Oblivious Routing for Fat-Tree Based System Area Networks with Uncertain Traffic Demands

Xin Yuan Wickus Nienaber Zhenhai Duan
Department of Computer Science
Florida State University
Tallahassee, FL 32306
{xyuan,nienaber,duan}@cs.fsu.edu

Rami Melhem
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
melhem@cs.pitt.edu

ABSTRACT

Fat-tree based system area networks have been widely adopted in high performance computing clusters. In such systems, the routing is often deterministic and the traffic demand is usually uncertain and changing. In this paper, we study routing performance on 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 and the optimal routing algorithm for any given traffic demand. We consider both single path routing where the traffic between each source-destination pair follows one path, and multipath routing where multiple paths can be used for the traffic between a source-destination pair. We derive lower bounds of the oblivious performance ratio of any single path routing scheme for fat-tree topologies and develop single path oblivious routing schemes that achieve the optimal oblivious performance ratio for commonly used fat-tree topologies. These oblivious routing schemes provide the best performance guarantees among all single path routing algorithms under uncertain traffic demands. For multi-path routing, we show that it is possible to obtain a scheme that is optimal for any traffic demand (an oblivious performance ratio of 1) on the fat-tree topology. These results quantitatively demonstrate that single path routing cannot guarantee high routing performance while multi-path routing is very effective in balancing network loads on the fat-tree topology.

Categories and Subject Descriptors

C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design*

General Terms

Performance

SIGMETRICS'07, June 12–16, 2007, San Diego, California, USA.
Copyright 2007 ACM 978-1-59593-639-4/07/0006 ...$5.00.

Keywords

Oblivious routing, fat-tree, system area networks

1. INTRODUCTION

The fat-tree topology has many properties that make it attractive for large scale interconnects and system area networks [8, 9]. Most importantly the bisection bandwidth of the fat-tree topology scales linearly with the network size. The topology is also inherently highly resilient with a large number of redundant paths between two processing nodes. The fat-tree topology is very popular for building medium and large system area networks [6, 12]. In particular, it has been widely adopted by high performance computing (HPC) clusters that employ the off-the-shelf high speed system area networking technology, such as Myrinet [13] and Infiniband [7]. The fat-tree topology is used in many of the top 500 fastest supercomputers listed in the June 2006 release [16].

Although the fat-tree topology provides rich connectivity, having a fat-tree topology alone does not guarantee high network 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 balance in the network [9]. However, the routing in the current major system area networking technology such as Infiniband and Myrinet is deterministic [7, 13]. For a fat-tree based system area network with deterministic routing, it is important to employ an efficient load balance routing scheme in order to fully exploit the rich connectivity provided by the fat-tree topology.

Traditional load balance 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 usually uncertain and changing. Consider, for example, the traffic demands in a large HPC cluster. Since many users share such a system and can run many 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 load balance routing schemes, which have recently been shown to promise excellent performance guarantees with changing and uncertain traffic demands in the Internet environment [1, 2, 18].

In this paper, we investigate routing performance on fat-tree based system area networks with deterministic routing under the assumption that the traffic demand is uncertain and changing. For a given traffic demand that can be represented by a traffic matrix, 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 the next section. Informally, a routing algorithm, r, with an oblivious performance ratio of x means that for *any* traffic demand, the performance (maximum link load) of r on the demand is at most x times that of the optimal routing algorithm for this demand. An oblivious performance ratio of 1 means that the routing algorithm is optimal for all traffic demands.

System area networks, including Infiniband and Myrinet, typically support single path routing and some form of multipath routing. It is well known that single path routing is simple, but may not be as effective as multi-path routing in balancing network loads. On the other hand, multi-path routing introduces complications such as packet reordering that the network system must handle. However, the performance difference between single path routing and multipath routing on the fat-tree topology 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 multi-path routing for its performance. This paper resolves this problem: it provides a concrete quantitative comparison between the performance of single path routing and multi-path routing on the fat-tree topology.

This study focuses on fat-tree topologies 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 analyzing techniques, can be easily extended to other types of fat-tree topologies. The major conclusions in this paper include the following. For a 3-level fat-tree, we prove that the oblivious performance ratio of any single path routing algorithm is at least $\sqrt{\frac{m}{2}}$. For a 4-level fat-tree, we prove that the oblivious performance ratio of any single path routing algorithm is at least $\frac{m}{2}$. For a fat-tree of height H, $H > 4$, we show that the oblivious performance ratio of any single path routing algorithm is at least $(\frac{m}{2})^{\lfloor\frac{H-2}{3}\rfloor}$. These lower bounds indicate that single path routing cannot guarantee high routing performance on the fat-tree topology. For example, for any single path routing algorithm on a 4-level fat-tree formed by 16-port switches, there always exists a traffic demand such that this routing algorithm is $\frac{16}{2} = 8$ times worse than the optimal algorithm for that traffic demand. We show that the lower bounds are tight for 3-level and 4-level fat-trees by developing optimal single path oblivious routing schemes that achieve these bounds. These algorithms provide the best performance guarantees among all single path routing algorithms under uncertain traffic demands. It must be noted that practical fat-tree topologies are usually no more than 4 levels: depending on the number of ports in the switches forming the fat-tree, a 4-level fat-tree can easily support more than ten thousands processing nodes. Hence, the single path routing schemes developed in this paper are sufficient for most practical fat-tree based networks. For multi-path routing, we show that it is possible to obtain a scheme that is optimal for any traffic demand (an oblivious performance ratio of 1) on the fat-tree topology. This suggests that multipath routing is much more effective than single path routing in providing the worst case performance guarantees on the fat-tree topology.

The rest of the paper is organized as follows. In Section 2, we formally define routing and the metrics for evaluating routing schemes and specify the fat-tree topology. In Section 3, we study the single path oblivious routing schemes for the fat-tree topology. In Section 4, we present the results for multi-path routing. Section 5 reports the results of our performance study of the proposed algorithms and other routing algorithms designed for the fat-tree topology. Section 6 describes the related work. Finally, Section 7 concludes the paper.

2. BACKGROUND

2.1 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, $0 \le i \le N-1$, $0 \le i \le N-1$, denotes the amount of traffic from node i to node j. Let A be a set, $|A|$ denotes the size of the set.

The definitions of routing and the performance metrics in this paper are modeled after [1]. A *routing* specifies how the traffic of each Source-Destination (SD) pair is routed across the network. We consider two types of routing schemes: single path routing where only one path can be used for each SD pair, and multi-path routing where multiple paths can be used. In multi-path routing, each path for an SD pair routes a fraction of the traffic for the SD pair.

The multi-path routing can be characterized by a set of paths $MP_{i,j} = \{MP^1_{i,j}, MP^2_{i,j}, ..., MP^{|MP_{i,j}|}_{i,j}\}$ for each SD pair (i,j), and the fraction of the traffic routed through each path $f_{i,j} = \{f^k_{i,j} | k = 1, 2, ..., |MP_{i,j}|\}$. $\sum_{k=1}^{|MP_{i,j}|} f^k_{i,j} = 1$. Hence, a multi-path routing, mr, is specified by a set of paths $MP_{i,j}$ and a vector representing the fraction of the traffic routed through each path $f_{i,j}$ for each SD pair (i,j), $0 \le i \le N-1$ and $0 \le j \le N-1$. Let link $l \in MP^k_{i,j}$, the contribution of the traffic $tm_{i,j}$ to link l through path $MP^k_{i,j}$ is thus $tm_{i,j} \times f^k_{i,j}$. Notice that link l may be in more than one path in $MP_{i,j}$. In this case, multiple paths for the same SD pair can contribute to the traffic on link l. Single path routing is a special case of multi-path routing where $|MP_{i,j}| = 1$ and all traffic from node i to node j is routed through $MP^1_{i,j}$ ($f^1_{i,j} = 1$). Hence, a single path routing can be specified by a path $MP^1_{i,j}$ for each SD pair (i,j).

