

# Oblivious Routing in Fat-Tree Based System Area Networks With Uncertain Traffic Demands

Xin Yuan, *Member, IEEE*, Wickus Nienaber, Zhenhai Duan, *Member, IEEE*, and Rami Melhem, *Fellow, IEEE*

**Abstract**—We study oblivious routing in fat-tree-based system area networks with deterministic routing under the assumption that the traffic demand is uncertain. The performance of a routing algorithm under uncertain traffic demands is characterized by the oblivious performance ratio that bounds the relative performance of the routing algorithm with respect to the optimal algorithm for any given traffic demand. We consider both single-path routing, where only one path is used to carry the traffic between each source–destination pair, and multipath routing, where multiple paths are allowed. For single-path routing, we derive lower bounds of the oblivious performance ratio for different fat-trees and develop routing schemes that achieve the optimal oblivious performance ratios for commonly used topologies. Our evaluation results indicate that the proposed oblivious routing schemes not only provide the optimal worst-case performance guarantees but also outperform existing schemes in average cases. For multipath routing, we show that it is possible to obtain an optimal scheme for all traffic demands (an oblivious performance ratio of 1). These results quantitatively demonstrate the performance difference between single-path routing and multipath routing in fat-trees.

**Index Terms**—Fat-tree, oblivious routing, system area networks.

## I. INTRODUCTION

THE FAT-TREE topology has many properties that make it attractive for large-scale interconnects and system area networks [17], [18]. Most importantly, the bisection bandwidth of the fat-tree topology scales linearly with the number of network ports.<sup>1</sup> The topology is also inherently highly resilient with a large number of redundant paths. The fat-tree topology is very popular for building medium and large system area networks [15], [21]. In particular, it has been widely adopted in high-performance computing (HPC) clusters that employ the off-the-shelf high-speed system area networking technology, such as Myrinet [22] and InfiniBand [16].

Manuscript received August 27, 2007; revised May 28, 2008; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor A. Orda. First published May 26, 2009; current version published October 14, 2009. This work is supported in part by the National Science Foundation (NSF) under Grants CCF-0342540, CCF-0541096, and CCF-0551555.

X. Yuan, W. Nienaber and Z. Duan are with the Department of Computer Science, Florida State University, Tallahassee, FL 32306 USA (e-mail: xyuan@cs.fsu.edu; nienaber@cs.fsu.edu; duang@cs.fsu.edu).

R. Melhem is with the Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260 USA (e-mail: melhem@cs.pitt.edu).

Digital Object Identifier 10.1109/TNET.2009.2012853

<sup>1</sup>As originally defined [17], fat-trees are a very broad class of networks with substantial flexibility regarding the bisection bandwidth. The fat-tree topologies used in the current system area networks, which are also known as constant bisectional bandwidth (CBB) networks [21], scale bisection bandwidth linearly with the number of network ports. This paper focuses on this type of fat-trees.

Although the fat-tree topology provides rich connectivity, having a fat-tree topology alone does not guarantee high performance; the routing mechanism also plays a crucial role. Historically, adaptive routing, which dynamically builds the path for a packet based on the network condition, has been used with the fat-tree topology to achieve load balancing [18]. However, the routing in the current major system area networking technology, such as InfiniBand and Myrinet, is deterministic [16], [22]. For a fat-tree-based system area network with deterministic routing, it is important to employ an efficient load-balanced routing scheme in order to fully exploit the rich connectivity provided by the fat-tree topology.

Traditional load-balanced routing schemes usually optimize the network usage for a given traffic demand. Such *demand-specific* schemes may not be effective for system area networks where the traffic demand is uncertain and changing. Consider, for example, the traffic demand in a large HPC cluster. Since many users share such a system and can run different applications, the traffic demand depends both on how the processing nodes are allocated to different applications and on the communication requirement within each application. Hence, an ideal routing scheme should provide load balancing across all possible traffic patterns. This requirement motivates us to study *demand-oblivious routing schemes* that determine routes independent of the traffic demand. Our focus is on oblivious routing with routes being set deterministically since most current system area networks use deterministic routing. It has recently been demonstrated that oblivious routing can promise excellent performance guarantees with uncertain traffic demands in the Internet environment [1], [2], [28].

We investigate oblivious routing in fat-tree networks with deterministic routing under the assumption that the traffic demand is uncertain. For a given traffic demand, the performance of a routing scheme is measured by the *maximum link load* metric. The performance of a routing algorithm under uncertain traffic demands is characterized by the *oblivious performance ratio* [1]. The formal definition of the oblivious performance ratio will be introduced in Section II. Informally, a routing algorithm  $r$  has an oblivious performance ratio of  $x$  if for *any* traffic demand, the performance (maximum link load) of  $r$  is at most  $x$  times that of the optimal routing algorithm for this demand. An oblivious performance ratio of 1 means that the algorithm is optimal for all traffic demands.

This study focuses on the fat-tree topologies that are formed with  $m$ -port switches, where  $m$  is a parameter that is restricted to be a multiple of 2. Although the results are obtained for this type of fat-trees, the results, as well as our analysis techniques, can be extended to other types of fat-tree topologies. We consider both single-path routing, where only one path is used to

carry the traffic between each source-destination pair, and multipath routing, where multiple paths are allowed. Let  $H$  be the height of the fat-tree. The major results of this paper include the following.

- For single-path routing, we prove that 1) when  $H = 3$ , the oblivious performance ratio of any single-path routing algorithm is at least  $\sqrt{\frac{m}{2}}$ ; 2) when  $H = 4$ , the oblivious performance ratio of any single-path routing algorithm is at least  $\frac{m}{2}$ ; and 3) when  $H > 4$ , the oblivious performance ratio of any single-path routing algorithm is at least  $(\frac{m}{2})^{\lfloor \frac{H-2}{3} \rfloor}$ . We develop optimal single-path oblivious routing schemes that achieve the oblivious performance ratio lower bounds for the cases when  $H = 3$  and  $H = 4$ , which implies that the lower bounds for these two cases are tight. These routing schemes are sufficient for most practical cases since the heights of most practical fat-trees are no more than 4. The results of our performance study indicate that the proposed optimal oblivious routing schemes not only provide the optimal worst-case performance guarantees among all single-path routing schemes but also outperform existing schemes in average cases.
- For multipath routing, we show that it is possible to obtain a scheme with an oblivious performance ratio of 1, that is, an optimal scheme for any traffic demand. This suggests that multipath routing is much more effective in balancing network loads than single-path routing in fat-trees. Note that although it is well known that single-path routing is simple but not as effective as multipath routing in balancing network loads in general, the performance difference between single-path routing and multipath routing in fat-trees is not well understood. Without a clear understanding of the performance difference, it is difficult to make a wise decision about whether a system should use single-path routing for its simplicity or multipath routing for its performance. This paper quantifies the performance difference and resolves this issue.

The rest of the paper is organized as follows. In Section II, we formally define routing and the metrics for evaluating routing schemes, specifically the fat-tree topology. In Section III, we present the results for multipath routing. In Section IV, we derive the lower bounds of the oblivious performance ratio for any single-path routing scheme in fat-trees. In Section V, we give the optimal oblivious routing schemes for commonly used fat-trees. Section VI reports the results of our performance study. Section VII describes the related work. Finally, Section VIII concludes the paper.

## II. BACKGROUND

### A. Routing and Its Performance Metrics

Let the system have  $N$  processing nodes, numbered from 0 to  $N - 1$ . The traffic demand is described by an  $N \times N$  traffic matrix,  $TM$ . Each entry  $tm_{i,j}$  in  $TM$ ,  $0 \leq i \leq N - 1$ ,  $0 \leq j \leq N - 1$ , denotes the amount of traffic from node  $i$  to node  $j$ . Let  $A$  be a set and  $|A|$  be the size of the set.

A routing specifies how the traffic for each source-destination (SD) pair is routed across the network. We consider single-path routing, where only one path can be used for each SD pair, and

multipath routing, where multiple paths can be used. In multipath routing, each path for an SD pair routes a fraction of the traffic for the SD pair.

A multipath routing is characterized by a set of paths  $MP_{i,j} = \{MP_{i,j}^1, MP_{i,j}^2, \dots, MP_{i,j}^{|MP_{i,j}|}\}$  for each SD pair  $(i, j)$ , and the fraction of the traffic routed through each path  $f_{i,j} = \{f_{i,j}^k | k = 1, 2, \dots, |MP_{i,j}|\}$ .  $\sum_{k=1}^{|MP_{i,j}|} f_{i,j}^k = 1$ . Let link  $l \in MP_{i,j}^k$ . The contribution of the traffic  $tm_{i,j}$  to link  $l$  through path  $MP_{i,j}^k$  is thus  $tm_{i,j} \times f_{i,j}^k$ . Single-path routing is a special case of multipath routing where  $|MP_{i,j}| = 1$  and all traffic from node  $i$  to node  $j$  is routed through  $MP_{i,j}^1$  ( $f_{i,j}^1 = 1$ ). Hence, a single-path routing can be specified by a path  $MP_{i,j}^1$  for each SD pair  $(i, j)$ .

For a given traffic matrix, the performance of a routing is measured by the maximum link load. Since all links in a fat-tree network have the same capacity, the maximum link load is equivalent to the maximum link utilization. Let  $Links$  denote the set of all links in the network. For a multipath routing  $mr$ , the maximum link load is given by

$$\text{MLOAD}(mr, TM) = \max_{l \in Links} \left\{ \sum_{i,j,k \text{ such that } l \in MP_{i,j}^k} tm_{i,j} \times f_{i,j}^k \right\}.$$

For a single-path routing  $sr$ , the formula simplifies to

$$\text{MLOAD}(sr, TM) = \max_{l \in Links} \left\{ \sum_{i,j \text{ such that } l \in MP_{i,j}^1} tm_{i,j} \right\}.$$

An optimal routing for a given traffic matrix  $TM$  is a routing that minimizes the maximum link load. Formally, the optimal load for a traffic matrix  $TM$  is given by

$$\text{OPTU}(TM) = \min_{r \text{ is a routing}} \{\text{MLOAD}(r, TM)\}.$$

The performance ratio of a routing  $r$  on a traffic matrix  $TM$  measures how far  $r$  is from being optimal on  $TM$ . It is defined as the maximum link load of  $r$  divided by the smallest possible maximum link load on  $TM$  [1].

$$\text{PERF}(r, TM) = \frac{\text{MLOAD}(r, TM)}{\text{OPTU}(TM)}.$$

$\text{PERF}(r, TM)$  is at least 1. It is exactly 1 if and only if the routing is optimal for  $TM$ . The definition of the performance ratio follows the “competitive analysis” framework where performance guarantees of a certain solution are provided relative to the best possible solution. As such, the performance ratio is not directly related to the absolute network performance. The performance ratio of a routing scheme may be better when the network is under heavy loads and worse when the network is under light loads. The definition of performance ratio of a routing is extended to be with respect to a set of traffic matrices [1]. Let  $\Gamma$  be a set of traffic matrices. The performance ratio of a routing  $r$  on  $\Gamma$  is defined as

$$\text{PERF}(r, \Gamma) = \max_{TM \in \Gamma} \{\text{PERF}(r, TM)\}.$$

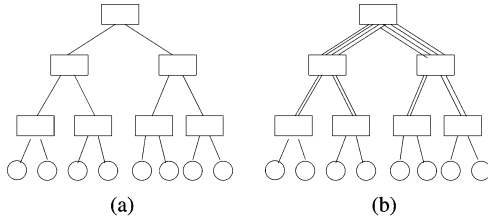


Fig. 1. Topologies of (a) binary tree and (b) binary fat-tree.

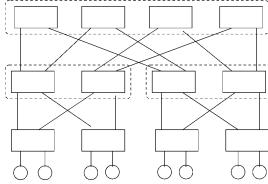


Fig. 2. Approximate the topology in Fig. 1(b).

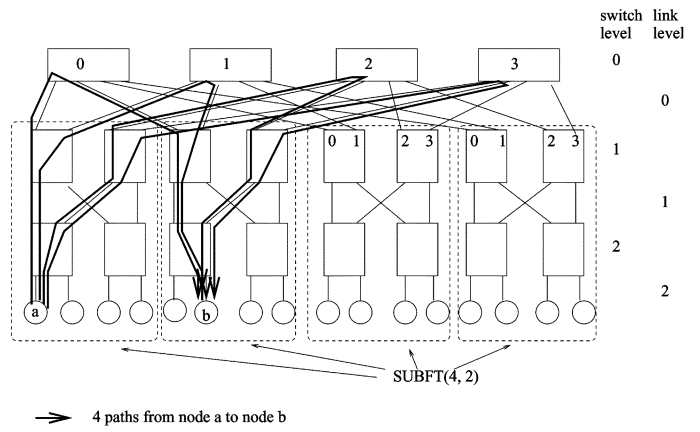
When the set  $\Gamma$  includes all possible traffic matrices, the performance ratio is referred to as the **oblivious performance ratio** [1]. The oblivious performance ratio of a routing  $r$  is denoted by  $\text{PERF}(r)$ . The oblivious performance ratio is the worst performance ratio that a routing obtains with respect to all traffic matrices. A routing with a minimum oblivious ratio is an *optimal oblivious routing* scheme, and its oblivious ratio is the optimal oblivious ratio of the network.

