Capacity Provisioning, Failure Recovery and Throughput Analysis for Low Earth Orbit Satellite Constellations

by

Jun Sun

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2002

Author .....
Department of Electrical Engineering and Computer Science
August 9, 2002

Certified by .....
Eytan Modiano
Assistant Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Capacity Provisioning, Failure Recovery and Throughput Analysis for Low Earth Orbit Satellite Constellations

by

Jun Sun

Submitted to the Department of Electrical Engineering and Computer Science
on August 9, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

We investigate the capacity needed to build a restorable satellite network and design routing schemes to achieve high throughput. Specifically, the first part of this thesis considers the link capacity requirement for a 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. In the second part of this thesis, we consider a packet switching satellite network in which each node independently generates packets with a fixed probability during each time slot. With a limited number of transmitters and buffer space onboard each satellite, contention for transmission inevitably occurs as multiple packets arrived at a node. We consider three routing schemes in resolving these contentions: *Shortest Hops Win, Random Packet Win* and *Oldest Packet Win*; and evaluate their performance in terms of throughput. Under no buffer case, the throughput of the three schemes are significantly different. However, there is no appreciable difference in the throughput when buffer is available at each node. Also, a small buffer size at each node can achieve the same throughput performance as that of infinite buffer size. Simulations suggests that our theoretical throughput analysis is very accurate.

Thesis Supervisor: Eytan Modiano
Title: Assistant Professor

Acknowledgments

I would like to express my gratitude, first and foremost, to my advisor, Professor Eytan Modiano, for continuously giving me insights on the topic. His guidance has definitely refined my approach to research and problem solving.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Satellite networks provide global access to information, especially for users located in remote area where the communication infrastructure is inadequate. Recently, demand for satellite communication bandwidth for government, business, and individuals increases significantly. In military alone, it is projected that at least 16 gigabits per second of satellite communication bandwidth (more than three-fold of current bandwidth requirement) is required [22]. Satellite networks can also act as a safety valve for the Next Generation Internet. For example, failures in the fiber infrastructure or network congestion problems can be recovered easily by routing traffic through a satellite channel. For these reasons, here we investigate a future generation of satellite networks that are based on a constellation of low earth orbit (LEO) satellites.

Our work consists of two parts. The first part deals with the capacity provisioning and failure recovery in the LEO satellite network with a connection-oriented (circuit switching) network structure. Link failures and node (satellite) failures are not uncommon for satellite networks due to the potentially hazardous space weather (e.g., coronal mass ejections, solar flares, geomagnetic storm) which they are exposed to. Because of the diffculty associated with repairing the failed link or node, spare capacity is embeded in the network for restoration. To minimize the cost of adding such spare capacity in the 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. The second part of our work analyzes the network

throughput under various scheduling schemes in the LEO satellite network with a datagram (packet switching) network structure. Due to the increased popularity of the internet, there is an increase emphasis on the use of IP routing technology for both commercial and military satellites. With limited transmitters and buffer space on-board each satellite, contention for transmission inevitably occurs as multiple packets arrived at a node. We investigate several scheduling schemes for resolving contention and compare their performance in terms of throughput.

The thesis is organized as follows. In Chapter 2, we describe the network topology used to represent the satellite network, along with necessary definitions and problem statements. Capacity provisioning for satellite without any failure is also given. Then, we find the minimum spare capacity needed on each link in case of a single link or node failure. An algorithm for achieving the minimum spare capacity for a link failure is also presented. In Chapter3, we investigate the throughput of a packetized satellite network by using several different scheduling schemes during contention. The effect of buffer size on the throughput is also invetigated. Chapter 4 concludes the thesis.

Chapter 2

Capacity Provisioning and Failure Recovery for Low Earth Orbit Satellite Constellation

2.1 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, 2, 3, 5, 6]. 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, 2, 3].

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 this chapter, 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 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

	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 2.1: Capacity requirements under link based and path based restoration for a link failure.

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 2.1 summarizes capacity requirements under link based and path based restoration for link failure.

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 quantifies the capacity benefits of path over link restoration. In [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 work addresses the mesh structure for which we can get a closed form results for the spare capacity.

The structure of this chapter 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 summarizes this paper.

2.2 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 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

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

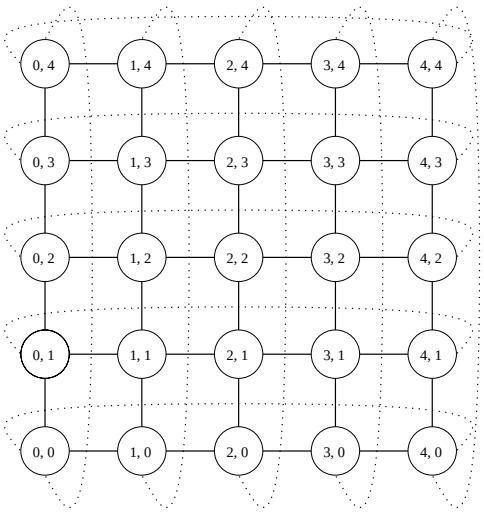


Figure 2-1: A 2-dimensional 5-mesh.

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. 3-1 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*

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

2.3 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 [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.

2.3.1 A Lower Bound on the Primary Capacity

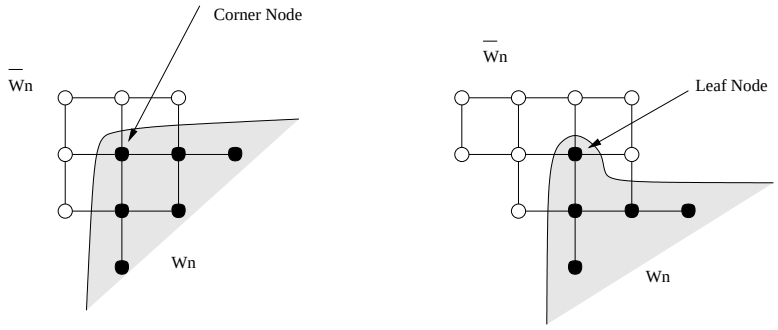


Figure 2-2: Representation of corner node and leaf node.

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. 2-2). 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. 2-3(a) is square and the shape of the set in Fig. 2-3(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*

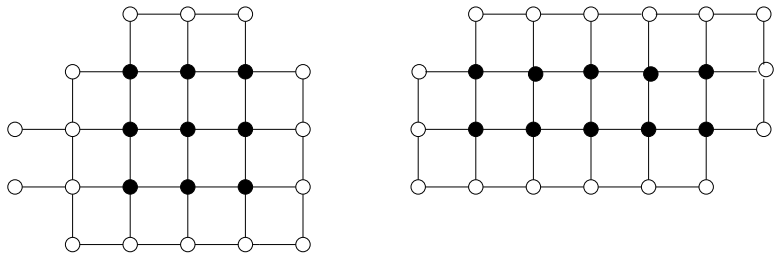


Figure 2-3: An illustration of the square shape and the rectangular shape.

$\varepsilon_\infty(n) = C(W_n, \overline{W_n})$ *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

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

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. 2-4. 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 that $\varepsilon_\infty(n^2 + k) \geq 4n + 2$ for $1 \leq k \leq n$.

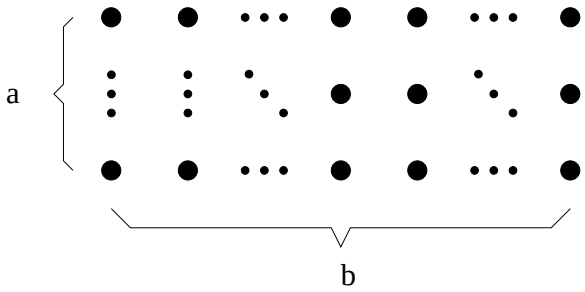


Figure 2-4: An arrangement of n^2 nodes in rectangular shape.

Then, by construction, $\varepsilon_\infty(n^2+k) = 4n+2$ for $1 \le k \le n$. 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, $\varepsilon_\infty(n^2+1) \ge 4n+2$. By the monotonicity of $\varepsilon_\infty(\cdot)$, $\varepsilon_\infty(n^2+k) \ge 4n+2$ for $1 \le k \le n$. 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 $1 \le k \le n$.

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

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

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

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

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

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

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

and

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

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:

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

$$\mathcal{O}_2 \equiv \{x \mid x \in \{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\} \text{ and } (x \; mod \; N) \neq 0\}, \text{ and}$$

$$\mathcal{O}_3 \equiv \{x \mid x \in \{\frac{N^2-1}{4}+1, \ldots, \frac{N^2+1}{2}\} \text{ and } (x \; mod \; N) = 0\}.$$

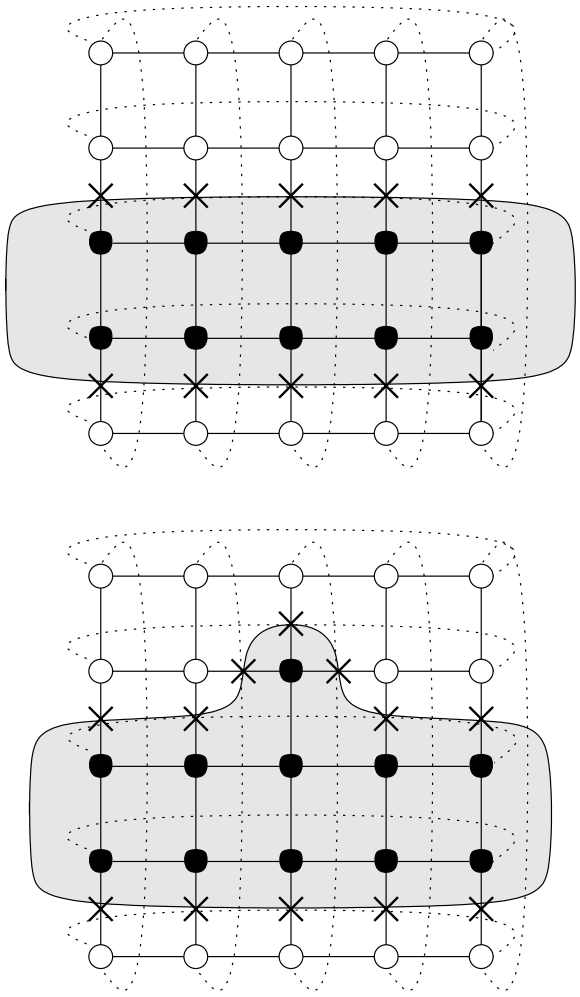


Figure 2-5: Ways of splitting the N-mesh into two disjoint parts.

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

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

for N odd,

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

Proof. From Fig. 2-5, we see that $\varepsilon_N(n) \leq 2N \ \forall n$ such that $(n \ mod \ N) = 0$ and $\varepsilon_N(n) \leq 2N+2$ if $(n \ mod \ N) \neq 0$. 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. 2-5, 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. 2-5 to separate the sets. □

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 $B_{max}^{C_1}$ 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

$$B_{max}^{C_1} = \max_{n \in \{1,\ldots,N^2-1\}} \left\lceil \frac{2(N^2-n)n}{\varepsilon_N(n)} \right\rceil \tag{2.1}$$

$$= \max\left\{\max_{n\in\mathcal{A}_1}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)}\right], \max_{n\in\mathcal{A}_2}\left[\frac{2(N^2-n)n}{2N+2}\right], \max_{n\in\mathcal{A}_3}\left[\frac{2(N^2-n)n}{2N}\right]\right\}. \quad (2.2)$$

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

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

where α_e (α_o) in the above equation is the result of the first term of equation (2.2) 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

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

□

Corollary 3. *On a 2-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 $\frac{N^4}{2(2N-1)}$ 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

$$B_{max}^{C_2} = \max_{n\in\{1,\ldots,N^2-1\}} \left[\frac{2(N^2-n)n}{\varepsilon_N(n)-1}\right] \tag{2.3}$$

$$= \max\left\{\max_{n\in\mathcal{A}_1}\left[\frac{2(N^2-n)n}{\varepsilon_\infty(n)-1}\right], \max_{n\in\mathcal{A}_2}\left[\frac{2(N^2-n)n}{2N+2-1}\right], \max_{n\in\mathcal{A}_3}\left[\frac{2(N^2-n)n}{2N-1}\right]\right\} \tag{2.4}$$

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

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

where α_e (α_o) in the above equation is the result of the first term of equation (2.4) 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

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

□

2.3.2 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 $\vec{p} = (p_1, p_2)$, 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.

