Load-Balanced Birkhoff-von Neumann Switches and Fat-Tree Networks

Hung-Shih Chueh, Ching-Min Lien, Cheng-Shang Chang, Jay Cheng, and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
E-mail: d929607@oz.nthu.edu.tw; cmlien@ee.nthu.edu.tw;
cschang@ee.nthu.edu.tw; jcheng@ee.nthu.edu.tw; lds@cs.nthu.edu.tw;

***Abstract*—Fat-tree networks have been widely used in the field of Network-on-Chip. One of the key issues in a fat-tree network is that the degree of a node has to be increased rapidly from the bottom of the tree to the root. As such, the complexity of implementing the switches near the root could be extremely high, and this poses a serious scalability issue. To cope with the scalability issue in fat-tree networks, many previous works require changing the tree topology and adding buffers in nodes. Unlike the existing arts, we adopt a different approach that can still maintain the original tree topology without adding any buffers in internal nodes. Our key idea is to explore various nice features of the load-balanced Birkhoff-von Neumann switches. Such switches have been shown to achieve 100% throughput for all admissible traffic and have comparable delay performance to the ideal output-buffered switch when traffic is heavy and bursty. We show that the implementation complexity can be greatly reduced if a fat-tree network is only required to realize a set of N permutations needed for the $N \times N$ load-balanced Birkhoff-von Neumann switches. For this, we first derive a lower bound on the required degree for each node in a fat-tree network. By using the uniform mapping property of the bit-reverse permutation, we show that there exists a set of N permutations that achieve the lower bound.**

***Index Terms*—Fat-tree networks, Load-balanced Birkhoff-von Neumann switches.**

I. Introduction

Fat-tree networks [9] (and many variants of them) have been widely used in the field of Network-on-Chip (NoC) [12], [4], [3], [8], [11]. As suggested by its name, a fat-tree network is constructed from a tree, where each node in the tree is a switch and each leaf is an input/ouput port. To accommodate the traffic multiplexed into the tree, the degree (or capacity in [9] or radix in [8]) of a node (in terms of the number of links connected to the node) has to be increased rapidly from the bottom of the tree to the root. In particular, if a fat-tree network with N input/output ports is required to be *nonblocking*, i.e., all the $N!$ permutations can be realized by this fat-free network, then the degree of a node has to be increased *exponentially* from the bottom of the tree to the root. This poses a serious scalability issue in designing and connecting the switches near the root when the number of input/output ports N is very large.

On the other hand, the load-balanced Birkhoff-von Neumann switches (see e.g., [1], [6], [7], [2], [5]) have received a lot of attention recently as they are much more *scalable* than other existing switch architectures in the literature. Regarding the throughput and the delay performance of the load-balanced Birkhoff-von Neumann switches, it has been shown that they achieve 100% throughput for all admissible traffic and have comparable delay performance to the ideal output-buffered switch when the traffic is heavy and bursty [1]. The operation of a generic load-balanced Birkhoff-von Neumann switch consists of two stages (see Figure 1): the first stage performs *load balancing* that converts any admissible (and nonuniform) traffic into the uniform traffic, and the second stage performs *switching* for the uniform traffic. One of the most salient features of load-balanced Birkhoff-von Neumann switches is that the connection patterns for the switch fabrics in both stages in Figure 1 are *deterministic* and *periodic*. As such, there is no need to find matchings as required in most input-buffered switches, and there are no computational overheads in load-balanced Birkhoff-von Neumann switches. For an load-balanced Birkhoff-von Neumann switch with N input/output ports, the switch fabrics in both stages only need to realize a set of N $N \times N$ permutation matrices $P_1, P_2, \ldots, P_N$ that satisfy

$$P_1 + P_2 + \cdots + P_N = \mathbf{e}, \tag{1}$$

where $\mathbf{e}$ is the $N \times N$ matrix with all its elements being 1. As these two stages use the same set of permutation matrices, one can further fold these two stages by using a single $N \times N$ switch fabric that realizes the N permutation matrices in (1).

