Dichotomy Slot Allocation: A QoS Guaranteed Scheduling Algorithm for Input-Queued Switches

Jingjing Zhang, *Student Member, IEEE*, Nirwan Ansari, *Fellow, IEEE*, Yaohui Jin, *Associate Member, IEEE*, and Weisheng Hu, *Member, IEEE*

***Abstract*—With the rapid increase of real-time applications, jitter, delay, and throughput have become the three important QoS criteria in the scheduling of input-queued (IQ) switches with virtual output queues (VOQ). In this paper, we propose a novel frame-based scheduling algorithm, referred to as *dichotomy slot allocation* (DSA), to achieve high throughput, bounded delay, and bounded jitter. DSA possesses three major characteristics. First, DSA schedules the switch on a per-VOQ basis, and strives to provision QoS guarantees for each traffic stream corresponding to each VOQ. Second, DSA allocates the exact amount of slots to schedule the actual traffic of each VOQ, and decides the time slot for each packet. Third, DSA equally allocates the slots in a frame to each port pair, and then adjusts the scheduling to meet the actual traffic demand. The adjustment process employs a specially designed order named Dichotomy Order to guarantee QoS. Both analysis and simulation results demonstrate that DSA achieves good performance in jitter and throughput.**

***Index Terms*—Dichotomy slot allocation, input-queued switch, jitter, throughput, QoS.**

I. Introduction

RECENTLY, input-queued (IQ) crossbar switch fabric with virtual output queues (VOQ) has attracted broad attention in designing high speed routers and switches [2]–[4]. In such a switch fabric, the arriving packets are queued in front of the inputs according to their destination ports, referred to as virtual output queues. As compared to conventional IQ or output queuing (OQ) schemes, IQ with VOQ has its superior advantages. On one hand, the IQ with VOQ scheme overcomes the poor scalability problem of output queuing. On the other hand, it avoids the performance degradation due to the well known head-of-line (HOL) blocking problem existing in input queuing schemes. With the rapidly increasing real-time applications, scheduling the IQ switch with VOQ needs to guarantee quality of service (QoS), e.g., low jitter, fairness, high throughput, and low delay [1], [5]–[8].

Many scheduling schemes have been proposed to provision low jitter and fairness in packet scheduling, including weighted fair queuing (WFQ) [9], worst-case weighted fair queuing (WF^{2}Q) [10], deficit found robin (DRR) [11], smoothed round robin (SRR) [12], Huffman fair queuing (HuFQ) [13], call admission control with a grouping architecture [14], and adaptive distributed fair scheduling [15]. Recently, low-jitter scheduling has received a lot of attention in the research community of switches and wireless mesh networks. Lee [16] has shown that the smoothness of the scheduling of a decomposed permutation matrix is bounded by the entropy of BV decomposition, and satisfies the Kraft's inequality in Clos networks. He *et al.* [17] proposed an approach of rate-based smoothed switching, and designed a smoothed buffered crossbar CICQ switch. Szymanski [18] proposed a technique to optimize throughput and jitter of a nonuniform multichannel wireless mesh network, in which the scheduling problem can be transformed into an IQ switch scheduling problem.

For IQ switches with VOQ, one low-jitter and fairness scheduling referred to as, Greedy Low-jitter Decomposition (GLJD), was proposed by Keslassy *et al.* [8]. Similar to the well known Birkhoff Von-Neumann (BV) decomposition algorithm [19], GLJD decomposes the rate matrix into a sum of permutation matrices. Different from BV decomposition, GLJD adds an additional constraint of nonoverlapping positions of nonzero entries in the decomposed permutation matrices. The intuition here is the relatively easier controllability of jitter when a port pair appears in only one permutation matrix. After decomposition, one weighted round robin (WRR) scheme was adopted to schedule these decomposed matrices [20]. Another related work was conducted by Mohanty and Bhuya [21]. They proposed the Binary Matrix Decomposition (BMD) algorithm and applied SRR [12] to schedule these decomposed matrices. For the sake of convenience, the two algorithms are denoted as $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$ in the rest of this paper.

Note that port pairs are sometimes granted with more allocations than their requirements in $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$. These surplus allocations may entail unexpected high jitter and unfairness even if the decomposed permutation matrices are ideally scheduled.

For instance, in a frame period of eight time-slots, say, $1, 2, \ldots, 8$, the traffic demand of a 2×2 switch is described by$\mathbf{T}$, where $T_{i,j}$ is the required slots for transmitting traffic from input i to output j.

$$T_{2\times 2}=\begin{pmatrix}4 & 4\\ 4 & 2\end{pmatrix}$$

Each of port pairs (1, 1), (1, 2), and (2, 1) has four packets to be transmitted while port pair (2, 2) has only two packets to be

Manuscript received November 03, 2008; revised July 31, 2009; accepted September 22, 2009. Date of publication February 08, 2010; date of current version April 07, 2010.