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

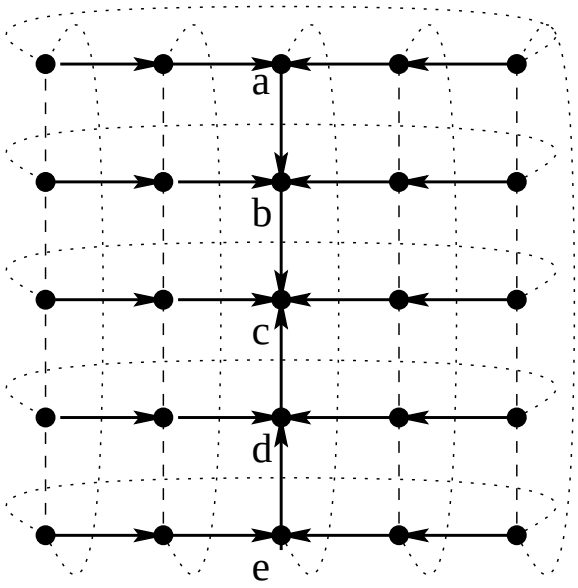


Figure 2-6: An illustration of traffic flow into node c by using Dimensional Routing Algorithm.

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. 2-6. 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. 2-6 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.

2.4 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.

2.4.1 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 $\frac{N^3-N}{4}$ unit of traffic on $l_{\vec{u}\vec{v}}$ have to be rerouted for N odd and $\frac{N^3}{4}$ for N 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. 2-7. 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. □

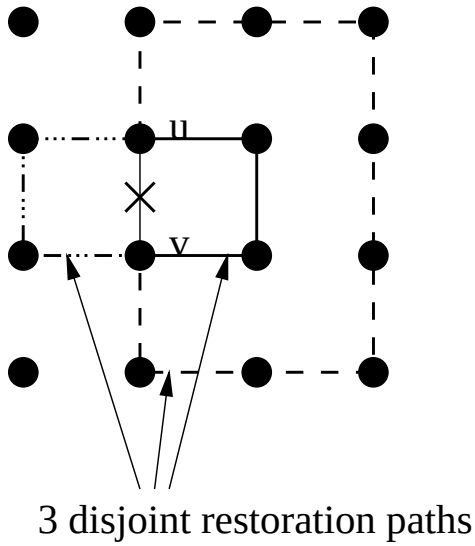


Figure 2-7: Restoration paths using link based recovery scheme.

2.4.2 Path Based Restoration Strategy

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

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

□

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).

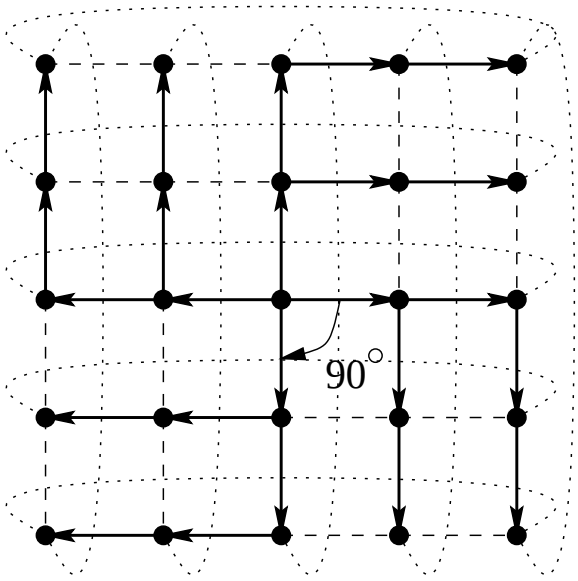


Figure 2-8: Routing path of the Rotational Symmetric routing algorithm. Rotating the graph by 90° does not change the configuration.

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 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 hori-

zontal 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. 2-8).

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 $T_{\vec{p}}(q_1, q_2) = (d_1, d_2)$, 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. 2-9 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. 2-9). Specfically, let

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

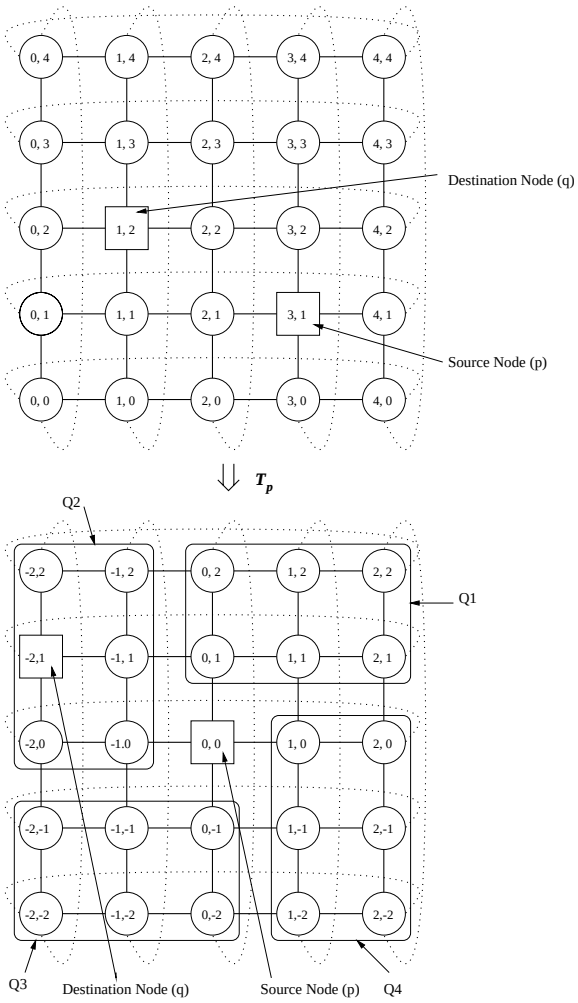


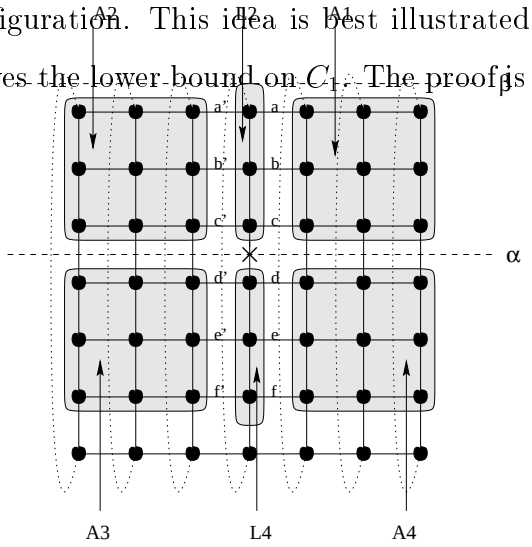
Figure 2-9: Change of coordinate by using transformation $T_{\vec{p}}$.

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.

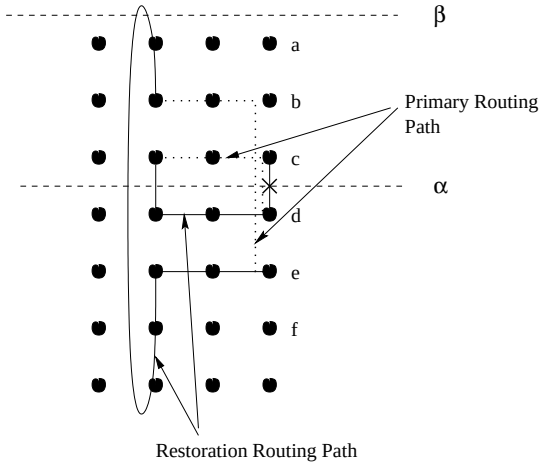
   If $\vec{d} = T_{\vec{p}}(\vec{q}) \in (\mathcal{Q}_2 \cup \mathcal{Q}_4)$, route the traffic horizontally in the direction of shortest cyclic distance to the destination node by $D_N(p_1, q_1)$ hops. Then, route the traffic vertically in the direction of shortest cyclic distance to the destination node 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. 2-8. *RS routing algorithm* also achieves the lower bound on C_1. The proof is straightforward and thus omitted here.



(a)



(b)

Figure 2-10: Routing path of the restoration algorithm

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 $l_{\vec{cd}}$ failed in the 2-dimensional 7-mesh shown in Fig. 2-10(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. Let F denote the set of all possible such S-D pairs. Then, we have $F = F_1 \cup F_2 \cup F_3 \cup F_4 \cup F_5 \cup F_6$ where

$$
\begin{aligned}
F_1 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in A_2 \text{ and } \vec{t} \in L_4\},\\
F_2 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in L_2 \text{ and } \vec{t} \in A_3\},\\
F_3 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in A_4 \text{ and } \vec{t} \in L_2\},\\
F_4 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in L_4 \text{ and } \vec{t} \in A_1\},\\
F_5 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in L_4 \text{ and } \vec{t} \in L_2\}, \text{ and}\\
F_6 &= \{(\vec{s},\vec{t}) \in F \mid \vec{s} \in L_2 \text{ and } \vec{t} \in L_4\}.
\end{aligned}
$$

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. 2-10(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:

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

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. 2-10(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. 2-10(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. 2-11(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. 2-11(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. 2-11(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.

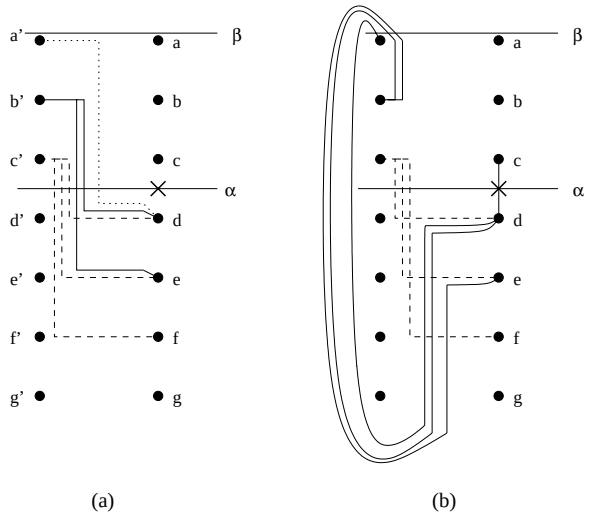


Figure 2-11: Restoration path for the 2-dimensional 7-mesh

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. 2-10(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 ($\vec{s} \in A_2, \vec{t} = \vec{d}$) 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.

Next, we present the full detail of the path based restoration algorithm. We also show that the lower bound on the spare capacity $(C_2 - C_1)$ is indeed achievable.

Path based restoration algorithm

Again, we focus on the case of N odd for simplicity. From the source node $\vec{p}$ to the destination node $\vec{q}$, 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 $\vec{u}$ and $\vec{v}$ with node $\vec{u}$ on the top of $\vec{v}$, i.e. $(v_2 + 1)\ mod\ N = u_2$. 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 coordinate by applying transformation $T_{\vec{v}}$ so that node $\vec{v}$ will be moved to the origin. Let $\vec{s} = (s_1, s_2) = T_{\vec{v}}(\vec{p})$ and $\vec{t} = (t_1, t_2) = T_{\vec{v}}(\vec{q})$.

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

   (a) For $\vec{s} \in A_2$ and $\vec{t} \in L_4$, let $w = \frac{N+1}{2} - |s_2|$.

   Case 1: $g(w) \leq \gamma$, choose v_l direction.

   Case 2: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|t_2| \in \{0, \cdots, (a-1)\}$, choose v_s direction.

   Case 3: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|t_2| \in \{a, \cdots, \frac{N-1}{2} - 1\}$, choose v_l direction.

   Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.

   (b) For $\vec{s} \in L_2$ and $\vec{t} \in A_3$, let $w = \frac{N+1}{2} - |t_2| - 1$.

   Case 1: $g(w) \leq \gamma$, choose v_l direction.

   Case 2: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|s_2| \in \{1, \cdots, b\}$, choose v_s direction.

   Case 3: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|s_2| \in \{b+1, \cdots, \frac{N-1}{2}\}$, choose v_l direction.

   Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.

   (c) For $\vec{s} \in L_4$ and $\vec{t} \in A_1$, let $w = \frac{N+1}{2} - |t_2|$.

   Case 1: $g(w) \leq \gamma$, choose v_l direction.

   Case 2: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|s_2| \in \{0, \cdots, (a-1)\}$, choose v_s direction.

   Case 3: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|s_2| \in \{a, \cdots, \frac{N-1}{2} - 1\}$, choose v_l direction.

   Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.

(d) For $\vec{s} \in A_4$ and $\vec{t} \in L_2$, let $w = \frac{N+1}{2} - |s_2| - 1$.

Case 1: $g(w) \leq \gamma$, choose v_l direction.

Case 2: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|t_2| \in \{1, \cdots, b\}$, choose v_s direction.

Case 3: $g(w) > \gamma$, $g(w-1) \leq \gamma$, and $|t_2| \in \{b+1, \cdots, \frac{N-1}{2}\}$, choose v_l direction.

Case 4: $g(w) > \gamma$ and $g(w-1) > \gamma$, choose v_s direction.

(e) For $\vec{s} \in L_2$ and $\vec{t} \in L_4$, route the traffic in the ring which contains the souce $\vec{s}$ and destination $\vec{t}$.