### B. Fat-Tree Topology

In a fat-tree network, all links are bidirectional with the same capacity. Fig. 1 compares a binary tree with a binary fat-tree. In the binary tree, the number of links (and thus the aggregate bandwidth) is reduced by half at each level from the leaves to the root. This can cause congestion toward the root. The binary fat-tree topology remedies this situation by maintaining the same bandwidth at each level of the network.

The fat-tree topology shown in Fig. 1(b) is not practical for building large networks due to the large nodal degree of the root. Alternatives were proposed to approximate such a topology using multistage networks that are formed by nodes with small nodal degrees [8], [15], [26]. For example, the fat-tree in Fig. 1(b) can be approximated by the topology in Fig. 2. These alternatives trade connectivity with implementation simplicity. In this paper, we focus on one such alternative: the fat-tree topologies formed by  $m$ -port switches, where  $m$  is a parameter that is restricted to a multiple of 2. Let an *internal node* in the fat-tree topology be a node with a degree more than 1. All internal nodes in our fat-tree topology have a degree of  $m$  (so that they can be realized by  $m$ -port switches). Such a topology is a minor generalization of the topology proposed in [15]. The technique we developed for this topology can easily be extended for other fat-tree variations.

We will follow the naming convention in [15]. The fat-tree is called  $m$ -port  $n$ -tree and denoted as  $\text{FT}(m, n)$ . The parameter  $m$  in  $\text{FT}(m, n)$ , which must be a multiple of 2, specifies the nodal degree of all internal nodes in the topology. The parameter  $n$  specifies the number of levels of internal nodes in the topology. Thus, the height of  $\text{FT}(m, n)$  is  $n + 1$ . That is,

Fig. 3. The 4-port 3-tree ( $\text{FT}(4, 3)$ ).

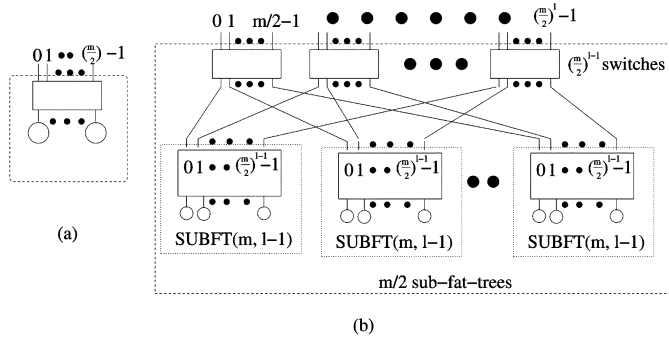
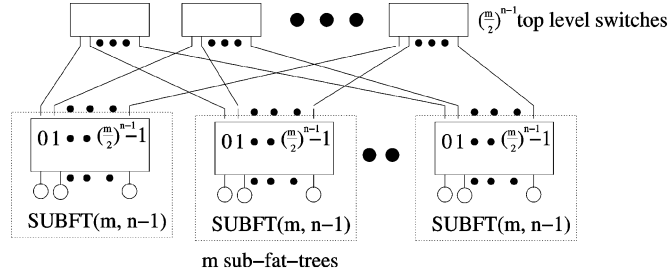
$\text{FT}(m, n)$  is an  $n + 1$  level tree. In the rest of this paper, internal nodes in  $\text{FT}(m, n)$  may also be referred to as *switches* since each of the internal nodes is realized by a switch when the topology is constructed. Similarly, leaf nodes may also be referred to as *processing nodes*. A 4-port 3-tree,  $\text{FT}(4, 3)$ , is shown in Fig. 3.

Next, we will describe how  $\text{FT}(m, n)$  is formed. More details can be found in [15].  $\text{FT}(m, n)$  is formed by connecting the root-level switches to  $m$  sub-fat-trees with  $n - 1$  levels of switches. We will use the notation  $\text{SUBFT}(m, n - 1)$  to denote the sub-fat-trees with  $n - 1$  levels of switches.  $\text{SUBFT}(m, l)$  is different from  $\text{FT}(m, l)$  in that  $\text{SUBFT}(m, l)$  must provide (open-ended) up links for the sub-fat-tree, while  $\text{FT}(m, l)$  does not have up links.  $\text{SUBFT}(m, l)$  is recursively constructed as follows.

When  $l = 1$ ,  $\text{SUBFT}(m, 1)$  contains one  $m$ -port switch.  $\frac{m}{2}$  of the ports in the switch connect to  $\frac{m}{2}$  processing nodes, and  $\frac{m}{2}$  ports remain open. We will call these opened ports *up-link ports* since they will be used to connect to the upper-level switches. We denote the number of up-link ports in  $\text{SUBFT}(m, l)$  as  $\text{nu}(m, l)$ .  $\text{nu}(m, 1) = \frac{m}{2}$ . As will be shown later,  $\text{nu}(m, l) = (\frac{m}{2})^l$ . The up-link ports in  $\text{SUBFT}(m, l)$  are numbered from 0 to  $\text{nu}(m, l) - 1$ .

$\text{SUBFT}(m, l)$  is formed by connecting  $\text{nu}(m, l - 1) = (\frac{m}{2})^{l-1}$   $m$ -port top-level (of the sub-fat-tree) switches with  $\frac{m}{2}$   $\text{SUBFT}(m, l - 1)$ 's. Each of the top-level switches uses  $\frac{m}{2}$  ports to connect to all  $\frac{m}{2}$  of the  $\text{SUBFT}(m, l - 1)$ 's. Let us number top-level switches from 0 to  $\text{nu}(m, l - 1) - 1$ . The up-link ports  $i$ ,  $0 \leq i < \text{nu}(m, l - 1)$ , in all of the  $\text{SUBFT}(m, l - 1)$ 's are connected to top-level switch  $i$ . The remaining  $\frac{m}{2}$  ports in a top-level switch are up-link ports of  $\text{SUBFT}(m, l)$ . Top-level switch  $i$  provides up-link ports  $\frac{m}{2} \times i$  to  $\frac{m}{2} \times (i + 1) - 1$  for  $\text{SUBFT}(m, l)$ . Fig. 4(a) shows  $\text{SUBFT}(m, 1)$ , and (b) shows the structure of  $\text{SUBFT}(m, l)$ . Clearly,  $\text{nu}(m, l) = \text{nu}(m, l - 1) \times \frac{m}{2} = (\frac{m}{2})^l$ . Hence, each  $\text{SUBFT}(m, l)$  has  $(\frac{m}{2})^l$  up-link ports and connects to  $(\frac{m}{2})^l$  processing nodes.

$\text{FT}(m, n)$  is formed by having  $\text{nu}(m, n - 1) = (\frac{m}{2})^{n-1}$  root-level switches connecting with  $m$   $\text{SUBFT}(m, n - 1)$ 's. Let us number top-level switches from 0 to  $(\frac{m}{2})^{n-1} - 1$ . The up-link port  $i$ ,  $0 \leq i < \text{nu}(m, n - 1)$ , in all of the  $\text{SUBFT}(m, n - 1)$ 's are connected to root-level switch  $i$ .

Fig. 4. (a)  $SUBFT(m, 1)$  and (b)  $SUBFT(m, n)$ .Fig. 5. The structure of  $FT(m, n)$ .

1)'s is connected to top-level switch  $i$ . Each of the  $m$  ports in the root-level switch connects to one  $SUBFT(m, n-1)$ . The structure of  $FT(m, n)$  is shown in Fig. 5.  $FT(m, n)$  supports  $m(\frac{m}{2})^{n-1}$  processing nodes. The root level contains  $nu(m, n-1) = (\frac{m}{2})^{n-1}$  switches, and each of the other  $n-1$  layers has  $2 \times (\frac{m}{2})^{n-1}$  switches. Hence, the total number of switches in  $FT(m, n)$  is  $(2n-1) \times (\frac{m}{2})^{n-1}$ .

Let us number the  $n$  levels of switches in  $FT(m, n)$  from 0 to  $n-1$  (root-level switches being level 0 switches). We will classify the links according to their levels. For  $0 \leq i < n-1$ , the links connecting level  $i$  switches with level  $i+1$  switches are called level  $i$  links. The links connecting level  $n-1$  switches with the processing nodes are level  $n-1$  links. All links in  $FT(m, n)$  are bidirectional links, with an up channel for communication from a lower-level switch to an upper-level switch and a down channel from an upper-level switch to a lower-level switch. We will use the term *level  $i$  up link* to denote an up channel from a level  $i+1$  switch to a level  $i$  switch, and *level  $i$  down link* to denote a down channel from a level  $i$  switch to a level  $i+1$  switch. A path between two processing nodes in  $FT(m, n)$  has two phases. The first phase contains only up channels, and the second phase contains only down channels.

From the definition of  $FT(m, n)$ , one can easily derive the following properties.

**Property 1:**  $FT(m, n)$  contains  $m$   $SUBFT(m, n-1)$ 's,  $m \times \frac{m}{2}$   $SUBFT(m, n-2)$ 's,  $\dots$ ,  $m \times (\frac{m}{2})^{n-2}$   $SUBFT(m, 1)$ 's.

Level 0 (root-level) switches do not belong to any sub-fat-tree. Each level 1 switch is in a  $SUBFT(m, n-1)$ , each level 2 switch is in a  $SUBFT(m, n-1)$  and a  $SUBFT(m, n-2)$ , and so on. A level  $i$  switch,  $1 \leq i \leq n-1$ , is in a  $SUBFT(m, n-1)$ , a  $SUBFT(m, n-2)$ ,  $\dots$ , and a  $SUBFT(m, n-i)$ . In  $FT(m, n)$ , we will call switches in levels  $0, 1, \dots$ , and  $n-i-1$  the upper-level switches for  $SUBFT(m, i)$ . The upper-

level switches for  $SUBFT(m, i)$  provide connectivity among all  $SUBFT(m, i)$ 's.

**Property 2:** Through upper-level switches for  $SUBFT(m, i)$ ,  $1 \leq i \leq n-1$ , an up-link port in a  $SUBFT(m, i)$  is only connected to an up-link port of the same port number in any other  $SUBFT(m, i)$ . More specifically, up-link port  $j$ ,  $0 \leq j \leq nu(m, i)-1$ , in one  $SUBFT(m, i)$  is only connected with the up-link port  $j$  of other  $SUBFT(m, i)$ 's (but not other ports) through upper-level switches for  $SUBFT(m, i)$ .

It is clear that this property is true for  $SUBFT(m, n-1)$ . The property for general  $SUBFT(m, i)$ ,  $1 \leq i < n-1$ , can be formally proven by induction on  $i$  (with base case  $i = n-1$ ) and by examining how the top-level switches in  $SUBFT(m, i)$ 's are connected.

**Property 3:** Let  $SUBFT(m, i)$  be the smallest sub-fat-tree in  $FT(m, n)$  that contains two processing nodes  $a$  and  $b$ ; there exist  $(\frac{m}{2})^{i-1}$  different shortest paths from  $a$  to  $b$ . If such a sub-tree does not exist, there are  $(\frac{m}{2})^{n-1}$  different shortest paths from  $a$  to  $b$ . In this case,  $a$  and  $b$  are in different top-level sub-fat-trees ( $SUBFT(m, n-1)$ 's).

Fig. 3 shows an example. From node  $a$  to node  $b$  in  $FT(4, 3)$ , there are  $(\frac{m}{2})^{n-1} = 2^2 = 4$  shortest paths. In both cases in Property 3, the number of shortest paths between any two nodes can be represented as  $(\frac{m}{2})^x$ , with the value of  $x$ ,  $0 \leq x \leq n-1$ , depending on the positions of the source and the destination.

**Property 4:** In  $FT(m, n)$ , let there exist  $(\frac{m}{2})^x$  different shortest paths from processing node  $s$  to processing node  $d$ . Each of the level  $n-1-i$  up/down links that carry traffic from  $s$  to  $d$  is used by  $(\frac{m}{2})^{x-i}$  shortest paths,  $0 \leq i \leq x$ .

This property is intuitive. For example, level  $n-1$  links are the links connecting processing nodes. Hence, all paths from the processing node connected by a level  $n-1$  link must use the link. This is the case when  $i = 0$ : All  $(\frac{m}{2})^x$  shortest paths use the link. For the next level ( $i = 1$ ), a source will have  $\frac{m}{2}$  choices (the fan-out from the first switch) to go to another node. Thus, each such link will be used by  $(\frac{m}{2})^x / \frac{m}{2} = (\frac{m}{2})^{x-1}$  shortest paths. The cases for links in other levels are similar. Consider the four paths from node  $a$  to node  $b$  in Fig. 3. All four paths use the level 2 up/down links (the link connecting the processing node), two paths use each of the level 1 up/down links that carry traffic from  $a$  to  $b$ , and one path uses each of the level 0 up/down links.

**Property 5:** In  $FT(m, n)$ , a level  $i$ ,  $0 \leq i \leq n-1$ , up link carries traffic from at most  $(\frac{m}{2})^{n-1-i}$  source nodes. A level  $i$  down link carries traffic to at most  $(\frac{m}{2})^{n-1-i}$  nodes.

This property is also intuitive. For example, when  $i = n-1$ , a level  $i = n-1$  link directly connects to a processing node. Therefore, such a link carries traffic to/from at most  $(\frac{m}{2})^{n-1-(n-1)} = 1$  node. When  $i = n-2$ , the link connects to a level  $n-1$  switch; and such a link carries traffic to/from the  $(\frac{m}{2})^{n-1-(n-2)} = \frac{m}{2}$  nodes directly connected to that switch.

