INTERNATIONAL JOURNAL OF SATELLITE COMMUNICATIONS AND NETWORKING
Int. J. Satell. Commun. Network. 2003; **21**:259–284 (DOI: 10.1002/sat.752)

Capacity provisioning and failure recovery for Low Earth Orbit satellite constellation

Jun Sun*,† and Eytan Modiano‡

Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, U.S.A.

SUMMARY

This paper considers the link capacity requirement for an LEO satellite constellation. We model the constellation as an $N \times N$ mesh-torus topology under a uniform all-to-all traffic model. Both primary capacity and spare capacity for recovering from a link or node failure are examined. In both cases, we use a 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. Copyright © 2003 John Wiley & Sons, Ltd.

1. 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 or a node 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 [1–5]. The obvious advantages of using the link restoration strategy are simplicity and ability to rapidly 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 [1–3].

*Correspondence to: Jun Sun, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, U.S.A.
† E-mail: junsun@mit.edu
‡ E-mail: modiano@mit.edu

Contract/grant sponsor: DARPA

Received 30 May 2002
Accepted 20 November 2002

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 (2D) n-ary hypercube and differs from a binary hypercube in that each node has a constant number of neighbours (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*) [6]. 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 [7–9], are equally applicable to other networks with a mesh topology such as multi-processor interconnect networks [10–12] 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 (S–D) 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 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 4, 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 or nodes 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. For a single link failure, 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 for link failure.

Communication on a mesh network has been studied in References [9, 12, 13]. In Reference [13], the authors consider processors communicating over a mesh network with the objective of broadcasting information. The work in Reference [9] presents routing algorithm generating minimum propagation delay for satellite mesh networks. In Reference [12], 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 [1–4]. In Reference [1], the authors study and compare spare capacity needed by using link-based and path-based schemes. The work of Reference [4] provides a method for capacity optimization of

Table I. Capacity requirements under link-based and path-based restoration for a link failure.

	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	$\frac{N^3}{12}$	$\frac{N^3}{4(2N-1)}$

path restorable networks and quantifies the capacity benefits of path over link restoration. In References [2, 3], the authors examine 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 2 gives necessary definitions and statement of the problem. In Section 3, a lower bound on C_1 is given along with a routing algorithm achieving this lower bound. The lower bound C_2 for link failure is presented also. We then show in Section 4 that the lower bound on C_{spare}, $C_2 - C_1$, can be achieved by a path-based restoration algorithm under a single link failure. In Section 5, we derive a lower bound on C_{spare} for the node failure case and present a restoration scheme. Section 6 concludes this paper.

2. 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 two-dimensional N-mesh is an undirected graph $G = (V, E)$, with vertex set

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

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

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

The above definition is from Reference [6]. A two-dimensional N-mesh has a total of N^2 nodes. Each node has two neighbours in the vertical and horizontal dimension, for a total of four neighbours. 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. Figure 1 shows a two-dimensional 5-mesh. The notion two-dimensional ∞-mesh is used to denote the case where N is arbitrarily large, and it is the same as an infinity grid.

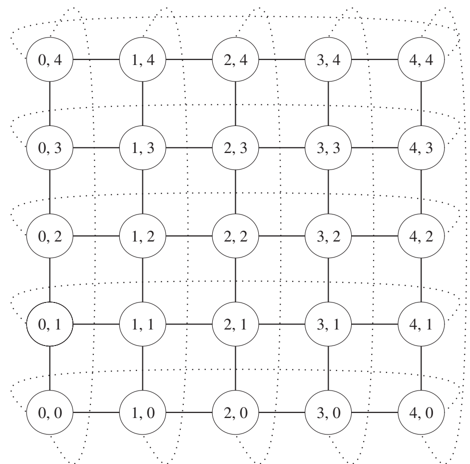

Figure 1. A two-dimensional 5-mesh.

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

Here the notation Cut-Set$(S, V - S) = \{(\mathbf{a}, \mathbf{b}) \in E \mid \mathbf{a} \in S, \mathbf{b} \in V - S\}$ 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 $C(S, V - S) = |\text{Cut-Set}(S, V - S)|$.

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

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

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

The function $\varepsilon_N(n)$ returns the minimum number of edges that must be removed in order to split the two-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 two-dimensional N-mesh are associated with the notion of *cyclic distance* which we will define next [14].

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

$$D_N(i,j) = \min\{(i-j) \bmod N), (j-i) \bmod N)\}$$

3. CAPACITY REQUIREMENT WITHOUT LINK OR NODE 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 Reference [13]) 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.

3.1. A lower bound on the primary capacity

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 neighbouring nodes are also in the set W while the other two are in $\bar{W}$. And of those two neighbouring nodes in W, they form a 90° angle with respect to node x (as shown in Figure 2). Similarly, a *leaf node* x of set W is defined to be a node $x \in W$ such that three of its four neighbouring nodes are in $\bar{W}$, 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 Figure 3(a) is square and the shape of the set in Figure 3(b) is rectangular. Lastly, we use the term *minimum set* W_n to refer any set such that $C(W_n, \overline{W_n}) = \varepsilon_\infty(n)$.

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 $n \in \mathscr{Z}^+$.
4. $\varepsilon_\infty(n)$ is a monotonically non-decreasing 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.

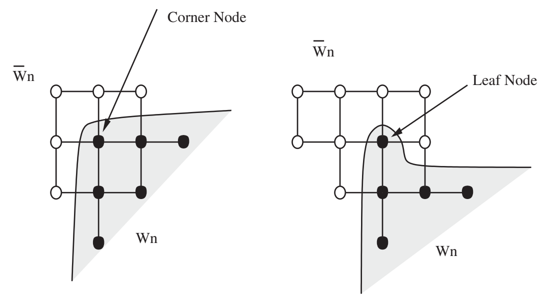

Figure 2. Representation of corner node and leaf node.

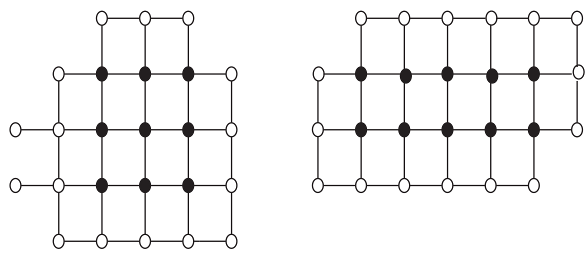

Figure 3. An illustration of the square shape and the rectangular shape.