(f) For $\vec{s} \in L_4$ and $\vec{t} \in L_2$, 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 $\vec{v'}$ belongs to with an integer number (shown in Fig. 2-12) and consider N such that $\frac{1}{2}(\sum_{i=1}^{\frac{N-1}{2}} i)$ is an integer. In Fig. 2-12, node $\vec{z}$ (associated with the number 1) will send one unit of traffic to nodes in D_4. Similarly, node $\vec{u'}$ (associated with the number $\frac{N-1}{2}$) will have $\frac{N-1}{2}$ 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 $l_{\vec{u}\vec{v}}$. After the link failure, these traffic will be routed in vertical direction first, and they have to go through either $l_{\vec{u'}\vec{v'}}$ or $l_{\vec{w}\vec{z}}$.

Without loss of generality, we consider the increment of the amount of traffic on an arbitrary vertical link $l_{\vec{m}\vec{n}}$. The distance (hops) between node $\vec{m}$ and $\vec{v'}$ is denoted

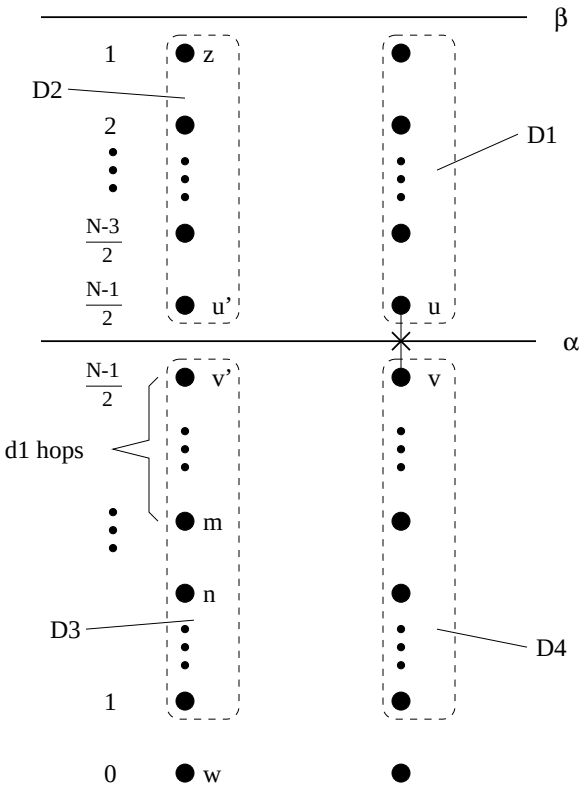


Figure 2-12: Numbering of nodes used in path based restoration algorithm

by d_1 (shown in Fig. 2-12). Since the link $l_{\vec{m}\vec{n}}$ is on the right side of the link $l_{\vec{u}\vec{v}}$, only the traffic in the set $F_1 \cup F_2$ contributes to the traffic increment on $l_{\vec{m}\vec{n}}$. Now, after rerouting the affected traffic in F_1 (traffic goes from D_2 to D_4), let's calculate the exact amount of traffic added on the link $l_{\vec{m}\vec{n}}$.

First, we dvide the nodes in D_2 into three subsets–$B_1 = \{\vec{s} \mid \vec{s} \in D_2 \text{ and } s_2 \in \{1, \cdots, \sigma - 1\}\}$, $B_2 = \{\vec{s} \mid \vec{s} \in D_2 \text{ and } s_2 \in \{\sigma\}\}$, and $B_3 = \{\vec{s} \mid \vec{s} \in D_2 \text{ and } s_2 \in \{\sigma + 1, \cdots, \frac{N-1}{2}\}\}$, where $\sigma = \lfloor \frac{1+\sqrt{1+4\alpha}}{2} \rfloor$ and $\alpha = \frac{1}{8}(N^2 - 1)$. σ is the largest integer such that $\sum_{i=1}^{\sigma-1} \leq \frac{1}{16}(N^2 - 1)$. 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 $l_{\vec{u'}\vec{v'}}$ and the other part go through $l_{\vec{w}\vec{z}}$.

The following equations give us the amount of traffic in F_1 added on the link $l_{\vec{m}\vec{n}}$. Let $\sigma_{up} = \frac{1}{2}\sum_{i=1}^{\frac{N-1}{2}} i - \sum_{i=1}^{\sigma-1} i$ and $\sigma_{down} = \sigma - \sigma_{up}$.

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

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

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

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

3. Traffic added on $l_{\vec{m}\vec{n}}$ with source node in B_2 through the link $l_{\vec{w}\vec{z}}$, denoted as $T_{B_{2a}}$, is

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

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

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 $l_{\vec{m}\vec{n}}$.

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

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.*

Proof. Again, we assume that an arbitrary vertical link connecting nodes $\vec{u}$ and $\vec{v}$ failed. Then, by showing separately that the rerouted traffic added on each horizontal link and on each vertical link are less or equal to $\frac{N^3-N}{4(2N-1)}$, we prove the minimum spare capacity needed on each link is $\frac{N^3-N}{4(2N-1)}$ 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 $l_{\vec{m}\vec{n}}$ (the two nodes connecting this link are called $\vec{m}$ and $\vec{n}$). From the primary routing algorithm, we know exactly what the affected traffic is and their routing paths. Let $n_{\vec{m}\vec{n}}$ 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 $l_{\vec{m}\vec{n}}$. After applying the restoration algorithm, $n_{\vec{m}\vec{n}}$ units of failed traffic are removed from link $l_{\vec{m}\vec{n}}$ and $n_{\vec{m}\vec{n}}$ units of rerouted traffic are added on link $l_{\vec{m}\vec{n}}$. 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 $l_{\vec{m}\vec{n}}$ (i.e. no spare capacity needed on $l_{\vec{m}\vec{n}}$ 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 $l_{\vec{m}\vec{n}}$. But its amount is small, and it is less than $\frac{N^3-N}{4(2N-1)}$. Thus, we have shown that a spare capacity of $\frac{N^3-N}{4(2N-1)}$ 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 $\frac{N^3-N}{4(2N-1)}$. Consider an arbitrary vertical link $l_{\vec{m}\vec{n}}$ which is d_1 hops away from node v'. For the case of N such that $d_1 + 1 \leq \sigma_{down}$ and $d_1 + 1 < \frac{N-1}{2} - \sigma$, we calculate the amount of traffic in the set $F_1 \cup F_2$ added on the link $l_{\vec{m}\vec{n}}$, which is called T_{F_1,F_2}.

$$\begin{aligned} T_{F_1,F_2} &= T_{B_1} + T_{B_{2a}} + T_{B_{2b}} + T_{B_3} + T_{D_4} \qquad (2.5) \\ &= \sum_{i=\sigma+1}^{\frac{N-1}{2}} i - (\frac{N-1}{2} - \sigma)(d_1 + 1) \\ &\quad + \sum_{i=\sigma-1-d_1}^{\sigma-1} i + (\sigma_{down} - d_1 - 1) + \sigma_{down} \end{aligned}$$

$$+ \sum_{i=1}^{(\frac{N-1}{2}-\sigma)-(d_1+1)} (\sigma+i) \tag{2.6}$$

$$= \sigma - N - d_1 + 2\sigma_{down} + \frac{1}{4}N^2 - \sigma^2$$
$$-Nd_1 + 2\sigma d_1 - \frac{5}{4} \tag{2.7}$$

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

$$\frac{1}{8}(N^2-1) - T_{F_1,F_2} = -\sigma + N + d_1(1+N) - 2\sigma_{down}$$
$$-\frac{1}{8}N^2 + \sigma^2 - 2\sigma d_1 + \frac{9}{8} \tag{2.8}$$
$$= (N-2\sigma)(d_1+1) + 1 \tag{2.9}$$

From Eq.2.8 to Eq.2.9, the formula $2(\sum_{i=1}^{\sigma-1} i + \sigma_{up}) = \frac{1}{8}(N^2-1)$ was used. Since $\sigma < \frac{N-1}{2}$, T_{F_1,F_2} is less than or equal to $\frac{1}{8}(N^2-1)$.

For the case of $d_1+1 \geq \sigma_{down}$, $d_1+1 > \frac{N-1}{2} - \sigma$, and $d_1+1 < \sigma$, we calculate that

$$T_{F_1,F_2} = T_{B_1} + T_{B_{2a}} + T_{B_{2b}} + T_{B_3} + T_{D_4} \tag{2.10}$$
$$= \sum_{i=\sigma+1}^{\frac{N-1}{2}} i - (\frac{N-1}{2} - \sigma)(d_1+1)$$
$$+ \sum_{i=\sigma-1-d_1}^{\sigma-1} i + (d_1 + 1 - \sigma_{down})$$
$$+\sigma_{up} + \sum_{i=1}^{d_1-(\frac{N-1}{2}-\sigma)} (\sigma - i) \tag{2.11}$$
$$= -\sigma - d_1 + \sigma_{up} - \sigma_{down} + 2\sigma d_1 - {d_1}^2 \tag{2.12}$$

and

$$\frac{1}{8}(N^2-1) - T_{F_1,F_2} = -2\sigma + d_1 + 2\sigma_{down} - 2\sigma d_1$$
$$+\frac{1}{8}N^2 + {d_1}^2 - \frac{1}{8} \tag{2.13}$$

$$= \quad (\sigma - d_1 - 1)(\sigma - d_1) \tag{2.14}$$

Eq.2.14 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've 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 thet 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 $\frac{1}{8}(N^2-1) + [\frac{1}{8}(N^2-1)]/(2N-1) = \frac{N^3-N}{4(2N-1)}$. Therefore, a spare capacity of $\frac{N^3-N}{4(2N-1)}$ on each link is enough for us to restore the original all-to-all traffic.

□

2.5 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 2-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 $\frac{N^2(N-4)}{4(2N-1)}$ for N even and $\frac{N(N^2-4N+3)}{4(2N-1)}$ for N odd.*

The proof of this theorem follows the similar steps in the proofs of theorem 1

and theorem 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 $\frac{1/2(N^2-1)N^2-N(N-1)}{2N-1}$. 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 $[\frac{1/2(N^2-1)N^2-N(N-1)}{2N-1}] - C_1$ 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 $\frac{(N-1)^2}{4}$. After rerouting the affected traffic, the newly added traffic on each link is at most $\lceil \frac{1}{8}N^2 - \frac{9}{8} + \frac{(N-1)^2}{4} \rceil$. 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 the appendix. 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.

2.6 Summary

This chapter 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 [5]).

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.

Chapter 3

Throughput Analysis in Satellite Network

3.1 Introduction

Satellite networks provide a global coverage and support a wide range of data communication needs of businesses, government, and individuals [11]. It is foreseeable that both LEO (Low Earth Orbit) and GEO (Geostationary Orbit) networks will constitute an essential part of the Next-Generation Internet. Thus, future generations of satellite networks are envisioned to provide integrated services that carry a wide range of data types. Currently, connection-oriented routing (circuit switching) has been the focus of LEO satellite networks. Little analysis has been done in the performance of packet switching satellite network. In this work, we address the throughput of a packet switching satellite network.

We model the satellite network as a $N \times N$ mesh-torus where each satellite has k transmitters and m receivers. We focus on the case of $k = 1$ and $m = 4$ (i.e., the satellite can transmit to only one of its neighbors and receive from all of its neighbors simultaneously). This assumption was used in [4] and follows from the use of optical beams or highly directive antennas for communication. The analysis of the more general m receivers and k transmitters case can be done by following the similar steps shown in this chapter. We further assume that each satellite uses its only

transmitter onboard for both inter-satellite communication and satellite-to-ground communication. However, as we show later, our results can also be applied to the case where each satellite has two transmitters: one for inter-satellite communication and the other for satellite-to-ground communication.

We consider fixed shortest path routing schemes (e.g., Rotational Symmetrical Routing in Sec. 2.4.2) for node-to-node communication in mesh satellite networks, and we analyze their performance under a stochastic traffic environment. In particular, we assume that packets having a single destination are generated at each node of the mesh according to some probabilistic rule. The destination of the new packet is uniformly distributed over all mesh nodes (except its source node).

The network operation is similar to one discussed in [18]. That is, the nodes operate synchronously: the time axis is divided into slots and each node can relay one packet per time slot. A *new* packet is generated independently at each node locally with probability p_0 during each time slot. Thus, the arrival process of new packets is modeled as a Bernoulli process with rate p_0 packets per time slot. At the end of a slot there are u *continuing* packets (received from other neighboring nodes during this time slot) offered to the node. Since at most one packet can be received at each receiver, u is less than or equal to four. As more than one packets arrive at a particular node, contention for transmission in the next time slot will occur. Hence, we need to develop a transmission scheduling scheme for resolving this conflict. After a packet arrives at its destination node at the end of a time slot, it is not immediately removed from the system since it has to be sent to the ground. Instead, this packet has to compete with other incoming packets for transmission in the next time slot. In the case of each satellite having two transmitters, a packet is removed from the system as soon as it arrives its destination node, provided that the dedicated transmitter for satellite-to-ground communication can send the data fast enough (i.e., no downlink contention).