### III. MULTIPATH OBLIVIOUS ROUTING

Let the traffic matrix be  $TM$  with entries  $tm_{i,j}$ ,  $0 \leq i \leq N-1$  and  $0 \leq j \leq N-1$ , specifying the amount of traffic from node  $i$  to node  $j$ . The total traffic sent from node  $i$  is  $\sum_j tm_{i,j}$ , and the total traffic received by node  $i$  is  $\sum_j tm_{j,i}$ .

Since there is only one link connecting each processing node to the network, such traffic must be carried on that link regardless of the routing scheme. Hence, for any routing scheme, the load on the link (which has two directions) connecting to node  $i$  is  $\max\{\sum_j tm_{i,j}, \sum_j tm_{j,i}\}$ . We define the base load of a traffic matrix  $TM$  as

$$\text{baseload}(TM) = \max_{0 \leq i \leq N-1} \left\{ \max \left\{ \sum_j tm_{i,j}, \sum_j tm_{j,i} \right\} \right\}.$$

Clearly, for any  $TM$  and any  $FT(m, n)$ , the minimum maximum link load with any routing scheme, single-path or multipath, is at least  $\text{baseload}(TM)$ :

$$\text{OPTU}(TM) \geq \text{baseload}(TM).$$

We will give a multipath routing algorithm, *OMRMN*, such that  $\text{MLOAD}(\text{OMRMN}, TM) = \text{baseload}(TM)$  for any  $FT(m, n)$  and any  $TM$  ( $\text{PERF}(\text{OMRMN}) = 1$ ). *OMRMN* works as follows. Let  $X$  be the number of shortest paths between processing nodes  $i$  and  $j$  (since processing nodes are in the same level of the fat-tree,  $X$  is fixed for all pairs of nodes), and let the  $X$  different shortest paths between nodes  $i$  and  $j$  be  $P_{i,j}^1, P_{i,j}^2, \dots, P_{i,j}^X$  (From Property 3 in Section II-B, these paths can be easily found). *OMRMN* makes use of all the paths and allocates exactly the same amount of traffic on each path. That is,  $MP_{i,j} = \{P_{i,j}^1, P_{i,j}^2, \dots, P_{i,j}^X\}$  and  $f_{i,j}^1 = \dots = f_{i,j}^X = \frac{1}{X}$ .

**Theorem 1:** For any  $FT(m, n)$ ,  $\text{PERF}(\text{OMRMN}) = 1$ .

*Proof:* Since in  $FT(m, n)$  the up links and down links are symmetrical, it is sufficient to show that, for any traffic matrix  $TM$ , the load on each up link  $l$  is no more than  $\text{baseload}(TM)$ . Consider a source  $s$  in  $FT(m, n)$ . Let us denote  $LD(s) = \sum_{j \neq s} tm_{s,j}$  the total amount of traffic sent from node  $s$ . For each source node  $s$ , using *OMRMN*, each level  $n-1-i$  link carries at most  $LD(s)/(\frac{m}{2})^i$  traffic since the traffic is evenly distributed among the  $(\frac{m}{2})^i$  links at level  $n-1-i$  that can carry traffic from node  $s$  (Property 4). In addition, each level  $n-1-i$  link carries traffic from at most  $(\frac{m}{2})^i$  source nodes (Property 5). Let the nodes be  $s_0, s_1, \dots, s_{(\frac{m}{2})^i-1}$  and the load on a level  $n-1-i$  link  $l$  be  $\text{load}(l)$ . We have  $\text{load}(l) \leq \sum_{j=0}^{(\frac{m}{2})^i-1} \frac{LD(s_j)}{(\frac{m}{2})^i}$ . Since  $LD(s_j) \leq \text{baseload}(TM)$ ,  $0 \leq j \leq (\frac{m}{2})^i - 1$ , we have  $\text{load}(l) \leq \text{baseload}(TM)$ . Note that there is no restriction on the link  $l$  and traffic matrix  $TM$ . Hence, for all links and all traffic matrices, we have  $\text{load}(l) \leq \text{baseload}(TM) \leq \text{OPTU}(TM)$ . Hence,  $\text{PERF}(\text{OMRMN}) = 1$ .  $\square$

**Corollary 1:** For any  $FT(m, n)$  and any traffic matrix  $TM$ ,  $\text{OPTU}(TM) = \text{baseload}(TM)$ .  $\square$

Theorem 1 states that *OMRMN* is optimal for all traffic patterns. However, *OMRMN* uses all of the shortest paths between two processing nodes. We will refer to it as an *unrestricted* multipath routing scheme. Practical system area networks cannot support such unrestricted multipath routing in large fat-trees, and the performance of more restricted forms of routing must be studied.

#### IV. LOWER BOUNDS OF OBLIVIOUS PERFORMANCE RATIO FOR SINGLE-PATH ROUTING

We will first introduce some concepts that will be used later in the lower bound derivation. Let  $A = \{(s_1, d_1), (s_2, d_2), \dots\}$  be a set of source-destination (SD) pairs.

**Definition 1:** The set of SD pairs  $A = \{(s_1, d_1), (s_2, d_2), \dots\}$  is said to be **node disjoint** if, for any  $(s_i, d_i) \in A$  and  $(s_j, d_j) \in A$  with  $i \neq j$ , we have  $s_i \neq s_j$  and  $d_i \neq d_j$ .

Basically, in a node disjoint set of SD pairs, each source (in the source-destination pairs) appears in the set as a source exactly once, and each destination appears as a destination exactly once. A node may appear as a source and as a destination in a node disjoint set. For example,  $\{(1, 2), (1, 3)\}$  is not a node disjoint set, while  $\{(1, 2), (3, 1)\}$  is.

**Definition 2:** For a set of SD pairs  $A$ , a set of SD pairs  $B$  is said to be a **node disjoint subset** of  $A$  when 1)  $B \subseteq A$  and 2)  $B$  is a node disjoint set.

**Definition 3:** For a given set of SD pairs  $A$ , a set of SD pairs  $B$  is said to be a **largest node disjoint subset** of  $A$  when 1)  $B$  is a node disjoint subset of  $A$ , and 2) for any node disjoint subset  $C$  of  $A$ ,  $|B| \geq |C|$ . We let  $L(A)$  be the size of a largest node disjoint subset of  $A$ .

Let  $S_s^A = \{(s, x) | (s, x) \in A\}$  be the set of SD pairs in  $A$  with source node  $s$  and  $D_d^A = \{(x, d) | (x, d) \in A\}$  be the set of SD pairs in  $A$  with destination node  $d$ .  $\text{SRC}(A) = \{s | \exists (s, d) \in A\}$  is the set of source nodes in  $A$  and  $\text{DST}(A) = \{d | \exists (s, d) \in A\}$  is the set of destination nodes in  $A$ . We denote by  $\text{LS}(A)$  the largest number of SD pairs in  $A$  either with the same source or with the same destination. Formally,

$$\text{LS}(A) = \max \left\{ \max_{s \in \text{SRC}(A)} |S_s^A|, \max_{d \in \text{DST}(A)} |D_d^A| \right\}.$$

For any node  $i$ ,  $|S_i^A| \leq \text{LS}(A)$  and  $|D_i^A| \leq \text{LS}(A)$ .

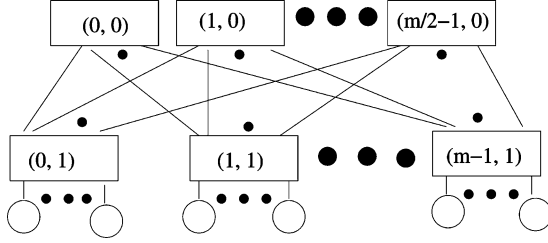
Consider for example  $A = \{(1, 2), (1, 3), (2, 1), (2, 4), (3, 1)\}$ . The set  $\{(1, 2), (2, 1)\}$  is a node disjoint subset of  $A$  but not a largest node disjoint subset. Both  $\{(1, 2), (2, 4), (3, 1)\}$  and  $\{(1, 3), (2, 4), (3, 1)\}$  are largest node disjoint subsets of  $A$ . Hence,  $L(A) = 3$ .  $\text{SRC}(A) = \{1, 2, 3\}$ , and  $\text{DST}(A) = \{1, 2, 3, 4\}$ .  $S_1^A = \{(1, 2), (1, 3)\}$ ,  $S_2^A = \{(2, 1), (2, 4)\}$ , and  $S_3^A = \{(3, 1)\}$ .  $D_1^A = \{(2, 1), (3, 1)\}$ ,  $D_2^A = \{(1, 2)\}$ ,  $D_3^A = \{(1, 3)\}$ , and  $D_4^A = \{(2, 4)\}$ . Hence,  $\text{LS}(A) = 2$ .

**Lemma 1:** Let  $A$  be a set of SD pairs,  $|\text{SRC}(A)| \geq L(A)$  and  $|\text{DST}(A)| \geq L(A)$ .

*Proof:* Straight-forward from the largest node disjoint subset definition.  $\square$

**Lemma 2:** Let  $A$  and  $B$  be two sets of SD pairs,  $L(A) + L(B) \geq L(A \cup B)$ .

*Proof:* Let  $C$  be a largest node disjoint subset of  $A \cup B$ .  $|C| = L(A \cup B)$ . Each element in  $C$  must either be in  $A$ , or in  $B$  (or in both  $A$  and  $B$ ). Let  $C_A = \{(s, d) | (s, d) \in C \cap A\}$  and  $C_B = \{(s, d) | (s, d) \in C \cap B\}$ . We have  $|C_A| + |C_B| \geq |C| = L(A \cup B)$ . Since  $C_A$  is a node disjoint subset of  $A$  and  $C_B$  is a node disjoint subset of  $B$ , by definition,  $L(A) \geq |C_A|$  and  $L(B) \geq |C_B|$ . Hence,  $L(A) + L(B) \geq L(A \cup B)$ .  $\square$

Fig. 6.  $FT(m, 2)$  topology.

**Lemma 3:** Let  $A$  be a set of SD pairs. If there is a source node  $s$  such that  $|S_s^A| > L(A)$ , then  $L(A - S_s^A) = L(A) - 1$ .

**Proof:** Since  $S_s^A$  has only one source node,  $L(S_s^A) = 1$ . From Lemma 2, we have  $L(A - S_s^A) + L(S_s^A) \geq L((A - S_s^A) \cup S_s^A) = L(A)$ . Hence,  $L(A - S_s^A) \geq L(A) - 1$ .

Next, we will prove  $L(A - S_s^A) \leq L(A) - 1$  by contradiction. Let  $B = \{(s_1, d_1), (s_2, d_2), \dots, (s_k, d_k)\}$  be a largest node disjoint subset of  $A - S_s^A$ . Assume that  $|B| = k = L(A - S_s^A) > L(A) - 1$ . Since  $A - S_s^A$  is a subset of  $A$ ,  $k \leq L(A)$ . Hence,  $k$  must be exactly equal to  $L(A)$ . Since  $|S_s^A| > L(A) = k$ , there exists at least one  $(s, d) \in S_s^A$  such that  $d \neq d_i, 1 \leq i \leq k$ . Hence, the set  $C = B \cup \{(s, d)\}$  is node disjoint and  $|C| = L(A) + 1$ . Since  $C$  is a node disjoint subset of  $A$ ,  $|C| \leq L(A)$ . This is the contradiction. Hence,  $L(A - S_s^A) = L(A) - 1$ .  $\square$

**Lemma 3a:** Let  $A$  be a set of SD pairs. If there is a node  $d$  such that  $|D_d^A| > L(A)$ , then  $L(A - D_d^A) = L(A) - 1$ .  $\square$

**Lemma 4:** Let  $A$  be a set of SD pairs. If there exist  $k$  source nodes  $s_i, 1 \leq i \leq k$ , such that  $|S_{s_i}^A| > L(A)$ , and  $l$  destination nodes  $d_j, 1 \leq j \leq l$ , such that  $|D_{d_j}^A| > L(A)$ , then  $L(A - \bigcup_{i=1}^k S_{s_i}^A - \bigcup_{j=1}^l D_{d_j}^A) = L(A) - k - l$ .

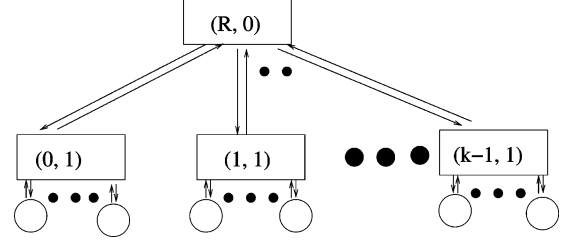
**Proof:** The conclusion in this lemma is obtained by repeatedly applying Lemmas 3 and 3a.  $\square$

**Lemma 5:** Let  $A$  be a set of SD pairs.  $|A| \leq L(A) \times LS(A)$ .

**Proof:** See Appendix.  $\square$

We use a topology called extended 2-layer fat-tree ( $EFT2(m, k)$ ) in the derivation of the lower bounds.  $EFT2(m, k)$  contains two levels of switches. The top level contains  $\frac{m}{2} k$ -port switches. The bottom level contains  $k m$ -port switches. Half of the  $m$  ports in the bottom-level switches are used to connect to processing nodes, and the other half connect to top-level switches. There is a link between each top-level switch and each bottom-layer switch. The structure of  $EFT2(m, k)$  is similar to  $FT(m, 2)$  (shown in Fig. 6) except that  $EFT2(m, k)$  allows different types of switches in the two levels.  $FT(m, 2)$  is the same as  $EFT2(m, m)$ . A subgraph of  $EFT2(m, k)$ , called  $SEFT2(m, k)$ , contains all lower-level switches and processing nodes in  $EFT2(m, k)$  but only one top-level switch. Fig. 7 shows the  $SEFT2(m, k)$  topology, which is basically a regular tree topology with the root having  $k$  children and each level 1 switch having  $\frac{m}{2}$  children. In Fig. 7, we separate the two directional channels.