A common metric for the performance of a routing scheme with respect to a certain traffic matrix, TM, is 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 multi-path routing mr, the maximum link load is given by

$$MLOAD(mr, TM) = \max_{l \in Links}\{\sum_{i,j,k \texttt{ such that } l \in MP^k_{i,j}} tm_{i,j} \times f^k_{i,j}\}.$$

For a single path routing sr, the maximum link load formula is degenerated to

$$MLOAD(sr, TM) = \max_{l \in Links}\{\sum_{i,j \texttt{ such that } l \in P^1_{i,j}} tm_{i,j}\}.$$

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

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

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

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

The value for $PERF(r, TM)$ is always at least 1. It is exactly 1 if and only if the routing is optimal for TM. When a routing is optimized for a specific traffic matrix, it does not provide any guarantees for other traffic matrices. 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. The definition of performance ratio of a routing is extended to be with respect to a set of traffic matrices [1]. Let Γ be a set of traffic matrices, The performance ratio of a routing r on Γ is defined as

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

When the set Γ 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 $PERF(r)$. 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.

2.2 Fat-tree topology

In a fat-tree network, all links are bidirectional with the same capacity. Figure 1 compares a binary tree with a binary fat-tree topology. 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 serious congestion towards the root. The binary fat-tree topology remedies this situation by maintaining the same bandwidth at each level of the network.

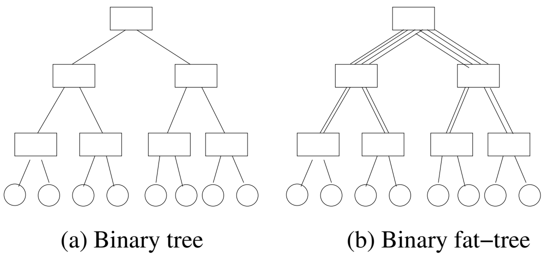

(a) Binary tree (b) Binary fat−tree

Figure 1: Binary tree and binary fat-tree topologies

The fat-tree topology shown in Figure 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 multi-stage networks that are formed by nodes with small nodal degrees [6, 17]. For example, the fat-tree in Figure 1 (b) can be approximated by the topology in Figure 2. These alternatives trade the connectivity with the implementation simplicity. In this paper, we focus on one of such alternatives: 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 has 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 [6]. The technique we developed for this topology can easily be extended for other fat-tree variations.

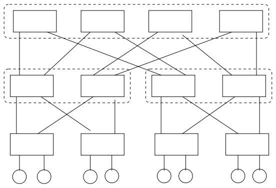


Figure 2: Approximate the topology in Figure 1 (b)

We will follow the naming convention in [6]: the fat-tree is called m-port n-tree and denoted as $FT(m, n)$. The parameter m in $FT(m, n)$, 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 $FT(m, n)$ is $n+1$, that is, $FT(m, n)$ is an $n+1$ level tree. $FT(m, n)$ is a minor generalization of the topology in [6] in that we allow m to be a multiple of 2 while in [6], m must be a power of 2. In the rest of this paper, internal nodes in $FT(m, n)$ 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, $FT(4, 3)$, is shown in Figure 3. Notice that in practice a processing element may have multiple network adapters and have multiple connections to the network. From the network point of view, however, supporting such a processing element is the same as supporting multiple processing nodes, each having one connection to the network with the combined traffic representing the traffic for the processing element. Hence, we will assume that each processing node has one connection to the network. Notice also that the topology in Figure 2 is not an m-port n-tree since there are two types of switches in the topology: top-level switches have a degree of 2 and the lower-level switches have a degree of 4.

Next, we will describe how $FT(m, n)$ is formed. More details can be found in [6]. $FT(m, n)$ is formed by connecting the root level switches to m sub-fat-trees with $n-1$ levels of switches. We will use the notion $SUBFT(m, n-1)$ to denote the sub-fat-trees with $n-1$ levels of switches. $SUBFT(m, l)$ is different from $FT(m, l)$ in that $SUBFT(m, l)$ must provide (open ended) up links for the sub-fat-tree while $FT(m, l)$ does not have up links. $SUBFT(m, l)$ is recursively constructed as follows.

When $l = 1$, $SUBFT(m, 1)$ contains 1 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

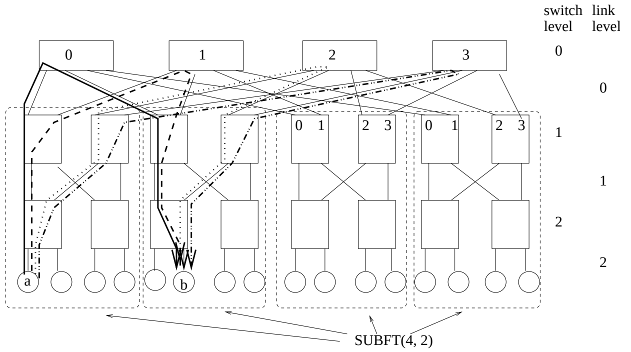
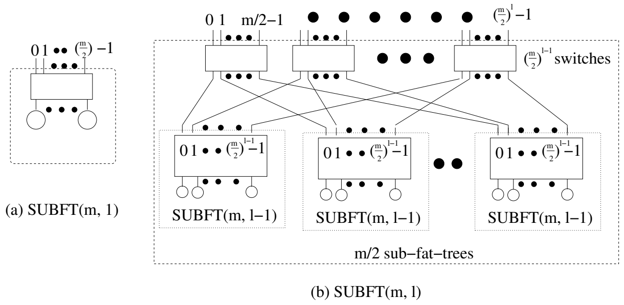


Figure 3: The 4-port 3-three ($FT(4,3)$)

$SUBFT(m,l)$ as $nu(m,l)$. $nu(m,1) = \frac{m}{2}$. As will be shown later, $nu(m,l) = (\frac{m}{2})^l$. The up-link ports in $SUBFT(m,l)$ are numbered from 0 to $nu(m,l) - 1$.

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


0 1 •• $(\frac{m}{2})$ −1
0 1 m/2−1
$(\frac{m}{2})^l$ −1
$(\frac{m}{2})^{l-1}$ switches
0 1 • • $(\frac{m}{2})^{l-1}$−1
SUBFT(m, l−1)
SUBFT(m, l−1)
SUBFT(m, l−1)
m/2 sub−fat−trees

(a) SUBFT(m, 1)

(b) SUBFT(m, l)

Figure 4: $SUBFT(m,1)$ and $SUBFT(m,n)$

$FT(m,n)$ is formed by having $nu(m,n-1) = (\frac{m}{2})^{n-1}$ root level switches connecting with m $SUBFT(m,n-1)$'s. Let us number top level switches from 0 to $(\frac{m}{2})^{n-1}-1$. The up-link port i, $0 \leq i < nu(m,n-1)$, in all of the $SUB(m,n-1)$'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 Figure 5. Note that $FT(m,n)$ does not need to maintain any open ended link and hence, each of the m ports in a root level switch can connect to a sub-fat-tree. $FT(m,n)$ supports $m(\frac{m}{2})^{n-1}$ processing nodes. It has n levels of switches. 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 $(2n-1) \times (\frac{m}{2})^{n-1}$. Figure 3 shows the complete topology of $FT(4,3)$ that has $(2 \times 3 - 1) \times (\frac{4}{2})^{3-1} = 20$ switches and $4 \times (\frac{4}{2})^{3-1} = 16$ processing nodes.