Routing schemes for solving packets' contention in a regular topology have been investigated by numerous researchers. In [16], Greenberg and Hajek provided an approximate analysis of the transient and steady state behavior of deflection routing

in hypercube network. Stamoulis and Tsitsiklis [17] studied the efficiency of greedy routing in hypercube network. In [18], the authors propose two different hypercube routing schemes and evaluate the throughput of both the buffered and the unbuffered version of these schemes. Their results are also approximate. In all the aforementioned papers, the topology that they used is hypercube.

In this chapter, we propose several scheduling schemes and compare, for each scheme, the average throughput of the network when it reaches steady state. Specifically, we study the throughput of *Shortest Hop Win* (SHW) scheme, *Oldest Packet Win* scheme (OPW), and *Random Packet Win* (RPW) scheme. Both the analytic and simulated results show that, in the case of no buffer at each node, SHW scheme attains the best throughput performance, OPW scheme the second, and RPW the worst. When there is a buffer at each node, the performance of the three schemes have no appreciable difference. Also, a small buffer size can achieve throughput close to that of an infinite buffer size. In all of the three schemes mentioned above, we give the newly generated packet the lowest priority (i.e., new packet can enter the system only if there is no continuing packet and no buffered packet). Therefore, most of the packet drops occur because new packets cannot enter the system.

The structure of this chapter is as follows. In section 2, we describe the stability region of the network under uniform traffic. Section 3.1 and 3.2 provide an approximate theoretical analysis of the thoughput of Shortest Hop Win scheme for both the buffer and no buffer cases. Simulation results of the throughput are also presented. Theoretical analysis of the throughput of Oldest Packet Win scheme is given in sections 3.3 and 3.4. In section 3.5, we compare the throughput of these three schemes in case of no buffer at each node. In section 3.6, for the buffer case, we describe a few additional routing schemes of interest and compare their performance with the three aforementioned schemes. Section 3.7 investigates the throughput performance in relation with the buffer size. Section 4 summarizes this chapter.

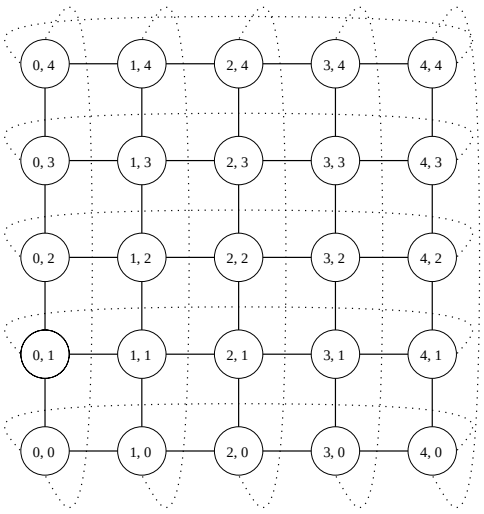


Figure 3-1: A 2-dimensional 5-mesh.

3.2 Stability Analysis

We consider now an $N \times N$ mesh (shown in Fig. 3-1) with N^2 nodes, each of which generates packets independently according to a random process of rate λ packets per second to be sent to a uniformly chosen destination node. The packet takes exactly one unit of time to be transmitted. Each node can only transmit to one of its neighbors during a given slot, but can receive packets from all neighbors simultaneously. We first derive a necessary condition for stability.

Theorem 7. *A necessary condition for the system to be stable is*

$$\lambda(E[d] + 1) < 1 \tag{3.1}$$

where $E[d]$ is the expected number of hops from source node to destination node.

Proof. The total number of new packets generated in the network per unit time is λm, where $m = N^2$. During each time unit, an average total demand of $\lambda \cdot m \cdot (E[d] + 1)$ packet transmissions are generated in the system, where $E[d] + 1$ is the expected number of hops from source node to destination node plus the last hop from satellite to ground. Since at most m transmissions may take place per unit of time, we have $\lambda m(E[d] + 1) < m$, or $\lambda(E[d] + 1) < 1$. □

Next, we will show that by employing a fixed shortest path routing scheme(i.e., every node sends out traffic according to the same set of routing rules via the shortest path to the destination node, for example, the *Rotational Symmetrical Routing Algorithm* described in the previous chapter), the network is stable for all $\lambda(E[d]+1) < 1$. We first present several lemmas that will be useful. The following lemma is from [4].

Lemma 4. *Consider any start node x and let $n_x(i)$ be the number of nodes exactly i hops away from node x. Then*

$$n_x(i) = \begin{cases} 1 & , \ i = 0 \\ 4i & , \ 0 < i < \frac{N}{2} \\ 4i - 2 & , \ i = \frac{N}{2} \\ 4(N-i) & , \ \frac{N}{2} < i < N \\ 1 & , \ i = N \end{cases}$$

Proof. See [4]. □

Next, consider a scenario in which every node of the network sends out one unit of traffic to every other node (also known as *complete exchange* or *all-to-all communication*) [7] by using a fixed shortest path routing algorithm. Each source-destination pair uniquely defines a different class of traffic. The load of a particular link is defined to be the number of different classes of traffic that pass through that link. We are interested in the average load of a link under all-to-all traffic.

For a 2-dimensional N-mesh, the total number of nodes in the network is N^2; the total number of unordered node pairs is $\frac{N(N-1)}{2}$; and the total number of links is $2N^2$. The following lemma gives us the average load of a link under all-to-all traffic.

Lemma 5. *For a 2-dimensional N-mesh under all-to-all traffic, the average load of a link is $\frac{1}{4}(N^3 - N)$ for N odd, and $\frac{1}{4}N^3$ for N even by using a fixed shortest path routing algorithm.*

Proof. We first consider the case where N is odd. From Lemma 4, we see that there are a total of $\frac{1}{2}(4i)N^2$ unordered pairs that are i hops away from each other for

$0 < i \leq \frac{N-1}{2}$, and a total of $\frac{1}{2}(4(N-i))N^2$ unordered pair for $\frac{N-1}{2} < i \leq N-1$. Here, the maximum length between two nodes in the network is $N-1$.

Let D denote the average path length between two nodes. We then have

$$D = \frac{\sum_{i=1}^{\frac{N-1}{2}} i(2iN^2) + \sum_{i=\frac{N-1}{2}+1}^{N-1} i(2(N-i)N^2)}{(\frac{N(N-1)}{2})} = \frac{N^3-N}{2(N^2-1)}.$$

The total traffic in this network is $N^2(N^2-1)\frac{N^3-N}{2(N^2-1)} = \frac{1}{2}(N^5-N^2)$. Thus, since all links have the same load due to the symmetry of the network and the fixed shortest path routing, the average load on a link is the total traffic divided by the number of links $\frac{1}{2}(N^5-N^2)/2N^2 = \frac{1}{4}(N^3-N)$.

The case for N even can be shown similarly. □

Now, assuming there is a separate buffer for each class of traffic that is going to be served at a node, we define a service policy u_0 to be the round-robin service discipline. That is, the transmitter serves each queue with an equal amount of time. In the case of an empty queue, the transmitter will be idle for a period of time that is allocated to that queue. Then, we have the following theorem:

Theorem 8. *With packet's arrival rate λ and the destination node uniformly chosen, the 2-dimensional N-mesh network is stable for all $\lambda < \frac{1}{E[d]+1}$, where $E[d] = \frac{N}{2}$ is the expected length between two nodes, under a fixed shortest path routing scheme and policy u_0.*

Proof. Consider a 2-dimensional N-mesh for N odd. There is a total of $N^2(N^2-1)$ classes of traffic in this queueing network, each corresponding to a unique source-destination pair (i,j). For an arbitrary node k in the network, since the packets are arriving at a rate of λ externally and destinations are uniformly chosen, packets of class (k,j) arrives at the rate of $\frac{\lambda}{N^2-1}$ for all nodes $j \neq k$. Because of the fixed shortest path routing scheme, we know exactly how many classes of traffic need to be served at node k. Specifically, from Lemma 5, for the four links connecting node k, each of them has $\frac{1}{2} \cdot \frac{1}{4}(N^3-N)$ classes of traffic that are required to go to or through node k (the term $\frac{1}{2}$ is there because we only consider the traffic *coming into* node k). We

call these classes of traffic internal arrivals, and the $N^2 - 1$ classes of traffic which are generated locally external arrivals. Since policy u_0 serves each class in a round robin fashion, a constant fraction of service is allocated to each class of traffic. Under policy u_0, node k can be viewed as having many dedicated servers (one for each class of traffic) with identical service rate. Hence, all queues at node k are independent, and they are stable as long as the service rate is greater than the arrival rate for each class of traffic. A paticular class of traffic may go though several nodes to reach its destination. If all nodes on its path to the destination are serving this class of traffic at a rate greater than the arrival rate, the series of queues are also stable (Theorem 7.4.12,[21]). Then the total internal arrival rate is

$$4 \cdot \frac{1}{8}(N^3 - N) \cdot (\text{arrival rate of a single class}) = \frac{1}{8}(N^3 - N) \cdot \frac{\lambda}{N^2 - 1}$$

Therefore, the total arrival rate to node k (the sum of the external and internal arrival rates) is

$$4 \cdot \frac{1}{8} \cdot (N^3 - N) \cdot \frac{\lambda}{N^2 - 1} + \lambda = \lambda(\frac{N}{2} + 1)$$

Consider a service discipline with service rate of 1. Thus, for the queue to be stable, we must have

$$\lambda < \frac{1}{E[d] + 1}$$

□

Moreover, for routing schemes that choose a random shortest path between the source and destination node, it can be shown that the stability region is still $\lambda < \frac{1}{E[d]+1}$.

3.3 Analysis and Simulation of Throughput

In this section, we present the main results of this work. Serveral scheduling schemes for resolving contention for transmission are discussed. Detailed theoretical analysis and simulation results of throughput are provided. First, we give a general overview of these transmission schemes which will be analyzed in the later sections. We assume that, at each node, there is buffer which can hold up to k packets, in addition to the packet under transmission. Because only one transmitter is available at each node, conflicts result from simultaneous arrivals of more than one packet from the neighboring nodes or a new packet generated in the current node. Even if a packet has reached its destination node, this packet has to compete with other packets to be sent to the ground in the next time slot. Contention may be resolved by assigning different priority to the incoming packets (both the continuing packets and the new packet). Below we propose several schemes to resolve the contention. In all schemes, packets follow fixed shortest paths to their destination nodes.

1. Shortest Hop Win (SHW): If more than one continuing packets arrive at a node, SHW chooses the one with the shortest hop distance to its destination node to be transmitted in the next time slot. The other packets are stored in the buffer if there is space available. When the buffer space cannot accommodate all of the continuing packets that need to be stored in the buffer, SHW randomly picks packets among these continuing packets to fill up the buffer (the other packets are dropped). In case of no continuing packet arriving, SHW picks the head of buffer packet to be transmitted in the next time slot. If the buffer is empty, SHW sends the newly generated packet (if there is one) in the next time slot. In case of contention, new packets are discarded.

2. Random Packet Win (RPW): If more than one continuing packets arrive at a node, RPW randomly chooses the one to be transmitted in the next time slot. The other packets are stored in the buffer if there is space available. When the buffer space cannot accommodate all of the continuing packets that need to be stored in the buffer, RPW randomly selects packets among these

continuing packets to fill up the buffer (the other packets are dropped). In case of no continuing packet arriving, RPW picks the head of buffer packet to be transmitted in the next time slot. If the buffer is empty, RPW sends the newly generated packet (if there is one) in the next time slot. In case of contention, new packets are discarded.

3. Oldest Packet Win (OPW): If more than one continuing packets arrive at a node, OPW chooses the one that has travelled the most hops to be transmitted in the next time slot. The other packets are stored in the buffer if there is space available. When the buffer space cannot accommodate all of the continuing packets that need to be stored in the buffer, OPW randomly selects packets among these continuing packets to fill up the buffer (the other packets are dropped). In case of no continuing packet arriving, OPW selects the head of buffer packet to be transmitted in the next time slot. If the buffer is empty, OPW transmits the newly generated packet (if there is one) in the next time slot. In case of contention, new packets are discarded.

4. Shortest Hops Win 2 (SHW2): Among the continuing packets, the head of the buffer packet, and the new packet (if there is one) at a node, SHW2 chooses the one with shortest hop distance to its destination to be transmitted in the next time slot. Packets that did not win the contention are stored in the buffer if there is space in the buffer.

5. Shortest Hops Win 3 (SHW3): The packet at the head of the buffer (if there is one) is always transmitted at the beginning of next time slot. The continuing packets and the new packet (if there is one) are stored in the buffer if there is enough space available. In case of not enough buffer space, SHW3 drops the packets with greatest hop distance to their destination.

We will give a detailed analysis on the throughput of Shortest Hop Win scheme and Oldest Packet Win scheme in the subsequent sections. The analysis of Random Packet Win scheme is similar to the Shortest Hop Win scheme and, therefore, omitted for brevity.

We also introduce the following notation which will be useful in the later sections. For an arbitrary packet $\mathcal{P}$, let $s_{\mathcal{P}}$ denote its source node; $t_{\mathcal{P}}$ denote its destination node; and $d_H(s_{\mathcal{P}}, t_{\mathcal{P}})$ the shortest hop distance between $s_{\mathcal{P}}$ and $t_{\mathcal{P}}$.