There are several existing approaches for coping with the scalability issue of fat-tree networks. Most of them require

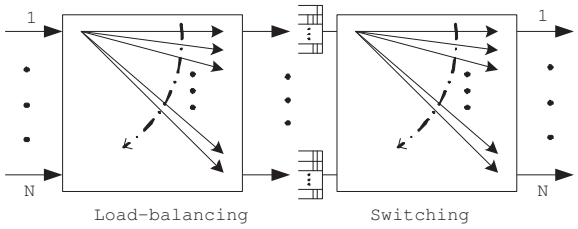

Fig. 1. A generic load-balanced Birkhoff-von Neumann switch in [1].

changing the tree topology and adding buffers in nodes, e.g., generalized fat trees [12], Dragonfly [8], and Unidirectional Load-Balanced Multistage Interconnection Networks [3]. In this paper, we adopt a different approach that can still maintain the original tree topology without adding any buffers in internal nodes. The advantage of keeping the tree topology includes simple routing (as there is a unique self avoiding path from a node to another node in a tree) and easy layout in a chip (as it is a planar graph). Also, buffers in internal nodes are costly to implement in a chip as they usually require a fabrication process different from that for combinational logic circuits. Our key idea is to explore various nice features of the load-balanced Birkhoff-von Neumann switches. Specifically, we can operate a fat-tree network as a folded load-balanced Birkhoff-von Neumann switch that consists of the two phases: the load balancing phase (that converts the incoming traffic into the uniform traffic) and the switching phase (that performs switching for the uniform traffic). By so doing, we can then explore the possibility of reducing the implementation complexity of the fat-tree network as it is now only required to realize a set of N $N \times N$ permutation matrices $P_1, P_2, \ldots, P_N$ that satisfy (1) (instead of all the $N!$ permutations). As the load-balanced Birkhoff-von Neumann switches have comparable delay performance to the ideal output-buffered switch for heavy and bursty traffic, one might not suffer too much performance degradation from implementing N such permutations in a fat-tree network. On the other hand, the gain in reducing implementation complexity (in terms of the degrees of the nodes) is substantial as will be shown later in the paper.

There are many choices of permutation matrices $P_1, P_2, \ldots, P_N$ that have been proposed in the literature for implementing $N \times N$ load-balanced Birkhoff-von Neumann switches, e.g., rotators and symmetric TDM switches in [10]. Not every choice can lead to reduction of the degrees of nodes. In Sec. II, we first show a lower bound on the degree for each switch in a fat-tree network that is capable of implementing any set of N permutations satisfying the condition in (1). The lower bound is derived based on averaging the traffic flow needed to go through a link. Unfortunately, both rotators and symmetric TDM switches require the degrees that are substantially higher than those from the lower bound.

In this paper, we propose a new set of permutation matrices $P_1, P_2, \ldots, P_N$ that not only satisfy the condition in (1) but also achieve the lower bound. The idea is based on the bit-reverse permutation previously proposed by Wu and Feng [13]. One key property of the bit-reverse permutation is the *uniform mapping property* that maps a set of inputs in a subtree uniformly to the outputs in other subtrees. We show that any permutation that satisfies the uniform mapping property can be realized by a fat-tree network with the degrees specified by the lower bound. The bit-reverse permutation and its variants obtained by circular shifts all have the uniform mapping property and thus can be realized by a fat-tree network with the degrees specified by the lower bound.

This paper is organized as follows. In Sec. II, we introduce the fat-tree network and prove a lower bound on the degree for the fat-tree network to realize any set of permutations needed for the load-balanced Birkhoff-von Neumann switches. In Sec. III, we introduce the bit-reverse permutation and prove that any permutation that satisfies the uniform mapping property can be realized by a fat-tree network with the degrees specified by the lower bound. The paper is then concluded in Sec. IV.

II. Fat-tree networks

A fat-tree network, first proposed in [9], is a switching network constructed from a complete binary tree. To explain how a fat-tree network works, we first consider a complete binary tree with 2^n leaves (see Figure 2 for a complete binary tree with 16 leaves). In such a binary tree, there are $n+1$ levels, indexed from $0, 1, \ldots, n$. The root is the only node at level 0 and a node is at level j if it is a child of a node at level $j-1$. Index the root as node $(0,0)$, and index recursively the two children of node (j,k) as nodes $(j+1, 2k)$ and $(j+1, 2k+1)$ for $0 \le k \le 2^j - 1$ and $0 \le j \le n-1$. Clearly, there are 2^j nodes at level j and we will also call node (j,k) as the k^{th} node at level j in this paper. Note that the 2^n leaves are simply nodes $(n,0), \ldots, (n, 2^n - 1)$ and we will simply call node (n,x) as leaf x (by omitting the index of level n).

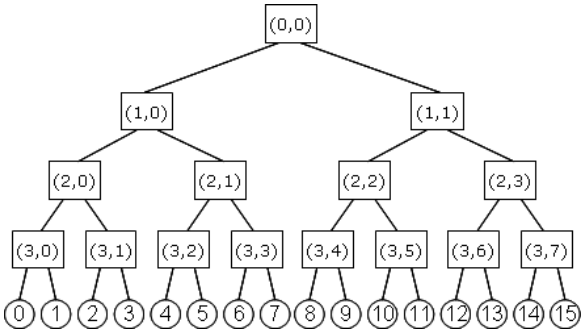

Fig. 2. A complete binary tree with 16 leaves.