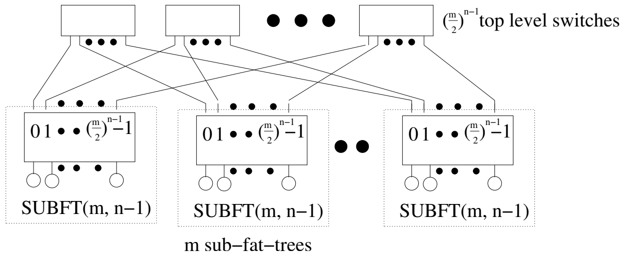

$(\frac{m}{2})^{n-1}$top level switches
0 1 • • $(\frac{m}{2})^{n-1}$−1
SUBFT(m, n−1)
SUBFT(m, n−1)
SUBFT(m, n−1)
m sub−fat−trees

Figure 5: The structure of $FT(m,n)$

$FT(m,n)$ has n levels of switches, which are numbered from 0 to $n-1$ (root level switches being level 0 switches). Similarly, we will classify the links according to the 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 bi-directional links: with an up channel for communication from a lower level switch to an upper level switch, and a down channel for communication 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, ..., $m \times (\frac{m}{2})^{n-2}$ $SUBFT(m,1)$'s.

Level 0 (root level) switches do not belong to any sub-fat-trees. 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 $SUBFT(m,n-2)$, ..., and a $SUBFT(m,n-i)$. In $FT(m,n)$, we will call switches in levels 0, 1, ..., 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 with the up-link ports with the same port number in other $SUBFT(m,i)$'s. 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-trees 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).

Figure 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 (when the path uses such a link). Thus, each of such links 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 4 paths from node s to node d in Figure 3, all 4 paths use the level 2 up/down links (the link connecting the processing node), 2 paths use each of the level 1 up/down links that carries traffic from a to b, and 1 path uses each of the level 0 up/down links carrying traffic from a to b.

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 $(\frac{m}{2})^{n-1-i}$ destination nodes.

This property is also intuitive. For example, when $i = n-1$, a level $i = n-1$ link directly connects to a processing nodes. So 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.

3. SINGLE PATH OBLIVIOUS ROUTING

3.1 Lower bounds of oblivious performance ratio for single path routing

In this section, we derive the lower bounds of oblivious performance ratio for single path routing on $FT(m,n)$. The following concepts will be used in the derivation of the lower bounds. Let $A = \{(s_1,d_1),(s_2,d_2),...\}$ be a set of SD pairs.

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

Basically, in a node disjoint set of SD pairs, each source (in the source-destination pair) appears in the set as a source exactly once; and each destination appears in the set as a destination exactly once. It must be noted that 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 given set of SD pairs A, a set of SD pairs B is said to be a **node disjoint subset** of A when the following conditions are met: (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 the following two conditions are met: (1) B is a node disjoint subset of A; and (2) let C be a node disjoint subset of A, $|B| \geq |C|$. 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. $SRC(A) = \{s|\exists(s,d) \in A\}$ is the set of source nodes in A and $DST(A) = \{d|\exists(s,d) \in A\}$ is the set of destination nodes in A. We denote $LS(A)$ the largest number of SD pairs in A either with the same source or with the same destination. Formally,

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

For any node i, $|S_i^A|$ and $|D_i^A|$ are at most $LS(A)$.

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, $L(A) = 3$. $SRC(A) = \{1,2,3\}$; and $DST(A) = \{1,2,3,4\}$. $S_1^A = \{(1,2),(1,3)\}$; $S_2^A = \{(2,1),(2,4)\}$; 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, $LS(A) = 2$.

The following lemmas give some properties of these concepts.

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

Proof: Straight-forward from the largest node disjoint subset definition. □

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

Proof: Let C be a largest node disjoint subset of $A \bigcup B$. $|C| = L(A \bigcup B)$. 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 $|C_A| + |C_B| \geq |C| = L(A \bigcup B)$. 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, $L(A) + L(B) \geq L(A \bigcup B)$. □

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 $L(A - S_s^A) + L(S_s^A) \geq L((A - S_s^A) \bigcup S_s^A) = L(A)$. 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 $B = \{(s_1,d_1),(s_2,d_2),...,(s_k,d_k)\}$ 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 $C = B \bigcup \{(s,d)\}$ 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, $L(A - S_a^A) = L(A) - 1$. □

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

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 a_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 Lemma 3 and Lemma 3a. □.

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

Proof: See Appendix. □

Besides using the concepts introduced in the above definitions and lemmas, we will use a topology, called extended 2-layer fat-tree, in the derivation of the lower bounds for oblivious performance ratio on $FT(m,n)$. The extended 2-

layer fat-tree, denoted as $EFT2(m,k)$, 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 processing nodes and the other half connecting 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 $FT(m,2)$, which is shown in Figure 6. The difference is that $FT(m,2)$ uses the same kind of switches in both levels while $EFT2(m,k)$ has more flexibility: the switches in the top level can be different from the switches in the bottom level. The $FT(m,2)$ topology is the same as $EFT2(m,m)$. We will also use a sub-graph of $EFT2(m,k)$, which we call $SEFT2(m,k)$. $SEFT2(m,k)$ contains all lower level switches and processing nodes in $EFT2(m,k)$, but only one root level switch. Figure 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 this figure, we separate the two directional channels.

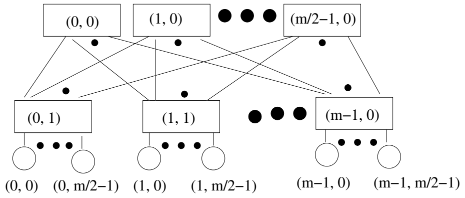


Figure 6: $FT(m,2)$ **topology**

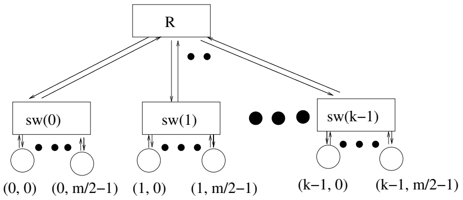


Figure 7: $SEFT2(m,k)$ **topology**

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

Proof: In $EFT2(m,k)$, each of the top level switches has a link with each of the bottom level switches. Since $|A| \leq k$, we can assign a different top level switch for each SD pair $(s_i,d_i) \in A$. For each (s_i,d_i), if s_i and d_i are in the same switch, there is only one path between s_i and d_i (from s_i to the switch connecting both s_i and d_i, and then to d_i). Since A is node disjoint, these links are not used by other paths for other SD pairs. If s_i and d_i are not in the same switch, the path for (s_i,d_i) is: from s_i to the bottom level switch connecting s_i to the top level switch assigned to (s_i,d_i) to the bottom level switch connecting d_i to d_i. This way, all the SD pairs in A are routed with link disjoint paths. □

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, $|A| \leq k$, Then, $PERF(sr) \geq |A|$.