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 non-decreasing function, suppose there exists $k \in \mathscr{Z}^+$ 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 nodes a or b are removed from W_{k+1}, the resulting set, say W'_k, will have k nodes remaining. We get $C(W'_k, \overline{W'_k}) \leqslant m_1$ 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

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

for $n, k \in \mathscr{Z}^+$ where $\mathscr{Z}^+$ 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 $\varepsilon_\infty(n^2) = 4n$, $\forall n \in \mathscr{Z}^+$, 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 Figure 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 $1 \leqslant k \leqslant n$, we prove that $\varepsilon_\infty(n^2+k) \geqslant 4n+2$ for $1 \leqslant k \leqslant n$. Then, by construction, $\varepsilon_\infty(n^2+k) = 4n+2$ for $1 \leqslant k \leqslant n$. 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 $\varepsilon_\infty(n^2+1) \geqslant 4n+2$. By the monotonicity of $\varepsilon_\infty(\cdot)$, $\varepsilon_\infty(n^2+k) \geqslant 4n+2$ for $1 \leqslant k \leqslant n$. 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 $1 \leqslant k \leqslant n$.

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

Corollary 1
For $\varepsilon_\infty(n)$ defined in above lemma, $\varepsilon_\infty(n) \geqslant 4\sqrt{n}$ for $n \in \mathscr{Z}^+$.

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

$$n - m^2 > m \Rightarrow \varepsilon_\infty(n) = 4m + 4$$

$$n - m^2 < m \Rightarrow \varepsilon_\infty(n) = 4m + 2$$

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, $\varepsilon_\infty(n) \geqslant 4\sqrt{n}$ for $n \in \mathscr{Z}^+$. □

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

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

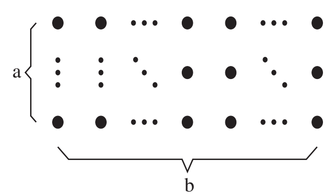

Figure 4. An arrangement of n^2 nodes in rectangular shape.

and

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

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

Proof
The proof of this corollary 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 two-dimensional N-mesh, it is essential to know $\varepsilon_N(n)$. In a two-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 two-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 two-dimensional N-mesh will result in $\varepsilon_N(n) < \varepsilon_\infty(n)$.

Now, let us define the following sets:

$$\mathscr{A}_1 \equiv \left\{1, 2, \ldots, \frac{N^2}{4}\right\}$$

$$\mathscr{A}_2 \equiv \left\{x \mid x \in \left\{\frac{N^2}{4}+1, \ldots, \frac{N^2}{2}\right\} \text{ and } (x \bmod N) \neq 0\right\}$$

$$\mathscr{A}_3 \equiv \left\{x \mid x \in \left\{\frac{N^2}{4}+1, \ldots, \frac{N^2}{2}\right\} \text{ and } (x \bmod N) = 0\right\}$$

$$\mathscr{O}_1 \equiv \left\{1, 2, \ldots, \frac{N^2-1}{4}\right\}$$

$$\mathscr{O}_2 \equiv \left\{x \mid x \in \left\{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\right\} \text{ and } (x \bmod N) \neq 0\right\}, \text{ and}$$

$$\mathscr{O}_3 \equiv \left\{x \mid x \in \left\{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\right\} \text{ and } (x \bmod N) = 0\right\}$$

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

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

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 Figure 5, we see that $\varepsilon_N(n) \leqslant 2N$ $\forall n$ such that $(n \bmod N) = 0$ and $\varepsilon_N(n) \leqslant 2N+2$ if $(n \bmod N) \neq 0$. For n small, $\varepsilon_N(n) = \varepsilon_\infty(n)$. When $n = N^2/4 + k$ for $k \geqslant 1$, we have $\varepsilon_\infty(N^2/4 + k) \geqslant 2N+2$. Therefore, we can use the splitting method in Figure 5, which will result in a cut size of $2N+2$, to separate the two sets. For N odd, $\varepsilon_\infty((N^2-1)/4+1) = \varepsilon_\infty(((N-1)/2)^2 + (N-1)/2+1) = 4((N-1)/2)+4 = 2N+2$. Again, we can use the method in Figure 5 to separate the sets. □

Theorem 1

On a two-dimensional N-mesh, the minimum capacity, C_1, that each link must have in order to support all-to-all traffic is at least $N^3/4$ for N even, and $(N^3-N)/4$ 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

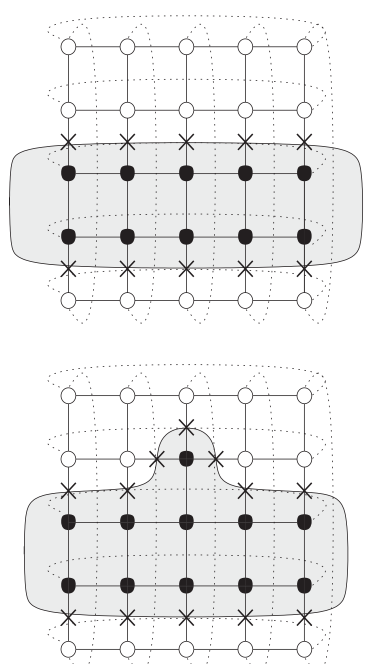

Figure 5. Ways of splitting the N-mesh into two disjoint parts.

nodes. Based on the all-to-all traffic model, we know the exact amount of traffic, $C_{\text{cross}} = 2n(N^2 - n)$, 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 us 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 $B_{\max}^{C_1}$ which is the maximum of B_n over all $n \in \{1, \ldots, N^2 - 1\}$. We say that $B_{\max}^{C_1}$ 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

$$B_{\max}^{C_1} = \max_{n \in \{1,\ldots,N^2-1\}} \left\lceil \frac{2(N^2 - n)n}{\varepsilon_N(n)} \right\rceil \tag{1}$$

$$= \max\left\{ \max_{n \in \mathscr{A}_1} \left\lceil \frac{2(N^2 - n)n}{\varepsilon_\infty(n)} \right\rceil, \max_{n \in \mathscr{A}_2} \left\lceil \frac{2(N^2 - n)n}{2N + 2} \right\rceil, \max_{n \in \mathscr{A}_3} \left\lceil \frac{2(N^2 - n)n}{2N} \right\rceil \right\} \tag{2}$$

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

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

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 $\varepsilon_\infty(n) \geqslant 4\sqrt{n}$ for $n \in \mathscr{Z}^+$, the following equation holds:

$$\alpha_e = \max_{n \in \mathscr{A}_1} \left\lceil \frac{2(N^2 - n)n}{\varepsilon_\infty(n)} \right\rceil \leqslant \max_{n \in \mathscr{Z}^+} \left\lceil \frac{2(N^2 - n)n}{\varepsilon_\infty(n)} \right\rceil$$

$$\leqslant \max_{n \in \mathscr{Z}^+} \left\lceil \frac{2(N^2 - n)n}{4\sqrt{n}} \right\rceil = \frac{3N^3}{16} < \frac{N^3}{4}$$