For $0 \le j \le n$ and $0 \le k \le 2^j - 1$, we now define the k^{th} subtree at the j^{th} level as the subtree constructed by node (j,k) and all its descendants. Such a subtree is called subtree $T(j,k)$. Let $S(j,k)$ be the set of all the leaves in subtree $T(j,k)$, namely,

$$S(j,k) = \{x | k \cdot 2^{n-j} \le x \le (k+1)2^{n-j} - 1\} \tag{2}$$

for each $0 \le k \le 2^j - 1$, $0 \le j \le n$. Thus, the total number of leaves in subtree $T(j,k)$ is

$$|S(j,k)| = 2^{n-j}. \tag{3}$$

For example, for the complete binary tree with 16 leaves in Figure 2, we have $S(3,2) = \{4,5\}$, $S(3,3) = \{6,7\}$ and $S(2,1) = S(3,2) \cup S(3,3) = \{4,5,6,7\}$.

On the other hand, for each integer $0 \le x \le 2^n - 1$, the n-tuple $(I_n(x), I_{n-1}(x), \ldots, I_1(x))$ is called the binary

representation of x if

$$x = \sum_{m=1}^{n} I_m(x) 2^{m-1}, \tag{4}$$

where $I_m(x)$ is the m^{th} least significant bit of x. With the binary representation, for fixed j and k, the set $S(j,k)$ can be alternatively represented by

$$S(j,k) = \{x | 0 \le x \le 2^n - 1, \\ \text{and } I_{n-j+m}(x) = I_m(k) \text{ for } 1 \le m \le j\}. \tag{5}$$

That is, subtree $T(j,k)$ contains all the leaves of which the first j most significant bits are the same as the last j least significant bits of k for each $0 \le k \le 2^j - 1$ (note that the first $n-j$ most significant bits of k are simply 0).

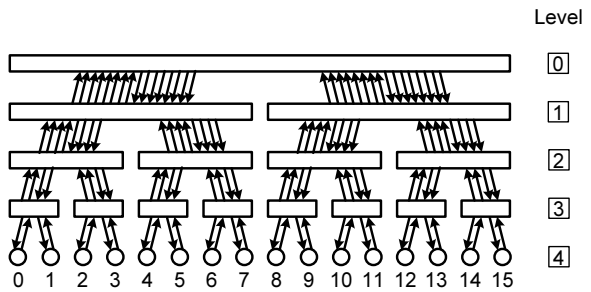

Fig. 3. A (nonblocking) fat-tree network with 16 input/output ports.

Now we show how one constructs a $2^n \times 2^n$ fat-tree network (with 2^n input/output ports) by using the binary tree with 2^n leaves. For this, we view each leaf as both an input port and an output port of a $2^n \times 2^n$ switching network and every other node as a (nonblocking) switch (see Figure 3 for a fat-tree network with 16 input/ouptut ports). The upward (resp. downward) degree of node (j,k), denoted by $C_u(j,k)$ (resp. $C_d(j,k)$), is the number of *parallel upward (resp. downward) links* that connect from node (j,k) to its parent (resp. its two children). In this paper, the downward degree of a node is *evenly* split between its two children. For such a $2^n \times 2^n$ fat-tree network to be nonblocking, it has to realize every $2^n \times 2^n$ permutation, i.e., for every $2^n \times 2^n$ permutation there is a *non-conflicting* path for every pair of input/output ports. Clearly, this requires that the upward degree of node (j,k) must not be smaller than the total number of leaves in its subtree. Thus, in order for a $2^n \times 2^n$ fat-tree network to be nonblocking, we must have

$$C_u(j,k) \ge |S(j,k)| = 2^{n-j} \tag{6}$$

for $0 \le k \le 2^j - 1$ and $1 \le j \le n$, and

$$C_d(j,k) \ge |S(j,k)| = 2^{n-j} \tag{7}$$

for $0 \le k \le 2^j - 1$ and $0 \le j \le n-1$. On the other hand, as long as both the upward degree and the downward degree satisfy the conditions in (6) and (7), there is always a non-conflicting path from an input to the root and a non-conflicting path from the root to an output. Thus, the conditions in (6) and (7) are the necessary and sufficient conditions for a fat-tree network to be nonblocking. As such, in this paper we define a $2^n \times 2^n$ *nonblocking* fat-tree network to be the fat-tree network with both the upward degree and the downward degree equal to 2^{n-j} for node (j,k). Note that the degree of a switch in a nonblocking fat-tree network grows exponentially from the leaves to the root. Moreover, the construction complexity of the switches near the root is also very high. For example, the two nodes at level 1 are $2^n \times 2^n$ switches that need to split the 2^{n-1} inputs from the root to their two children. The number of connection patterns (permutations) of doing that is