Proof: To show that $PERF(sr) \geq |A|$, we must show that there exists a traffic matrix TM such that $\frac{MLOAD(sr,TM)}{OPTU(TM)} \geq |A|$. Consider a traffic matrix TM where $tm_{i,j} = 1$ for all $(i,j) \in A$ and all other entries are 0 (no other traffic). 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,

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

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. The *maximum disjoint size* of routing r, $mds(r)$, is defined as $mds(r) = \max_{l \in Links} mds(r,l)$. Notice that in $EFT2(m,k)$ and $SEFT2(m,k)$, a level 1 link directly connected to a processing node. From Lemma 1, the maximum disjoint size on such a link is at most 1.

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: In $SEFT2(m,k)$, at most $k(k-1)(\frac{m}{2})^2$ SD pairs can be routed through the root. The lemma is always true when $X \geq \frac{m}{2}$.

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 and the k level 1 switches $sw(0)$, $sw(1)$, ..., $sw(k-1)$ as shown in Figure 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 $sw(i)$ and destination nodes in switch $sw(j)$. $S = \bigcup_{i,j \text{ such that } 0\leq i\neq j\leq k-1} S_{i,j}$. Let us denote

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

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

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

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

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

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

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

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

Combining these two in-equations, we obtain $L(S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}) \leq X - (\sum_{x \neq i} E_{i,x} + F_{i,j} + E_{i,j} + \sum_{x \neq j} F_{x,j})/2$. Each source or destination node in $S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}$ can have no more than X SD pairs in the set (otherwise, these SD pairs would be included in either $LX_{i,j}^{src}$ or $LX_{i,j}^{dst}$). Hence, $LS(S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}) \leq X$. From Lemma 5, $|S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}| \leq L(S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}) \times LS(S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}) \leq (X - (\sum_{x \neq i} E_{i,x} + F_{i,j} + E_{i,j} + \sum_{x \neq j} F_{x,j})/2) \times X$. Hence, 9

$$\begin{aligned}
&|\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} (S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst})| \\
&\leq \sum_{i=0}^{k-1} \sum_{j \neq i} |S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}| \\
&\leq \sum_{i=0}^{k-1} \sum_{j \neq i} X \times (X - (\sum_{x \neq i} E_{i,x} + F_{i,j} \\
&\qquad\qquad + E_{i,j} + \sum_{x \neq j} F_{x,j})/2) \\
&\leq k(k-1)X^2 - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j}
\end{aligned}$$

Since each switch connects to $\frac{m}{2}$ processing nodes, $LX_{i,j}^{src} \leq E_{i,j} \times \frac{m}{2}$ and $LX_{i,j}^{dst} \leq F_{i,j} \times \frac{m}{2}$. Hence,

$$\begin{aligned}
|S| &= |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} S_{i,j}| \\
&\leq |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} ((S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst}) \bigcup LX_{i,j}^{src} \bigcup LX_{i,j}^{dst})| \\
&\leq |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} (S_{i,j} - LX_{i,j}^{src} - LX_{i,j}^{dst})| \\
&\quad + |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} LX_{i,j}^{src}| + |\bigcup_{i=0}^{k-1} \bigcup_{j \neq i} LX_{i,j}^{dst}| \\
&\leq k(k-1)X^2 - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} - \frac{kX}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j} \\
&\quad + \frac{m}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} + \frac{m}{2} \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j} \\
&= k(k-1)X^2 \\
&\quad - (\frac{kX}{2} - \frac{m}{2})(\sum_{i=0}^{k-1} \sum_{j \neq i} E_{i,j} + \sum_{i=0}^{k-1} \sum_{j \neq i} F_{i,j})
\end{aligned}$$

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

Let us denote 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 by T(X). Obviously, when $X > Y$ and $T(Y)$ is a subset of all 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 $X < \frac{m}{k}$, $T(X) < T(\frac{m}{k}) \leq k(k-1)(\frac{m}{k})^2$.

Lemma 9: Let r be a single path routing algorithm on $EFT2(m,k)$. If $k \geq \sqrt{2m}$, $PERF(r) \geq \sqrt{\frac{m}{2}}$.

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 $EFT2(m,k)$, 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 $SEFT2(m,k)$ formed by that 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 must be routed through the switch ($T(X) = k(k-1)\frac{m}{2}$), $X < \sqrt{\frac{m}{2}}$ cannot be true. Thus, $X \geq \sqrt{\frac{m}{2}}$. Since $k \geq \sqrt{2m} \geq \sqrt{\frac{m}{2}}$, from Lemma 7, $PERF(r) \geq \sqrt{\frac{m}{2}}$. □

The following three lemmas are the main results in this section.

Lemma 10: Let r be a single path routing algorithm for $FT(m,2)$ $(m \geq 2)$, $PERF(r) \geq \sqrt{\frac{m}{2}}$.

Proof: $FT(m,2)$ is equivalent to $EFT2(m,m)$. Since in $FT(m,2)$, $m > 2$ and $m \geq \sqrt{2m}$. From Lemma 9, $PERF(r) \geq \sqrt{\frac{m}{2}}$. □

Lemma 11: Let r be a single path routing algorithm for $FT(m,3)$, $PERF(r) \geq \frac{m}{2}$.

Proof: $FT(m,3)$ is composed by top level switches (m-port) with m $SUBFT(m,2)$'s. Let us consider the maximum disjoint sizes on the links that connect the $SUBFT(m,2)$'s with the root level switches (Level 0 links). By treating each $SUBFT(m,2)$ as one $2(\frac{m}{2})^2$-port switch, $FT(m,3)$ is approximated as $EFT2(2(\frac{m}{2})^2, m)$. Following the proof of Lemma 9, since $m \geq \sqrt{2 \times 2(\frac{m}{2})^2}$, the largest of the maximum disjoint sizes of the level 0 links is at least $\sqrt{\frac{2 \times (\frac{m}{2})^2}{2}} = \frac{m}{2}$. From Lemma 7, $PERF(r) = \frac{m}{2}$. □

Lemma 12: Let r be a single path routing algorithm for $FT(m,n)$, $PERF(r) \geq (\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}$.