3.3.1 Throughput analysis for Shortest Hop Win (SHW) scheme with buffer

The arrivals of packets on different links to a particular node may not be independent. However, under our uniform traffic and random destination assumption, they should behave in an almost independent way. Hence, we make two approximating assumptions here.

1. Packet arrivals on each of the different incoming links to a particular node are independent during a time slot.

2. The arrivals of packets to a node in one slot is independent of the arrivals to the node during previous slot.

At the beginning of a time slot, the transmitter at an arbitrary node, say node a, sends a packet $\mathcal{P}$ to one of its neighbor, say node k. Before the start of the transmission at node a , if the packet $\mathcal{P}$ is i hops away from its destination node, we say the packet is of type i; more precisely, $d_H(a, t_{\mathcal{P}}) = i$. When the packet $\mathcal{P}$ arrives at node k, it competes with other arriving packets for transmission during the next time slot. A packet is said to be the winning packet if it will be transmitted in the next time slot. The SHW scheme selects amongst continuing packets at node k the one with the shortest hop distance to its destination to be the winning packet. If there are $j, j > 1$, packets have the same shortest hop distance to their respective destination nodes, SHW randomly selects one packet to be the winning packet among these j packets. If $\mathcal{P}$ has the shortest hop distance to its destination node among the continuing packets at node k, it is said to be a winning packet of type $(i-1)$ at node k. If no continuing packet arrives at node k during a time slot, the winning packet is the head of buffer packet if the buffer is nonempty. Similarly, the newly generated packet is the winning packet if there are no continuing packets and buffered packets

at node k.

In the steady state, due to the same externally arrival rate p_0 and uniform destination for each newly generated packet, by symmetry each node has the same statistics (i.e., the probability that a winning packet is of type i, $0 \leq i \leq d$, is the same for all nodes) without the approximation assumption. However, to get the exact value of these statistics, we have to utilize the two approximations made above. Specifically, by considering only one node k in the network, let A_i, $0 \leq i \leq d$, denote the event that node k has a winning packet of type i. Similarly, let E denote the event that node k is empty. We can then write the probabilities $P(A_i)$'s and $P(E)$ recursively, in terms of the same probabilities at neighboring nodes, by using the property that each node has the same statistics and by considering the interactions between node k and its neighboring nodes. Throughout this section, we focus on finding $P(A_i)$'s. The throughput is thus obtained as $P(A_0)$ in the Shortest Hop Win scheme.

Again, considering an arbitrary node k, we define

- B_i, $0 \leq i \leq 4$, to be the event that node k received packets from i out of the four neighboring nodes.
- H_i, $0 \leq i \leq d$, to be the event that the head of the buffer packet is of type i.
- U_i, $1 \leq i \leq d$, to be the event that a new packet that is i hops away from the destination node is generated at node k.
- BE to be the event that the buffer at node k is empty.
- BE^c to be the event that the buffer at node k is nonempty.

With the relevant events defined, we now write the equations for solving $P(A_i)$ in terms of these events. For $1 \leq i \leq d-1$, we have

$$\begin{aligned}\gamma(i) = P(A_i) = {} & P(A_i|B_1)P(B_1) + P(A_i|B_2)P(B_2) + P(A_i|B_3)P(B_3) \\ & + P(A_i|B_4)P(B_4) + P(H_i)P(BE^c)P(B_0) + P(U_i)P(BE)P(B_0).\end{aligned} \tag{3.2}$$

Similarly, for $i = 0$

$$\begin{aligned}\gamma(0) = P(A_0) = P(A_0|B_1)P(B_1) + P(A_0|B_2)P(B_2) + P(A_0|B_3)P(B_3) \\ + P(A_0|B_4)P(B_4) + P(H_0)P(BE^c)P(B_0)\end{aligned} \tag{3.3}$$

and for $i = d$

$$\gamma(d) = P(U_d)P(BE)P(B_0).$$

To derive the above equations, consider the events that take place at node k. Since we give the first priority to the continuing packets, next priority to the buffered packet, and the lowest priority to the new packet, event A_i occurs if and only if one of the following events occur:

- A continuing packet of type i arrives at node k and wins the contention.
- The head of buffer packet is of type i, and no continuing packet arrives.
- A new packet of type i is generated at node k; no continuing packet arrives; and the buffer is empty.

Eq.[3.2] enumerates all of the above events. Now, we write the individual terms out. The probability that a new packet with i hops to its destination is generated is the following:

$$P(U_i) = \frac{n_x(i)}{N^2 - 1} \cdot p_0$$

where $n_x(i)$ denotes the number of nodes that are i hops away (see Lemma 1) and $N^2 - 1$ is the total number of possible destination node. We also get for $0 \leq n \leq 4$

$$\beta_n = P(B_n) = \binom{4}{n} \left[\frac{1}{4}\sum_{j=1}^{d} \gamma(j)\right]^n \left[1 - \frac{1}{4}\sum_{j=1}^{d} \gamma(j)\right]^{4-n} \tag{3.4}$$

The term $[\frac{1}{4}\sum_{j=1}^{d} \gamma(j)]$ denotes the probability that a neighboring node of k sends a packet to node k. Similarly, $\frac{\gamma(i+1)}{\sum_{j=1}^{d} \gamma(j)}$ is the probability that a node is sending

a packet of type $i+1$ given that node is sending a packet; and $\frac{\sum_{j=i+2}^{d}\gamma(j)}{\sum_{j=1}^{d}\gamma(j)}$ is the probability that a node is sending a packet of type m, where $i+2 \leq m \leq d$, given that node is sending a packet.

Then, letting $a_i = \frac{\gamma(i+1)}{\sum_{j=1}^{d}\gamma(j)}$ and $c_i = \frac{\sum_{j=i+2}^{d}\gamma(j)}{\sum_{j=1}^{d}\gamma(j)}$, we have for $0 \leq i \leq d-1$

$$
\begin{aligned}
P(A_i|B_1) &= a_i && (3.5)\\
P(A_i|B_2) &= a_i^2 + \binom{2}{1} a_i c_i && (3.6)\\
P(A_i|B_3) &= a_i^3 + \binom{3}{2} a_i^2 c_i + \binom{3}{1} a_i c_i^2 && (3.7)\\
P(A_i|B_4) &= a_i^4 + \binom{4}{3} a_i^3 c_i + \binom{4}{2} a_i^2 c_i^2 + \binom{4}{1} a_i c_i^3 && (3.8)
\end{aligned}
$$

To interpret the above equation, consider $P(A_i|B_2)$. Recall that packet with shorter distance to its destination has priority. Given that exactly two packets arrived from two of the four neighboring nodes of node k, the event that the winning packet is i hops away from its destination, or type i packet, is the union of the following two disjoint events:

- at node k, both of these two arriving packets are type i packets (The first term in Eq.[3.6], a_i^2, for example).

- at node k, one of these two packets is a type i packet and the other one is of type j, where $i+1 < j \leq d$ (The second term in Eq.[3.6], $\binom{2}{1} a_i c_i$, for example).

Next, we will investigate the probability that a head of buffer packet is of type i. Let G_i denote the event that an arbitrary packet, say $\mathcal{P}$, that is $i+1$ hops away from its destination node before the start of its transmission to node k, subsequently loses the contention with other packets at node k. Assuming node a is a neighbor of node k, we then have

$$
\begin{aligned}
P(H_i) &= P(\text{a packet in the buffer is of type } i) && (3.9)\\
&= \frac{P(\text{type } i \text{ packet gets sent to the buffer})}{P(\text{packet gets sent to buffer})} && (3.10)
\end{aligned}
$$

$$= \frac{\gamma(i+1)P(G_{i+1})}{\sum_{j=1}^{d}\gamma(j)P(G_j)} \tag{3.11}$$

Notice also that a packet of type d (newly generated packet with d hops to its destination node) will never be stored in the buffer by the priority rule. Packet $\mathcal{P}$, which just became a type $i-1$ packet after reaching node k, may lose the contention if one of the following events occur:

- Event $E_0(i)$: Out of the three remaining neighboring nodes of node k, there is at least one of them which is sending a packet of type j, where $j < i$, to node k.

- Event $E_1(i)$: Out of the three remaining neighboring nodes of node k, there is exactly one of them which is also sending a packet of type i to node k, while the others are either not sending a packet to node k or sending packets of type j $(j > i)$ to node k.

- Event $E_2(i)$: Out of the three remaining neighbor of node k, there are exactly two of them which are also sending packets of type i to node k, while the other neighboring node is either not sending a packet to node k or sending packet of type j, $j > i$, to node k.

- Event $E_3(i)$: Out of the three remaining neighbor of node k, each one of them is sending a packet of type i to node k.

From the above description, we get for $2 \leq i \leq d$

$$P(E_0(i)) = 1 - \left[1 - \frac{1}{4}\sum_{j=1}^{i-1}\gamma(j)\right]^3, \ \forall\ i > 1, \text{ and } P(E_0(1) = 0) \tag{3.12}$$

$$P(E_1(i)) = \binom{3}{1}\left[\frac{\gamma(i)}{4}\right]\left[1 - \frac{1}{4}\sum_{j=1}^{i}\gamma(j)\right]^2 \tag{3.13}$$

$$P(E_2(i)) = \binom{3}{2}\left[\frac{\gamma(i)}{4}\right]^2\left[1 - \frac{1}{4}\sum_{j=1}^{i}\gamma(j)\right] \tag{3.14}$$

$$P(E_3(i)) = \left[\frac{\gamma(i)}{4}\right]^3 \tag{3.15}$$

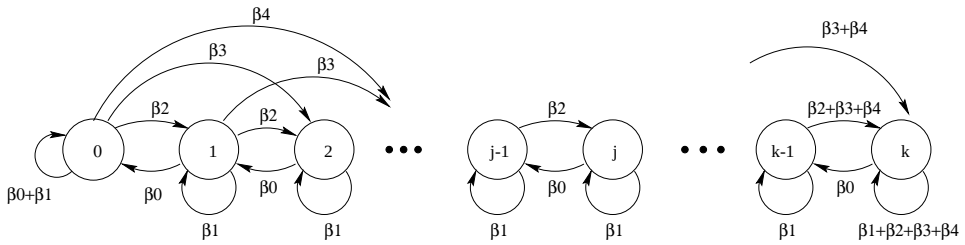


Figure 3-2: Markov chain of the number of packets in a buffer of size k.

When event $E_0(i)$ occurs, packet $\mathcal{P}$ will be *sent* to the buffer with probability one (although it may be dropped due to buffer overflow). Likewise, when event $E_1(i)$, or $E_2(i)$, or $E_3(i)$ occurs, packet $\mathcal{P}$ will be sent to the buffer with probability $\frac{1}{2}$, $\frac{2}{3}$, $\frac{3}{4}$ respectively. Now, $P(G_i)$ can be obtained as:

$$P(G_i) = P(E_0(i)) + \frac{1}{2}P(E_1(i)) + \frac{2}{3}P(E_2(i)) + \frac{3}{4}P(E_3(i))$$

To get $P(BE)$, we denote by b_i, $i = 0, 1, \cdots, m$, the probability that there are i packets at a node's buffer at the beginning of slot. Since there are four receivers at a node, at most three continuing packets may arrive at the buffer during a time slot. Fig. 3-2 is a finite state markov chain which describes the evolution of the number of packets in a buffer of size k. The state represents the number of packets in the buffer. Thus, we have

$$\begin{aligned}
P(BE) = b_0 &= b_0\beta_1 + (b_0 + b_1)\beta_0 && (3.16)\\
b_1 &= b_0\beta_2 + b_1\beta_1 + b_2\beta_0 && (3.17)\\
b_2 &= b_0\beta_3 + b_1\beta_2 + b_2\beta_1 + b_3\beta_0 && (3.18)\\
b_j &= b_{j-3}\beta_4 + b_{j-2}\beta_3 + b_{j-1}\beta_2 + b_j\beta_1 + b_{j+1}\beta_0 && (3.19)\\
b_k &= b_k(\beta_1 + \beta_2 + \beta_3 + \beta_4) + b_{k-1}(\beta_2 + \beta_3 + \beta_4)\\
&\quad + b_{k-2}(\beta_3 + \beta_4) + b_{k-3}\beta_4 && (3.20)
\end{aligned}$$

p_0	Theoretical Throughput	Simulation Throughput
0.1	0.0645	0.0649
0.2	0.0952	0.0960
0.3	0.1128	0.1153
0.4	0.1241	0.1228
0.5	0.1318	0.1302
0.6	0.1373	0.1378
0.7	0.1414	0.1392
0.8	0.1446	0.1446
0.9	0.1472	0.1461
0.95	0.1483	0.1477
0.99	0.1491	0.1488

Table 3.1: A comparison of simulation result and theoretical result for 2-dimensional 11-mesh using Shortest Hop Win scheme 1 with buffer.