$\alpha_o < (N^3 - N)/4$ can be shown similarly. Thus, we have

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

□

Corollary 3

On a two-dimensional N-mesh with an arbitrary link failure, the lower bound, C_2, on the minimum capacity that each link must have in order to support all-to-all traffic is $N^4/2(2N - 1)$ for N even, and $N^2(N^2 - 1)/2(2N - 1)$ 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

$$B_{\max}^{C_2} = \max_{n\in\{1,\ldots,N^2-1\}} \left\lceil \frac{2(N^2-n)n}{\varepsilon_N(n)-1} \right\rceil \tag{3}$$

$$= \max\left\{ \max_{n\in\mathscr{A}_1} \left\lceil \frac{2(N^2-n)n}{\varepsilon_\infty(n)-1} \right\rceil, \max_{n\in\mathscr{A}_2} \left\lceil \frac{2(N^2-n)n}{2N+2-1} \right\rceil, \max_{n\in\mathscr{A}_3} \left\lceil \frac{2(N^2-n)n}{2N-1} \right\rceil \right\} \tag{4}$$

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

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

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 $4\sqrt{n}-1 \geqslant 3.5 \times \sqrt{n}\ \forall n \geqslant 5$, an upbound on α_e and α_o will provide us the essential information to solve the maximization problem. Since $\varepsilon_\infty(n) \geqslant 4\sqrt{n}$ for $n \in \mathscr{Z}^+$, the following equation holds:

$$\begin{aligned} \alpha_e &= \max_{n\in\mathscr{A}_1} \left\lceil \frac{2(N^2-n)n}{\varepsilon_\infty(n)-1} \right\rceil \leqslant \max_{n\in\mathscr{Z}^+} \left\lceil \frac{2(N^2-n)n}{\varepsilon_\infty(n)-1} \right\rceil \\ &\leqslant \max\left\lceil \max_{n\in\{1,\ldots,4\}} \frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}, \max_{n\geqslant 5} \frac{2(N^2-n)n}{3.5\sqrt{n}} \right\rceil \\ &< \frac{N^4}{2(2N-1)} \end{aligned}$$

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

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

□

3.2. *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 $\mathbf{p} = (p_1, p_2)$, move horizontally in the direction of shortest cyclic distance to the destination node $\mathbf{q} = (q_1, q_2)$; if there is more than one way to route the traffic, pick the one that moves in the $(+)$ direction (mod N), i.e. $(p_1, p_2) \rightarrow ((p_1 + 1) \bmod N, p_2) \rightarrow ((p_1 + 2) \bmod N, p_2) \rightarrow \cdots \rightarrow (q_1, p_2)$. 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 $\mathbf{p}$ and $\mathbf{q}$ 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 $\mathbf{p}$ and $\mathbf{q}$ in vertical direction.

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

Theorem 2
Let $G = (V, E)$ be a two-dimensional N-mesh, by using the *dimensional routing algorithm* above, to satisfy the all-to-all traffic, the maximum load on each link is $N^3/4$ for N even and $(N^3 - N)/4$ 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 $l_{\mathbf{bc}}$ in Figure 6. We see that ten units of traffic heading for node **c** must go through $l_{\mathbf{bc}}$. By the symmetry of the mesh topology and *dimensional routing algorithm*, five units of traffic heading for node **d** must go through $l_{\mathbf{bc}}$ since five units of traffic heading for

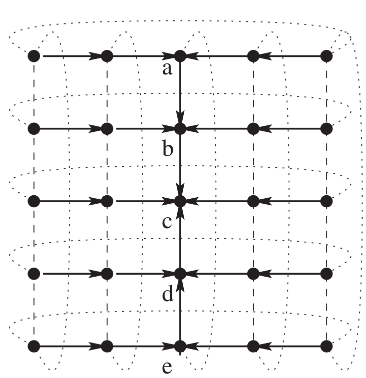

Figure 6. An illustration of traffic flow into node c by using dimensional routing algorithm.

node **c** go through $l_{\mathbf{ab}}$. Extending this argument, we see from Figure 6 that an additional ten units of traffic destined for node **b** and five units of traffic headed to node **a** must communicate through $l_{\mathbf{bc}}$. 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:

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

For N even, using the same routing algorithm, we get $T_l = N^3/4$. □

Clearly, using the *dimensional routing algorithm*, we see that the lower bound of link capacity in 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.

4. 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.

4.1. Link based restoration strategy

Consider that an arbitrary link, $l_{\mathbf{uv}}$ (connecting nodes **u** and **v**), failed in the two-dimensional N-mesh. We know from the previous section that there are $(N^3 - N)/4$ unit of traffic on $l_{\mathbf{uv}}$ have to be rerouted for N odd and $N^3/4$ for N even. Since the link-based restoration strategy is used here, these $(N^3 - N)/4$ units of traffic in and out of node **u** have to be rerouted through the remaining three links connecting to node **u** ($l_{\mathbf{uv}}$ 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 $(N^3 - N)/12$ for N odd and $N^3/12$ for N even.

Proof
By using link-based restoration scheme, a lower bound on spare capacity is $(N^3 - N)/12$ for N odd and $N^3/12$ for N even from the argument stated in the previous paragraph. To show achievability, we refer to Figure 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. □

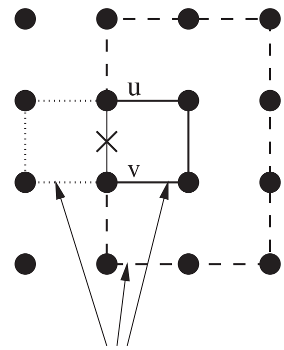

Figure 7. Restoration paths using link-based recovery scheme.

4.2. *Path-based restoration strategy*

4.2.1. *Lower bound on the minimum spare capacity.*

Theorem 4
On a two-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 $N^3/4(2N-1)$ for N even, and $(N^3-N)/4(2N-1)$ for N odd.

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

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

4.2.2. Algorithm using minimum spare capacity. In this section, we will show that the minimum spare capacity needed on each link is $N^3/4\ (2N-1)$ for N even and $(N^3-N)/4(2N-1)$ 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).

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 simplifies 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 Figure 8).