Proof: Let us consider the maximum disjoint sizes on links connecting the to up-link ports of $SUBFT(m,i)$'s, $1 \leq i \leq n-1$, in $FT(m,n)$. From Property 1 and Property 2 of $FT(m,n)$, the connectivity in $FT(m,n)$ can be partitioned into two levels (with respective to such links): the lower level connectivity provided by $SUBFT(m,i)$'s and the upper level connectivity provided by the upper level switches for $SUBFT(m,i)$'s. The connectivity in $SUBFT(m,i)$ can be approximated as a $2(\frac{m}{2})^i$-port switch; and the upper level switches the connects the up-link ports with the same port number in each of the $SUBFT(m,i)$ (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 $EFT2(2(\frac{m}{2})^{\lfloor \frac{2(n-1)}{3} \rfloor}, m(\frac{m}{2})^{\lceil \frac{n-1}{3} \rceil})$. Since

$$m(\frac{m}{2})^{\lceil \frac{n-1}{3} \rceil} \geq \sqrt{2 \times 2(\frac{m}{2})^{\lfloor \frac{2(n-1)}{3} \rfloor}},$$

following the proof in Lemma 9, the largest of the maximum disjoint sizes on such links is at least

$$\sqrt{\frac{2(\frac{m}{2})^{\lfloor \frac{2(n-1)}{3} \rfloor}}{2}} = (\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}.$$

From Lemma 7, $PERF(r) \geq (\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}$. □

Note that the tree height of $FT(m,n)$ is $n+1$: $FT(m,2)$ is a 3-level fat-tree; $FT(m,3)$ is a 4-level fat-tree; and $FT(m,n)$ is $n+1$-level. This sub-section establishes that the lower bounds of the oblivious performance ratio for any single path routing is $\sqrt{\frac{m}{2}}$ for a 3-level fat-tree, $\frac{m}{2}$ for a 4-level fat-tree, and $(\frac{m}{2})^{\lfloor \frac{H-2}{3} \rfloor}$ for an H-level fat-tree, $H > 4$. Notice also that while our results are obtained for m-port n-trees ($FT(m,n)$), the techniques can be easily extended to other types of tree (e.g. the ones with root level switches being different from switches in other levels such as the fat-tree shown in Figure 2).

3.2 Optimal single path oblivious routing for $FT(m,2)$ and $FT(m,3)$

Most practical fat-tree topologies have no more than three levels of switches. This is because it is common to use 32-port or 48-port switches to construct large fat-tree topologies. Using 32-port switches, $FT(32,2)$ supports up to 512 processing nodes while $FT(32,3)$ support up to 8192 processing nodes. With 48-port switches, $FT(48,3)$ can support $24\times 24\times 48 = 27648$ processing nodes. Hence, optimal oblivious routing schemes for $FT(m,2)$ and $FT(m,3)$ bear most practical significance. Moreover, the development of these algorithms also bears theoretical significance by making the lower bounds on the oblivious performance ratio for $FT(m,2)$ and $FT(m,3)$ (Lemma 10 and Lemma 11) tight bounds.

Let N be the number of processing nodes. Let the traffic matrix be TM with entries $tm_{i,j}$, $0 \le i \le N-1$ and $0 \le j \le N-1$, 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 (single path or multi-path) 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

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

The minimum maximum link load on the fat-tree topology using any routing scheme, single path or multi-path, is at least $baseload(TM)$ for any traffic matrix TM. In other words,

$$OPTU(TM) \ge baseload(TM).$$

Our optimal single path oblivious routing schemes are based on the following Lemma.

Lemma 13: If a single path routing scheme r routes SD pairs such that the SD pairs in each of the links in $FT(m,n)$ are either from at most X sources or towards at most X destinations, then $PERF(r) \le X$.

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

Optimal oblivious routing for $FT(m,2)$

As defined in Section 2, $FT(m,2)$ contains $\frac{3m}{2}$ switches and supports $\frac{m^2}{2}$ processing nodes. In this topology, $\frac{m}{2}$ switches are in the level 0 and m switches are in level 1. There is one link from each switch in level 1 to each of the switch in level 0. In order to describe the oblivious routing algorithm, we will give a non-recursive description of $FT(m,2)$. The $\frac{m}{2}$ top level switch are labeled switches $(0,0)$, $(1,0)$, ..., $(\frac{m}{2}-1,0)$. The m level 1 switches are labeled switches $(0,1)$, $(1,1)$, ..., $(m-1,1)$. Each level 1 switch $(i,1)$, $0\le i\le m-1$, is connected with $\frac{m}{2}$ processing nodes numbered as $(i,0)$, $(i,1)$, ..., $(i,\frac{m}{2}-1)$. There is a link between switch $(i,0)$, $0 \le i \le \frac{m}{2}-1$ and switch $(j,1)$, $0\le j\le m-1$. For $0\le i\le m-1$, there is a link between processing node (i,x), $0\le x\le \frac{m}{2}-1$, and switch $(i,1)$. Figure 6 depicts the $FT(m,2)$ topology as well as the switch and processing node labeling.

Lemma 10 states that any single path routing r, $PERF(r) \ge \sqrt{\frac{m}{2}}$ for $FT(m,2)$. To ease exposition, let us assume that $\sqrt{\frac{m}{2}}$ is an integer. The cases when $\sqrt{\frac{m}{2}}$ is not an integer can be handled with some minor modifications. We give the algorithm that deals with the cases when $\sqrt{\frac{m}{2}}$ is not an integer in Figure 8. Following Lemma 13, the optimal oblivious routing algorithm schedules the SD pairs such that the traffic in each up link from a bottom level switch to a top level switch has exactly $\sqrt{\frac{m}{2}}$ sources while each down from a top level switch to a bottom level switch has exactly $\sqrt{\frac{m}{2}}$ destinations. Note that in $FT(m,2)$, each of the level 1 links carries traffic either to 1 node or from 1 node (the node that the link is connected to) and we do not have to consider such links in the design of the optimal oblivious routing algorithm. Let $Z=\sqrt{\frac{m}{2}}$. The routing algorithm 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 $(i,1)$, $0\le i\le m-1$, are partitioned into $Z=\sqrt{\frac{m}{2}}$ groups: group j, $0\le j\le Z-1$, includes nodes $(i, j*Z)$, $(i, j*Z+1)$, ..., $(i, j*Z+Z-1)$.

The SD pairs are scheduled as follows. The up-link $(i,1)\rightarrow(j,0)$, $0\le i\le m-1$ and $0\le j\le \frac{m}{2}-1$, carries SD pairs with sources nodes in group j/Z in switch $(i,1)$ and destination nodes in group $j\ mod\ Z$ in all other switches. More specifically, The up-link $(i,1)\rightarrow(0,0)$ carries traffic from group 0 processing nodes (in switch $(i,1)$) to group 0 processing nodes in other switches; $(i,1)\rightarrow(1,0)$ carries traffic from group 0 processing nodes to group 1 processing nodes in other switches; ...; $(i,1)\rightarrow(Z-1,0)$ carries traffic from group 0 processing nodes to group $Z-1$ processing nodes in other switches; $(i,1)\rightarrow(Z,0)$ carries traffic from group 1 processing nodes to group 0 processing nodes in other switches; ...; $(i,1)\rightarrow(2Z-1,0)$ carries traffic from group 1 processing nodes to group $Z-1$ processing nodes in other switches; ...; $(i,1)\rightarrow((Z-1)Z,0)$ carries traffic from group $Z-1$ processing nodes to group 0 processing nodes in other switches; ...; $(i,1)\rightarrow(Z^2-1,0)$ carries traffic from group $Z-1$ processing nodes to group $Z-1$ processing nodes in other switches. The traffic in the down link $(j,0)\rightarrow(i,1)$, $0\le i\le m-1$ and $0\le j\le\frac{m}{2}-1$, which is fixed once the traffic in the up-link is decided, carries traffic with source nodes in group j/Z in all other switches than switch $(i,1)$ to destination nodes in group $j\ mod\ Z$ in switch $(i,1)$. Hence, each of the up-links carries traffic from exactly Z source nodes and each of the down links carries traffic to exactly Z destination nodes.

The detailed routing algorithm, called $OSRM2$, is shown in Figure 8. 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 switch 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 1: When $\sqrt{\frac{m}{2}}$ is an integer, $PERF(OSRM2) = \sqrt{\frac{m}{2}}$.