With the above equations, we can solve for $\gamma(i)$ numerically. For our simulation, a 2-dimensional 11-mesh with a buffer size of four at each node is used. As Table 3.1 shows, our numerical results is very accurate compare with the simulation result.

3.3.2 Throughput analysis for Shortest Hop Win scheme without buffer

In this section, we consider the throughput of the Shortest Hop Win scheme without buffer at each node. The analysis is similar to the one in the previous section. The notation, if not specified, will be the same as the one defined previously. Again, we give priority to the continuing packet over the newly generated packet (i.e. the new packet can be transmitted only if there is no continuing packet arrives at that node). Thus, we have for $1 \leq i \leq d-1$

$$\begin{aligned}\gamma(i) = P(A_i) = P(A_i|B_1)P(B_1) + P(A_i|B_2)P(B_2) + P(A_i|B_3)P(B_3) \\ + P(A_i|B_4)P(B_4) + P(U_i)P(B_0)\end{aligned} \tag{3.21}$$

p_0	Theoretical Throughput	Simulation Throughput
0.1	0.0450	0.0449
0.2	0.0635	0.0631
0.3	0.0756	0.0780
0.4	0.0845	0.0841
0.5	0.0917	0.0914
0.6	0.0976	0.0984
0.7	0.1027	0.1028
0.8	0.1071	0.1061
0.9	0.1110	0.1102
0.95	0.1129	0.1140
0.99	0.1142	0.1144

Table 3.2: A comparison of simulation result and theoretical result for 2-dimensional 11-mesh using Shortest Hop Win scheme without buffer.

Similarly, we have for $i = 0$,

$$\begin{aligned}\gamma(0) = P(A_0) = P(A_0|B_1)P(B_1) + P(A_0|B_2)P(B_2) + P(A_0|B_3)P(B_3) \\ + P(A_0|B_4)P(B_4)\end{aligned} \tag{3.22}$$

and for $i = d$,

$$\gamma(d) = P(U_d)P(B_0) \tag{3.23}$$

$P(A_i|B_1), \cdots, P(A_i|B_3)$ and $P(U_i)$ can be calculated by using the exact same formulas given in the previous section.

Again, we calculate the theoretical throughput for a 2-dimensional 11-mesh and compare with simulation results. Fig. 3-3 is a plot of the throughput of a system with buffer and a system without buffer under SHW. The throughput increases significantly when every node has a buffer. This can be explained by noting that packets can be put in the buffer temporarily if it lost the competition instead of just dropping them in the case of no buffer. Dropping a packet which has already travelled a certain number of hops waste the previous transmissions of that packet become wasted, thus decreases the throughput of system. Intuitively, we would like to minimize the wasted work and

hope that every transmission is going to contribute to the increase of throughput.

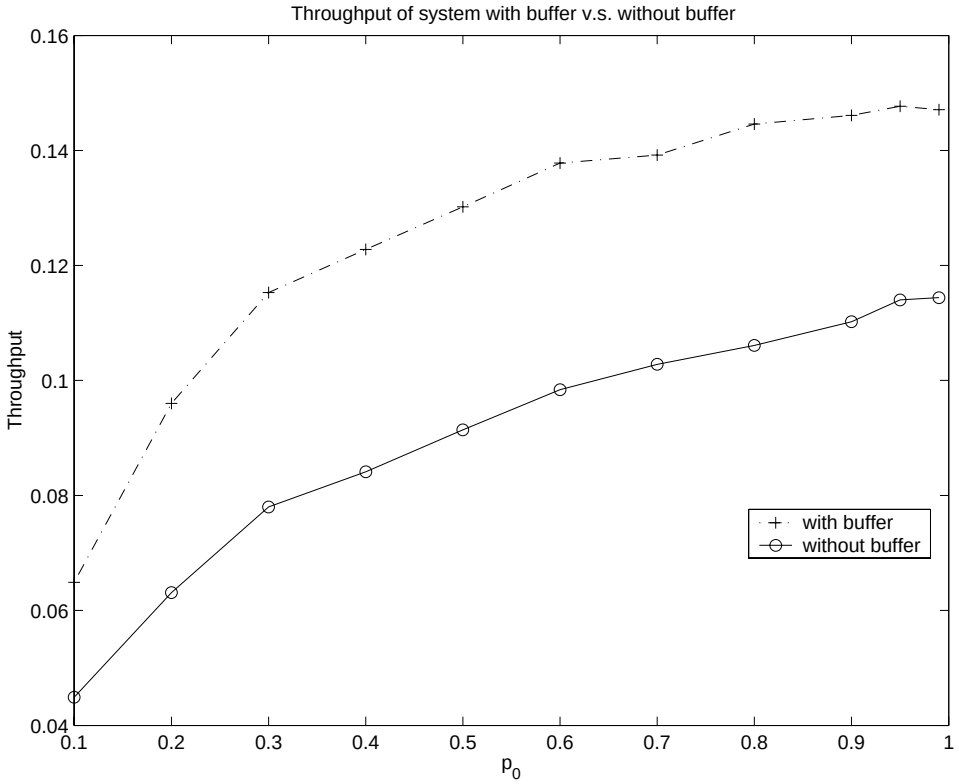


Figure 3-3: A comparison of throughput of system with or without buffer using SHW.

3.3.3 Throughput analysis for Oldest Packet Win scheme with buffer

We continue to use the approximations made in the analysis of the throughput of Shortest Hop Win scheme. At the beginning of a time slot, the transmitter at an arbitrary node, say node a, sends a packet $\mathcal{P}$ to one of its neighbor, say node k. Before the start of the transmission at node a , if the packet $\mathcal{P}$ has already travelled i hops from its starting node ($d_H(a, s_{\mathcal{P}}) = i$), we say the packet is of *type* i. Notice that the definition of the type of a packet here is different from that in the previous section. In the analysis of the SHW, a packet of type i implies that it is i hops *away* from its destination node. During the transmission, we say that the packet is travelling on its $(i+1)$th hop from its starting node. When $\mathcal{P}$ arrives at node k, it becomes a type

$(i+1)$ packet and has to compete with other arriving packets for the transmission right of the next time slot. Among all of the continuing packets at node k, a packet is said to be the winning packet if it travelled the longest hop distance from its origin node. In case of a tie, the winning packet is selected at random from the packets that have travelled the longest distance. When no continuing packets arrive at node k during a time slot, the winning packet is the head of buffer packet if the buffer is nonempty. Similarly, the newly generated packet is the winning packet if there are no continuing packets and no buffered packets at node k.

In the steady state, similar to the analysis in the previous section, each node still has the same statistics without the approximation assumption. Let A_i denote the event that an arbitrary node has a winning packet of type i, and E denote the event that node k is empty. Again, we use the two approximations made previously to get a set of $P(A_i)$'s and $P(E)$, which solve the equations below and sum to one. The throughput can thus be obtained from $P(A_i), 0 \leq j \leq d$.

Let $\alpha(i) = P(A_i)$ and $C_{pass}(i) = \Pr(\text{a packet must travel at least one additional hop on its way to the destination node} \mid \text{it has already travelled } i \text{ hops})$.

$$C_{pass}(i) = \frac{\sum_{j=i+1}^{d} n_x(j)}{\sum_{j=i}^{d} n_x(j)}$$

To get $\alpha(i)$, notice that a node has a winning packet of type i if and only if one of the following events occur during a time slot:

- Event O_1: No continuing packet is transmitted to node k, and the head of buffer packet at node k is of type i.

- Event O_2: Of the four receivers at node k, there are at least one of them received a packet of type i, while the others either did not receive any packet or received packet of type j $(j < i)$.

We then have for $1 \leq i \leq d$

$$\begin{aligned}
\alpha(i) = & \left[1 - \frac{1}{4}\sum_{j=0}^{d-1} \alpha(j) C_{pass}(j)\right]^4 (1-b_0) Pr(H_i|\text{buffer nonempty}) \\
& + \binom{4}{1} \left[1 - \frac{1}{4}\sum_{j=i-1}^{d-1} \alpha(j) C_{pass}(j)\right]^3 \left[\frac{1}{4}\alpha(i-1) C_{pass}(i-1)\right] \\
& + \binom{4}{2} \left[1 - \frac{1}{4}\sum_{j=i-1}^{d-1} \alpha(j) C_{pass}(j)\right]^2 \left[\frac{1}{4}\alpha(i-1) C_{pass}(i-1)\right]^2 \\
& + \binom{4}{3} \left[1 - \frac{1}{4}\sum_{j=i-1}^{d-1} \alpha(j) C_{pass}(j)\right] \left[\frac{1}{4}\alpha(i-1) C_{pass}(i-1)\right]^3 \\
& + \left[\frac{1}{4}\alpha(i-1) C_{pass}(i-1)\right]^4
\end{aligned} \tag{3.24}$$

and

$$\alpha(0) = p_0 b_0 \left[1 - \frac{1}{4}\sum_{j=0}^{d-1} \alpha(j) C_{pass}(j)\right]^4 \tag{3.25}$$

The first term in Eq.[3.24] represents the probability of event O_1, and the rest terms denotes the probability of event O_2. The term $\left[1 - \frac{1}{4}\sum_{j=i-1}^{d-1} \alpha(j) C_{pass}(j)\right]$ represents the probability that there is no type j ($j \geq i-1$) packet travelling on a particular link.

Let the events B_i and H_i be similarly defined as in the previous section. Then, we have for $0 \leq n \leq 4$

$$P(B_n) = \binom{4}{n} \left[\frac{1}{4}\sum_{j=0}^{d} \alpha(j) C_{pass}(j)\right]^n \left[1 - \frac{1}{4}\sum_{j=0}^{d} \alpha(j) C_{pass}(j)\right]^{4-n}$$

Similar to the previous analysis on the throughput of Shortest Hop Win scheme, the probability that a buffered packet is of type i, $1 \leq i \leq d$, is

$$P(H_i) = \frac{\alpha(i-1) C_{pass}(i-1) P(G_i)}{\sum_{j=0}^{d-1} \alpha(j) C_{pass}(j) P(G_{j+1})} \tag{3.26}$$

Again, G_i is the event that an arbitrary packet has already travelled i hops from its starting node after reaching node k, it subsequently lost the contention with other packets at node k.

A packet, transmitted from one of node k's neighbors (say node a), just finished travelling its ith hop may lose the contention at node k if one of the following events occur:

- Event $E_0(i)$: Out of the three remaining neighboring nodes of node k, there is at least one of them which is sending a packet of type j, where $j \geq i$, to node k.

  - Comment: Since $j \geq i$, after reaching node k, that packet will be type $j+1$. The packet from node a will definitely lose in the competition since its hop distance to its source node, i, is strictly shorter.

- Event $E_1(i)$: Out of the three remaining neighboring node of node k, there is exactly one neighboring node, say b, which is also sending a packet of type $i-1$ to node k, while the other neighboring nodes are either not sending a packet to node k or sending packets of type j $(j < i-1)$ to node k.

  - Comment: Packet from node a will compete with packet from node b for the transmission right of next slot. Since both of packets are of the same type, each one wins the competition with probability one half.

- Event $E_2(i)$: Out of the three remaining neighbor of node k, there are exactly two of them which are also sending a packet of type $i-1$ to node k, while the other neighboring node is either not sending a packet to node k or sending packet of type j $(j < i-1)$ to node k.

- Event $E_3(i)$: For the three remaining neighbor of node k, each of them is sending a packet of type $i-1$ to node k.

We have for $1 \leq i \leq d$

$$P(E_0(i)) = 1 - \left[1 - \frac{1}{4}\sum_{j=i}^{d} \alpha(j) C_{pass}(j)\right]^3 \quad (3.27)$$

$$P(E_1(i)) = \binom{3}{1}\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]\left[1 - \frac{1}{4}\sum_{j=i-1}^{d} \alpha(j) C_{pass}(j)\right]^2 \quad (3.28)$$

$$P(E_2(i)) = \binom{3}{2}\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]^2\left[1 - \frac{1}{4}\sum_{j=i-1}^{d} \alpha(j) C_{pass}(j)\right] \quad (3.29)$$

$$P(E_3(i)) = \left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]^3 \quad (3.30)$$

Now, $P(G_i)$ is obtained from the following equation:

$$P(G_i) = P(E_0(i)) + \frac{1}{2}P(E_1(i)) + \frac{2}{3}P(E_2(i)) + \frac{3}{4}P(E_3(i))$$

The probability that there are i packets at a node's buffer, $b_i(i = 0, 1, \cdots, m)$, has exactly the same form as in the previous section (see equations [3.16]-[3.20]). Lastly, we introduce the event that a packet reached its destination node given it has already travelled i hops. More precisely,

$$\begin{aligned} C_{end}(i) &= Pr(\text{packet } \mathcal{P} \text{ reached its destination node } \mid \text{ it has already travelled } i \text{ hops}) \\ &= \frac{Pr(d_H(s_{\mathcal{P}}, t_{\mathcal{P}}) = i)}{Pr(d_H(s_{\mathcal{P}}, t_{\mathcal{P}}) \geq i)} = \frac{n_x(i)}{\sum_{j=i}^{d} n_x(j)} \end{aligned}$$

With all equations available, the throughput of Oldest Packet Win scheme can be computed as follows

$$Throughput = \sum_{j=1}^{d} \alpha(j) C_{end}(j)$$

Solving for the values of $\alpha(i)$ and subsequently the throughput numerically for a 2-dimensional 11-mesh with a buffer size of four at each node, we again obtain accurate results as compared with simulations, as shown in Table 3.3.

p_0	Theoretical Throughput	Simulation Throughput
0.1	0.0645	0.0645
0.2	0.0951	0.0951
0.3	0.1126	0.1102
0.4	0.1237	0.1235
0.5	0.1311	0.1310
0.6	0.1364	0.1342
0.7	0.1402	0.1398
0.8	0.1432	0.1427
0.9	0.1454	0.1448
0.95	0.1463	0.1456
0.99	0.1470	0.1462

Table 3.3: A comparison of simulation result and theoretical result for 2-dimensional 11-mesh using Oldest Packet Win scheme.