In  $EFT2(m, k)$ , there are  $\frac{m}{2}$  upper-level switches, and each of the  $k$  lower-level switches connects directly to each of the upper-level switches. Hence, by routing different SD

Fig. 7.  $SEFT2(m, k)$  topology.

pairs through different upper-level switches, this topology can provide  $\frac{m}{2}$  link disjoint paths for any  $\frac{m}{2}$  node disjoint SD pairs. As a result, if a scheme routes  $X, X \leq \frac{m}{2}$ , node disjoint SD pairs on a link, the performance ratio of this routing scheme is at least  $X$  since the  $X$  node disjoint SD pairs can be routed using a set of link disjoint paths. Lemmas 6 and 7 formally capture this intuition.

**Lemma 6:** Let the processing nodes in  $EFT2(m, k)$  be numbered from 0 to  $N - 1$ . Let  $A = \{(s_1, d_1), \dots, (s_{|A|}, d_{|A|})\}$  be a set of node disjoint SD pairs (for  $1 \leq i \leq |A|$ ,  $s_i \in \{0, \dots, N - 1\}$ , and  $d_i \in \{0, \dots, N - 1\}$ ). When  $|A| \leq \frac{m}{2}$ , the SD pairs in  $A$  can be routed in  $EFT2(m, k)$  with  $|A|$  link disjoint paths.

**Proof:** Since  $|A| \leq \frac{m}{2}$ , we can assign a different top-level switch  $SW_i$  for each SD pair  $(s_i, d_i) \in A$ . For each  $(s_i, d_i)$ , if  $s_i$  and  $d_i$  connect to the same (lower-level) switch  $SW$ , the path from  $s_i$  to  $d_i$  is  $s_i \rightarrow SW \rightarrow d_i$ . If  $s_i$  and  $d_i$  are not in the same switch, let  $s_i$  connect to  $SW_1$  and  $d_i$  to  $SW_2$ . The path for  $(s_i, d_i)$  is  $s_i \rightarrow SW_1 \rightarrow SW_i \rightarrow SW_2 \rightarrow d_i$ . Since  $A$  is node disjoint, all of the paths are link disjoint.  $\square$

**Lemma 7:** Let  $sr$  be a single-path routing on  $EFT2(m, k)$ . Assume that under routing  $sr$ , there exists a link  $l$  that carries traffic for a set  $A$  of node disjoint SD pairs,  $|A| \leq \frac{m}{2}$ . Then,  $PERF(sr) \geq |A|$ .

**Proof:** Consider a traffic matrix  $TM$  where  $tm_{i,j} = 1$  for all  $(i, j) \in A$  and all other entries are 0. From Lemma 6, there exists a routing scheme  $sr'$  that routes the SD pairs in  $A$  using link disjoint paths. Hence,  $MLOAD(sr', TM) = 1$  and  $OPTU(TM) \leq 1$ . Since using routing  $sr$ , the load on link  $l$  is  $|A|$  and  $MLOAD(sr, TM) \geq |A|$ . Hence,

$$PERF(sr) \geq \frac{MLOAD(sr, TM)}{OPTU(TM)} \geq \frac{|A|}{1} = |A|.$$

$\square$

For a single-path routing  $r$ , let us define the *maximum disjoint size* on link  $l$ ,  $mds(r, l)$ , to be the size of the largest node disjoint subset of the set of SD pairs routed on  $l$ . In  $EFT2(m, k)$  and  $SEFT2(m, k)$ , the SD pairs on a level 1 link directly connected to a processing node have at most one source node or one destination node. From Lemma 1, the maximum disjoint size on such a link is at most 1. The *maximum disjoint size* of routing  $r$ ,  $mds(r)$ , is defined as  $mds(r) = \max_{l \in Links} mds(r, l)$ .

Lemma 8 and Theorem 2 prove the lower bound of the oblivious performance ratio for single-path routing on  $EFT2(m, k)$ . The proof follows the following logic. To realize routes for all SD pairs in  $EFT2(m, k)$ , there are  $k(k-1)(\frac{m}{2})^2$  SD pairs that must be routed through upper-level switches. Since there are  $\frac{m}{2}$

upper-level switches in EFT2( $m, k$ ), at least one upper-level switch must carry at least  $k(k-1)\frac{m}{2}$  SD pairs. This particular upper-level switch, the lower switches, and processing nodes form an SEFT2( $m, k$ ). Lemma 8 proves a more general result, implying that to carry traffic for  $k(k-1)\frac{m}{2}$  SD pairs through the root in the SEFT2( $m, k$ ), at least one link must carry traffic for  $\sqrt{\frac{m}{2}}$  node disjoint SD pairs. Combining these results with Lemma 7, we derive the lower bound for EFT2( $m, k$ ).

**Lemma 8:** Consider using the SEFT2( $m, k$ ) topology to route a subset of all possible SD pairs. If the largest of the maximum disjoint sizes of all links is at most  $X$ , the number of SD pairs routed through the root is at most  $k(k-1)X^2$  when  $X \geq \frac{m}{k}$ .

*Proof:* Let  $(s, d)$  be a SD pair. The pair must be routed through the root only when nodes  $s$  and  $d$  are connected to different switches. We will call the root switch in SEFT2( $m, k$ ) switch  $(R, 0)$  and the  $k$  level 1 switches  $(0, 1), (1, 1), \dots, (k-1, 1)$  as shown in Fig. 7. Let  $S$  be a largest set of SD pairs that are routed through the root when the largest of the maximum disjoint sizes of all links is at most  $X$ . Let  $S_{i,j}$ ,  $0 \leq i \neq j \leq k-1$ , be the set of SD pairs in  $S$  with source nodes in switch  $(i, 1)$  and destination nodes in switch  $(j, 1)$ .  $S = \bigcup_{i,j \text{ such that } 0 \leq i \neq j \leq k-1} S_{i,j}$ . Let us denote

$$LX_{i,j}^{\text{src}} = \bigcup_{\substack{a \text{ such that } a \in \text{SRC}(S_{i,j}) \text{ and } |S_a^{S_{i,j}}| > X}} S_a^{S_{i,j}}.$$

Let  $E_{i,j} = |\text{SRC}(LX_{i,j}^{\text{src}})|$ . For the SD pairs in  $S_{i,j}$ ,  $E_{i,j}$  is the number of source nodes in switch  $(i, 1)$ , each of which having more than  $X$  destination nodes in switch  $(j, 1)$ .  $LX_{i,j}^{\text{src}}$  contains all such SD pairs. Similarly, we will denote

$$LX_{i,j}^{\text{dst}} = \bigcup_{\substack{d \text{ such that } d \in \text{DST}(S_{i,j}) \text{ and } |D_d^{S_{i,j}}| > X}} D_d^{S_{i,j}}.$$

Let  $F_{i,j} = |\text{DST}(LX_{i,j}^{\text{dst}})|$ . For the SD pairs in  $S_{i,j}$ ,  $F_{i,j}$  is the number of destination nodes in switch  $(j, 1)$ , each of which having more than  $X$  source nodes in switch  $(i, 1)$ .  $LX_{i,j}^{\text{dst}}$  contains all such SD pairs.

All SD pairs in  $S_{i,j}$  must pass through links  $(i, 1) \rightarrow (R, 0)$  and  $(R, 0) \rightarrow (j, 1)$ . First, let us consider link  $(i, 1) \rightarrow (R, 0)$ . Let all SD pairs with source nodes in  $(i, 1)$  be  $All_{(i,1) \rightarrow (R,0)} = \bigcup_{j \neq i} S_{i,j}$ . All SD pairs in  $All_{(i,1) \rightarrow (R,0)}$  must go through link  $(i, 1) \rightarrow (R, 0)$ . Hence,  $L(All_{(i,1) \rightarrow (R,0)}) \leq X$ . From Lemma 4,  $L(All_{(i,1) \rightarrow (R,0)} - \bigcup_{x \neq i} LX_{i,x}^{\text{src}}) \leq X - \sum_{x \neq i} E_{i,x}$ . Since  $S_{i,j} - LX_{i,j}^{\text{src}} \subseteq All_{(i,1) \rightarrow (R,0)} - \bigcup_{x \neq i} LX_{i,x}^{\text{src}}$ , we have  $L(S_{i,j} - LX_{i,j}^{\text{src}}) \leq L(All_{(i,1) \rightarrow (R,0)} - \bigcup_{x \neq i} LX_{i,x}^{\text{src}}) \leq X - \sum_{x \neq i} E_{i,x}$ . Hence, applying Lemma 4,

$$L(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \leq X - \sum_{x \neq i} E_{i,x} - F_{i,j}.$$

Using the similar logic, by considering link  $(R, 0) \rightarrow (j, 1)$ , we can obtain

$$L(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \leq X - E_{i,j} - \sum_{x \neq j} F_{x,j}.$$

Combining these two inequalities, we obtain

$$L(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \leq X - (\sum_{x \neq i} E_{i,x} + F_{i,j} + E_{i,j} + \sum_{x \neq j} F_{x,j})/2.$$

Each source or destination node in  $S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}$  can have no more than  $X$  SD pairs in the set (otherwise, these SD pairs would be included in either  $LX_{i,j}^{\text{src}}$  or  $LX_{i,j}^{\text{dst}}$ ). Hence,  $LS(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \leq X$ . From Lemma 5,  $|S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}| \leq L(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \times LS(S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \leq (X - (\sum_{x \neq i} E_{i,x} + F_{i,j} + E_{i,j} + \sum_{x \neq j} F_{x,j})/2) \times X$ .

Hence,  $|\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} (S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}})| \leq \sum_{i=0}^{k-1} \sum_{j \neq i} |S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}| \leq \sum_{i=0}^{k-1} \sum_{j \neq i} X \times (X - (\sum_{x \neq i} E_{i,x} + F_{i,j} + E_{i,j} + \sum_{x \neq j} F_{x,j})/2) = k(k-1)X^2 - \frac{X}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} (\sum_{x \neq i} E_{i,x} + E_{i,j}) - \frac{X}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} (\sum_{x \neq j} F_{x,j} + F_{i,j})$ .

The values of  $j$  and  $x$  are in the range  $0..k-1$  and are not equal to  $i$ . Thus,  $\sum_{j \neq i} \sum_{x \neq i} E_{i,x} = (k-1) \sum_{j \neq i} E_{i,j}$ . This can be seen by expanding the summation form. Each term  $E_{i,j}$ ,  $j \neq i$ , happens  $k-1$  times in the summation. Hence,  $\sum_{j \neq i} (\sum_{x \neq i} E_{i,x} + E_{i,j}) = k \sum_{j \neq i} E_{i,j}$ . Similarly,  $\sum_{j \neq i} (\sum_{x \neq j} F_{x,j} + F_{i,j}) = k \sum_{j \neq i} F_{i,j}$ . Hence,  $|\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} (S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}})| \leq k(k-1)X^2 - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j}$ .

Since each switch connects to  $\frac{m}{2}$  processing nodes,  $LX_{i,j}^{\text{src}} \leq E_{i,j} \times \frac{m}{2}$  and  $LX_{i,j}^{\text{dst}} \leq F_{i,j} \times \frac{m}{2}$ . Hence,  $|S| = |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} S_{i,j}| \leq |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} ((S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}}) \cup LX_{i,j}^{\text{src}} \cup LX_{i,j}^{\text{dst}})| \leq |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} (S_{i,j} - LX_{i,j}^{\text{src}} - LX_{i,j}^{\text{dst}})| + |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} LX_{i,j}^{\text{src}}| + |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} LX_{i,j}^{\text{dst}}| \leq k(k-1)X^2 - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j} + \frac{m}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} + \frac{m}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j} = k(k-1)X^2 - (\frac{kX}{2} - \frac{m}{2}) (\sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} + \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j})$ .

When  $X \geq \frac{m}{k}$ ,  $\frac{kX}{2} \geq \frac{m}{2}$ . Thus,  $|S| \leq k(k-1)X^2$ .  $\square$

Let us denote by  $T(X)$  the maximum number of SD pairs routed through SEFT2( $m, k$ ) when the largest of the maximum disjoint sizes of the links in SEFT2( $m, k$ ) is  $X$ . Obviously, when  $X > Y$  and  $T(Y)$  is less than the total number of SD pairs that can be routed,  $T(X) > T(Y)$  regardless of the relation among  $X$ ,  $m$ , and  $k$ . Lemma 8 states that when  $X \geq \frac{m}{k}$ ,  $T(X) \leq k(k-1)X^2$ . Hence, when  $X < \frac{m}{k}$ ,  $T(X) < T(\frac{m}{k}) \leq k(k-1)(\frac{m}{k})^2$ .

**Theorem 2:** Let  $r$  be a single-path routing algorithm on  $\text{EFT2}(m, k)$ . If  $k \geq \sqrt{2m}$ ,  $\text{PERF}(r) \geq \sqrt{\frac{m}{2}}$ .