Proof: As discuss earlier, using $OSRM2$, each link carries traffic either from $\sqrt{\frac{m}{2}}$ sources or to $\sqrt{\frac{m}{2}}$ destinations. From Lemma 13, $PERF(OSRM2)\le\sqrt{\frac{m}{2}}$. From Lemma 10, $PERF(OSRM2)\ge\sqrt{\frac{m}{2}}$. Hence, $PERF(OSRM2)=\sqrt{\frac{m}{2}}$ and $OSRM2$ is an optimal oblivious routing algorithm for

Algorithm OSRM2:

```
Route from node $(s_0, s_1)$ to node $(d_0, d_1)$
Let $m2 = \frac{m}{2}$;
Let $Z_s = \lceil\sqrt{m2}\rceil$, $Z_d = \lfloor\sqrt{m2}\rfloor$;
Let $N_s = \lceil\frac{m2}{Z_s}\rceil$, $N_d = \lceil\frac{m2}{Z_d}\rceil$;
if $(s_0 == d_0)$
  use route: $node(s_0, s_1) \rightarrow switch(s_0, 1) \rightarrow node(d_0, d_1)$
if $(s_0! = d_0)$
  use route: $node(s_0, s_1) \rightarrow switch(s_0, 1)$
             $\rightarrow switch(s_1/N_s * N_d + d_1/N_d, 0)$
             $\rightarrow switch(d_0, 1) \rightarrow node(d_0, d_1)$
```

Figure 8: Optimal oblivious routing for $FT(m, 2)$

$FT(m, 2)$ when $\sqrt{\frac{m}{2}}$ is an integer. □

Optimal oblivious routing for $FT(m, 3)$

We will now consider $FT(m, 3)$. From Section 2, $FT(m, 3)$ contains three levels of switches, with the top level having $nu(m, 2) = \frac{m}{2} \times \frac{m}{2}$ 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 $((i_0, i_1), level)$: the top level switches are labeled as $((i_0, i_1), 0)$, $0 \le i_0 \le \frac{m}{2} - 1$ and $0 \le i_1 \le \frac{m}{2} - 1$; the level 1 switches are labeled as $((i_0, i_1), 1)$, $0 \le i_0 \le m - 1$, and $0 \le i_1 \le \frac{m}{2} - 1$; the level 2 switches are labeled as $((i_0, i_1), 2)$, $0 \le i_0 \le m - 1$ and $0 \le i_1 \le \frac{m}{2} - 1$. 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), $0 \le p_0 \le m - 1$, $0 \le p_1 \le \frac{m}{2} - 1$, and $0 \le p_2 \le \frac{m}{2} - 1$. A processing node (p_0, p_1, p_2) is attached to switch $((p_0, p_1), 2)$, $0 \le p_0 \le m - 1$, $0 \le p_1 \le \frac{m}{2} - 1$, and $0 \le p_2 \le \frac{m}{2} - 1$. A level 2 switch $((i_0, i_1), 2)$, $0 \le i_0 \le m - 1$ and $0 \le i_1 \le \frac{m}{2} - 1$, has a link to each of the level 1 switches $((i_0, X), 1)$, $0 \le X \le \frac{m}{2} - 1$. A level 1 switch $((i_0, i_1), 1)$, $0 \le i_0 \le m - 1$ and $0 \le i_1 \le \frac{m}{2} - 1$, has a link to each of the level 0 switches $((i_1, X), 0)$, $0 \le X \le \frac{m}{2} - 1$.

From Lemma 11, we can see that for any single path routing algorithm r on $FT(m, 3)$, $PERF(r) \ge \frac{m}{2}$. Like in the $FT(m, 2)$ case, our optimal routing algorithm ensures that the SD pairs in each link are either from at most $\frac{m}{2}$ sources or towards at most $\frac{m}{2}$ destinations. From Property 5 in Section 2.2, each level 1 or level 2 link in $FT(m, 3)$ carries traffic either from no more than $\frac{m}{2}$ sources 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 links connecting layer 0 and layer 1 switches). 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 $(\frac{m}{2})^2 = Z^2$ 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 $SUBFT(m, 2)$.

Algorithm OSRM3:

```
Route from node $(s_0, s_1, s_2)$ to $(d_0, d_1, d_2)$:
if $(s_0 == d_0$ 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: $node(s_0, s_1, s_2) \rightarrow switch((s_0, s_1), 2)$
                                   $\rightarrow switch((s_0, s_2), 1)$
                                   $\rightarrow switch((s_2, d_2), 0)$
                                   $\rightarrow switch((d_0, s_2), 1)$
                                   $\rightarrow switch((d_0, d_1), 2)$
                                   $\rightarrow node(d_0, d_1, d_2)$
```

Figure 9: Optimal oblivious single routing for $FT(m, 3)$

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; ...; $((i_0, 0), 1) \rightarrow ((0, Z-1), 0)$ carries traffic from group 0 processing nodes to group $Z-1$ processing nodes in other $SUBFT(m, 2)$'s; $((i_0, 1), 1) \rightarrow ((1, 0), 0)$ carries traffic from group 1 processing nodes to group 0 processing nodes in other $SUBFT(m, 2)$'s; ...; $((i_0, 1), 1) \rightarrow ((1, Z - 1), 0)$ carries traffic from group 1 processing nodes to group $Z - 1$ processing nodes in other $SUBFT(m, 2)$'s; ...; $((i_0, Z - 1), 1) \rightarrow ((Z - 1, 0), 0)$ carries traffic from group $Z - 1$ processing nodes to group 0 processing nodes in other $SUBFT(m, 2)$'s; ...; $((i_0, Z - 1), 1) \rightarrow ((Z - 1, Z - 1), 0)$ carries traffic from group $Z - 1$ processing nodes to group $Z - 1$ processing nodes in other $SUBFT(m, 2)$'s. This way, each up-link only carries SD pairs with exactly $Z = \frac{m}{2}$ sources. Similarly, each down link only carries SD pairs with exactly Z destinations. The detailed routing algorithm, called $OSRM3$, is shown in Figure 9

Theorem 2: $PERF(OSRM3) = \frac{m}{2}$ and $OSRM3$ is an optimal oblivious routing algorithm for $FT(m, 3)$.

Proof: From above discussion, using $OSRM3$, the SD pairs in each link have either at most $\frac{m}{2}$ source nodes or at most $\frac{m}{2}$ destination nodes. From Lemma 13, $PERF(OSRM3) \le \frac{m}{2}$. From Lemma 11, an performance oblivious ratio of $\frac{m}{2}$ is the low bound for any single path routing scheme on $FT(m, 3)$. Hence, $OSRM3$ is an optimal oblivious routing algorithm for $FT(m, 3)$. □

4. MULTI-PATH OBLIVIOUS ROUTING

In the previous section, it is shown that any single path routing would have at best a $(\frac{m}{2})^{\lfloor\frac{n-1}{3}\rfloor}$ oblivious performance ratio on $FT(m, n)$. This indicates that single path routing may not be effective in exploiting the fat-tree topology when the traffic pattern is uncertain since any single path algorithm may perform $(\frac{m}{2})^{\lfloor \frac{n-1}{3} \rfloor}$ times worse than the optimal routing algorithm on some traffic matrix. In this section, we show that it is almost trivial to obtain a multi-path routing algorithm with an oblivious performance ratio equal to 1. This not only indicates that the multi-path routing algorithm is optimal for any traffic matrix, but also means that multi-path routing is much more effective than single path routing in providing performance guarantees on the fat-tree topology.