J. Zhang and N. Ansari are with the Advanced Networking Laboratory, New Jersey Institute of Technology, University Heights, Newark, NJ 07032 USA (e-mail: nirwan.ansari@njit.edu).

Y. Jin and W. Hu are with the State Key Laboratory of Advanced Optical Communication Systems and Networks, Shanghai Jiao Tong University, Shanghai 200240, China

Digital Object Identifier 10.1109/JSYST.2010.2040226

transmitted. Both BV and GLJD decomposition will generate the following weighted sum of permutation matrices:

$$\begin{bmatrix}4 & 4\\4 & 2\end{bmatrix} \prec 4\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} + 4\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}.$$

Symbol "$\prec$" means that each entry in the right is no less than the corresponding entry in the left. Then, the two decomposed permutation matrices are scheduled with equal interval of two slots—see the equation shown at the bottom of the page. Note that port pair (2, 2), which only needs two slots, was granted four slots, 1, 3, 5, 7. The usual scheme of (2, 2) is to choose the earliest two slots, slots 1 and 3. In this case, the interval between these two allocations is two slots, while the interval between the second allocation and the allocation in the next frame is no less than six slots. This introduces high jitter. On the other hand, 100% of the traffic of port pair (2, 2) is scheduled during the first four slots while only 50% of the traffic of port pairs (1, 1), (1, 2), and (2, 1) is scheduled in the same time period. This causes unfairness. However, choosing slots 1 and 5 will produce lower jitter and better fairness performance.

In this paper, we present a novel scheduling scheme referred to as *Dichotomy Slot Allocation* (DSA) to guarantee quality of service for each port pair. Following the assumption made in [4], [8], [19], we also focus on frame-based scheduling in this paper. In frame-based scheduling, the delay is bounded by the frame length [4]. So, we do not focus on delay performance. Moreover, fairness requirement is generally consistent with jitter. Low jitter requires allocations with near-equal intervals. It discourages one port pair greedily occupying the resources in continuous time slots, thereby avoiding short-term unfairness to other pairs in some degree. We particularly focus on jitter and throughput when designing DSA.

Specifically, DSA first equally allocates slots to each port pair in the assumption of uniform traffic. These allocations are referred to as pre-allocations. Then, based on the real nonuniform traffic demands, DSA adjusts this ideal scheduling port pair by port pair. For port pairs with traffic less than the pre-allocations, DSA selects proper allocations from pre-allocated slots and release those unselected allocations; for port pairs with traffic exceeding the pre-allocations, DSA allocates these newly released slots to them. The adjustment process employs a specially designed order referred to as *Dichotomy Order* for the purpose of QoS. Besides switches, DSA can be tailored to many other networks, such as slotted WDM [20].

The rest of the paper is organized as follows: Section II describes the system model. Section III presents our proposed algorithm along with theoretical analysis. Section IV provides simulation results, and concluding remarks are included in Section V.

II. System Model

Consider an $N \times N$ input-output switch fabric. Data are scheduled from each input to outputs with the fixed-size cells (or packets) in each time slot by a central arbiter. We make the similar assumption [8] that we have some prior knowledge of the rates required for each input/output pair. The cell arriving rate is described by the matrix $\mathbf{\Lambda}$, where the element $\lambda_{i,j}$ is the arrival rate of pair (i,j) from input i to output j. In the frame-based scheduling, L consecutive time slots are grouped into one frame. The traffic demand in one frame is denoted by matrix $\mathbf{T}$, where the element $T_{i,j}$ is the number of the cells queued in input i which need to be scheduled to output j.

Mneimneh and Siu [22] analyzed the relationship between the arriving rate and the traffic demand under three types of traffic models. For simplicity, we consider the constant traffic scenario, in which the traffic conforms to the following:

$$\mathbf{T} = L \cdot \mathbf{\Lambda} \tag{1}$$

Regarding to the dynamic scenarios, if the arriving rate is changed, traffic demand $\mathbf{T}$ can be derived by the current Internet architecture, e.g., bandwidth broker or MPLS signaling. If the traffic arrives in a burst mode, it can be smoothed in the input queues as the frame size L increases. Previously, Lee and Lam [23] indicated that such a quasi-static statistic also exists in the 3-stage Clos ATM switches.

The frame-based switch scheduling problem is formulated as follows: given the frame size L as well as a traffic demand $\mathbf{T}$, find a scheduling table to achieve high throughput and low jitter. Generally, the scheduling table is changed in every frame as the traffic demand changes. However, in the case of constant traffic demand, the same scheduling table is repeated frame by frame. We use a three dimensional Boolean matrix $\mathbf{R}$ to denote the scheduling table, where $R_{i,j,l}$ is set one when slot l is allocated for switching a cell of the pair (i,j). The basic switching constraints that a port can not send to or receive from more than one port simultaneously are mathematically formulated as follows:

$$\sum_{j=1}^{N} R_{i,j,l} \le 1, \forall i, l \tag{2}$$

$$\sum_{i=1}^{N} R_{i,j,l} \le 1, \forall j, l \tag{3}$$

$$R_{i,j,l} \in \{0,1\}, \forall i, j, l. \tag{4}$$

Different from the model established in [8] and [15], redundant scheduling is not permitted in this paper, it must satisfy

$$\sum_{l=1}^{L} R_{i,j,l} \le T_{i,j}, \forall i, j. \tag{5}$$

$$\begin{array}{lcccccccc} & \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\Bigg\| & \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\Bigg\| & \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\Bigg\| & \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\Bigg\| & \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\Bigg\| & \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\Bigg\| & \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\Bigg\| & \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\Bigg\| \\ \text{slot}: & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8. \end{array}$$

$$\mathbf{T} = \begin{bmatrix} 3 & 1 & 2 & 2 \\ 1 & 4 & 2 & 3 \\ 1 & 2 & 2 & 1 \\ 3 & 1 & 1 & 0 \end{bmatrix}$$

S =

input \ slot	1	2	3	4	5	6	7	8	9	10
1		1	4	3	2	1	4	3	1	
2	4		2	1	4	3	2	4	3	2
3	3	2		4	3	2	1			
4	1			2	1		3	1		

Requests for some port pairs may not be fully satisfied, thus leading to smaller throughput. To measure the scheduling loss, we now introduce the following definition.

Definition 1: The throughput of a scheduling table $\mathbf{R}$ for traffic demand $\mathbf{T}$ is

$$\text{throughput} = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{L} R_{i,j,l} / \sum_{i=1}^{N}\sum_{j=1}^{N} T_{i,j} \tag{6}$$

There are different ways to quantify jitter. In this paper, we use the measure of rate jitter, which is defined as the difference between the minimal and maximal inter-arrival time [7]. Denote $F_{i,j,k}$ as the arrival time of the kth cell in output j from input i

$$F_{i,j,k} = \begin{cases} t, & \text{if } R_{i,j,t} = 1 \quad \text{and} \sum_{l=1}^{t} R_{i,j,l} = k, \forall i,j \\ 0, & \text{others.} \end{cases} \tag{7}$$

$D_{i,j,k}$ denotes the inter-arrival time in output port j between the kth cell and the $(k+1)$th cell from input i. Note that the scheduling table repeats frame by frame for the constant traffic demand

$$D_{i,j,k} = (F_{i,j,(k \bmod T_{i,j}+1)} - F_{i,j,k}) \bmod L, \forall i,j,k. \tag{8}$$

Definition 2: The jitter of a scheduling table $\mathbf{R}$ for pair (i,j) is

$$J_{i,j} = \max_k(D_{i,j,k}) - \min_k(D_{i,j,k}), \quad \forall i,j. \tag{9}$$

The average jitter is

$$\text{jitter} = \sum_{i=1}^{N}\sum_{j=1}^{N} J_{i,j}/N^2. \tag{10}$$

Succinctly, $\mathbf{R}$ can be represented by an equivalent two-dimensional matrix $\mathbf{S}$ in which $S_{i,l}$ is the destination output of input i in slot l

$$S_{i,l} = j, \quad \text{if } \exists R_{i,j,l} = 1, \forall i,l.$$

We refer to $\mathbf{S}$ as the *scheduling table* for $\mathbf{T}$.

Below is an example of traffic demand matrix $\mathbf{T}$ and its scheduling table $\mathbf{S}$, where $N = 4$ and $L = 10$,

Pair (2, 2) requires four cells to be transmitted, but only three cells are scheduled and one cell is lost. For pair (1, 1), $F_{1,1,1} = 2, F_{1,1,2} = 6, F_{1,1,3} = 9; D_{1,1,1} = 4, D_{1,1,2} = 3, D_{1,1,3} = 3$; so $J_{1,1} = 1$. For this example, the throughput is 96.55% and the average jitter is 1 slot.

III. Dichotomy Order

Dichotomy Order, denoted by O, is the key idea behind our proposed DSA algorithm. We first show the generation process of O.

Dichotomy Order is derived from the idea of dichotomy. It can be considered as an operation order on a circular array. Given a circular array $\mathbf{B}$ with length M, we determine the operation positions $O_1, O_2, \ldots, O_M$ in the array one by one. The determining rule can be generalized as: every new operation position O_{k+1} on $\mathbf{B}$ is the middle position of the longest section of $\mathbf{B}$ divided by its formerly determined elements $O_1, O_2, \ldots, O_k$. Specifically, for the first determined k operation positions $O_1, \ldots, O_k$, assume O_i and $O_j (1 \le i,j \le k)$ are two consecutive positions. Owing to the circular property of array $\mathbf{B}$, the interval between O_i and O_j equals to $\mathrm{mod}(|O_i - O_j| + M, M + 1)$. Denote Ψ_k as the set containing intervals between any two consecutive operation positions in $\{O_1, \ldots, O_k\}$, H_k as the largest element in set Ψ_k, α_k and $\beta_k (\alpha_k, \beta_k \in \{O_1, \ldots, O_k\})$ as two consecutive operation positions with intervals equaling to $H_k (H_k = \mathrm{mod}(|\beta_k - \alpha_k| + M, M + 1))$. Then, the next operation position is selected as $(\alpha_k + \beta_k)/2$. Interval H_k is divided into two intervals: $|\beta_k - (\alpha_k + \beta_k)/2|$ and $|\alpha_k - (\alpha_k + \beta_k)/2|$. The detail generation process of O is described in Algorithm 1 below.