RS routing algorithm: Each node **a** in a two-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 **p** to the destination node **q**, do the following:

1. Change co-ordinate 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 $T_{\mathbf{p}}$. Here, the transformation $T_{\mathbf{p}} : \mathscr{Z}_N \times \mathscr{Z}_N \rightarrow \mathscr{Z}_N \times \mathscr{Z}_N$ is defined as $T_{\mathbf{p}}(q_1, q_2) = (d_1, d_2)$, where for $i = 1, 2$
$$d_i = \begin{cases} q_i - p_i & \text{if } -\frac{N-1}{2} \leqslant q_i - p_i \leqslant \frac{N-1}{2} \\ (q_i - p_i) \bmod N & \text{if } -(N-1) \leqslant q_i - p_i < -\frac{N-1}{2} \\ -([-(q_i - p_i)] \bmod N) & \text{if } \frac{N-1}{2} < q_i - p_i \leqslant N-1 \end{cases}$$
Here, $(-n) \bmod p$ is defined as $p - n \bmod p$ if $0 < n \bmod p < p$. Thus, we will have $T_{\mathbf{p}}(\mathbf{p}) = (0,0)$. Figure 9 illustrates this transformation.
2. Divide the nodes of the two-dimensional N-mesh into four quadrants with the source node as the origin (shown in Figure 9). Specifically, let
$$\mathscr{Q}_1 = \left\{(a,b) \mid a,b \in \mathscr{Z}_N \text{ and } 0 \leqslant a \leqslant \frac{N-1}{2},\ 0 < b \leqslant \frac{N-1}{2}\right\}$$
$$\mathscr{Q}_2 = \left\{(a,b) \mid a,b \in \mathscr{Z}_N \text{ and } -\frac{N-1}{2} \leqslant a < 0,\ -\frac{N-1}{2} \leqslant b \leqslant 0\right\}$$
$$\mathscr{Q}_3 = \left\{(a,b) \mid a,b \in \mathscr{Z}_N \text{ and } -\frac{N-1}{2} \leqslant a \leqslant 0, -\frac{N-1}{2} \leqslant b < 0\right\}, \text{ and}$$
$$\mathscr{Q}_4 = \left\{(a,b) \mid a,b \in \mathscr{Z}_N \text{ and } 0 < a \leqslant \frac{N-1}{2}, -\frac{N-1}{2} \leqslant b \leqslant 0\right\}$$
3. If $\mathbf{d} = T_{\mathbf{p}}(\mathbf{q}) \in (\mathscr{Q}_1 \cup \mathscr{Q}_3)$, 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 $\mathbf{d} = T_{\mathbf{p}}(\mathbf{q}) \in (\mathscr{Q}_2 \cup \mathscr{Q}_4)$, 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 **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

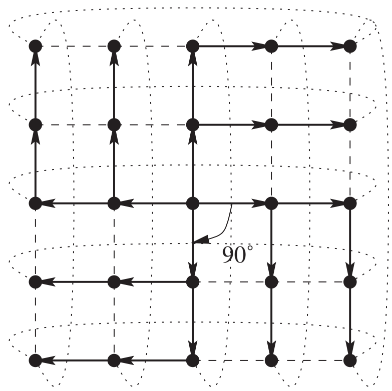

Figure 8. Routing path of the rotational symmetric routing algorithm. Rotating the graph by 90° does not change the configuration.

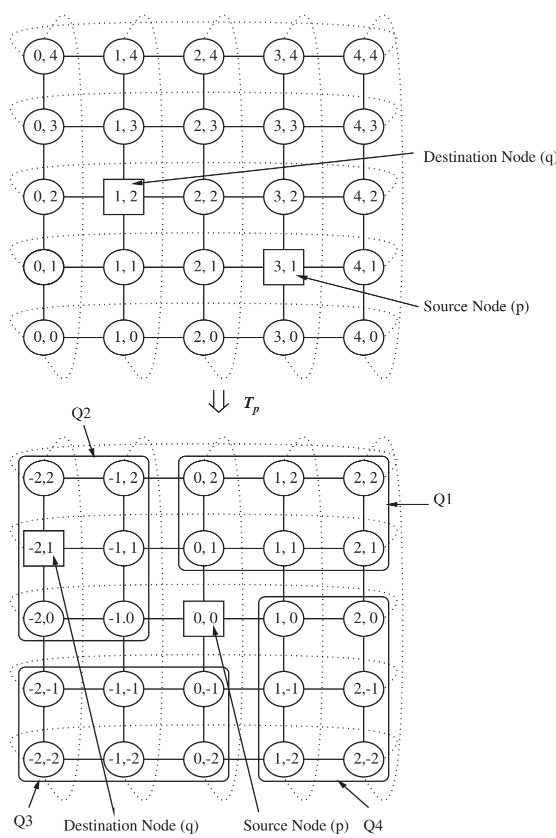

Figure 9. Change of co-ordinate by using transformation $T_{\mathbf{p}}$.

idea is best illustrated by Figure 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 $l_{\mathbf{cd}}$ failed in the two-dimensional 7-mesh shown in Figure 10(a). We need to find all possible 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 $\mathbf{s}$ and destination node be $\mathbf{t}$. 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

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

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

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

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

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

$$F_6 = \left\{ (\mathbf{s}, \mathbf{t}) \mid \mathbf{s} \in L_2 \text{ and } \mathbf{t} \in L_4;\ D_N(s_1, t_1) \leqslant \frac{N-1}{2} \text{ and } D_N(s_2, t_2) \leqslant \frac{N-1}{2} \right\}$$

In the two-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 Figure 10(a). More generally, with a failed vertical link connecting nodes $\mathbf{v} = (v_1, v_2)$ and $\mathbf{u} = (v_1, (v_2 + 1) \bmod N)$, after taking the transformation $T_{\mathbf{v}}$, we can define these sets as the following:

$$A_1 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } 1 \leqslant a \leqslant \frac{N-1}{2},\ 1 \leqslant b \leqslant \frac{N-1}{2} \right\}$$

$$A_2 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } -\frac{N-1}{2} \leqslant a \leqslant -1,\ 1 \leqslant b \leqslant \frac{N-1}{2} \right\}$$

$$A_3 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } -\frac{N-1}{2} \leqslant a \leqslant -1, -\left[\frac{N-1}{2} - 1\right] \leqslant b \leqslant 0 \right\}$$

$$A_4 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } 1 \leqslant a < \frac{N-1}{2}, -\left[\frac{N-1}{2} - 1\right] \leqslant b \leqslant 0 \right\}$$

$$L_2 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } a = 0, 1 \leqslant b \leqslant \frac{N-1}{2} \right\}, \text{ and}$$

$$L_4 = \left\{ (a, b) \mid a, b \in \mathscr{Z}_N \text{ and } a = 0, -\left[\frac{N-1}{2} - 1\right] \leqslant b \leqslant 0 \right\}$$

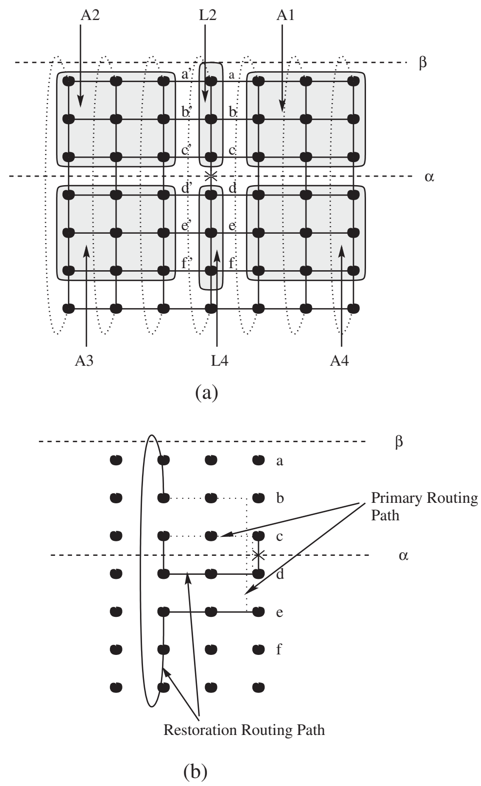

Figure 10. Routing path of the restoration algorithm.