*Proof:* Regardless of the single-path routing algorithm  $r$  used,  $k(k-1)(\frac{m}{2})^2$  SD pairs must be routed through top-level switches. Since there are  $\frac{m}{2}$  top-level switches in  $\text{EFT2}(m, k)$ , at least one top-level switch must carry  $\frac{k(k-1)(\frac{m}{2})^2}{\frac{m}{2}} = k(k-1)\frac{m}{2}$  SD pairs. Consider the  $\text{SEFT2}(m, k)$  formed by this particular top-level switch (with  $k(k-1)\frac{m}{2}$  SD pairs passing through) with all level 1 switches and all processing nodes. Let the maximum disjoint size of the links connecting to this switch be  $X$ . Under the assumption  $k \geq \sqrt{2m}$ , if  $X < \frac{m}{k} \leq \frac{m}{\sqrt{2m}} = \sqrt{\frac{m}{2}}$ ,  $T(X) < k(k-1)(\frac{m}{k})^2 \leq k(k-1)\frac{m}{2}$ . Since there are  $k(k-1)\frac{m}{2}$  SD pairs that must be routed through the switch ( $T(X) \geq k(k-1)\frac{m}{2}$ ),  $X < \sqrt{\frac{m}{2}}$  cannot be true. Thus,  $X \geq \sqrt{\frac{m}{2}}$ . Since  $\frac{m}{2} \geq \sqrt{\frac{m}{2}}$ , from Lemma 7,  $\text{PERF}(r) \geq \sqrt{\frac{m}{2}}$ .  $\square$

**Theorem 3:** Let  $r$  be a single-path routing algorithm for  $\text{FT}(m, 2)$ ,  $m \geq 2$ ,  $\text{PERF}(r) \geq \sqrt{\frac{m}{2}}$ .

*Proof:*  $\text{FT}(m, 2)$  is equivalent to  $\text{EFT2}(m, m)$ . Since in  $\text{FT}(m, 2)$ ,  $m \geq 2$  and  $m \geq \sqrt{2m}$ , from Theorem 2,  $\text{PERF}(r) \geq \sqrt{\frac{m}{2}}$ .  $\square$

**Theorem 4:** Let  $r$  be a single-path routing algorithm for  $\text{FT}(m, 3)$ ,  $\text{PERF}(r) \geq \frac{m}{2}$ .

*Proof:*  $\text{FT}(m, 3)$  is composed of top-level switches ( $m$ -port) with  $m$   $\text{SUBFT}(m, 2)$ 's. Let us consider the maximum disjoint sizes on the links that connect the  $\text{SUBFT}(m, 2)$ 's with the root-level switches (Level 0 links). By treating each  $\text{SUBFT}(m, 2)$  as one  $2(\frac{m}{2})^2$ -port switch,  $\text{FT}(m, 3)$  is approximated as  $\text{EFT2}(2(\frac{m}{2})^2, m)$ . The top-level links in  $\text{EFT2}(2(\frac{m}{2})^2, m)$  correspond to the top-level links in  $\text{FT}(m, 3)$ . For any routing algorithm  $r$  on  $\text{FT}(m, 3)$ , there is a routing algorithm  $r'$  on  $\text{EFT2}(2(\frac{m}{2})^2, m)$  such that the corresponding top-level links are used exactly the same. Following the proof of Theorem 2, since  $m \geq \sqrt{2 \times 2(\frac{m}{2})^2}$ , for any  $r'$  on  $\text{EFT2}(2(\frac{m}{2})^2, m)$ , the largest of the maximum disjoint size of the level 0 links is at least  $\sqrt{\frac{2 \times 2(\frac{m}{2})^2}{2}} = \frac{m}{2}$ . Hence, for the any routing  $r$  on  $\text{FT}(m, 3)$ , there exists a link carrying at least  $\frac{m}{2}$  node disjoint SD pairs. Let this set of node disjoint SD pairs be  $A$ . Consider the traffic matrix  $TM$ , where  $tm_{i,j} = 1$  for all  $(i, j) \in A$  and all other entries are 0. Clearly,  $\text{MLOAD}(r, TM) \geq \frac{m}{2}$ . Since  $A$  is node disjoint, each node sends and receives at most 1 unit of traffic and  $\text{baseload}(TM) = \text{OPTU}(TM) = 1$  (Corollary 1). Thus,

$$\text{PERF}(r) \geq \frac{\text{MLOAD}(r, TM)}{\text{OPTU}(TM)} = \frac{m}{2}.$$

$\square$

**Theorem 5:** Let  $r$  be a single-path routing algorithm for  $\text{FT}(m, n)$ ,  $\text{PERF}(r) \geq (\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}$ .

*Proof:* Let us consider the maximum disjoint sizes on links connecting to up-link ports of  $\text{SUBFT}(m, i)$ 's,  $1 \leq i \leq n-1$ , in  $\text{FT}(m, n)$ . From Property 1 and Property 2 of  $\text{FT}(m, n)$ , the connectivity in  $\text{FT}(m, n)$  can be partitioned into two levels (with respect to such links): the lower-level connectivity provided by  $\text{SUBFT}(m, i)$ 's and the upper-level connectivity provided by the upper-level switches

for  $\text{SUBFT}(m, i)$ 's. The connectivity in  $\text{SUBFT}(m, i)$  can be approximated as a  $2(\frac{m}{2})^i$ -port switch, and the upper-level switches that connect the up-link ports with the same port number in each of the  $\text{SUBFT}(m, i)$  (Property 2), which approximates a  $m(\frac{m}{2})^{n-1-i}$ -port switch. Consider the case when  $i = \lfloor \frac{2(n-1)}{3} \rfloor$ , the topology can be approximated by  $\text{EFT2}(2(\frac{m}{2})^{\lfloor \frac{2(n-1)}{3} \rfloor}, m(\frac{m}{2})^{\lceil \frac{n-1}{3} \rceil})$ . Following the same logic as the proof of Theorem 4, for any  $r$  on  $\text{FT}(m, n)$ , there exists a link carrying at least  $\sqrt{2(\frac{m}{2})^{\lfloor \frac{2(n-1)}{3} \rfloor}}/2 = (\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}$  node disjoint pairs and

$$\text{PERF}(r) \geq \left(\frac{m}{2}\right)^{\lfloor \frac{n-1}{3} \rfloor}.$$

$\square$

## V. OPTIMAL SINGLE-PATH OBLIVIOUS ROUTING FOR $\text{FT}(m, 2)$ AND $\text{FT}(m, 3)$

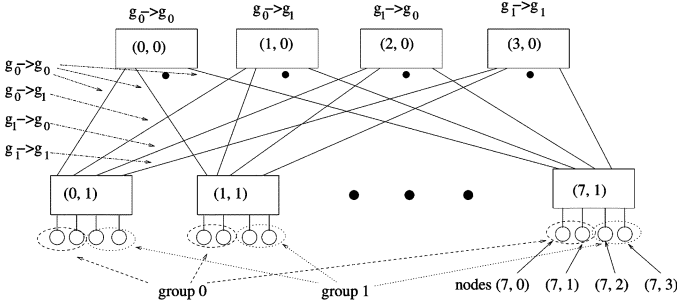
Most practical fat-trees have no more than three levels of switches since such topologies can already support thousands of processing nodes. For example,  $\text{FT}(24, 3)$  supports 3456 process nodes,  $\text{FT}(32, 3)$  supports 8192 processing nodes, and  $\text{FT}(48, 3)$  can support 27648 processing nodes. Hence, developing routing schemes for  $\text{FT}(m, 2)$  and  $\text{FT}(m, 3)$  bears most practical significance. Moreover, the development of these algorithms also has theoretical significance by making the lower bounds of the oblivious performance ratio for  $\text{FT}(m, 2)$  and  $\text{FT}(m, 3)$  (Theorems 3 and 4) tight bounds. The proposed optimal single-path oblivious routing schemes are based on the following lemma.

**Lemma 9:** If a single-path routing scheme  $r$  routes SD pairs such that the SD pairs in each of the links in  $\text{FT}(m, n)$  are either from at most  $X$  sources or toward at most  $X$  destinations, then  $\text{PERF}(r) \leq X$ .

*Proof:* As discussed earlier, for any traffic demand  $TM$ , on  $\text{FT}(m, n)$ ,  $\text{OPTU}(TM) \geq \text{baseload}(TM)$ . Since each link carries traffic either from at most  $X$  sources or toward  $X$  destinations, the load of the link is no more than  $X \times \text{baseload}(TM)$ . Hence,  $\text{PERF}(r, TM) \leq \frac{X \times \text{baseload}(TM)}{\text{baseload}(TM)} = X$ . Since this applies for any traffic demand  $TM$ ,  $\text{PERF}(r) \leq X$ .  $\square$

Existing routing schemes for the fat-tree topology [9], [15] try to balance the link load by spreading the traffic among different links. However, all of these schemes spread the traffic with a "locally optimal" heuristic: They make sure that the traffic from one node to all other nodes is spread out uniformly among all possible links. However, such a locally optimal heuristic is not globally optimal in the sense that a link can potentially carry traffic from many sources and to many destinations, which can potentially make the link a hot-spot for some particular traffic patterns. The proposed optimal oblivious routing schemes achieve global optimality by routing traffic either from at most  $\sqrt{\frac{m}{2}}$  sources or to at most  $\sqrt{\frac{m}{2}}$  destinations on each link in  $\text{FT}(m, 2)$ , and traffic either from at most  $\frac{m}{2}$  sources or to at most  $\frac{m}{2}$  destinations on each link in  $\text{FT}(m, 3)$ . From Lemma 9, this ensures that the performance ratios of our schemes are at most  $\sqrt{\frac{m}{2}}$  for  $\text{FT}(m, 2)$  and  $\frac{m}{2}$  for  $\text{FT}(m, 3)$ , which are optimal (Theorems 3 and 4).



Fig. 8. Optimal oblivious routing for  $FT(8, 2)$ .

Route from node  $(s_0, s_1)$  to node  $(d_0, d_1)$   
 Let  $m_2 = \frac{m}{2}$ ;  
 Let  $Z_s = \lceil \sqrt{m_2} \rceil$ ,  $Z_d = \lfloor \sqrt{m_2} \rfloor$ ;  
 Let  $N_s = \lceil \frac{m_2}{Z_s} \rceil$ ,  $N_d = \lceil \frac{m_2}{Z_d} \rceil$ ;  
 if  $(s_0 == d_0)$   
   use route:  $node(s_0, s_1) \rightarrow switch(s_0, 1) \rightarrow node(d_0, d_1)$   
 if  $(s_0 \neq d_0)$   
   use route:  $node(s_0, s_1) \rightarrow switch(s_0, 1)$   
      $\rightarrow switch(s_1/N_s * N_d + d_1/N_d, 0)$   
      $\rightarrow switch(d_0, 1) \rightarrow node(d_0, d_1)$

Fig. 9. Optimal oblivious routing for  $FT(m, 2)$ .

### A. Optimal Oblivious Routing for $FT(m, 2)$

To describe the oblivious routing algorithm, we will give a nonrecursive description of  $FT(m, 2)$ .  $FT(m, 2)$  contains  $\frac{3m}{2}$  switches and supports  $m^2/2$  processing nodes. In this topology,  $\frac{m}{2}$  switches are in the level 0 and  $m$  switches are in level 1. The  $\frac{m}{2}$  top-level switches are labeled switches  $(0, 0), (1, 0), \dots, (\frac{m}{2} - 1, 0)$ . The  $m$  level 1 switches are labeled switches  $(0, 1), (1, 1), \dots, (m - 1, 1)$ . Each level 1 switch  $(i, 1)$ ,  $0 \leq i \leq m - 1$ , is connected with  $\frac{m}{2}$  processing nodes numbered as  $(i, 0), (i, 1), \dots, (i, \frac{m}{2} - 1)$ . Notice that the process nodes and switches are numbered independently. There is a link between switch  $(i, 0)$ ,  $0 \leq i \leq \frac{m}{2} - 1$ , and switch  $(j, 1)$ ,  $0 \leq j \leq m - 1$ . For  $0 \leq i \leq m - 1$ , there is a link between processing node  $(i, x)$ ,  $0 \leq x \leq \frac{m}{2} - 1$ , and switch  $(i, 1)$ . Fig. 8 depicts the  $FT(8, 2)$  topology as well as the switch and processing node labeling.

To ease exposition, let us assume that  $\sqrt{\frac{m}{2}}$  is an integer. Our algorithm *OSRM2*, described in Fig. 9, can also handle the case when  $\sqrt{\frac{m}{2}}$  is not an integer. In  $FT(m, 2)$ , each level 1 link connects to one processing node and can only carry traffic to and from one node (Property 5). Thus, we only need to make sure that the traffic on each level 0 link has no more than  $\sqrt{\frac{m}{2}}$  sources or destinations.

Let  $Z = \sqrt{\frac{m}{2}}$ . *OSRM2* partitions the  $\frac{m}{2} = Z^2$  processing nodes in each bottom-level switch into  $Z$  groups, each group having  $Z$  nodes. More specifically, the  $\frac{m}{2}$  processing nodes connected to switch  $(i, 1)$ ,  $0 \leq i \leq m - 1$ , are partitioned into  $Z = \sqrt{\frac{m}{2}}$  groups. Group  $j$ ,  $0 \leq j \leq Z - 1$ , includes nodes  $(i, j * Z), (i, j * Z + 1), \dots, (i, j * Z + Z - 1)$ . Let us denote  $g_i \rightarrow g_j$  as SD pairs from nodes in group  $i$  in the one switch to nodes in group  $j$  in all other switches. Fig. 8 shows how the routing algorithm works on  $FT(8, 2)$ . In  $FT(8, 2)$ ,  $\frac{m}{2} = 4$  and  $\sqrt{\frac{m}{2}} = 2$ . As shown in the figure, the  $\frac{m}{2} = 4$  nodes attached to each lower-level switch are partitioned into two groups: group

0 and group 1, with each group having two nodes. The SD pairs are scheduled such that  $g_0 \rightarrow g_0$  goes through switch  $(0, 0)$ ,  $g_0 \rightarrow g_1$  goes through switch  $(1, 0)$ ,  $g_1 \rightarrow g_0$  goes through switch  $(2, 0)$ , and  $g_1 \rightarrow g_1$  goes through switch  $(3, 0)$ . Since each upper-level switch carries SD pairs from nodes in one group to nodes in another group, each up link (to a top-level switch) carries SD pairs with exactly  $\sqrt{\frac{m}{2}}$  sources, and each down link (to a lower-level switch) carries SD pairs with exactly  $\sqrt{\frac{m}{2}}$  destinations.