Algorithm 1: **Generation of Dichotomy Order**

```
// set the first element $\nu$ in the order, $\nu$ can be any position from 1 to M
$O_1 = v, \Psi_1 = \{M\}, \alpha_1 = v, \beta_1 = v, H_1 = M;$
// calculate the other elements in the order
For $k = 2, \ldots, M$
    // decide the $k$th element $O_k$ in the order
    $O_k = (\alpha_{k-1} + \beta_{k-1})/2$
    // update the interval set $\Psi_k$
    $\Psi_k = \Psi_k \backslash \{H_k\} \cup \{|\alpha_{k-1} - (\alpha_{k-1} + \beta_{k-1})/2|, |\beta_{k-1} - (\alpha_{k-1} + \beta_{k-1})/2|\}$
    // obtain the largest interval $H_k$ in set $\Psi_k$
    $H_k = \max\{x | x \in \Psi_k\}$
    update $\alpha_k$ and $\beta_k$ accordingly
End
```

Ψ_k and H_k are directly related to the jitter performance when O is employed as the scheduling order. They exhibit the following quantitative properties for O with length M. (see Lemma 1 and 2).

Lemma 1: $\forall c \in Z$,if $2^c < k \le 2^{c+1} \le M$, then we get the first equation shown at the bottom of the page.

$$\Psi_k \subseteq \begin{cases} \{\lceil M/2^c \rceil, \lfloor M/2^c \rfloor\}, k = 2^c \\ \{\lceil M/2^c \rceil, \lfloor M/2^c \rfloor, \lceil M/2^{c+1} \rceil, \lfloor M/2^{c+1} \rfloor\}, 2^c < k < 2^{c+1} \end{cases}$$

Proof: We prove this by induction. Assume $\Psi_k = \{\lfloor M/2^c \rfloor, \lceil M/2^c \rceil\}$ when $k = 2^c$. Then, the first k operation positions divide the circular array into k sections, each with length of either $\lfloor M/2^c \rfloor$ or $\lceil M/2^c \rceil$. Assume a of the k sections are with length of $\lceil M/2^c \rceil$, and $k - a$ remaining sections with length of $\lfloor M/2^c \rfloor$. After the $k = 2^c$ operations, the longest section length equals $\lceil M/2^c \rceil$ so far. Hence, the $(2^c + 1)$th operation position is selected as the middle position of one of these a sections with length of $\lceil M/2^c \rceil$, and it will further divide that section into two sections with length of $\lfloor \lceil M/2^c \rceil /2 \rfloor$ or $\lceil \lceil M/2^c \rceil /2 \rceil$. After the $(2^c + 1)$th, $(2^c + 2)$th, $\ldots, (2^c + a - 1)$th positions are determined, the longest section length will remain as $\lceil M/2^c \rceil$. After the determination of the $(2^c + a)$th position, the longest section length is changed to $\lfloor M/2^c \rfloor$. Then, the next $k - a$ operation positions, i.e., the $(2^c + a + 1)$th, $(2^c + a + 2)$th, $\ldots, 2^{c+1}$th position, will be selected as the middle position of one of these $k - a$ intervals with length of $\lfloor M/2^c \rfloor$, and each of them will divide the section with length of $\lfloor M/2^c \rfloor$ into two sections with length of $\lfloor \lfloor M/2^c \rfloor /2 \rfloor$ or $\lceil \lfloor M/2^c \rfloor /2 \rceil$. Therefore, after the determination of the first 2^{c+1} positions, the circular array will be divided into sections with length of $\lfloor \lceil M/2^c \rceil /2 \rfloor, \lceil \lceil M/2^c \rceil /2 \rceil, \lfloor \lfloor M/2^c \rfloor /2 \rfloor$, or $\lceil \lfloor M/2^c \rfloor /2 \rceil$. Since $\lfloor \lceil M/2^c \rceil /2 \rfloor, \lceil \lceil M/2^c \rceil /2 \rceil, \lfloor \lfloor M/2^c \rfloor /2 \rfloor$, and $\lceil \lfloor M/2^c \rfloor /2 \rceil$ equals to either $\lceil M/2^{c+1} \rceil$ or $\lfloor M/2^{c+1} \rfloor, \Psi_{2^{c+1}} \subseteq \{\lceil M/2^{c+1} \rceil, \lfloor M/2^{c+1} \rfloor\}$, and $\Psi_k = \{\lceil M/2^c \rceil, \lfloor M/2^c \rfloor, \lceil M/2^{c+1} \rceil, \lfloor M/2^{c+1} \rfloor\}$, if $2^{c+1} > k > 2^c$. In summary, we get the second equation shown at the bottom of the page.