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 Figure 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 Figure 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. Figure 11(a) illustrates the recovering paths of the traffic (originating from nodes $\vec{a}'$, $\vec{b}'$ and $\vec{c}'$) in the set F_1, which are being rerouted through the link $l_{\mathbf{c}'\mathbf{d}'}$. Recovering paths for the traffic in F_2, although not shown here, is just a flip of Figure 11(a) with respect to the line α. The total amount of rerouted traffic in $F_1 \cup F_2$ added on link $l_{\mathbf{c}'\mathbf{d}'}$, which is 12, exceeds the lower bound of spare capacity,

$$C_2 - C_1 = \left\lceil \frac{N^3 - N}{4(2N-1)} \right\rceil = 7$$

However, utilizing the ring structure of the mesh topology, we can reroute half of the affected traffic through links crossing line β (illustrated in Figure 11(b)). This way, we have a total of six units traffic through the link $l_{\mathbf{c}'\mathbf{d}'}$ (three from F_1 and three from F_2). For the traffic in the set

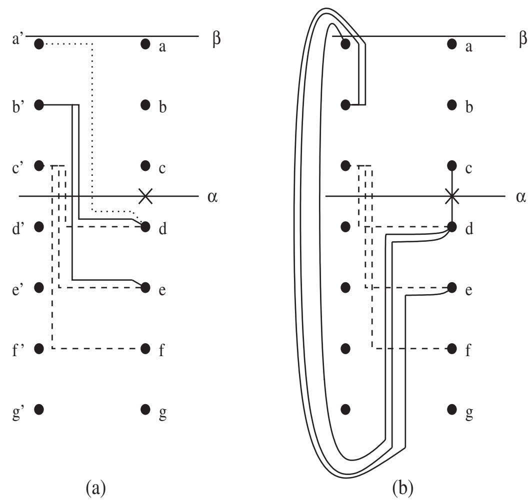

Figure 11. Restoration path for the two-dimensional 7-mesh.

$F_5 \cup F_6$, we can reroute half of them (six units) through the link $l_{\mathbf{ga}}$. 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.

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 $l_{\mathbf{d'd}}$ in Figure 10(a), one may find that the amount 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 ($\mathbf{s} \in A_2, \mathbf{t} = \mathbf{d}$) on link $l_{\mathbf{d'd}}$ but also take nine units of traffic ($\mathbf{s} \in L_2, \mathbf{t} \in L_3$) away from link $l_{\mathbf{d'd}}$. Overall, we have zero additional rerouted traffic from the set $F_1 \cup F_2$ go through link $l_{\mathbf{d'd}}$. Nevertheless, traffic in the set $F_5 \cup F_6$ does add extra units of traffic on the link $l_{\mathbf{d'd}}$. By rerouting half of the traffic in $F_5 \cup F_6$ (six) through the link $l_{\mathbf{ga}}$ (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 include the full details of the path-based restoration algorithm in Appendix A. We also state the following theorem which shows that the lower bound on the spare capacity ($C_2 - C_1$) is indeed achievable.

Theorem 5

On a two-dimensional N-mesh, to restore the original all-to-all traffic in the event of a link failure, we need a spare capacity of $(N^3 - N)/4(2N - 1)$ on each link for N odd and $N^3/4(2N - 1)$ for N even by using the restoration algorithm (proof in Appendix).

5. CAPACITY REQUIREMENT FOR RECOVERING FROM A NODE FAILURE

In this section, we investigate the spare capacity needed to fully restore the original traffic in the case of an arbitrary node failure. When a node failed in the network, all of the traffic destined for or generated from that node are terminated. And all of the traffic that passed through the failed node need to be rerouted. Next, we present the following theorem which gives us a lower bound on the spare capacity needed to restore the original traffic.

Theorem 6
On a two-dimensional N-mesh with an arbitrary node failure, the minimum spare capacity, C_{spare}, that each link must have in order to support all-to-all traffic is at least $N^2(N-4)/4(2N-1)$ for N even and $N(N^2-4N+3)/4(2N-1)$ for N odd.

The proof of this theorem follows the similar steps in the proofs of Theorems 1 and 4. Specifically, under an arbitrary node failure, the lower bound on the minimum capacity each link must have in order to support the all-to-all traffic is $[1/2(N^2-1)N^2 - N(N-1)]/(2N-1)$. Here, the numerator represents the total traffic across the cut, and the denominator is the size of the cut. The lower bound on the spare capacity follows from $[(1/2(N^2-1)N^2 - N(N-1))/2N-1] - C_1$ where $C_1 = \frac{1}{4}(N^3-N)$.

Again, we use RS routing algorithm as the primary routing algorithm.

Restoration algorithm: For traffic that goes through the failed node, reverse the routing order. Specifically, if the original traffic goes vertically first in the direction of shortest cyclic distance to the destination node and then moves horizontally to the destination node, we reroute the traffic horizontally in the direction of shortest cyclic distance first and then reroute the traffic vertically.

To calculate the spare capacity required by using the above restoration scheme, we consider the spare capacity needed on the set of links surrounding the failed node. By examining the rerouted traffic, we can see that those links are the ones that require the most spare capacity. First, we calculate the relinquished capacity on each of these links to be $(N-1)^2/4$. After rerouting the affected traffic, the newly added traffic on each link is at most

$$\left\lceil \frac{1}{8}N^2 - \frac{9}{8} + \frac{(N-1)^2}{4} \right\rceil$$

Therefore, a total of $\lceil \frac{1}{8}N^2 - \frac{9}{8} \rceil$ spare capacity is needed to fully restore the original traffic. A more rigorous proof of these statements will follow the line of proof shown in Appendix A. We can see that the spare capacity required by our restoration algorithm is asymptotically equal to the lower bound on spare capacity in Theorem 6.

6. 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. The 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 [4]).