For a general  $FT(m, 2)$ , the SD pairs are scheduled as follows. The up-link  $(i, 1) \rightarrow (j, 0)$ ,  $0 \leq i \leq m - 1$  and  $0 \leq j \leq \frac{m}{2} - 1$ , carries SD pairs with source nodes in group  $j/Z$  in switch  $(i, 1)$  and destination nodes in group  $j \bmod Z$  in all other switches. *OSRM2* is shown in Fig. 9. When  $\sqrt{\frac{m}{2}}$  is an integer, the algorithm works exactly as just described. When  $\sqrt{\frac{m}{2}}$  is not an integer, the algorithm partitions the  $\frac{m}{2}$  sources attached with each of the level 1 switches into  $Z_s = \lceil \sqrt{\frac{m}{2}} \rceil$  groups and the  $\frac{m}{2}$  destinations into  $Z_d = \lfloor \sqrt{\frac{m}{2}} \rfloor$  groups. It then uses the same logic as the cases when  $\sqrt{\frac{m}{2}}$  is an integer to schedule the SD pairs.

**Theorem 6:** When  $\sqrt{\frac{m}{2}}$  is an integer,  $\text{PERF}(\text{OSRM2}) = \sqrt{\frac{m}{2}}$ .

**Proof:** As discussed earlier, using *OSRM2*, each link carries traffic either from  $\sqrt{\frac{m}{2}}$  sources or to  $\sqrt{\frac{m}{2}}$  destinations. From Lemma 9,  $\text{PERF}(\text{OSRM2}) \leq \sqrt{\frac{m}{2}}$ . From Theorem 3,  $\text{PERF}(\text{OSRM2}) \geq \sqrt{\frac{m}{2}}$ . Hence,  $\text{PERF}(\text{OSRM2}) = \sqrt{\frac{m}{2}}$  and *OSRM2* is an optimal oblivious routing algorithm for  $FT(m, 2)$  when  $\sqrt{\frac{m}{2}}$  is an integer.  $\square$

### B. Optimal Oblivious Routing for $FT(m, 3)$

We will now consider  $FT(m, 3)$ .  $FT(m, 3)$  contains three levels of switches, with the top level having  $\text{nu}(m, 2) = \frac{m}{2} \times \frac{m}{2}$  switches and each of the other levels having  $m \times \frac{m}{2}$  switches ( $m$  SUBFT( $m, 2$ )'s, each SUBFT( $m, 2$ ) having  $\frac{m}{2}$  switches at each level). We label the switches by  $((i_0, i_1), \text{level})$ . The top-level switches are labeled as  $((i_0, i_1), 0)$ ,  $0 \leq i_0 \leq \frac{m}{2} - 1$  and  $0 \leq i_1 \leq \frac{m}{2} - 1$ . The level 1 switches are labeled as  $((i_0, i_1), 1)$ ,  $0 \leq i_0 \leq m - 1$ , and  $0 \leq i_1 \leq \frac{m}{2} - 1$ . The level 2 switches are labeled as  $((i_0, i_1), 2)$ ,  $0 \leq i_0 \leq m - 1$  and  $0 \leq i_1 \leq \frac{m}{2} - 1$ . Notice that in the switch labeling, for levels 1 and 2,  $i_0$  identifies the columns corresponding to the  $i_0$ th SUBFT( $m, 2$ ) and  $i_1$  identifies the column corresponding to the  $i_1$ th SUBFT( $m, 1$ ) within the  $i_0$ th SUBFT( $m, 2$ ). A  $FT(m, 3)$  has  $m \times \frac{m}{2} \times \frac{m}{2}$  processing nodes, which are labeled as  $(p_0, p_1, p_2)$ ,  $0 \leq p_0 \leq m - 1$ ,  $0 \leq p_1 \leq \frac{m}{2} - 1$ , and  $0 \leq p_2 \leq \frac{m}{2} - 1$ . A processing node  $(p_0, p_1, p_2)$  is attached to switch  $((p_0, p_1), 2)$ ,  $0 \leq p_0 \leq m - 1$ ,  $0 \leq p_1 \leq \frac{m}{2} - 1$ , and  $0 \leq p_2 \leq \frac{m}{2} - 1$ . A level 2 switch  $((i_0, i_1), 2)$ ,  $0 \leq i_0 \leq m - 1$  and  $0 \leq i_1 \leq \frac{m}{2} - 1$ , has a link to each of the level 1 switches  $((i_0, X), 1)$ ,  $0 \leq X \leq \frac{m}{2} - 1$ . A level 1 switch  $((i_0, i_1), 1)$ ,  $0 \leq i_0 \leq m - 1$  and  $0 \leq i_1 \leq \frac{m}{2} - 1$ , has a link to each of the level 0 switches  $((i_1, X), 0)$ ,  $0 \leq X \leq \frac{m}{2} - 1$ .

Like in the  $FT(m, 2)$  case, our optimal routing algorithm ensures that the SD pairs on each link are either from at most  $\frac{m}{2}$  sources or toward at most  $\frac{m}{2}$  destinations. From Property 5 in Section II-B, each level 1 or level 2 link in  $FT(m, 3)$  carries traffic either from no more than  $\frac{m}{2}$  sources

```

Route from node  $(s_0, s_1, s_2)$  to  $(d_0, d_1, d_2)$ :
if  $(s_0 == d_0 \text{ and } s_1 == d_1)$ 
  /* within one  $SUBFT(m, 2)$  */
  /* routing won't affect the oblivious ratio */
  Use route:  $node(s_0, s_1, s_2) \rightarrow switch((s_0, s_1), 2)$ 
                $\rightarrow node(d_0, d_1, d_2)$ 
else if  $(s_0 == d_0)$ 
  /* within one  $SUBFT(m, 2)$  */
  /* routing won't affect the oblivious ratio */
  Use route:  $node(s_0, s_1, s_2) \rightarrow switch((s_0, s_1), 2)$ 
                $\rightarrow switch((s_0, s_2), 1)$ 
                $\rightarrow switch((s_0, d_1), 2)$ 
                $\rightarrow node(d_0, d_1, d_2)$ 
else
  /* must be careful about links to/from level 0 switches */
  Use route:  $node(s_0, s_1, s_2) \rightarrow switch((s_0, s_1), 2)$ 
                $\rightarrow switch((s_0, s_2), 1)$ 
                $\rightarrow switch((s_2, d_2), 0)$ 
                $\rightarrow switch((d_0, s_2), 1)$ 
                $\rightarrow switch((d_0, d_1), 2)$ 
                $\rightarrow node(d_0, d_1, d_2)$ 

```

Fig. 10. Optimal oblivious single routing for  $FT(m, 3)$ .

or to no more than  $\frac{m}{2}$  destinations. Hence, routing within each  $SUBFT(m, 2)$  does not affect the performance oblivious ratio. Hence, we only need to focus on level 0 links. The idea is similar to that in *OSRM2*: The routing algorithm ensures that each up link out of the sub-fat-tree  $SUBFT(m, 2)$  carries traffic from  $\frac{m}{2}$  sources and each down link to a  $SUBFT(m, 2)$  carries traffic to  $\frac{m}{2}$  destinations. Basically, we can treat each  $SUBFT(m, 2)$  as if it is a  $2(\frac{m}{2})^2$ -port switch that connects to  $(\frac{m}{2})^2$  processing nodes and has  $(\frac{m}{2})^2$  up links. The routing algorithm partitions the  $\frac{2(\frac{m}{2})^2}{2} = Z^2$  processing nodes in a  $SUBFT(m, 2)$  into  $Z = \frac{m}{2}$  groups, each group having  $Z = \frac{m}{2}$  nodes. Node  $(p_0, p_1, p_2)$  is in group  $p_2$  of the  $p_0$ th  $SUBFT(m, 2)$ . The routing for links between  $SUBFT(m, 2)$  and the top-level switch is similar to that for links between level 1 switches to level 0 switches in  $FT(m, 2)$ : The up link  $((i_0, 0), 1) \rightarrow ((0, 0), 0)$  carries traffic from group 0 processing nodes (in the  $i_0$ -th  $SUBFT(m, 2)$ ) to group 0 processing nodes in other  $SUBFT(m, 2)$ 's,  $((i_0, 0), 1) \rightarrow ((0, 1), 0)$  carries traffic from group 0 processing nodes to group 1 processing nodes in other  $SUBFT(m, 2)$ 's, and so on. The detailed routing algorithm, called *OSRM3*, is shown in Fig. 10

**Theorem 7:**  $PERF(OSRM3) = \frac{m}{2}$ .

**Proof:** From above discussion, using *OSRM3*, the SD pairs on each link have either at most  $\frac{m}{2}$  source nodes or at most  $\frac{m}{2}$  destination nodes. From Lemma 9,  $PERF(OSRM3) \leq \frac{m}{2}$ . From Theorem 4, a performance oblivious ratio of  $\frac{m}{2}$  is the low bound for any single-path routing scheme on  $FT(m, 3)$ . Hence,  $PERF(OSRM3) = \frac{m}{2}$  and *OSRM3* is an optimal oblivious routing algorithm for  $FT(m, 3)$ .  $\square$

The proposed oblivious routing algorithms, *OSRM2*, *OSRM3*, and *OMRMN*, are quite simple. The paths between two nodes can basically be enumerated and can thus be established with either a centralized or a distributed scheme. Detailed realization of the routing schemes is beyond the scope of this paper. Interested readers can refer to other literature, such as [10], for more details.

TABLE I  
OBLIVIOUS PERFORMANCE RATIOS OF DIFFERENT ROUTING ALGORITHMS

	$FT(m, 2)$	$FT(m, 3)$
<i>OMRMN</i>	1	1
<i>OSRM2</i>	$\sqrt{\frac{m}{2}}$	-
<i>OSRM3</i>	-	$\frac{m}{2}$
<i>MLID</i>	$\frac{m}{2}$	$m - 1$
<i>WSR</i>	$\frac{m}{2}$	$m - 1$

In fat-trees with uncertain traffic demands, the performance of (unrestricted) multipath routing is much better than that of single-path routing. These results argue strongly that in a large fat-tree-based system area network, the unrestricted multipath routing should be used to alleviate the network contention problem. Moreover, these results raise questions in the current system area networks that only support a limited form of multipath routing. One example is the InfiniBand, where only a limited number of paths (128) between any two processing nodes are supported. With such a restriction, it is difficult to achieve optimal load balancing with multipath routing in fat-trees.

## VI. PERFORMANCE STUDY

We compare the performance of several known single-path routing algorithms, including the Multiple LID algorithm (*MLID*) in [16] and the widest shortest routing (*WSR*) algorithm. *WSR* was designed to achieve load balancing in the Internet environment. It works as follows. We first generate a traffic matrix where each SD pair has one unit of traffic. All links in the network are initialized with the same weight. The algorithm then computes routes for each SD pair in the following order:  $(0, 1), (0, 2), \dots, (0, N - 1), (1, 0), \dots, (1, N - 1), \dots, (N - 1, 1), (N - 1, 1), \dots, (N - 1, N - 2)$ . Every time a route is computed, the weight of each of the links along the route is increased by 1. When computing the route for each SD pair, the path with the smallest accumulated weight is selected. Note that the “shortest” heuristic enforces that only the shortest paths between two nodes are selected, and the “widest” heuristic spreads traffic from the same source among all links in the fat-tree. A recently proposed scheme [10] yields exactly the same routes as *WSR*.

Table I shows the oblivious performance ratios for different routing algorithms. The worst case oblivious performance ratios for *MLID* and *WSR* are obtained by analyzing the paths computed by the algorithms. This table shows that 1) our optimal single-path oblivious routing algorithms provide better performance guarantees than other existing single-path routing algorithms and 2) multipath routing (*OMRMN*) is significantly better than single-path routing.

We design experiments to investigate the performance of single-path routing algorithms with practical traffic patterns. In particular, our optimal oblivious routing algorithms (*OSRM2* and *OSRM3*) group SD pairs in a particular way so as to guarantee the best performance in the worst-case condition. This, however, might yield lower performance on typical uniform traffic demands (average-case performance). In fact, both *MLID* and *WSR* fully spread traffic among all links in the fat-tree topology and should perform well (among single-path

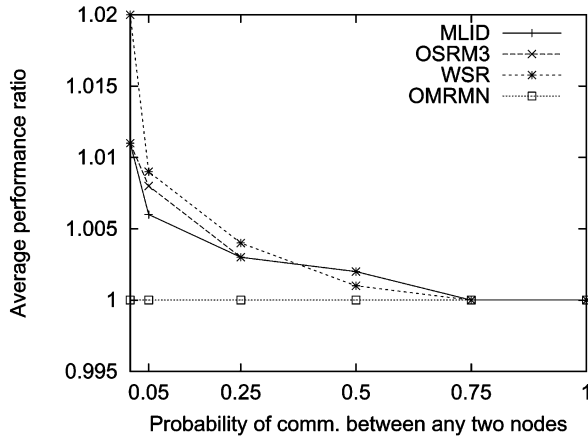


Fig. 11. Random uniform traffic on FT(16, 3).

routing schemes) for typical traffic demands. In this section, we will use the average case performance in the comparison and show that the proposed optimal oblivious algorithms not only provide the optimal worst case performance guarantees but also often perform better in average cases. We will report results for FT(32, 2), FT(8, 3), and FT(16, 3) that support 512, 128, and 1024 processing nodes, respectively.