Lemma 2: $\forall c \in \mathrm{Z}$, if $2^c < k \leq 2^{c+1} \leq M$, then

$$H_k = \begin{cases} 1 \text{ or } 0, & k = 2^c \\ \lceil M/2^{c+1} \rceil \textit{ or } \lfloor M/2^{c+1} \rfloor, & 2^c < k < 2^{c+1} \end{cases}$$

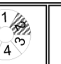
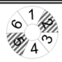
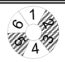
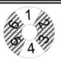
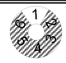
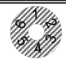
Proof: According to Lemma 1, when $k = 2^c, \Psi_k = \{\lfloor M/2^c \rfloor, \lceil M/2^c \rceil\}$. So, $H_k = 1$ or 0. Among the 2^c sections divided by the first 2^c positions, assume a sections are with length of $\lceil M/2^c \rceil$ and $2^c - a$ intervals with length of $\lfloor M/2^c \rfloor$. Then, the $(2^c + 1)$th, $\ldots, (2^c + a)$th positions in O further divide/partition sections with length of $\lceil M/2^c \rceil$ into sections with length of $\lfloor \lceil M/2^c \rceil /2 \rfloor$ or $\lceil \lceil M/2^c \rceil /2 \rceil$; the $(2^c + a + 1)$th, $\ldots, 2^{c+1}$th positions in O further divide sections with length of $\lfloor M/2^c \rfloor$ into sections with length of $\lfloor \lfloor M/2^c \rfloor /2 \rfloor$ or $\lceil \lfloor M/2^c \rfloor /2 \rceil$. Then

$$H_k = \begin{cases} \lceil M/2^c \rceil - \lfloor \lceil M/2^c \rceil /2 \rfloor, 2^c < k < 2^c + a \\ \lfloor M/2^c \rfloor - \lfloor \lceil M/2^c \rceil /2 \rfloor, k = 2^c + a \\ \lfloor M/2^c \rfloor - \lfloor \lfloor M/2^c \rfloor /2 \rfloor, 2^c + a < k < 2^{c+1}. \end{cases}$$

Corresponding to whether $\lceil M/2^c \rceil$ and $\lfloor M/2^c \rfloor$ are odd or even, H_k is either $\lceil M/2^{c+1} \rceil$ or $\lfloor M/2^{c+1} \rfloor$.

Assume the first element O_1 in order O is 2. Taking $M = 6$ for example, the generation process of$O$ is illustrated in Table I.

TABLE I
AN EXAMPLE FOR GENERATING O WITH $\mathrm{M} = 6, \mathrm{j} = 2$

					
O_1=2	O_2=5	O_3=3	O_4=6	O_5=4	O_6=1
Ψ_1={6}	Ψ_2={3}	Ψ_3={3,2,1}	Ψ_4={2,1}	Ψ_1={2,1}	Ψ_6={1}
H_1 =0	H_2 =0	H_3 =2	H_4 =1	H_5 =1	H_6 =0

IV. DICHOTOMY SLOT ALLOCATION

This section presents the proposed DSA algorithm, which employs O to smoothly schedule each port pair with small cell loss.

Intuitively, when $\mathbf{T}$ represents uniform traffic, the traffic can be easily scheduled without jitter (i.e., jitter free). However, the traffic is nonuniform in most practical scenarios. DSA rewrites the nonuniform traffic demand $\mathbf{T}$ into $\mathbf{T} = \mathbf{T}^0 - \mathbf{T}^- + \mathbf{T}^+$, where $\mathbf{T}^0_{i,j} = K, \forall i, j$. The determination of K will be discussed next

$$\mathbf{T}^-_{i,j} = \begin{cases} K - \mathrm{T}_{i,j}, & \text{if } \mathrm{T}_{i,j} < K \\ 0, & \text{else} \end{cases}$$
$$\mathbf{T}^+_{i,j} = \begin{cases} \mathrm{T}_{i,j} - K, & \text{if } \mathrm{T}_{i,j} > K \\ 0, & \text{else} \end{cases}.$$

$\mathbf{T}^-$ denotes deficit traffic (uniform traffic K minus the actual traffic) while $\mathbf{T}^+$ denotes the surplus traffic (actual traffic minus K). The basic process of DSA is as follows.

Step 1) Determine K and construct a jitter-free scheduling $\mathbf{S}^0$ for $\mathbf{T}^0$. A port pair's allocated slots in $\mathbf{S}^0$ are referred to as the pair's pre-allocations.

Step 2) For each port pair (i, j) with $\mathbf{T}^-_{i,j} > 0$, select proper pre-allocations in $\mathbf{S}^0$ and release the other unselected $\mathbf{T}^-_{i,j}$ pre-allocations of this pair.