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

APPENDIX A: PATH-BASED RESTORATION ALGORITHM

Again, we focus on the case of N odd for simplicity. From the source node $\mathbf{p}$ to the destination node $\mathbf{q}$, we consider the case that its routing path includes the failed link. Without loss of generality, we assume an arbitrary vertical link failed (the case of a horizontal link failure is the same because of symmetry provided by the primary routing algorithm). The two nodes connected by the failed link are referred to as node $\mathbf{u}$ and $\mathbf{v}$ with node $\mathbf{u}$ on the top of $\mathbf{v}$, i.e. $(v_2+1) \bmod N = u_2$. When we route a unit of traffic vertically along the column of the destination node, there are two disjoint paths leading to the destination node. One path is in the direction of the shortest cyclic distance to the destination node which will be called the v_s direction. The opposite of v_s direction will be called the v_l direction. Below are the steps of the recovering algorithm:

1. Shift co-ordinate by applying transformation $T_{\mathbf{v}}$ so that node $\mathbf{v}$ will be moved to the origin. Let $\mathbf{s}=(s_1,s_2)=T_{\mathbf{v}}(\mathbf{p})$ and $\mathbf{t}=(t_1,t_2)=T_{\mathbf{v}}(\mathbf{q})$.
2. Reverse the routing order of the primary routing path.
3. When route the traffic vertically, the direction (v_s or v_l) is determined by the following criteria:

Let $g(w)=\sum_{i=1}^{w} i$, $\gamma=\frac{1}{2}\sum_{i=1}^{(N-1)/2} i$, $a=\sum_{i=1}^{w} i-\left\lfloor \frac{1}{2}\sum_{i=1}^{(N-1)/2} i \right\rfloor$, and $b=\sum_{i=1}^{w} i-\left\lceil \frac{1}{2}\sum_{i=1}^{(N-1)/2} i \right\rceil$ where w is defined below:

(a) For $\mathbf{s}\in A_2$ and $\mathbf{t}\in L_4$, let $w=(N+1)/2-|s_2|$.
Case 1: $g(w)\leqslant\gamma$, choose v_l direction.
Case 2: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|t_2|\in\{0,\dots,(a-1)\}$, choose v_s direction.
Case 3: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|t_2|\in\{a,\dots,(N-1)/2-1\}$, choose v_l direction.
Case 4: $g(w)>\gamma$ and $g(w-1)>\gamma$, choose v_s direction.
(b) For $\mathbf{s}\in L_2$ and $\mathbf{t}\in A_3$, let $w=(N+1)/2-|t_2|-1$.
Case 1: $g(w)\leqslant\gamma$, choose v_l direction.
Case 2: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|s_2|\in\{1,\dots,b\}$, choose v_s direction.
Case 3: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|s_2|\in\{b+1,\dots,(N-1)/2\}$, choose v_l direction.
Case 4: $g(w)>\gamma$ and $g(w-1)>\gamma$, choose v_s direction.
(c) For $\mathbf{s}\in L_4$ and $\mathbf{t}\in A_1$, let $w=(N+1)/2-|t_2|$.
Case 1: $g(w)\leqslant\gamma$, choose v_l direction.
Case 2: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|s_2|\in\{0,\dots,(a-1)\}$, choose v_s direction.
Case 3: $g(w)>\gamma$, $g(w-1)\leqslant\gamma$, and $|s_2|\in\{a,\dots,(N-1)/2-1\}$, choose v_l direction.

Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.
(d) For $\mathbf{s} \in A_4$ and $\mathbf{t} \in L_2$, let $w = (N+1)/2 - |s_2| - 1$.
Case 1: $g(w) \leqslant \gamma$, choose v_l direction.
Case 2: $g(w) > \gamma$, $g(w-1) \leqslant \gamma$, and $|t_2| \in \{1, \ldots, b\}$, choose v_s direction.
Case 3: $g(w) > \gamma$, $g(w-1) \leqslant \gamma$, and $|t_2| \in \{b+1, \ldots, (N-1)/2\}$, choose v_l direction.
Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.
(e) For $\mathbf{s} \in L_2$ and $\mathbf{t} \in L_4$, route the traffic in the ring which contains the source **s** and destination **t**.
(f) For $\mathbf{s} \in L_4$ and $\mathbf{t} \in L_2$, route the traffic in a way such that the traffic cross-line α and β are evenly distributed.

With the restoration algorithm presented, we now investigate the additional amount of traffic added on each vertical link after rerouting the affected traffic. For a particular vertical link, the newly added traffic comes from rerouting the affected traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$ (traffic such that its source and destination nodes are not in the same vertical ring) and the affected traffic in the set $F_5 \cup F_6$ (traffic such that its source and destination nodes are in the same vertical ring). We first consider the amount of traffic added on an arbitrary vertical link by rerouting the traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$. To facilitate the calculation of the additional traffic added on the vertical link, we associate each node in the vertical ring which node $\mathbf{v}'$ belongs to with an integer number (shown in Figure 12) and consider N such that $\frac{1}{2}(\sum_{i=1}^{(N-1)/2} i)$ is an integer. In

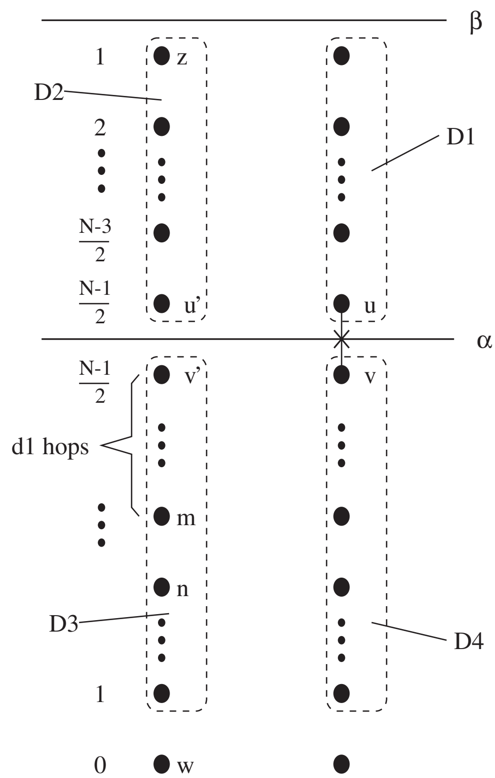

Figure 12. Numbering of nodes used in path-based restoration algorithm.

Figure 12, node **z** (associated with the number 1) will send one unit of traffic to nodes in D_4. Similarly, node $\mathbf{u}'$ (associated with the number $(N-1)/2$) will have $(N-1)/2$ units of traffic destined to nodes in D_4 by the primary routing algorithm. Also, before the link failure, traffic with source node in D_2 and destination node in D_4 will go through link $l_{\mathbf{uv}}$. After the link failure, these traffic will be routed in vertical direction first, and they have to go through either $l_{\mathbf{u'v'}}$ or $l_{\mathbf{wz}}$.

Without loss of generality, we consider the increment of the amount of traffic on an arbitrary vertical link $l_{\mathbf{mn}}$. The distance (hops) between node **m** and $\mathbf{v}'$ is denoted by d_1 (shown in Figure 12). Since the link $l_{\mathbf{mn}}$ is on the right side of the link $l_{\mathbf{uv}}$, only the traffic in the set $F_1 \cup F_2$ contributes to the traffic increment on $l_{\mathbf{mn}}$. Now, after rerouting the affected traffic in F_1 (traffic goes from D_2 to D_4), let us calculate the exact amount of traffic added on the link $l_{\mathbf{mn}}$.