$$\frac{(2^{n-1})!}{(2^{n-2})!(2^{n-2})!} \approx 2^{2^{n-1}}.$$

This poses a serious scalability problem in designing and connecting the switches near the root.

For the scalability issue in fat-tree networks, our idea is to explore various nice features of the load-balanced Birkhoff-von Neumann switches (see e.g., [1], [6], [7], [2], [5]). Our approach is different from several existing approaches that require either changing the tree topology or adding buffers in nodes, e.g., generalized fat trees [12], Dragonfly [8], and Unidirectional Load-Balanced Multistage Interconnection Networks [3]. Instead, we still maintain the original tree topology (or an equivalent topology to the original tree topology in operation) without adding any buffers in internal nodes. The advantage of keeping the tree topology includes simple routing (as there is a unique self avoiding path from a node to another node in a tree) and easy layout in a chip (as it is a planar graph). Also, buffers in internal nodes are costly to implement in a chip as they usually require a fabrication process different from that for combinational logic circuits. As pointed out in Sec. I, one of the nice features of an $N \times N$ load-balanced Birkhoff-von Neumann switch is that we do not need to realize all the $N!$ permutations (in order to achieve 100% throughput). We only need to realize any N permutations that satisfy the condition in (1). As such, it seems possible that the degrees of the nodes in a $2^n \times 2^n$ fat-tree network for implementing the needed 2^n permutations of a $2^n \times 2^n$ load-balanced Birkhoff-von Neumann switch can be greatly reduced. In the following, we first show lower bounds for the requirements of both the upward and downward degrees.

Lemma 1 *Suppose that $P_1, \ldots, P_{2^n}$ are $2^n \times 2^n$ permutation matrices such that the sum of these 2^n matrices is a $2^n \times 2^n$ matrix with all its elements being 1. If a $2^n \times 2^n$ fat-tree network is capable of realizing these 2^n permutation matrices, then the upward (resp. downward) degree of node (j,k) is lower bounded by $\lceil 2^{n-j}(1 - 1/2^j) \rceil$ (resp. $2\lceil 2^{n-j-1}(1 - 1/2^{j+1}) \rceil$), $0 \le k \le 2^j - 1$ and $1 \le j \le n$ (resp. $0 \le j \le n-1$).*

Proof. We only prove the lower bound for the upward degree of node (j,k). The proof for the lower bound of the downward degree of node (j,k) is similar.

Our idea for the proof is by *averaging*. We consider the usual discrete-time setting and partition time into time slots so

that a fixed size packet can be transmitted over a link within a time slot. In such a discrete-time setting, the upward degree of node (j,k) is simply the total number of packets that can be transmitted to its parent in a time slot.

Consider a frame of 2^n time slots, indexed from 1 to 2^n. In the i^{th} time slot, we realize permutation matrix P_i and transmit a packet from every input port. Since the sum of these 2^n matrices is a $2^n \times 2^n$ matrix with all its elements being 1, there is exactly one packet transmitted from every input to every output in this frame of 2^n time slots. In particular, for every leaf in the subtree $T(j,k)$, it sends a packet to every leaf that is *not* in the subtree $T(j,k)$ in that frame, and every one of such packets has to go through an upward link of node (j,k). As there are 2^{n-j} leaves in the subtree $T(j,k)$, the total number of packets that go through the upward links of node (j,k) in that frame of 2^n time slots is $2^{n-j}(2^n - 2^{n-j})$. On average, there are $2^{n-j}(2^n - 2^{n-j})/2^n$ packets going through the upward links of node (j,k) per time slot. As the upward degree of node (j,k) must not be smaller than the average number of packets going through its upward links, we then have

$$C_u(j,k) \geq \left\lceil \frac{2^{2n-j} - 2^{2n-2j}}{2^n} \right\rceil = \lceil 2^{n-j} - 2^{n-2j} \rceil. \qquad (8)$$

■

Note that for $n/2 < j \leq n$, we have $2^{n-2j} < 1$ and

$$\lceil 2^{n-j}(1 - 1/2^j) \rceil = 2^{n-j}.$$

Thus, for the lower half of the tree (when $n/2 < j \leq n$), both the upward degree and the downward degree of node (j,k) are the same as those of nonblocking fat-tree networks in (6) and (7). In view of this, the reduction of the degrees of the nodes in Lemma 1 is only possible for the upper half of the tree (when $j \leq \lfloor n/2 \rfloor$). But, as we mentioned before, the scalability problem is mainly due to the design of the switches in the upper half of the tree. Thus, reducing the degrees for switches in the upper half of the tree is still of high importance.