Step 3) For each port pair (i, j) with $\mathbf{T}^+_{i,j} > 0$, allocate proper released slots for the extra $\mathbf{T}^+_{i,j}$ traffic demand.

Details of these three steps are further deliberated next.

A. Determining and Scheduling $\mathbf{T}^0$

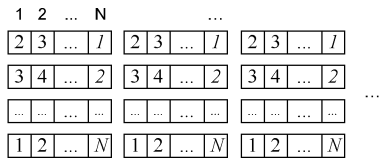
We first determine $\mathbf{T}^0$ for the given frame length L and switch size N, and construct a jitter-free schedule for $\mathbf{T}^0$.

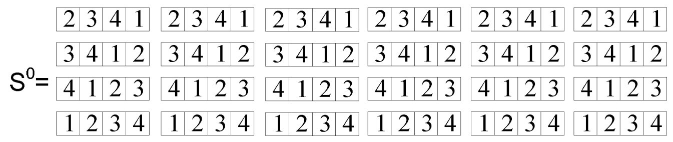
$\mathbf{T}^0$ depends on the frame length L. Given frame length L, Properties 1 and 2 show the maximum admissible uniform traffic with jitter-free scheduling.

Property 1: The maximum admissible uniform traffic rate is $\Lambda = 1/N * U$, where $\mathbf{U}$ is the all one matrix.

Proof: The condition of admissible traffic is $\max\{\max_k \sum_i \lambda_{i,k}, \max_k \sum_j \lambda_{k,j}\} \leq 1$. So, $\Lambda = 1/N * U$ is the maximum admissible uniform traffic rate.

$$\Psi_k \subseteq \begin{cases} \{\lceil M/2^c \rceil, \lfloor M/2^c \rfloor\}, k = 2^c \\ \{\lceil M/2^c \rceil, \lfloor M/2^c \rfloor, \lceil M/2^{c+1} \rceil, \lfloor M/2^{c+1} \rfloor\}, 2^c < k < 2^{c+1} \end{cases}$$





Property 2: Given the frame length L, if L is an integer multiple of N, there exists jitter-free schedules for uniform traffic $\mathbf{T}^0 = (L/N) \cdot U$, which is the maximum admissible uniform traffic.

Proof: One way to construct a scheduling table is as follows: From slot 1 to N, employ a round robin search, and then repeat such scheduling every N slots. For example, It is seen that jitter of every port pair is zero.

DSA intentionally selects the frame size L to be an integer multiple of N and set $\mathbf{T}^0 = (L/N) \cdot U$. After having determined $\mathbf{T}^0$, $\mathbf{T}^-$ and $\mathbf{T}^+$ can be derived.

The following is an example for $L = 24, N = 4$:

$$\overset{\mathrm{T}}{\begin{bmatrix} 6 & 5 & 5 & 6 \\ 7 & 6 & 4 & 5 \\ 8 & 6 & 6 & 3 \\ 2 & 5 & 6 & 10 \end{bmatrix}} = 6 \overset{\mathrm{T}^0}{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}$$

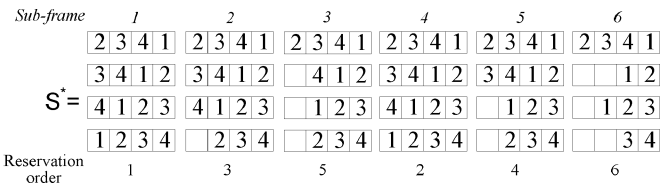
$$- \overset{\mathrm{T}^-}{\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 3 \\ 4 & 1 & 0 & 0 \end{bmatrix}} + \overset{\mathrm{T}^+}{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}}.$$

Below is one of the jitter-free schedules $\mathbf{S}^0$ for $\mathbf{T}^0$:

To facilitate analysis, we name N consecutive time slots as a *subframe* (a frame is composed of $K = L/N$ subframes). The scheduling in one subframe has a total of $N!(N-1)! \ldots 1!$ kinds of combinations.

B. Scheduling $\mathbf{T}^0 - \mathrm{T}^-$

For traffic $\mathbf{T}_{i,j} < T^0_{i,j}$, pre-allocations in the first step exceed the actually desired. Therefore, we select $\mathbf{T}_{i,j}$ slots from $\mathbf{T}^0_{i,j}$ pre-allocations and release the other $\mathbf{T}^-_{i,j}$ pre-allocations. DSA employs dichotomy order O as the selection order. The benefits are twofold. On one hand, O helps achieve relatively low jitter.



On the other hand, to ensure small cell loss, O aligns as many released allocations of different port pairs as possible in the same slot in order to accept more traffic of $\mathbf{T}^+$. Denote $\Phi(i,j)$ as the set of the selected subframes for pair (i,j). O is generated with the length $M = K$ and the initial position $O_1 = 1$. The selection process for $\boldsymbol{T}^{\mathbf{0}} - \boldsymbol{T}^-$ is illustrated by Algorithm 2 below.