3.3.4 Throughput analysis for Oldest Packet Win scheme without buffer

The case of no buffer at each node is very similar to the case with buffer in terms of throughput analysis. With a few minor modifications on Eq.[3.24], we get the following equation ($1 \leq i \leq d$):

$$\begin{aligned}\alpha(i) =& \binom{4}{1}\left[1-\frac{1}{4}\sum_{j=i-1}^{d-1}\alpha(j)C_{pass}(j)\right]^3\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right] \\ &+\binom{4}{2}\left[1-\frac{1}{4}\sum_{j=i-1}^{d-1}\alpha(j)C_{pass}(j)\right]^2\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]^2 \\ &+\binom{4}{3}\left[1-\frac{1}{4}\sum_{j=i-1}^{d-1}\alpha(j)C_{pass}(j)\right]\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]^3 \\ &+\left[\frac{1}{4}\alpha(i-1)C_{pass}(i-1)\right]^4\end{aligned} \tag{3.31}$$

and

$$\alpha(0) = p_0\left[1-\frac{1}{4}\sum_{j=0}^{d-1}\alpha(j)C_{pass}(j)\right]^4 \tag{3.32}$$

For a 2-dimensional 11-mesh, we again calculate the theoretical throughput of the system without buffer and compare it with the simulation results. We also see that the throughput for a system with buffer is significantly greater than the throughput of a system without buffer, shown in Fig. 3-4.

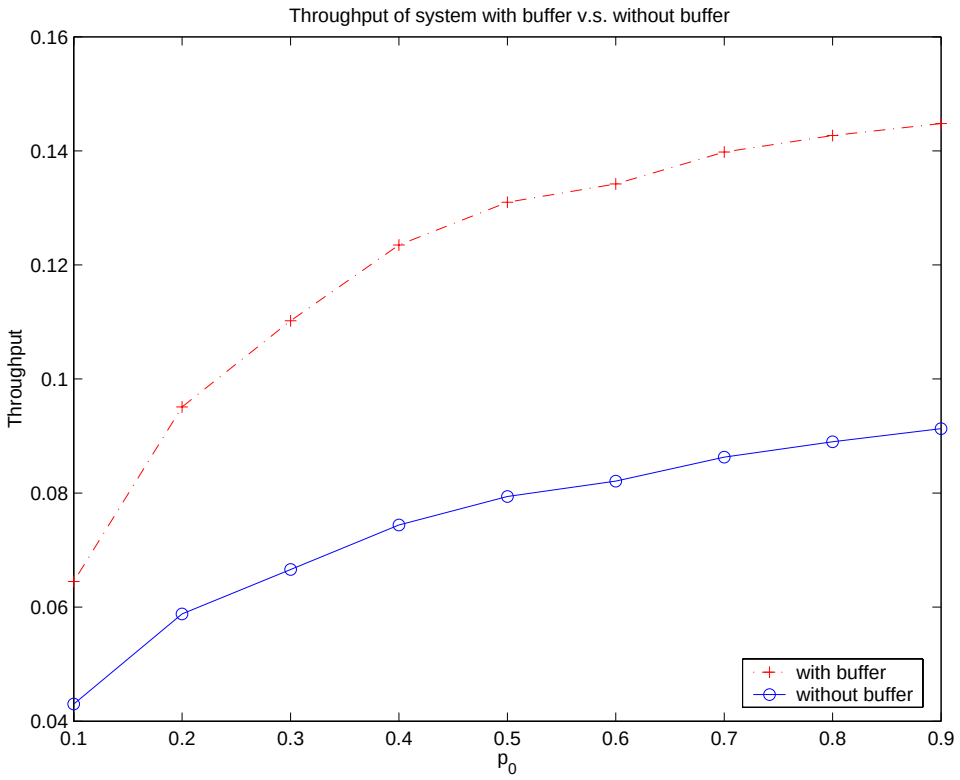


Figure 3-4: A comparison of throughput of system with or without buffer using OPW.

3.3.5 A comparison of different schemes in the no buffer case

Following the analysis of previous two sections, we can also get the throughput for the Random Packet Win scheme (with buffer or without buffer). For system without buffer, of the three schemes dicussed so far (SHW, OPW, RPW), we expect that SHW to perform better than the other two schemes in terms of throughput since the continuing packets in the system are likely to have a shorter distance to the destination node. Also, the Oldest Packet Win scheme should perform better than the Random Packet Win scheme since it tries to minimize the amount of wasted work

done for a continuing packet. Fig. 3-5 below substantiates the above statements.

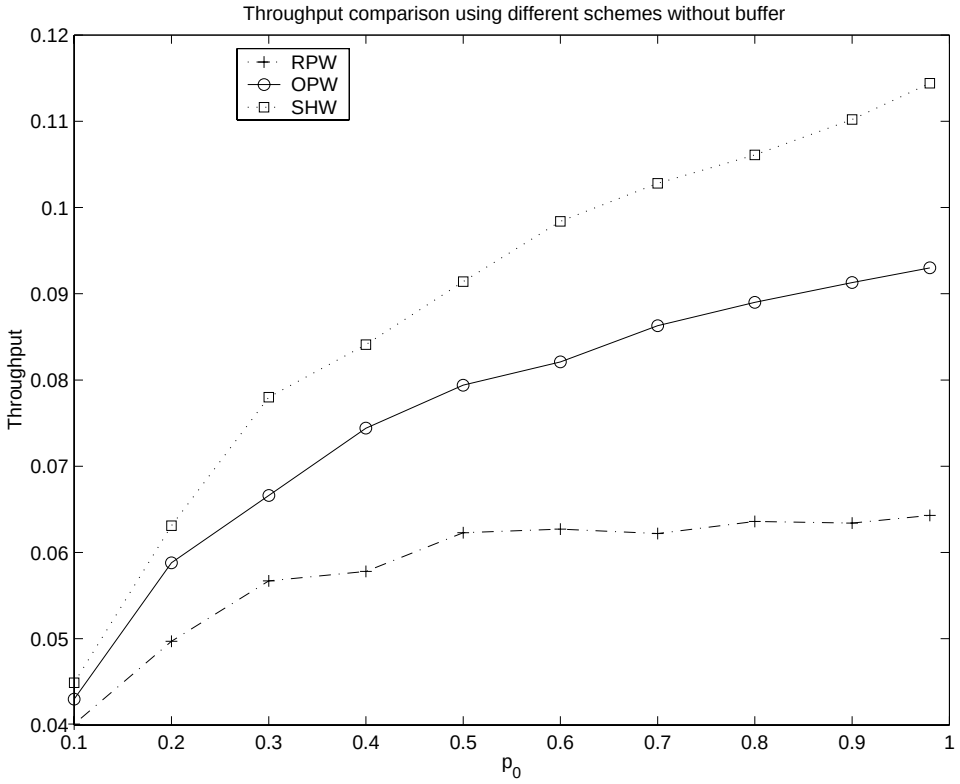


Figure 3-5: A comparison of throughput of system without buffer using different schemes.

3.3.6 Simulation of other schemes

The theoretical analysis of the Random Packet Win scheme, Shortest Hops Win scheme 2 and Shortest Hops Win scheme 3 can be carried out by following the analysis in the previous two sections. Here we provide simulation results of the aforementioned schemes and compare their performance. First, we want to compare the throughput of a system with buffer under the Random Packet Win scheme, Oldest Packet Win scheme, and Shortest Packet Win scheme. For all of the three schemes, the continuing packet is given the hightest priority. Buffered packet will be transmitted only if there are no continuing packets. Likewise, the newly generated packet will be transmitted only if there are no continuing packets and no buffered packets. Fig. 3-6 plots the throughput for the three scheme. We expect that SHW scheme would perform

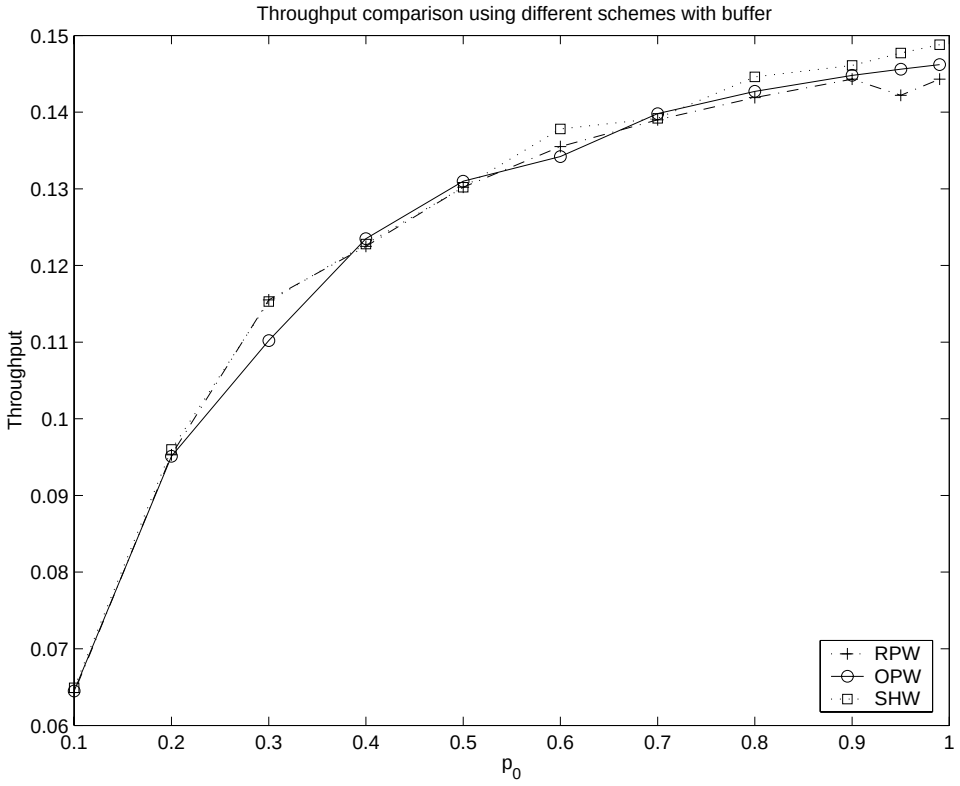


Figure 3-6: A comparison of throughput of system with buffer using different schemes.

significantly better than the other two schemes just as it did in the no buffer case. Surprisely, however, we see that the throughput for three schemes is about the same, although SHW performs slightly better than the other two schemes in the high p_0 region. It seems that the buffer has a neutralizing effects on the system's throughput (i.e., the choice of which scheme to use becomes less important). An explanation to the rather counterintuitive result is the following. After a packet arrived at the receiving node, the packet which lost the contention is stored in the buffer if there is any space available. Notice that we do not decide which packet to put in the buffer. If there is enough space for all of the packets which did not win the contention, all of them will be stored in the buffer. In the event that there is not enough buffer space for all losing packets, we randomly pick amongst them to be placed in the remaining spots of the buffer. The packets in the buffer will eventually be transmitted. Unlike the system without buffer, these packets are not dropped immediately, although they did not win the contention. It is in this sense that the contention is not a strict com-

petition (since they are still in the system). Therefore, the difference in throughput using different schemes is not very significant. To increase the throughput, one may want to develop an additional scheme in choosing which packet to be sent to the buffer instead of choosing it randomly.

To verify the above explanation, we also investigate the throughput of a rather "bad" scheme called Furthest Hops Win scheme. This scheme is identical to the SHW scheme except that during a contention the latter scheme chooses the packet with shortest hop distance to the destination to win while the former scheme chooses the packet with longest hop distance to the destination to win. We expect that the Furthest win scheme would perform much worse than all of the schemes mentioned so far. However, as Fig.3-7 shows, the throughput of Furthest Hops Win scheme performs only slightly worse than other schemes.