There are many choices of permutation matrices $P_1, P_2, \ldots, P_{2^n}$ that have been proposed for the load-balanced Birkhoff-von Neumann switches in the literature, e.g., rotators and symmetric TDM switches in [10]. We note that both rotators and symmetric TDM switches do not lead to reduction of the degrees for switches in the upper half of the tree. The question is then whether it is possible to find a set of 2^n $2^n \times 2^n$ permutations that satisfy the condition in Lemma 1 and achieve the lower bounds in Lemma 1. This problem will be answered in the next section.

III. Bit-Reverse Permutation

In the previous section, we derive lower bounds for both the upward degree and the downward degree of a fat-tree network to realize the needed permutations for a load-balanced Birkhoff-von Neumann switch. In this section, we will show that these lower bounds are indeed achievable by using the bit-reverse permutation introduced by Wu and Feng in [13].

First, we introduce some notations that will be used for describing the bit-reverse permutation. Let Z_N be the set of integers $\{0, 1, \ldots, N-1\}$. For a permutation σ on Z_N, we use P_σ to denote the $N \times N$ permutation matrix that corresponds to the permutation σ. Also, for any set $S \subset Z_N$, $\sigma(S)$ is defined as the range of S, i.e.,

$$\sigma(S) = \{y \in Z_N | y = \sigma(x) \text{ for some } x \in S\}.$$

Let σ_c be the circular shift permutation on Z_N, namely, $\sigma_c(x) = (x+1) \bmod N$ for all $0 \leq x \leq N-1$. Also, let $\sigma_c^i = \sigma_c$ for $i = 1$ and $\sigma_c^i = \sigma_c^{i-1} \circ \sigma_c$ for $i > 1$. Clearly, $\sigma_c^i(x) = (x+i) \bmod N$ for all $0 \leq x \leq N-1$ and it is the permutation that performs circular shift i times. Note that $P_{\sigma_c^N}$ is simply the $N \times N$ identity matrix. One can easily see that

$$P_{\sigma_c^1} + P_{\sigma_c^2} + \ldots + P_{\sigma_c^N} = \mathbf{e},$$

and the condition in (1) is satisfied. Moreover, for any permutation σ on Z_N, if we define $\sigma_i = \sigma_c^i \circ \sigma$ for $1 \leq i \leq N$, then we have

$$P_{\sigma_i} = (P_{\sigma_c^i}) P_\sigma$$

and

$$\begin{aligned} &P_{\sigma_1} + P_{\sigma_2} + \ldots + P_{\sigma_N} \\ &= (P_{\sigma_c^1} + P_{\sigma_c^2} + \ldots + P_{\sigma_c^N}) P_\sigma \\ &= \mathbf{e} P_\sigma = \mathbf{e}. \end{aligned}$$

Thus, the condition in (1) is also satisfied. This implies that we can use an arbitrary permutation σ on Z_N and all its circular shifts to construct the needed permutations for an $N \times N$ load-balanced Birkhoff-von Neumann switch. However, to use them in a fat-tree network, we need to choose the permutation σ carefully and in fact, as we will show later, the bit-reverse permutation proposed by Wu and Feng in [13] is right for the job.

Definition 2 *Let $N = 2^n$ and $(I_n(x), I_{n-1}(x), \ldots, I_1(x))$ be the binary representation of $x \in Z_N$, where $I_m(x)$ is the m^{th} least significant bit of x. The bit-reverse permutation π on Z_N is the permutation with*

$$I_m(\pi(x)) = I_{n+1-m}(x) \qquad (9)$$

for all $1 \leq m \leq n$. Moreover, we define the permutation $\pi_i = \sigma_c^i \circ \pi$ for $1 \leq i \leq N$, i.e.,

$$\pi_i(x) = \sigma_c^i(\pi(x)) = (\pi(x) + i) \bmod N \qquad (10)$$

for all $0 \leq x \leq N-1$.