Algorithm 2: Selection

```
% generate $O$ with $M = K, O_1 = 1$
For $T^-_{i,j} > 0$
    $\Phi(i,j) = \{O_1, \ldots, O_{T_{i,j}}\}$
    Select $(i,j)$'s pre-allocations in the subframes $\Phi(i,j)$
    and release others
End
```

Let $\mathbf{S}^*$ be the schedule obtained after the selection process. Then, $\mathbf{S}^*$ exhibits the following jitter properties.

Theorem 1: The jitter of $\mathbf{S}^*$ for pair (i,j) is as shown in the first equation at the bottom of the page.

Proof: When $T_{i,j} \geq K$, $J_{i,j} = 0$ since all its pre-allocations are selected. When $T_{i,j} < K$, we get the second equation shown at the bottom of the page, with slots according to Lemmas 1 and 2.

Computational Complexity: The dichotomy order only depends on K. It is the same for all VOQs, irrespective of the actual traffic $T_{i,j}$. Hence, for VOQ j at input i, the complexity of selection is $O(1)$. The worst case running time of the selection for all VOQs is $O(N^2)$.

Next, we will illustrate an example of selection of the $\mathbf{T}$ mentioned earlier. When $M = 6$ and $O_1 = 1, O = [1,4,2,5,3,6]$. Below is $\mathbf{S}^*$.

The total jitter of $\mathbf{S}^*$ is 28 slots.

C. Scheduling $\mathbf{T}^+$

We shall first explain the intuition to acquire small cell loss and low jitter. After a slot is allocated to pair (i,j), it cannot be allocated to other pairs from input port i or to output port j. This will cause cell loss for these port pairs. The cell loss shall

$$J_{i,j} = \begin{cases} 0, & T_{i,j} \geq K \\ N \text{ or } 0, & T_{i,j} < K \text{ and } T_{i,j} = 2^k \\ \lceil K/2^{k+1} \rceil * N \text{ or } \lfloor K/2^{k+1} \rfloor * N, & T_{i,j} < K \text{ and } 2^k < T_{i,j} < 2^{k+1} \end{cases}$$

$$J_{i,j} = \begin{cases} N \text{ or } 0, & T_{i,j} = 2^k \\ \lceil K/2^{k+1} \rceil N \text{ or } \lfloor K/2^{k+1} \rfloor N, & 2^k < T_{i,j} < 2^{k+1} \end{cases}$$

be minimized. On the other hand, each pair with $T^+_{i,j} > 0$ was allocated K slots with one subframe interval. Re-allocated slots decrease the scheduling intervals, and break the constant scheduling interval. Avoiding rapid decrease of scheduling intervals helps achieve lower jitter. Again, we use O to select re-allocated slots. Note that O here is different from O used in selection (Algorithm 2). Since the re-allocated slots are between two consecutive pre-allocations with constant interval N slots, O is generated with the length $M = N$ and $O_1 = l$, where slot l is the pre-allocated slots for port pair (i, j). The re-allocation algorithm is summarized as Algorithm 3 below.

Algorithm 3: Re-Allocation for $\mathbf{T}^+$

```
$m = 2$..........................................................................(a)
While $T^+_{i,j} > 0$ and $m \le N$
    generate $O$ with length $M = N, O_1 = l$
    $k = 1$
    While $T^+_{i,j} > 0$ and $k \le K$
        If slot $O_m$ in subframe $k$ can be allocated to $(i, j)$..........(b)
            allocate the slot to $(i, j)$
            $T^+_{i,j} = T^+_{i,j} - 1$
            $k = k + 1$
        End
    End
    $m = m + 1$
End
```

In Step (a), we set $m = 2$ since the pre-allocated slot of port pair (i, j), has already been allocated. Step (b) is to check whether the slots in O can be allocated to the port pair. If either input port i or output port j is busy in this slot, this slot cannot be allocated to (i, j).

DSA always attempts to re-allocate traffic into the slot in the middle position of the largest interval. This introduces relatively low jitter. The following theorem provides the jitter bound of the re-allocation.

Theorem 2: The jitter upper bound of port pairs with$T^+_{i,j} > 0$ is $(N - 1)$ slots in the final scheduling table $\mathbf{S}$.

Proof: Intervals are decreased after the re-allocation. The maximum interval is at most one subframe while the minimum one is at least one slot. So, the maximum jitter is N-1slots.

Computational Complexity: In Step (b) of the re-allocation, slot O_m in K subframes have to be checked. Thus, the worst case running time of the re-allocation for a pair is $O(N * K)$. To check the total of N^2 port pairs, the complexity of the re-allocation algorithm is $O(N^3 * K) = O(N^2 * L)$.

For a particular port pair, the re-allocation process does not need to check all L slots one by one to find out the suitable ones.