We will show the results for four types of traffic patterns: random uniform traffic, regular traffic, clustered traffic, and hot-spot traffic. In a random uniform traffic demand, each entry in the traffic matrix has an equal probability to send one unit of traffic (or not send any traffic). The regular traffic consists of five different patterns on all nodes in the system: ring, 2-dimensional mesh, 3-dimensional mesh, hypercube, and binary tree. These communication patterns are frequently used in high-performance applications. In a clustered traffic demand, the processing nodes are partitioned into groups of the same size (size = 2, 4, 8, 16, 32, 64, 128 nodes). Each processing node in the system is in one group. The members in each group are randomly selected from all processing nodes. One unit of data is communicated between each pair of nodes in one group (all-to-all communication pattern within each group). This pattern represents the cases when the nodes in the system are allocated to different jobs, with each job having the all-to-all communication. The hot-spot traffic pattern is created as follows. The system contains a number of hot-spots, which are a group of processing nodes performing the all-to-all communication (one unit of data between each pair in a hot-spot). The rest of the system is quiet. The number of hot-spots and the size of the hot-spots are parameters of this traffic pattern. For each data point, we produce 32 random instances and report the average performance ratio for the 32 instances. For regular traffic patterns, the node assignment is randomly generated for each instance. For example, in different ring patterns, the nodes can be in different positions of the ring.

The results for the random uniform traffic on FT(16, 3) are depicted in Fig. 11. The results on FT(32, 2) and FT(8, 3) are very similar. For the random traffic with different probability values, all of the single-path routing algorithms achieve a similar performance, and their performance ratios are very close to 1. This indicates that single-path routing is effective in dealing

TABLE II  
AVERAGE PERFORMANCE RATIO FOR REGULAR TRAFFIC ON THE WHOLE  
FT(32, 2)

Traffic Pattern	Average performance ratio	
	MLID/WSR	OSRM2
Ring	3.47	2.97
2D Mesh	1.88	1.74
3D Mesh	2.34	2.14
Hypercube	2.03	1.90
Binary Tree	2.37	2.20

TABLE III  
AVERAGE PERFORMANCE RATIO FOR REGULAR TRAFFIC ON THE WHOLE  
FT(8, 3) AND FT(16, 3)

Topology	Traffic Pattern	Average performance ratio	
		MLID/WSR	OSRM3
FT(8, 3)	Ring	2.84	2.80
	2D Mesh	1.64	1.63
	3D Mesh	2.04	1.99
	Hypercube	1.90	1.90
	Binary Tree	2.07	2.03
FT(16, 3)	Ring	3.78	3.78
	2D Mesh	2.83	2.83
	3D Mesh	2.48	2.43
	Hypercube	2.11	2.10
	Binary Tree	2.69	2.67

with such demands. All of the routing schemes considered are able to evenly distribute the load when all pairs are communicating. When the load is high (the probability of the communication between two nodes is larger than 0.75), all schemes have a performance ratio of 1. When the communication is more sparse, since *MLID*, *OSRM3*, and *WSR* are deterministic and demand-oblivious, they may not perform as good as the optimal routing scheme, which is typically demand-dependent. Hence, their performance ratios are slightly higher (when the network load is lower). Note again that the performance ratio is relative and is not directly related to the absolute network performance.

Tables II and III show the performance of different routing algorithms for regular traffic patterns. The regular traffic patterns are imposed on all the nodes in the system. The *MLID* and *WSR* always have exactly the same performance ratio for symmetrical traffics on the fat-tree topology even though paths between a pair of nodes are different under the two routing schemes. They give different performance ratios only when the traffic matrix is asymmetrical, like the random traffic in Fig. 11. Thus, for all results presented in the rest of this section where the traffic matrices are symmetrical, the results for *MLID* and *WSR* will be given together. As can be seen from Table II, *OSRM2* offers a fairly large performance improvement over *MLID/WSR* for the regular traffic on FT(32, 2), ranging from 6.8% for the hypercube pattern to 16.8% for the ring pattern. For FT(16, 3) and FT(8, 3), *OSRM3* has slightly better performance, but the difference is statistically insignificant. This shows that the proposed optimal oblivious routing schemes do not sacrifice the average case performance for these traffic patterns.

Fig. 12 shows the results for clustered traffic on FT(32, 2), and Fig. 13 shows the results for FT(8, 3). Results for FT(16, 3) are similar to those in FT(8, 3). As can be seen from the figures, the single-path routing algorithms are not effective in dealing with such traffic demands: The average performance

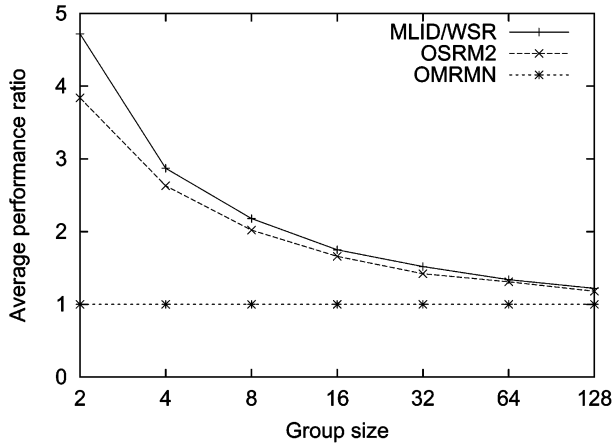


Fig. 12. Clustered traffic on FT(32, 2).

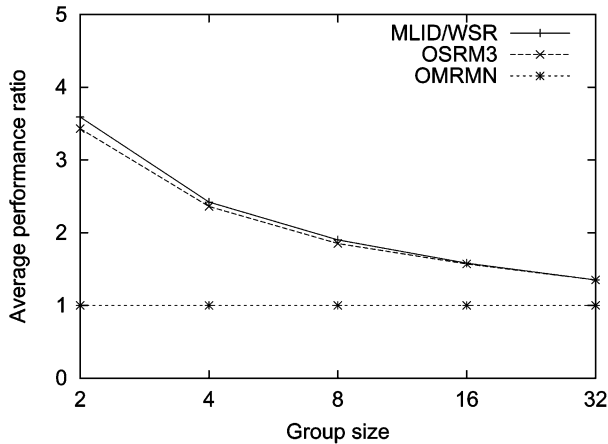


Fig. 13. Clustered traffic on FT(8, 3).

ratios for all single-path routing schemes are much larger than 1, especially when the group size is small. This indicates that with single-path routing, the network contention can be a problem with such traffic demands. The advantage of our optimal oblivious routing scheme is manifested in this experiment. *OSRM2* performs noticeably better than *MLID/WSR*. Notice that when the group size is equal to 2, the average performance ratio for *MLID/WSR* is larger than 4 on FT(32, 2). *OSRM2* guarantees that the performance ratio for any traffic pattern is no more than  $\sqrt{\frac{32}{2}} = 4$ . Our schemes improve the performance noticeably on FT(32, 2) but only slightly on FT(8, 3) and FT(16, 3).

Figs. 14 and 15 show the results for hot-spot traffic. In the experiments, there are four hot-spots in the system, with the size of the hot-spots varying. The performance ratios of all schemes are close to 1, which indicates that all schemes can handle this traffic pattern effectively. The performance of *OSRM3* is similar to other single-path routing schemes in FT(16, 3), while *OSRM2* is slightly (about 3% on average among all cases in the experiment) worse than *WSR/DLID* in FT(32, 2). This is the only case in all of our experiments that *OSRM2* performs worse on average. There exist traffic patterns that our proposed algorithms cannot handle as well as other algorithms. This is a common limitation of all deterministic routing schemes.

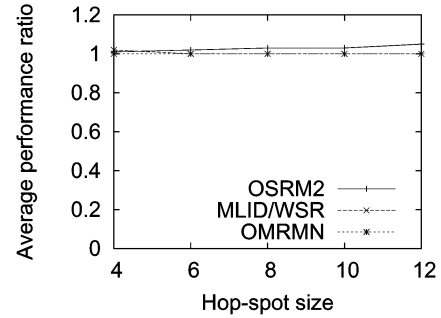


Fig. 14. Results for four hot-spots with different hot-spot sizes on FT(32, 2).

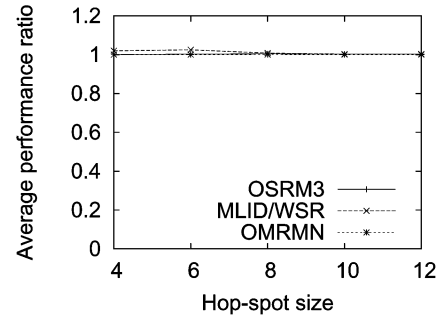


Fig. 15. Results for four hot-spots with different hot-spot sizes on FT(16, 3).

We have performed many other experiments with different traffic demands, including clustered traffic with random group sizes, clustered traffic with a random regular pattern in each group, and clustered traffic with random traffic in each group. We have also carried out experiments on other  $FT(m, 2)$ 's and  $FT(m, 3)$ 's. All of these experiments have similar results: Our optimal oblivious routing algorithms are either comparable to or better than *MLID* and *WSR* on average. Moreover, the improvement on  $FT(m, 2)$  is quite noticeable in most cases, as we showed in the figures and tables, while the improvement on  $FT(m, 3)$  is small. This is because *OSRM2* improves the worst-case performance ratio over *MLID/WSR* on  $FT(m, 2)$  by a factor of  $\frac{m}{\sqrt{\frac{m}{2}}} = \sqrt{\frac{m}{2}}$ , while *OSRM3* improves the worst-case performance ratio over *MLID/WSR* on  $FT(m, 3)$  by a factor of  $\frac{m-1}{\frac{m}{2}} \approx 2$ . The hot-spot traffic is the only case in all of our experiments that the proposed schemes perform slightly worse than other schemes on average. These results indicate that our optimal single-path oblivious routing algorithms can provide performance guarantees without sacrificing the average-case performance, and they often provide better performance in average cases.

## VII. RELATED WORK

The research most related to this work falls into three areas: the development of system area networks, routing on fat-trees, and oblivious routing. System area networks with off-the-shelf networking technology such as InfiniBand [16] and Myrinet [22] have become more common recently. The load balancing problems in such networks motivated this research. Most routing research for system area networks (see, for example, [5], [6], [19], [24], and [25]) has focused on developing techniques for computing and establishing routes. In [9] and [15],

routing algorithms were developed for fat-tree-based InfiniBand networks. We show that algorithms in [9] and [15] are not optimal oblivious routing schemes. Routing performance with various routing algorithms, such as randomized routing and adaptive routing, and various performance metrics on the fat-tree topology has also been studied [8], [17], [18]. However, we are unaware of any work studying the routing performance on fat-trees with deterministic routing when the traffic demand is uncertain and changing.

Oblivious routing has recently attracted much attention due to its effectiveness in guaranteeing routing performance under uncertain and changing traffic demands in the Internet environment [1], [2], [28]. The bounds of the competitive ratios of oblivious routing on the general directed and undirected topologies have been analyzed [3], [10]–[12], [23]. In [23], it was shown that for any undirected network, there exists an oblivious randomized routing algorithm that can achieve a polylogarithmic competitive ratio. The results in [3] show that for a directed network, the optimal competitive ratio of an oblivious routing algorithm is at least  $O(\sqrt{n})$ , where  $n$  is the number of nodes in the network. Works such as [10] and [11] considered the cases when demands are randomly chosen from a known distribution and showed that routing algorithms with polylogarithmic competitive ratios can be obtained for directed networks and that the competitive ratio cannot be significantly improved. These lower bounds for general networks are much higher than the lower bound for fat-trees. Various polynomial time algorithms for computing the optimal oblivious routing were also developed [1]–[4], [13]. These algorithms are still too computational-intensive to be applied to find the optimal oblivious routing scheme for large fat-trees. Moreover, these methods can only be used to compute optimal oblivious routing for unrestricted multipath routing and cannot be used to compute optimal deterministic oblivious single-path routing. In this paper, we consider deterministic oblivious routing in fat-trees. We show that in fat-trees, the competitive ratio for multipath oblivious routing is 1, which is much smaller than what lower bounds obtained for general networks. Moreover, optimal oblivious routing for fat-trees with unrestricted multipath routing is given (OMRMN) without running the high-complexity algorithms. We further give the bounds for the competitive ratios of single-path oblivious routing and develop optimal single-path oblivious routing schemes for FT( $m, 2$ ) and FT( $m, 3$ ). Oblivious routing on specific topologies has also been studied [7], [14], [27]. However, the fat-tree topology has not been investigated.