Note that

$$\pi_N(x) = (\pi(x) + N) \bmod N = \pi(x)$$

and thus π_N is simply the bit-reverse permutation π.

One important feature of the bit-reverse permutation is the uniform mapping property defined below.

Definition 3 *(Uniform Mapping Property) Let $N = 2^n$. Recall from (5) that the set $S(j,k)$ contains all the 2^{n-j} elements of which the first j most significant bits are the same as the last j least significant bits of k for each $0 \le k \le 2^j - 1$. Then, for fixed $0 \le j \le n$, a permutation σ on Z_N is said to satisfy the uniform mapping property if*

$$|\sigma(S(j,k)) \cap S(n-j,\ell)| = 1 \tag{11}$$

for all $0 \le k \le 2^j - 1$ and $0 \le \ell \le 2^{n-j} - 1$.

When one realizes a permutation σ in a fat-tree network, $\sigma(S(j,k))$ is the set of outputs with their inputs in the subtree $T(j,k)$. As there are 2^{n-j} leaves in the subtree $T(j,k)$ and 2^{n-j} subtrees at the $(n-j)^{th}$ level for $0 \le j \le n$, the uniform mapping property implies that all the 2^{n-j} leaves in the subtree $T(j,k)$ are mapped *uniformly* to the 2^{n-j} subtrees at the $(n-j)^{th}$ level.

According to (5) and (9), we have that

$$\begin{aligned}\pi(S(j,k)) = \{y | 0 \le y \le 2^n - 1,\\ \text{and } I_m(y) = I_{j+1-m}(k)\ , \text{for } 1 \le m \le j\}.\end{aligned} \tag{12}$$

On the other hand, we have from (5) that

$$\begin{aligned}S(n-j,\ell) = \{y | 0 \le y \le 2^n - 1,\\ \text{and } I_{j+m}(y) = I_m(\ell)\ , \text{for } 1 \le m \le n-j\}.\end{aligned} \tag{13}$$

Thus, we see from (12) and (13) that there is only element in $\pi(S(j,k)) \cap S(n-j,\ell)$ and it is uniquely determined by the binary representation

$$(I_{n-j}(\ell), \ldots, I_1(\ell), I_1(k), \ldots, I_j(k)).$$

This shows that the bit-reverse permutation π satisfies the uniform mapping property. Moreover, we show in the following lemma that the permutation π_i defined in Definition 2 also satisfies the uniform mapping property.

Lemma 4 *Let $N = 2^n$. The permutation π_i satisfies the uniform mapping property for $1 \le i \le N-1$.*

Proof. In view of (12), we note that for each $y \in \pi(S(j,k))$, y can be represented as

$$y = q \cdot 2^j + \sum_{m=1}^{j} I_{j+1-m}(k) 2^{m-1} = q \cdot 2^j + k' \tag{14}$$

for some $0 \le q \le 2^{n-j} - 1$, where

$$k' = \sum_{m=1}^{j} I_{j+1-m}(k) 2^{m-1}$$

is a function of k. In other words,

$$\begin{aligned}\pi(S(j,k)) = \{y | y = q \cdot 2^j + k',\\ \text{for some } 0 \le q \le 2^{n-j} - 1\}.\end{aligned} \tag{15}$$

As $\pi_i(x) = (\pi(x) + i) \bmod 2^n$, it then follows that

$$\begin{aligned}\pi_i(S(j,k)) = \{y | y = (q \cdot 2^j + k' + i) \bmod 2^n,\\ \text{for some } 0 \le q \le 2^{n-j} - 1\}.\end{aligned} \tag{16}$$

Now let $\alpha = ((k'+i) \bmod 2^j)$ and $q' = (k' + i - \alpha)/2^j$. In other words, q' is the quotient and α is the reminder of $k'+i$ divided by 2^j. Thus, $k' + i = q' \cdot 2^j + \alpha$. Then

$$\begin{aligned}\pi_i(S(j,k)) &= \{y | y = ((q+q') \cdot 2^j + \alpha) \bmod 2^n,\\ &\qquad \text{for some } 0 \le q \le 2^{n-j} - 1\}\\ &= \{y | y = q \cdot 2^j + \alpha,\\ &\qquad \text{for some } 0 \le q \le 2^{n-j} - 1\}.\end{aligned} \tag{17}$$

This then implies

$$\begin{aligned}\pi_i(S(j,k)) = \{y | 0 \le y \le 2^n - 1,\\ \text{and } I_m(y) = I_m(\alpha)\ , \forall\, 1 \le m \le j\},\end{aligned} \tag{18}$$

where $\alpha = (k'+i) \bmod 2^j$. Thus, we see from (13) and (18) that there is only element in $\pi_i(S(j,k)) \cap S(n-j,\ell)$ and it is uniquely determined by the binary representation

$$(I_{n-j}(\ell), \ldots, I_1(\ell), I_j(\alpha), \ldots, I_1(\alpha)).$$

■

In the following lemma, we show that any permutation that satisfies the uniform mapping property can be realized by a fat-tree network with the degrees specified by the lower bounds in Lemma 1.

