).

Interesting extensions include the consideration of node failures, for which finding an efficient restoration algorithm is challenging, as well as considering the impact of multiple link failures. Finally, for the application to satellite networks, it would also be interesting to examine the impact of different cross-link architectures.

References

[1] Y. Xiong and L. Mason, “Restoration Strategies and Spare Capacity Requirements in Self-Healing ATM Networks,” in *Proceedings of INFOCOM ’97*, vol. 1, pp 353-360, 1997.

[2] S. Ramamurthy and B. Mukherjee, “Survivable WDM Mesh Networks, Part I – Protection,” in *Proceedings of INFOCOM ’99*, vol. 2, pp. 744-751, Mar. 1999.

[3] S. Ramamurthy and B. Mukherjee, “Survivable WDM Mesh Networks, Part II – Restoration,” in *ICC ’99 Proceedings*, pp. 2023-2030, 1999.

[4] E. Modiano and A. Ephremides, “Efficient algorithms for performing packet broadcasts in a mesh network,” *IEEE/ACM Trans. on Networking*, vol. 4, no. 4, pp 639-648, Aug. 1996.

[5] R. R. Iraschko, M. H. MacGregor, and W. D. Grover, “Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks,” *IEEE/ACM Trans. on Networking*, vol. 6, Jun. 1998.

[6] S.S. Lumetta and M. Medard, “Towards a deeper understanding of link restoration algorithms for mesh networks,” in *Proceedings of INFOCOM ’01*, vol. 1, pp. 367-375, 2001.

[7] M.C. Azizoglu and O. Egecioglu “Lower bounds on communication loads and optimal placements in torus networks”, *IEEE Trans. on Computers*, vol. 49, no. 3, pp. 259-266, Mar. 2000.

[8] B. Bose, R. Broeg, Y. Kwon, and Y. Ashir, “Lee distance and topological properties of k-ary n-cubes,” *IEEE Trans. on Computers*, vol. 44, no. 8, pp. 1021-1030, Aug. 1995.

[9] P. W. Lemme, S. M. Glenister, and A. W. Miller, “Iridium aeronautical satellite communications ,” *IEEE Aerospace and Electronics Systems Magazine*, vol. 14, no. 11, pp. 11-16, Nov. 1999.

[10] D. P. Patterson, “Teledesic: a global broadband network,” *1998 IEEE Aerospace Conference*, vol. 4, pp. 547-552, 1998

[11] E. Ekici, I. F. Akyildiz, and M. D. Bender, “A distributed routing algorithm for datagram traffic in LEO satellite networks,” *IEEE/ACM Trans. on Networking*, vol. 9, no. 2, pp. 137-147, Apr. 2001.

[12] G. D. Stamoulis and J. N. Tsitsiklis, “Efficient routing schemes for multiple broadcasts in hypercubes,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 4, no. 7, pp. 725-739, Jul. 1993.

[13] E. Varvarigos, “Efficient routing algorithms for folded-cube networks ,” in *Proceedings of the 1995 IEEE 14th Annual International Phoenix Conference on Computers and Communications*, pp. 143 -151, 1995.

[14] Y. J. Suh and K. G. Shin, “All-to-all personalized communication in multidimensional torus and mesh networks,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 12, no. 1, Jan. 2001.

[15] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*, Athena Scientific, 1998.