## VIII. CONCLUSION

We study oblivious routing in fat-tree-based system area networks with deterministic routing under the assumption that the traffic demand is uncertain and changing. We show that single-path routing cannot provide good performance guarantees, while unrestricted multipath routing is effective in balancing network loads in fat-trees. We develop optimal single-path oblivious routing schemes for FT( $m, 2$ ) and FT( $m, 3$ ) and demonstrate that these optimal oblivious routing schemes can not only provide the optimal worst-case performance guarantees but also offer better performance than

existing single-path routing schemes in average cases. These results may directly influence the design of systems with large-scale fat-tree-based networks such as large HPC clusters.

## APPENDIX

*Lemma 5:* Let  $A$  be a set of SD pairs.  $|A| \leq L(A) \times LS(A)$ .

*Proof:* We will prove this lemma by induction on  $L(A)$ .

*Base case:* when  $L(A) = 1$ ,  $|A| \leq LS(A)$ . Let  $B = \{(s_1, d_1)\}$  be one largest node disjoint subset of  $A$ . If  $A$  only contains  $(s_1, d_1)$ , the case is proven. Otherwise, there exists another SD pair  $(s_2, d_2)$  in  $A$ . Since  $L(A) = 1$ , either  $s_1 = s_2$  or  $d_1 = d_2$ . We will show that if  $s_1 = s_2$ ,  $A = S_{s_1}^A$ . Similar logic can be used to show that if  $d_1 = d_2$ ,  $A = D_{d_1}^A$ . In both case,  $|A| \leq L(A) \times LS(A) = LS(A)$ .

Let us now prove that if  $s_1 = s_2$ ,  $A = S_{s_1}^A$ . Assume that when  $A \neq S_{s_1}^A$ , there exists a SD pair  $(s, d)$  such that  $s \neq s_1$ . In this case, if  $d \neq d_1$ , then  $\{(s_1, d_1), (s, d)\}$  is a node disjoint subset of  $A$ ;  $d = d_1$  and  $\{(s_1, d_2), (s, d)\}$  is a node disjoint subset of  $A$ , otherwise. Hence,  $L(A) \geq 2$  (contradiction).

*Induction case:* Assume that  $|A| \leq L(A) \times LS(A)$  when  $L(A) \leq k$  (induction hypothesis). We will prove that  $|A| \leq L(A) \times LS(A)$  when  $L(A) = k + 1$ .

Let  $B = \{(s_1, d_1), (s_2, d_2), \dots, (s_{k+1}, d_{k+1})\}$  be a largest node disjoint subset of  $A$ . If  $\text{SRC}(A) = \{s_1, s_2, \dots, s_{k+1}\}$ ,  $|A| \leq (k + 1) \times LS(A)$  (since each source node can at most have  $LS(A)$  SD pairs in  $A$  and there are  $k + 1$  source nodes) and the theorem is proven.

If  $\text{SRC}(A) \supset \{s_1, s_2, \dots, s_{k+1}\}$ , there must exist a source node  $s \in \text{SRC}(A)$  such that  $s \neq s_i$ ,  $1 \leq i \leq k + 1$ . Let  $(s, d) \in A$ . We have  $d \in \text{DST}(B)$  (Otherwise,  $(s, d) \cup B$  is node disjoint and  $B$  is not a largest node disjoint subset). Without loss generality, let  $d = d_1$ . We have  $\{(s, d_1), (s_1, d_1)\} \subseteq D_{d_1}^A$ . Obviously  $L(D_{d_1}^A) = 1$ ,  $LS(D_{d_1}^A) \leq LS(A)$ , and  $LS(A - D_{d_1}^A) \leq LS(A)$ .

Next, we will show that  $L(A - D_{d_1}^A) = k$ . From Lemma 2,  $L(A - D_{d_1}^A) \geq L(A) - L(D_{d_1}^A) = k + 1 - 1 = k$ . Since  $L(A) = k + 1 \geq L(A - D_{d_1}^A)$ , to show that  $L(A - D_{d_1}^A) = k$ , we only need to show that  $L(A - D_{d_1}^A) \neq k + 1$ . We prove this by contradiction. Assume that  $L(A - D_{d_1}^A) = k + 1$ . Let  $C = \{(s'_1, d'_1), (s'_2, d'_2), \dots, (s'_{k+1}, d'_{k+1})\}$  be a largest node disjoint subset of  $A - D_{d_1}^A$ . We have  $s_1 \in \text{SRC}(C)$  (otherwise,  $C \cup \{(s_1, d_1)\}$  is a node disjoint subset of  $A$  and  $L(A) \geq k + 2$ ). Similarly,  $s \in \text{SRC}(C)$ . Let us assume that  $s_1 = s'_1$  and  $s = s'_2$ .  $d'_2$  must be in  $\text{DST}(B) - \{d_1\}$  (otherwise,  $(s, d'_2) \cup B$  is node disjoint and  $B$  is not the largest node disjoint subset). Similarly,  $d'_1$  must be in  $\text{DST}(B) - \{d_1\}$ . Let  $d_{k+1} = d'_1$  and  $d_2 = d'_2$ . We have  $s_2 \in \text{SRC}(C)$  (Otherwise,  $C - \{(s'_2, d'_2)\} \cup \{(s_2, d_2), (s, d_1)\}$  is a node disjoint subset of  $A$  and  $L(A) \geq k + 2$ ). This process (finding that a source node  $s_i$  in  $B$  belongs to  $\text{SRC}(C)$  and then finding that destination node  $d'$  such that  $(s_i, d') \in C$  belongs to  $\text{DST}(B)$ ) can be repeated. Once the process cannot continue, one can construct a node disjoint subset of  $A$  whose size is  $k + 2$ . Since there are a finite number of elements in  $B$  and  $C$ , this process will stop at some point (In the worst case, one of  $B$  or  $C$  runs out of elements). Thus,  $L(A) \geq k + 2$ , which contradicts the assumption

that  $L(A) = k + 1$ . Hence,  $L(A - D_{d_1}^A) = k$ . By the induction hypothesis

$$\begin{aligned} |A - D_{d_1}^A| &\leq L(A - D_{d_1}^A) \times LS(A - D_{d_1}^A) \leq k \times LS(A) \\ |D_{d_1}^A| &\leq L(D_{d_1}^A) \times LS(D_{d_1}^A) \leq LS(A). \end{aligned}$$

Hence,  $|A| = |A - D_{d_1}^A| + |D_{d_1}^A| \leq (k + 1) \times LS(A)$ .  $\square$

## REFERENCES

- [1] D. Applegate and E. Cohen, "Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs," in *Proc. ACM SIGCOMM*, 2003, pp. 313–324.
- [2] D. Applegate, L. Breslau, and E. Cohen, "Coping with network failures: Routing strategies for optimal demand oblivious restoration," in *Proc. ACM SIGMETRICS*, 2004, pp. 270–281.
- [3] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke, "Optimal oblivious routing in polynomial time," in *Proc. 35th ACM STOC*, 2003, pp. 383–388.
- [4] M. Bienkowski, M. Korzeniowski, and H. Räcke, "A practical algorithm for constructing oblivious routing schemes," in *Proc. 15th ACM SPAA*, 2003, pp. 24–33.
- [5] A. Bermudez, R. Casado, F. J. Quiles, and J. Duato, "Use of provisional routes to speed-up change assimilation in InfiniBand networks," in *Proc. 2004 IEEE Int. Workshop CAC'04*, Apr. 2004, p. 186b.
- [6] A. Bermudez, R. Casado, F. J. Quiles, and J. Duato, "Fast routing computation on InfiniBand networks," *IEEE Trans. Parallel Distributed Syst.*, vol. 17, no. 3, pp. 215–226, Mar. 2006.
- [7] A. Borodin and J. E. Hopcroft, "Routing, merging and sorting on parallel models of computation," in *Proc. 14th ACM STOC*, 1982, pp. 338–344.
- [8] R. I. Greenberg and C. E. Lerserson, "Randomized routing on fat-trees," in *Randomness and Computation*, S. Micali, Ed. Stanford, CT: JAI, 1989, vol. 5, Advances in Computing Research, pp. 345–374.
- [9] C. Gomez, F. Gilabert, M. E. Gomez, P. Lopez, and J. Duato, "Deterministic versus adaptive routing in fat-trees, the," in *Proc. IPDPS Workshop CAC*, Apr. 2007, pp. 1–8.
- [10] M. T. Hajiaghayi, J. H. Kim, T. Leighton, and H. Räcke, "Oblivious routing in directed graphs with random demands," in *Proc. 37th ACM STOC*, 2005, pp. 193–201.
- [11] M. T. Hajiaghayi, R. Kleinberg, T. Leighton, and H. Räcke, "New lower bounds for oblivious routing in undirected graphs," in *Proc. Symp. Discrete Algorithms*, 2006, pp. 918–927.
- [12] M. T. Hajiaghayi, R. Kleinberg, H. Räcke, and T. Leighton, "Oblivious routing on node-capacitated and directed graphs," *ACM Trans. Algorithms*, vol. 3, no. 4, Nov. 2007.
- [13] C. Harrelson, K. Hildrum, and S. B. Rao, "A polynomial-time tree decomposition to minimize congestion," in *Proc. 15th ACM SPAA*, 2003, pp. 34–43.
- [14] C. Kaklamanis, D. Krizanc, and T. Santilas, "Tight bounds for oblivious routing in the hypercube," in *Proc. ACM SPAA*, 1990, pp. 31–36.
- [15] X. Lin, Y. Chung, and T. Huang, "A multiple LID routing scheme for fat-tree-based InfiniBand networks," in *Proc. 18th IEEE IPDPS'04*, Santa Fe, NM, Apr. 2004, p. 11a.
- [16] "InfiniBand™ architecture specification," InfiniBand™ Trade Association™, 2004.
- [17] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Trans. Comput.*, vol. C-34, pp. 892–901, Oct. 1985.
- [18] C. E. Leiserson, Z. S. Abuhmdeh, D. C. Douglas, C. R. Feynman, M. N. Ganmukhi, J. V. Hill, W. D. Hillis, B. C. Kuszmaul, M. A. S. Pierre, D. S. Wells, M. C. Wong-Chan, S.-W. Yang, and R. Zak, "The network architecture of the connection machine cm-5," *J. Parallel Distributed Comput.*, vol. 33, no. 2, pp. 145–158, Mar. 1996.
- [19] P. Lopez, J. Flich, and J. Duato, "Deadlock-free routing in InfiniBand through destination renaming," in *Proc. 2001 ICPP*, Sep. 2001, pp. 427–434.
- [20] J. C. Martinez, J. Flich, A. Robles, P. Lopez, and J. Duato, "Supporting fully adaptive routing in InfiniBand networks," in *Proc. 17th IEEE IPDPS'03*, Nice, France, Apr. 2003, p. 44a.
- [21] "Infiniband in the Enterprise Data Center," White Paper, 2006 [Online]. Available: <http://www.mellanox.com/pdf/whitepapers/scaling10gb-clusters.pdf>
- [22] Myricom home page [Online]. Available: <http://www.myri.com>
- [23] H. Räcke, "Minimizing congestion in general networks," in *Proc. IEEE Symp. FOCS*, 2002, pp. 43–52.
- [24] J. C. Sancho, A. Robles, and J. Duato, "Effective strategy to computing forwarding tables for InfiniBand networks," in *Proc. ICPP*, Sep. 2001, pp. 48–57.
- [25] J. C. Sancho, A. Robles, and J. Duato, "Effective methodology for deadlock-free minimal routing in InfiniBand networks," in *Proc. ICPP*, 2002, pp. 409–418.
- [26] M. Valerio, L. Moser, and P. Melliar-Smith, "Recursively scalable fat-trees as interconnect networks," in *Proc. 13th IEEE Int. Phoenix Conf. Comput. Commun.*, 1994, pp. 40–46.
- [27] L. G. Valiant and G. J. Brebner, "Universal schemes for parallel communication," in *Proc. 13th ACM STOC*, 1981, pp. 263–277.
- [28] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, "Cope: Traffic engineering in dynamic networks," in *Proc. ACM SIGCOMM*, 2006, pp. 99–110.



**Xin Yuan** (M'98) received the Ph.D. degree in computer science from the University of Pittsburgh, Pittsburgh, PA, in 1998.

He is an Associate Professor at the Department of Computer Science, Florida State University, Tallahassee. His research interests include parallel and distributed systems, networking, and high-performance communication for clusters of workstations.



**Wickus Nienaber** received the M.S. degree in computer science in 2007 from Florida State University, Tallahassee, where he is currently a Ph.D. student in the Department of Computer Science.

His research interests include networking issues in system area networks.



**Zhenhai Duan** (S'97–M'03) received the Ph.D. degree from the University of Minnesota in 2003.

He is currently an Assistant Professor in the Computer Science Department at Florida State University, Tallahassee. His research interests include computer networks and multimedia communications, especially the scalable network resource control and management in the Internet, Internet routing protocols and service architecture, and networking security.



**Rami Melhem** (F'00) received the Ph.D. degree in computer science from the University of Pittsburgh, Pittsburgh, PA, in 1981.

He is currently a Professor of computer science and electrical engineering and the Chair of the Computer Science Department at the University of Pittsburgh. His research interests include optical interconnection networks, fault-tolerant systems, high-performance computing, and parallel computer architecture.

Prof. Melhem was on the Editorial Board of the IEEE TRANSACTIONS ON COMPUTERS and the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. He is on the Editorial Board of the *Journal of Parallel and Distributed Computing*, *Computer Architecture Letters*, and the *International Journal of Embedded Systems*. He is a Member of the Association for Computing Machinery.