Lemma 5 *Let $N = 2^n$. Consider a $2^n \times 2^n$ fat-tree network. Suppose that the upward degree of node (j,k) is $\lceil 2^{n-j}(1 - 1/2^j) \rceil$ and the downward degree of node (j,k) is $2\lceil 2^{n-j-1}(1 - 1/2^{j+1}) \rceil$. Then any permutation σ that satisfies the uniform mapping property can be realized in such a fat-tree network.*

Proof. For this, we need to show there is a *non-conflicting* path from every pair of input/output ports when realizing a permutation σ that satisfies the uniform mapping property. We consider the *shortest path routing* for connecting a path from an input x to its output $\sigma(x)$. The shortest path between x and $\sigma(x)$ is to first go up the tree from x to the first common ancestor of x and $\sigma(x)$ and then go down the tree to $\sigma(x)$. We only show the proof for the upward links. The proof for the downward links is similar.

Case 1. There is no conflict in the upward links of node (j,k) in the *lower* half of the tree when $n/2 < j \le n$.

The argument is the same as that for nonblocking fat-tree networks. Note that the total number of paths that go through the upward links of node (j,k) is bounded above by $|S(j,k)|$, i.e., the total number of leaves in the subtree $T(j,k)$. As the upward degree of node (j,k) is $\lceil 2^{n-j}(1 - 1/2^j) \rceil$, we then have

$$|S(j,k)| = 2^{n-j} = \left\lceil 2^{n-j} \left(1 - \frac{1}{2^j}\right) \right\rceil, \tag{19}$$

where we use the fact that $2^{n-2j} < 1$ for $n/2 < j \le n$. Thus, there is no conflict in the upward links for node (j,k) when $n/2 < j \le n$.

Case 2. There is no conflict in the upward links of node (j,k) in the *upper* half of the tree when $j \le \lfloor n/2 \rfloor$.

Under the shortest path routing, a path needs to go through one of the upward links of node (j,k) if its input x is a leaf of the subtree $T(j,k)$ and its output $\sigma(x)$ is a leaf outside the subtree $T(j,k)$. The leaves that are outside the subtree $T(j,k)$ can be written as $\cup_{k_1\neq k}S(j,k_1)$. Thus, the total number of paths that go through the upward links of node (j,k) is

$$\begin{aligned}&|\sigma(S(j,k))\cap(\cup_{k_1\neq k}S(j,k_1))|\\&=|\cup_{k_1\neq k}(\sigma(S(j,k))\cap S(j,k_1))|\\&=\sum_{k_1\neq k}|\sigma(S(j,k))\cap S(j,k_1)|.\end{aligned}\tag{20}$$

Since $j\leq\lfloor n/2\rfloor$, we can further decompose the leaves in subtree $T(j,k_1)$ into the union of the leaves in subtrees $T(n-j,\ell)$, $\ell=2^{n-2j}k_1,2^{n-2j}k_1+1,\ldots,2^{n-2j}(k_1+1)-1$. Specifically, we have

$$S(j,k_1)=\bigcup_{\ell=2^{n-2j}k_1}^{2^{n-2j}(k_1+1)-1}S(n-j,\ell).$$

Thus,

$$\begin{aligned}&|\sigma(S(j,k))\cap S(j,k_1)|\\&=\sum_{\ell=2^{n-2j}k_1}^{2^{n-2j}(k_1+1)-1}|\sigma(S(j,k))\cap S(n-j,\ell)|\\&=2^{n-2j},\end{aligned}\tag{21}$$

where we use the uniform mapping property of σ in the last identity.

Using this in (20) yields

$$\begin{aligned}&|\sigma(S(j,k))\cap(\cup_{k_1\neq k}S(j,k_1))|\\&=\sum_{k_1\neq k}|\sigma(S(j,k))\cap S(j,k_1)|=(2^j-1)2^{n-2j}.\end{aligned}\tag{22}$$

Since $2^{n-j}-2^{n-2j}$ is an integer for each $1\leq j\leq\lfloor n/2\rfloor$, we also have

$$(2^j-1)2^{n-2j}=2^{n-j}\left(1-\frac{1}{2^j}\right)=\left\lceil 2^{n-j}\left(1-\frac{1}{2^j}\right)\right\rceil,\tag{23}$$

where the last quantity is exactly the upward degree of node (j,k). In view of (22) and (23), we then conclude that there is no conflict in the upward links of node (j,k) when $j\leq\lfloor n/2\rfloor$. ■

From Lemma 4 and Lemma 5, we have the following corollary.