The optimal oblivious multi-path routing algorithm for $FT(m, n)$, $OMRMN$, works as follows: Let the X different shortest paths between nodes i and j be $P^1_{i,j}$, $P^2_{i,j}$, ..., $P^X_{i,j}$ (From Property 3 in Section 2.2, 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, $MP_{i,j} = \{P^1_{i,j}, P^2_{i,j}, ..., P^X_{i,j}\}$ and $f^1_{i,j} = ... = f^X_{i,j} = \frac{1}{X}$.

Theorem 3: $PERF(OMRMN) = 1$.

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 $baseload(TM)$. Consider a source s in $FT(m, n)$. Let us denote $L(s) = \sum_{j \neq s} d_{s,j}$ 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 $L(s)/(\frac{m}{2})^i$ 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 (derived from Property 4). In addition, each level $n - 1 - i$ link carries traffic from at most $(\frac{m}{2})^i$ source node (Property 5). Let the nodes be s_0, s_1, ..., $s_{(\frac{m}{2})^i - 1}$ and the load on a level $n - 1 - i$ link l be $load(l)$, we have $load(l) \leq \sum_{j=0}^{(\frac{m}{2})^i - 1} \frac{L(s_j)}{(\frac{m}{2})^i}$. Since $L(s_j) \leq baseload(TM)$, $0 \leq j \leq (\frac{m}{2})^i - 1$, we have $load(l) \leq baseload(TM)$. Note that there is no restriction on the link l and traffic matrix TM. Hence, for all links and all traffic matrices, we have $load(l) \leq baseload(TM)$. Hence, $PERF(OMRMN) = 1$. □

Notice that $OMRMN$ uses all the shortest paths between two processing nodes. We will refer to it as an *unrestricted* multi-path routing scheme. It is unrestricted in that the number of paths used for each SD pair is not limited. As can be seen from theorem 3, with unrestricted multi-path routing, the optimal multi-path routing scheme that minimizes the maximum link load can be obtained. The performance of multi-path routing is much better than that of single path routing. These results argue strongly that on a large fat-tree based system area network, the unrestricted multi-path routing should be used to alleviate the network contention problem. Moreover, these results raise questions in the current system area network design that only supports a limited form of multi-path 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 multi-path routing on the fat-tree topology.

5. PERFORMANCE STUDY

We compare the performance of several known single path routing algorithms designed for the fat-tree topology. The algorithms used in the comparison include the Multiple LID algorithm ($MLID$) in [6] and the widest shortest routing (WSR) algorithm. The details about the $MLID$ algorithm can be found in [6]. 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 $(0, 1)$, $(0, 2)$, ..., $(0, N - 1)$, $(1, 0)$, $(1, 2)$, ..., $(1, N - 1)$, ..., $(N - 1, 1)$, $(N - 1, 1)$, ..., $(N - 1, N - 2)$. 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.

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

Table 1: Oblivious performance ratios of different routing algorithms

Table 1 shows the oblivious performance ratio for different routing algorithms. In this table, the worst case oblivious performance ratio for $MLID$ and WSR is obtained by analyzing the paths computed by the algorithms. This table shows (1) that our optimal single path oblivious routing algorithms provides better performance guarantees than other existing single path routing algorithms for the fat-tree topology; and (2) that multi-path routing ($OMRMN$) is significantly better than single path routing.

The next experiment is designed 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 certain way so as to guarantee worst case performance. This might hinder their performance on typical traffic demands. Notice that both $MLID$ and WSR fully spread traffic among all links in the fat-tree topology and should perform well (among single path routing schemes) for typically traffic demands. We perform simulation on $FT(32, 2)$ and $FT(16, 3)$. $FT(32, 2)$ supports 512 processing nodes and $FT(16, 3)$ supports 1024 processing nodes.

We will show the results for two types of traffic demands: random uniform traffic demands and clustered traffic demands. In a random uniform traffic demand, each entry in the traffic matrix has an equal probability to send 1 unit of traffic (or not send any traffic). 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. 1 unit of data is communicated between each pair of nodes in one group (all-to-all communication pattern within each group). For each data point, we produce 50 random instances and report the average performance ratio for the 50 instances.

The results for the random uniform traffic demands on $FT(16, 3)$ are depicted in Figure 10. The results for $FT(32, 2)$

have a very similar trend. For this type of 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 with such demands on the fat-tree topology. Notice that all of the single path routing algorithms are designed to be optimal (with a performance ratio of 1) when each of the SD pairs has 1 unit of traffic. The average performance ratio increases (very slightly) when the random traffic is more sparse.

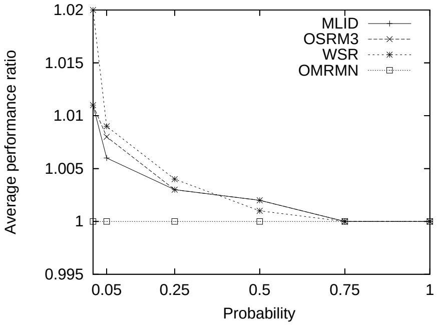


Figure 10: Random uniform traffic on $FT(16,3)$

Figure 11 shows the results for clustered traffic on $FT(32,2)$. For such traffic demands, WSR has exactly the same performance as $MLID$. As can be seen in the figure, the single path routing algorithms are not effective in dealing with such traffic demands: the average performance 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 manifests in this experiment: $OSRM2$ performs noticeable 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. $OSRM2$ guarantees that the performance ratio for any traffic pattern is no more than $\sqrt{\frac{32}{2}} = 4$. We have also performed many other experiments with different topologies and traffic demands. In all our experiments, our optimal oblivious routing algorithms are either comparable to or better than $MLID$ and WSR on average, which indicates that our optimal single path oblivious routing algorithms can provide performance guarantees without sacrificing the average case performance.

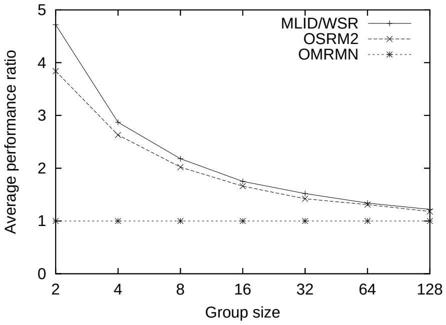


Figure 11: Clustered traffic on $FT(32,2)$

6. RELATED WORK

The research most related to this work falls into three areas: the development of system area networks, routing on fat-tree, and oblivious routing. System area networks with the off-the-shelf networking technology such as Infiniband [7] and Myrinet [13] have become more common recently. The load balance problems in such networks motivated this research. Most routing research for system area networks (see for example, [3, 4, 10, 14, 15]) has focused on developing techniques for computing and establishing routes. In [6], a routing algorithm was developed for fat-tree based Infiniband networks. It can be shown that the algorithm in [6] is not an optimal oblivious routing scheme. 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 [5, 8, 9]. 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 [1, 2, 18] due to its effectiveness in guaranteeing routing performance under uncertain and changing traffic demands in the Internet environment. In [1], it was shown that the problem of finding (unrestricted) optimal oblivious routing can be formulated as a linear programming problem. In this paper, we apply the idea of oblivious routing to the fat-tree topology and use the same performance metrics for evaluating routing schemes with uncertain traffic demands. The linear programming formulation in [1, 2] can only be used to compute optimal oblivious routing for unrestricted multi-path routing, but cannot be used to compute optimal oblivious single path routing. We also show that on the fat-tree topology, optimal oblivious routing for unrestricted multi-path routing can be obtained without solving the large linear programming formulation.