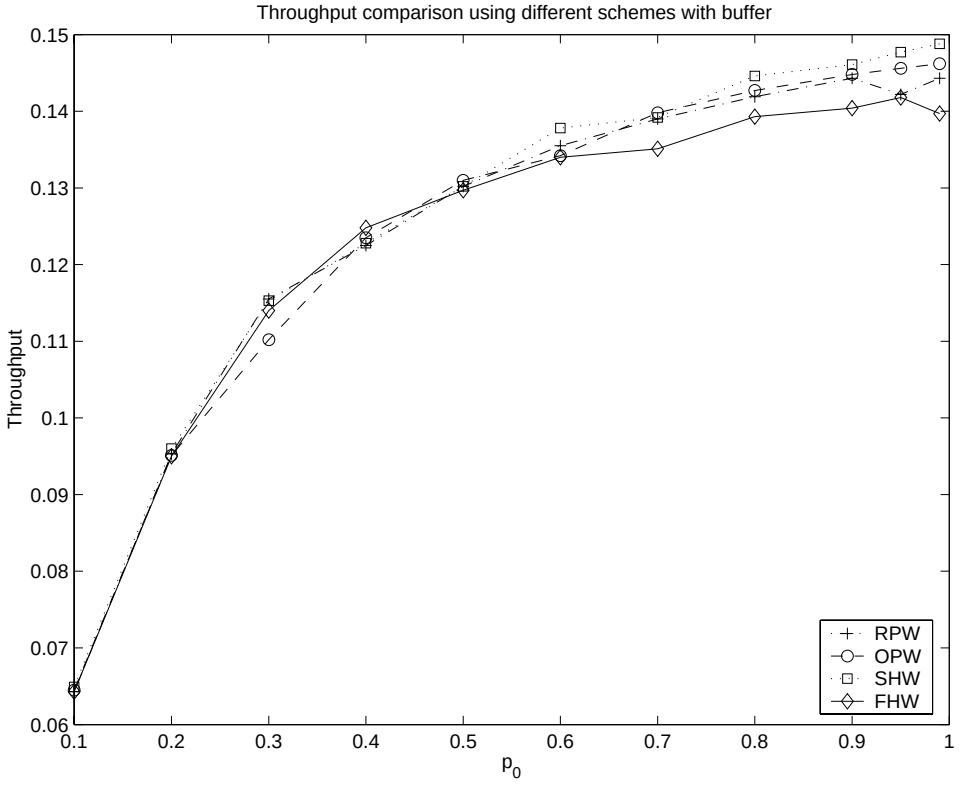


Figure 3-7: A comparison of throughput of system with buffer using FHS.

For all of the schemes mentioned so far, the highest priority is assigned to the continuing packets, and the new packets can only enter the system if there are no

continuing packets or buffered packets. This prompts us to think that the throughput can be improved if we allow all of the continuing packets, the head of buffer packet and the newly generated packet compete for the transmission right in the next slot instead of just letting the continuing packets to compete. A modified version of SHW scheme choose the packet with shortest hop distance to its destination node to win. Packets which loss the contention will be stored into the buffer if there is space available. We call this scheme Shortest Hops Win scheme 2. Again, an unexpected simulation result (see Fig. 3-8 below) shows that throughput is lower than the four schemes discussed so far. A closer look at the distribution of number of hops to the destination node for the head of buffer packet reveals that, with high probability, the head of buffer packet has a long hop distance to its destination node. This can be explained by noting that the head of buffer packet is transmitted only if it has the shortest hop distance to its destination. Consequently, packets with longer hop distance to their destination than the head of buffer packet will be placed in the buffer. Eventually, the buffer will be filled with packets with d hops (the maximum hop distance between a source and destination pair) to their destination nodes. As a result, we have a system with effectively no buffer, thus the througput is lower.

Examing the throughput results of OPW, RPW, and SHW schemes closely, we find that these schemes do not achieve high throughput when p_0 is inside the stability region. Ideally, we should be able to attain a throughput level which is the same as the arrival rate p_0 when p_0 is within the stability region. However, for all three schemes, the throughput is only about 0.065 when p_0 is 0.1. We develop next a scheme, called Shortest Hops Win scheme 3, that achieves high throughput when p_0 is relatively small. SHW3 works as follows: The new packet and the arriving packets are sent to the buffer if there is space available. If there is not enough space available for all the incoming packets (including the arriving packets and the new packet), we put packets with shorter hop distance to the destination node in the buffer first. At the beginning of a time slot, the packet at the head of buffer is going to be transmitted. Simulation shows that SHW3 achieve a throughput level close to the arrival rate when p_0 is within the stability region. However, as p_0 increases, the throughput of SHW3

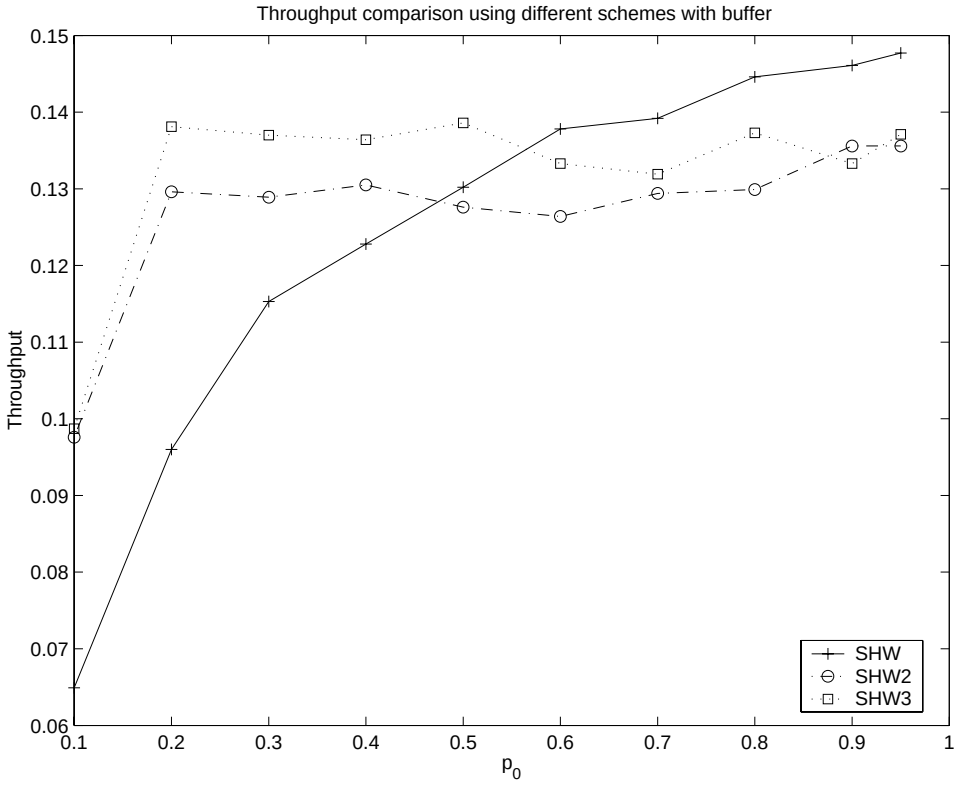


Figure 3-8: A comparison of throughput of system with buffer using SHW2 and SHW3.

is lower than that of the RPW, OPW, and SHW schemes.

3.3.7 Throughput and buffersize

To investigate the relationship between the buffer size and the throughput, we evaluate the througput for SHW and SHW3 at $p_0 = 0.1$, $p_0 = 0.5$, and $p_0 = 0.9$ by using vairous buffer size. Fig. 3-9 illustrates that a network with moderate buffer size such as four or eight can achieve the same level of throughput as a network with significantly larger buffer size. In other words, the throughput of system does not increase with the increase of buffer size.

3.4 Summary

In this chapter, we analyzed a problem where nodes of the 2-dimensional N-mesh generate packets independently at the beginning of each time slot with probability p_0. Each packet has unit transmission time and is destined for a uniformly selected node. We showed the stability region of such network. Then, we consider three routing schemes (Shortest Hop Win, Random Packet Win, and Oldest Packet Win) and compare their throughput performance. As multiple packets arrive at a particular node in a time slot, SHW chooses the one with shortest hop distance to its destination to be transmitted in the next slot; RPW randomly picks one to be transmitted; and OPW selects the one with longest hop distance to the destination. In all three schemes, continuing packets have priority over the buffered packets, and the buffered packets have priority over the new packets. Both the analytic and simulated results show that SHW scheme attains the best throughput performance, OPW scheme the second, and RPW the worst in the case of no buffer at each node. In the buffered case, the three schemes have similar performance. Also, a small buffer size can achieve throughput close to that of the infinite buffer size. Most of the packet drops occur because the new packet cannot enter the system.

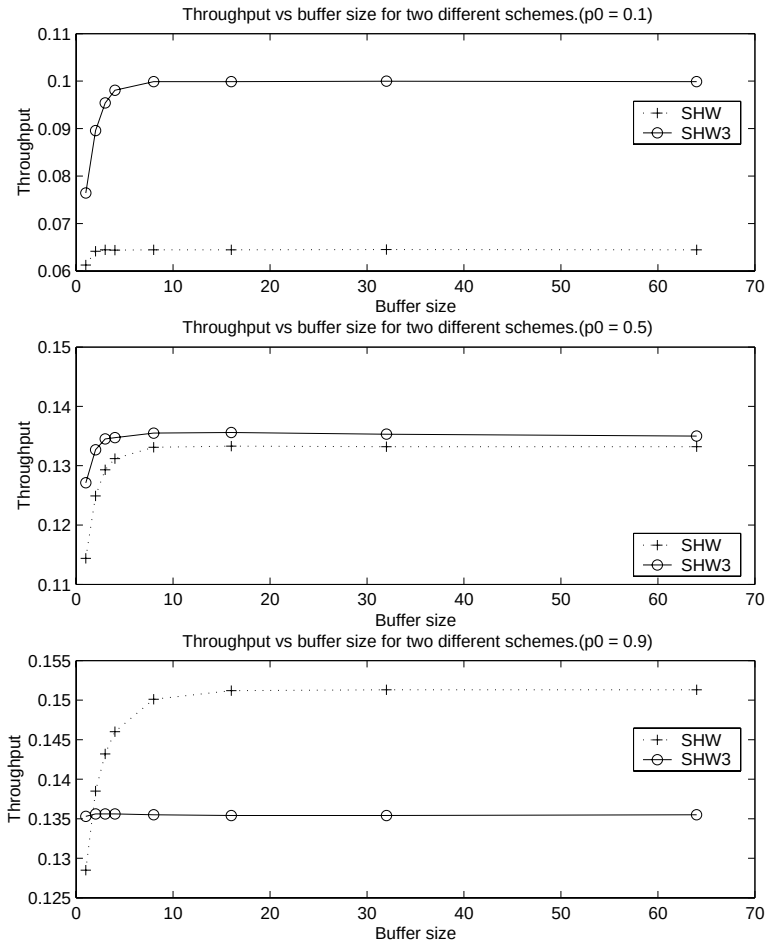


Figure 3-9: Relation of throughput and size of buffer using SHW and SHW3.

Chapter 4

Conclusion

This first part of this thesis 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 [5]).

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.

In the second part of this thesis, we analyzed a problem where nodes of the 2-dimensional N-mesh gnerate packet independently at the beginning of each time slot with probability p_0. Each packet has unit transmission time and is destined for a

uniformly selected node. We showed the stability region of such network. Then, we consider three routing schemes (SHW, RPW, and OPW) and compare their throughput performance. Both the analytic and simulated results show that SHW scheme attains the best throughput performance, OPW scheme the second, and RPW the worst in case of no buffer at each node. In the buffered case, the three schemes have similar performance. Also, a small buffer size can achieve throughput close to that of the infinite buffer size. Most of the packet drop occur because the new packet cannot enter the system. Once a new packet is admitted to the system, it has a rather high probability to reach its desintation node.

Bibliography

[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.

[16] A. G. Greenberg and Bruce Hajek, "Deflection routing in hypercube networks," *IEEE Trans. on Communications*, vol. 40, no. 6, pp. 1070-1081, Jun. 1992.

[17] G. D. Stamoulis and J. N. Tsitsiklis, "The efficiency of greedy routing in hypercubes and butterflies," *IEEE Trans. on Communications*, vol. 42, no. 11, pp. 3051-3061, Nov. 1994.

[18] E. A. Varvarigos and D. P. Bertsekas, "Performance of hypercube routing schemes with or without buffering," *IEEE/ACM Trans. on Networking*, vol. 2, no. 3, pp. 299-311, Jun. 1994.

[19] D. Bertsimas, *Stability, performance and optimization of queueing systems*, Lecture notes.

[20] E. Ekici, I. F. Akyildiz, and M. D. Bender, "A multicast routing algorithm for LEO satellite IP networks," *IEEE/ACM Trans. on Networking*, vol. 10, no. 2, pp. 183-192, Apr. 2002.

[21] J. Walrand, *An Introduction to Queueing Networks*, Prentice Hall, 1988.

[22] G. Jaffe, "Modern Warfare Strains Capacity to Communicate," *The Wall Street Journal*, Apr. 10, 2002.