Corollary 6 *Let $N=2^n$. Consider a $2^n\times 2^n$ fat-tree network. Suppose that the upward degree of node (j,k) is $\lceil 2^{n-j}(1-1/2^j)\rceil$ and the downward degree of node (j,k) is $2\lceil 2^{n-j-1}(1-1/2^{j+1})\rceil$. Then the 2^n permutations $\pi_1,\ldots,\pi_{2^n}$ can be realized in such a fat-tree network.*

IV. Conclusion

To cope with the scalability issue in designing and connecting the switches near the roots of fat-tree networks, we explore various nice features of the load-balanced Birkhoff-von Neumann switches. Specifically, we solved the problem of implementing the needed permutations of the load-balanced Birkhoff-von Neumann switches in fat-tree networks. For this, we derived a lower bound on the degree for each switch in a fat-tree network that is capable of realizing the needed permutations for load-balanced Birkhoff-von Neumann switches. By using the uniform mapping property of the bit-reverse permutation, we found a new set of permutations that achieve the lower bound.

References

[1] C. -S. Chang, D. -S. Lee and Y. -S. Jou, "Load balanced Birkhoff-von Neumann switches, part I: one-stage buffering, " *Computer Communicaitons,* vol. 25, pp. 611-622, 2002.

[2] H. J. Chao, J. Song, N. S. Artan, G. Hu, and S. Jiang, "Byte-focal: a practical load-balanced switch," *IEEE High Performance Switching and Routing*, 2005.

[3] C. Gómez, F. Gilabert, M.E. Gómez, P. López, and J. Duato, "Beyond Fat-tree: unidirectional load-balanced multistage interconnection network," *IEEE Computer Architecture Letters*, vol. 7, no. 2, pp. 49-52, 2008.

[4] H. Hossain, Md. M. Akbar and Md. M. Islam, "Extended-butterfly fat tree interconnection (EFTI) architecture for network on chip," *IEEE Pacific Rim Conference on Communications, Computers and signal Processing,* pp. 613-616, Aug. 2005.

[5] J. -J. Jaramillo, F. Milan and R. Srikant, "Padded Frames: A Novel Algorithms for Stable Scheduling in Load-Balanced Switches, " *IEEE/ACM Transactions on Networking,* vol. 16, no. 5, pp. 1212-1225, Oct. 2008.

[6] I. Keslassy, S. -T. Chung, K. Yu, D. Miller, M. Horowitz, O. Sloggard, and N. McKeown, "Scaling Internet Routers Using Optics, " *ACM SIGCOMM 2003*, Karlsruhe, Germany, Sep. 2003.

[7] I. Keslassy, S. -T. Chung, and N. McKeown, "A load-balanced switch with an arbitrary number of linecards," *Proceedings of IEEE INFOCOM*, 2004.

[8] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," *ISCA '08 Proceedings of the 35th Annual International Symposium on Computer Architecture*, pp. 77-88, 2008.

[9] C. E. Leiserson, "Fat-Trees: Universal networks for hardware-efficient supercomputing, " *IEEE Transaction on Computers,* vol. c-34, no. 10, pp. 892-901, Oct. 1985.

[10] C.-M. Lien, C.-S. Chang, J. Cheng, D.-S. Lee and J.-T. Liao, "Twister networks and their applications to load-balanced switches," *Proceedings of IEEE INFOCOM 2010.*

[11] H. Matsutani, M. Koibuchi, Y. Yamada, D. F. Hsu and H. Amano, "Fat H-Tree: A cost-efficient tree-based on-chip network, " *IEEE Transaction on Parallel and Distributed Systems,* vol. 20, no. 8, Aug. 2009.

[12] S. R. Ohring, M. Ibel, S. K. Das, and M. J. Kumar, "On generalized fat trees," *Proceedings of 9th International Parallel Processing Symposium*, Santa Barbara, CA, USA, pp. 37-44, 1995.

[13] C. -L. Wu and S. -Y. Feng, "The Reverse-Exchange Interconnection Network, " *IEEE Transaction on Computers,* vol. c-29, no. 9, pp. 801-811, Sep. 1980.