7. CONCLUSION

We study the routing performance on 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 multi-path routing is effective in balancing network load in such situations. These results extend the basic understanding of fat-tree based networks and may directly influence the design of systems with large scale fat-tree based networks such as large HPC clusters.

Acknowledgement

This work is supported in part by National Science Foundation (NSF) grants: CCF-0342540, CCF-0541096, and CCF-0551555.

8. REFERENCES

[1] D. Applegate and E. Cohen, “Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs.” *ACM SIGCOMM*, pages 313-324, 2003.

[2] D. Applegate, L. Breslau, and E. Cohen, “Coping with Network Failures: Routing Strategies for Optimal Demand Oblivious Restoration.” *ACM SIGMETRICS*, pages 270-281, 2004.

[3] A. Bermudez, R. Casado, F. J. Quiles, and J. Duato, "Use of Provisional Routes to Speed-up Change Assimilation in Infiniband Netrworks," *Proc. 2004 IEEE International Workshop on Communication Architecture for Clusters* (CAC'04), April 2004.

[4] A. Bermudez, R. Casado, F. J. Quiles, and J. Duato, "Fast Routing Computation on Infiniband Networks," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 17, No. 3, pp 215-226, March 2006.

[5] R. I. Greenberg and C. E. Lerserson, "Ramdonzied Routing on Fat-trees." In *26th Annual IEEE Symposium on Foundations of Computer Science*, pages 241-249, Oct. 1985.

[6] X. Lin, Y. Chung, and T. Huang, "A Multiple LID Routing Scheme for Fat-Tree-Based Infiniband Networks." *Proceedings of the 18th IEEE International Parallel and Distributed Processing Symposium* (IPDPS'04), p. 11a, Sana Fe, NM, April 2004.

[7] Infiniband^{TM} Trade Association, *Infiniband ^{TM} Architecture Specification*, Release 1.2, October 2004.

[8] C. E. Leiserson, "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing." *IEEE Transactions on Computers*, 34(10)892-901, October 1985.

[9] C. E. Leiserson, Z. S. Abuhamdeh, D. C. Douglas, C. R. Feynman, M. N. Ganmukhi, J. V. Hill, W. D. Hillis, B. C. Kuszmaul, M. A. St. Pierre, D. S. Wells, M. C. Wong-Chan, S-W. Yang, and R. Zak, "The network architecture of the Connection Machine CM-5." *Journal of Parallel and Distributed Computing*, 33(2):145–158, Mar 1996.

[10] P. Lopez, J. Flich, and J. Duato, "Deadlock-Free Routing in Infiniband through Destination Renaming," *Proc. 2001 International Conference on Parallel Processing* (ICPP), Sept. 2001.

[11] J.C. Martinez, J. Flich, A. Robles, P. Lopez, and J. Duato, "Supporting Fully Adaptive Routing in Infiniband Networks." *Proceedings of the 17th IEEE International Parallel and Distributed Processing Symposium* (IPDPS'03), p44a, Nice, France, April 2003.

[12] Mellanox Technologies, "Infiniband in the Enterprise Data Center." *White Paper*, 2006. Available at http://www.mellanox.com/pdf/whitepapers /scaling10gbsclusters.pdf.

[13] Myricom home page, http://www.myri.com.

[14] J. C. Sancho, A. Robles, and J. Duato, "Effective Strategy to Computing Forwarding Tables for Infiniband Networks," *Proc. International Conference on Parallel Processing* (ICPP), Sept. 2001.

[15] J. C. Sancho, A. Robles, and J. Duato, "Effective Methodology for Deadlock-Free Minimal Routing in Infiniband Networks," *Proc. International Conference on Parallel Processing* (ICPP), 2002.

[16] Top 500 supercomputer sites. http://www.top500.org

[17] M. Valerio, L. Moser, and P. Melliar-Smith, "Recursively Scalable Fat-trees as Interconnect Networks." *Proceedings of the 13th IEEE International Phoenix Conference on Computers and Communications*, pages 40-46, 1994.

[18] H. Wang, H. Xie, L. Qiu, Y.R. Yang, Y. Zhang, and A. Greenberg, "COPE: Traffic Engineering in Dynamic Networks." *ACM SIGCOMM*, 2006.

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 $s_1 = s_2$, $A = S^A_{s_1}$. Similar logic can be used to show that if $d_1 = d_2$, $A = D^A_{d_1}$. In both case, $|A| \leq L(A) \times LS(A) = LS(A)$.

Let us now prove that if $s_1 = s_2$, $A = S^A_{s_1}$. Assume that $A \neq S^A_{s_1}$, 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; otherwise, $d = d_1$ and $\{(s_1, d_2), (s, d)\}$ is a node disjoint subset of A. Hence, $L(A) \geq 2$, which contradicts the fact that $L(A) = 1$.

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

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

Next, we will show that $L(A - D^A_{d_1}) = k$. From Lemma 2, $L(A - D^A_{d_1}) \geq L(A) - L(D^A_{d_1}) = k + 1 - 1 = k$. Since $L(A) = k + 1 \geq L(A - D^A_{d_1})$, to show that $L(A - D^A_{d_1}) = k$, we only need to show that $L(A - D^A_{d_1}) \neq k + 1$. We prove this by contradiction. Assume that $L(A - D^A_{d_1}) = k + 1$. Let $C = \{(s'_1, d'_1), (s'_2, d'_2), ..., (s'_{k+1}, d'_{k+1})\}$ be a largest node disjoint subset of $A - D^A_{d_1}$. We have $s_1 \in SRC(C)$ (otherwise, $C \bigcup \{(s_1, d_1)\}$ is a node disjoint subset of A and $L(A) \geq k + 2$). Similarly, $s \in SRC(C)$. Let us assume that $s_1 = s'_1$ and $s = s'_2$. d'_2 must be in $DST(B) - \{d_1\}$ (otherwise, $(s, d'_2) \bigcup B$ is node disjoint and B is not the largest node disjoint subset). Similarly, d'_1 must be in $DST(B) - \{d_1\}$. Let $d_{k+1} = d'_1$ and $d_2 = d'_2$. We have $s_2 \in SRC(C)$ (otherwise, $C - \{(s'_2, d'_2)\} + \{(s_2, d_2), (s, d_1)\}$ 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 $SRC(C)$ and then finding that destination node d' such that $(s_i, d') \in C$ belongs to $DST(B)$) can be repeated. Once the process cannot continue, one can construct a node disjoint subset of A whose size is $k+2$ similar to the cases for $d'_2 \in B$ and $s_2 \in C$. 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, $L(A - D^A_{d_1}) = k$.

By the induction hypothesis,

$$|A - D^A_{d_1}| \leq L(A - D^A_{d_1}) \times LS(A - D^A_{d_1}) \leq k \times LS(A)$$

$$|D^A_{d_1}| \leq L(D^A_{d_1}) \times LS(D^A_{d_1}) \leq LS(A).$$

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