First, we divide the nodes in D_2 into three subsets–$B_1 = \{\mathbf{s} \mid \mathbf{s} \in D_2 \text{ and } s_2 \in \{1, \ldots, \sigma - 1\}\}$, $B_2 = \{\mathbf{s} \mid \mathbf{s} \in D_2 \text{ and } s_2 \in \{\sigma\}\}$, and $B_3 = \{\mathbf{s} \mid \mathbf{s} \in D_2 \text{ and } s_2 \in \{\sigma + 1, \ldots, (N-1)/2\}\}$, where

$$\sigma = \left\lfloor \frac{1 + \sqrt{1 + 4\alpha}}{2} \right\rfloor \quad \text{and} \quad \alpha = \frac{1}{8}(N^2 - 1)$$

σ is the largest integer such that $\sum_{i=1}^{\sigma-1} \leqslant \frac{1}{16}(N^2 - 1)$. The reason that we introduce σ here is that we need to split the traffic in F_1 into two equal parts, with one part go through link $l_{\mathbf{u'v'}}$ and the other part go through $l_{\mathbf{wz}}$.

The following equations give us the amount of traffic in F_1 added on the link $l_{\mathbf{mn}}$. Let $\sigma_{\text{up}} = \frac{1}{2}\sum_{i=1}^{(N-1)/2} i - \sum_{i=1}^{\sigma-1} i$ and $\sigma_{\text{down}} = \sigma - \sigma_{\text{up}}$.

1. Traffic added on $l_{\mathbf{mn}}$ with source node in B_3, denoted as T_{B_3}, is

$$T_{B_3} = \begin{cases} \sum_{i=\sigma+1}^{(N-1)/2} i - \left(\frac{N-1}{2} - \sigma\right)(d_1 + 1) & \text{for } 0 \leqslant d_1 \leqslant \frac{N-1}{2} \\ 0 & \text{otherwise} \end{cases}$$

2. Traffic added on $l_{\mathbf{mn}}$ with source node in B_1, denoted as T_{B_1}, is

$$T_{B_1} = \begin{cases} \sum_{i=\sigma-1-d_1}^{\sigma-1} i & \text{if } d_1 + 1 < \sigma \\ \sum_{i=1}^{\sigma-1} i & \text{otherwise} \end{cases}$$

3. Traffic added on $l_{\mathbf{mn}}$ with source node in B_2 through the link $l_{\mathbf{wz}}$, denoted as $T_{B_{2a}}$, is

$$T_{B_{2a}} = \begin{cases} 0 & \text{if } d_1 + 1 \leqslant \sigma_{\text{down}} \\ d_1 + 1 - \sigma_{\text{down}} & \text{if } d_1 + 1 \leqslant \sigma \text{ and } d_1 + 1 > \sigma_{\text{down}} \\ \sigma_{\text{up}} & \text{if } d_1 + 1 > \sigma \end{cases}$$

4. Traffic added on $l_{\mathbf{mn}}$ with source node in B_2 through the link $l_{\mathbf{u'v'}}$, denoted as $T_{B_{2b}}$, is $T_{B_{2b}} = \max(0, \sigma_{\text{down}} - d_1 - 1)$.

Similarly, the following equations give us the amount of traffic in F_2 (traffic goes from D_4 to D_2) added on the link $l_{\mathbf{mn}}$.

$$T_{D_4} = \begin{cases} \sigma_{\text{up}} & \text{if } d_1 = \frac{N-1}{2} - \sigma \\ \sigma_{\text{down}} & \text{if } d_1 = \frac{N-1}{2} - \sigma - 1 \\ \sigma_{\text{up}} + \sum_{i=1}^{d_1-[(N-1)/2-\sigma]} (\sigma - i) & \text{if } d_1 > \frac{N-1}{2} - \sigma \\ \sigma_{\text{down}} + \sum_{i=1}^{((N-1)/2-\sigma)-(d_1+1)} (\sigma + i) & \text{if } d_1 + 1 < \frac{N-1}{2} - \sigma \end{cases}$$

Proof of Theorem 5

Proof

Again, we assume that an arbitrary vertical link connecting nodes **u** and **v** failed. Then, by showing separately that the rerouted traffic added on each horizontal link and on each vertical link are less or equal to $(N^3 - N)/4(2N - 1)$, we prove the minimum spare capacity needed on each link is $(N^3 - N)/4(2N - 1)$ for N odd. The amount of rerouted traffic added on a horizontal link will be investigated first. Pick an arbitrary horizontal link in the mesh and call it $l_{\mathbf{mn}}$ (the two nodes connecting this link are called **m** and **n**). From the primary routing algorithm, we know exactly what the affected traffic is and their routing paths. Let $n_{\mathbf{mn}}$ denotes the number of failed traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$ that go through the link $l_{\mathbf{mn}}$. After applying the restoration algorithm, $n_{\mathbf{mn}}$ units of failed traffic are removed from link $l_{\mathbf{mn}}$ and $n_{\mathbf{mn}}$ units of rerouted traffic are added on link $l_{\mathbf{mn}}$. Overall, traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$ does not affect the amount of traffic flow through link $l_{\mathbf{mn}}$ (i.e. no spare capacity needed on $l_{\mathbf{mn}}$ to restore the affected traffic in the set $F_1 \cup F_2 \cup F_3 \cup F_4$). However, traffic in the set $F_5 \cup F_6$ does add extra units of traffic on link $l_{\mathbf{mn}}$. But its amount is small, and it is less than $(N^3 - N)/4(2N - 1)$. Thus, we have shown that a spare capacity of $(N^3 - N)/4(2N - 1)$ on each horizontal link is enough to restore the original traffic by using the restoration algorithm.

Now, we calculate the amount of rerouted traffic added on a vertical link and show that it is less than $(N^3 - N)/4(2N - 1)$. Consider an arbitrary vertical link $l_{\mathbf{mn}}$ which is d_1 hops away from node v'. For the case of N such that $d_1 + 1 \leqslant \sigma_{\text{down}}$ and $d_1 + 1 < (N - 1)/2 - \sigma$, we calculate the amount of traffic in the set $F_1 \cup F_2$ added on the link $l_{\mathbf{mn}}$, which is called T_{F_1,F_2}.

$$T_{F_1,F_2} = T_{B_1} + T_{B_{2a}} + T_{B_{2b}} + T_{B_3} + T_{D_4} \quad \text{(A1)}$$

$$= \sum_{i=\sigma+1}^{(N-1)/2} i - \left(\frac{N-1}{2} - \sigma\right)(d_1 + 1) + \sum_{i=\sigma-1-d_1}^{\sigma-1} i + (\sigma_{\text{down}} - d_1 - 1) + \sigma_{\text{down}} + \sum_{i=1}^{((N-1)/2-\sigma)-(d_1+1)} (\sigma + i) \quad \text{(A2)}$$

$$= \sigma - N - d_1 + 2\sigma_{\text{down}} + \frac{1}{4}N^2 - \sigma^2 - Nd_1 + 2\sigma d_1 - \frac{5}{4} \tag{A3}$$

We then show that T_{F_1,F_2} is less than or equal to $\frac{1}{8}(N^2 - 1)$. Specifically,

$$\frac{1}{8}(N^2 - 1) - T_{F_1,F_2} = -\sigma + N + d_1(1+N) - 2\sigma_{\text{down}} - \frac{1}{8}N^2 + \sigma^2 - 2\sigma d_1 + \frac{9}{8} \tag{A4}$$

$$= (N - 2\sigma)(d_1 + 1) + 1 \tag{A5}$$

From Equations (A4) to (A5), the formula $2(\sum_{i=1}^{\sigma-1} i + \sigma_{\text{up}}) = \frac{1}{8}(N^2 - 1)$ was used. Since $\sigma < (N-1)/2$, T_{F_1,F_2} is less than or equal to $\frac{1}{8}(N^2 - 1)$.

For the case of $d_1 + 1 \geqslant \sigma_{\text{down}}$, $d_1 + 1 > (N-1)/2 - \sigma$, and $d_1 + 1 < \sigma$, we calculate that

$$T_{F_1,F_2} = T_{B_1} + T_{B_{2a}} + T_{B_{2b}} + T_{B_3} + T_{D_4} \tag{A6}$$

$$= \sum_{i=\sigma+1}^{(N-1)/2} i - \left(\frac{N-1}{2} - \sigma\right)(d_1 + 1) + \sum_{i=\sigma-1-d_1}^{\sigma-1} i + (d_1 + 1 - \sigma_{\text{down}}) + \sigma_{\text{up}} + \sum_{i=1}^{d_1-((N-1)/2-\sigma)} (\sigma - i) \tag{A7}$$

$$= -\sigma - d_1 + \sigma_{\text{up}} - \sigma_{\text{down}} + 2\sigma d_1 - d_1^2 \tag{A8}$$

and

$$\frac{1}{8}(N^2 - 1) - T_{F_1,F_2} = -2\sigma + d_1 + 2\sigma_{\text{down}} - 2\sigma d_1 + \frac{1}{8}N^2 + d_1^2 - \frac{1}{8} \tag{A9}$$

$$= (\sigma - d_1 - 1)(\sigma - d_1) \tag{A10}$$

Equation (A10) is positive since $d_1 + 1 < \sigma$. The other cases of d_1 (i.e. whether d_1 is less than or greater than σ_{down}) can be shown similarly. Thus, we have proved that the rerouted traffic from the set $F_1 \cup F_2 \cup F_3 \cup F_4$ added on any arbitrary vertical link is less than or equal to $\frac{1}{8}(N^2 - 1)$. Now, for the rerouted traffic from that set $F_5 \cup F_6$ (S–D pairs in the same vertical ring), there are total of $\frac{1}{4}(N^2 - 1)$ units of them. Simply routing half of these traffic within the vertical ring, we have now on each vertical link of the mesh an additional amount of rerouted traffic no greater than $\frac{1}{8}(N^2 - 1)$. The other half of the traffic in the set $F_5 \cup F_6$ ($\frac{1}{8}(N^2 - 1)$ units of them) can be rerouted evenly through $2N - 1$ vertical links crossing line α and β. Thus, the total rerouted traffic on each vertical link is no greater than $\frac{1}{8}(N^2 - 1) + [\frac{1}{8}(N^2 - 1)]/(2N - 1) = (N^3 - N)/4(2N - 1)$. Therefore, a spare capacity of $(N^3 - N)/4(2N - 1)$ on each link is enough for us to restore the original all-to-all traffic. □

ACKNOWLEDGEMENTS

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

REFERENCES

1. Xiong Y, Mason L. Restoration strategies and spare capacity requirements in self-healing ATM networks. In *Proceedings of INFOCOM '97*, vol. 1, 1997; 353–360.
2. Ramamurthy S, Mukherjee B. Survivable WDM mesh networks, Part I—protection. In *Proceedings of INFOCOM '99*, vol. 2, March 1999; 744–751.
3. Ramamurthy S, Mukherjee B. Survivable WDM mesh networks, Part II—restoration. In *ICC '99 Proceedings*, 1999; 2023–2030.
4. Iraschko RR, MacGregor MH, Grover WD. Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks. *IEEE/ACM Transactions on Networking* 1998; **6**:325–336.
5. Lumetta SS, Medard M. Towards a deeper understanding of link restoration algorithms for mesh networks. In *Proceedings of INFOCOM '01*, vol. 1, 2001; 367–375.
6. Azizoglu MC, Egecioglu O. Lower bounds on communication loads and optimal placements in torus networks. *IEEE Transactions on Computers* 2000; **49**(3):259–266.
7. Lemme PW, Glenister SM, Miller AW. Iridium aeronautical satellite communications. *IEEE Aerospace and Electronics Systems Magazine* 1999; **14**(11):11–16.
8. Patterson DP. Teledesic: a global broadband network. *1998 IEEE Aerospace Conference*, vol. 4, 1998; 547–552.
9. Ekici E, Akyildiz IF, Bender MD. A distributed routing algorithm for datagram traffic in LEO satellite networks. *IEEE/ACM Transactions on Networking* 2001; **9**(2):137–147.
10. Stamoulis GD, Tsitsiklis JN. Efficient routing schemes for multiple broadcasts in hypercubes. *IEEE Transactions on Parallel and Distributed Systems* 1993; **4**(7):725–739.
11. Varvarigos E. Efficient routing algorithms for folded-cube networks. In *Proceedings of the 1995 IEEE 14th Annual International Phoenix Conference on Computers and Communications*, 1995; 143–151.
12. Suh YJ, Shin KG. All-to-all personalized communication in multidimensional torus and mesh networks. *IEEE Transactions on Parallel and Distributed Systems* 2001; **12**(1):38–59.
13. Modiano E, Ephremides A. Efficient algorithms for performing packet broadcasts in a mesh network. *IEEE/ACM Transactions on Networking* 1996; **4**(4):639–648.
14. Bose B, Broeg R, Kwon Y, Ashir Y. Lee distance and topological properties of k-ary n-cubes. *IEEE Transactions on Computers* 1995; **44**(8):1021–1030.
15. Bertsekas DP. *Network Optimization: Continuous and Discrete Models*. Athena Scientific: Belmont, MA 1998.