TABLE II
SCHEDULING $\mathbf{T}^+$

Pair	O	T<K	$\overline{\Phi(i, j')} \cap \overline{\Phi(i', j)}$	Slots
(2,1)	*3,1,4,2*	$T_{2,3} = 4, T_{4,1} = 2$	O[5,6]	9
(3,1)	*2,4,3,1*	$T_{3,4} = 3, T_{4,1} = 2$	O [4,5,6]	17, 21
(4,4)	*4,2,1,3*	$T_{4,2} = 5, T_{2,4} = 5$	O[6]	22
		$T_{4,1} = 2, T_{3,4} = 3$	O[4,5,6]	17, 9, 21

Sub-frame	*1*				*2*				*3*				*4*				*5*				*6*			
	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1
	3	4	1	2	3	4	1	2	1	4	1	2	3	4	1	2	3	4	1	2			1	2
S=	4	1	2	3	4	1	2	3		1	2	3	4	1	2	3	1	1	2	3	1	1	2	3
	1	2	3	4		2	3	4	4	2	3	4	1	2	3	4	4	2	3	4	4	4	3	4

Some of the unsuitable slots can be easily figured out. The following Lemma 2 provides a method to avoid some unnecessary checking.

Lemma 2: Let (i, j') and (i', j) be pairs with the pre-allocated slot l in a subframe, $i' \neq i, j' \neq j$; then slot l can be allocated to (i, j) only when they are in the subframes $\overline{\Phi(i, j')} \cap \overline{\Phi(i', j)}$.

Proof: (i, j') and (i', j) select pre-allocations in the subframe $\Phi(i, j')$ and $\Phi(i', j)$. Then, either input port i or output port j is busy in the subframes $\Phi(i, j')$ and $\Phi(i', j)$. Both i and j being in unselected subframes $\overline{\Phi(i, j')} \cap \overline{\Phi(i', j)}$ is the necessary condition for both of them to be idle. Thus, only if slot l is in these subframes, it may be allocated to (i, j).

When $T_{i',j} \ge K$ or $T_{i,j'} \ge K$, $\overline{\Phi(i, j')} \cap \overline{\Phi(i', j)} = \oslash$. After the selection step, $\overline{\Phi(i, j')} \cap \overline{\Phi(i', j)} = \{O_{\max\{T^*_{i,j'}, T^*_{i',j}\}}\}$.

Using the above mentioned $\mathbf{T}$ and $\mathbf{S}^*$ as an example, the re-allocation is shown in Table II. Specifically, insufficient scheduled traffic for (2,1), (3,1), and (4,4) need to be re-allocated. Note that after slot 9 is allocated to (2,1), slot 9 cannot be allocated to (3,1) any longer since output port 1 becomes busy.

The final schedule table $\mathbf{S}$ is shown below. No cell is lost. The re-allocation introduces extra jitter of 8 slots. Hence, the average jitter is 2.25 slots.

Note that, for some admissible traffic, DSA may not be able to guarantee 100% throughput as BV decomposition does. We will show the average throughput yielded by DSA in the simulation part.

In summary, DSA is composed of selection and re-allocation. As shown above, the complexity of the selection process is $O(N^2)$, and the complexity of the re-allocation process is $O(N^2 * L)$. Hence, the total complexity is $O(N^2) + O(N^2 * L) = O(N^2 * L)$.

V. Simulation Results and Discussion

In this section, we investigate the performances of DSA by simulations and compare them with that of GLJD + WRR [8]

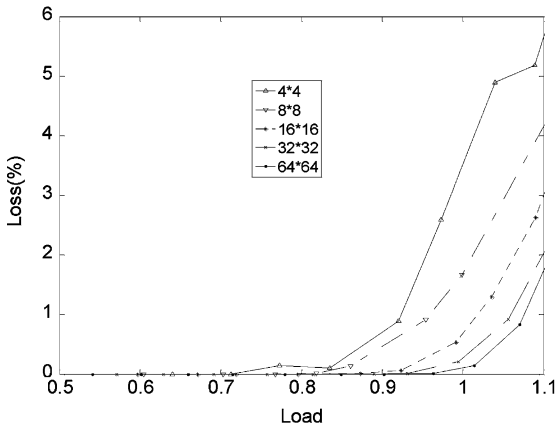


Fig. 1. Cell-loss ratio as a function of traffic load under uniform traffic scenario for various switch sizes.

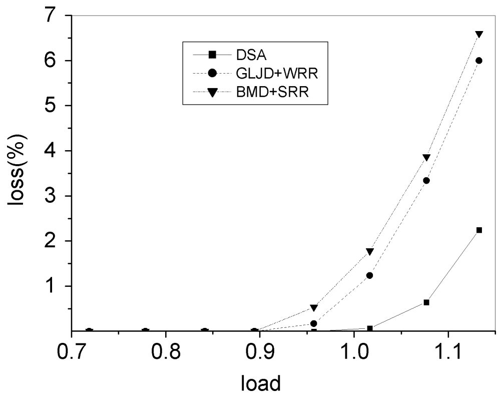


Fig. 3. Comparison of cell loss for DSA, $\mathrm{GLJD}+\mathrm{WRR}$, and $\mathrm{BMD}+\mathrm{SRR}$.

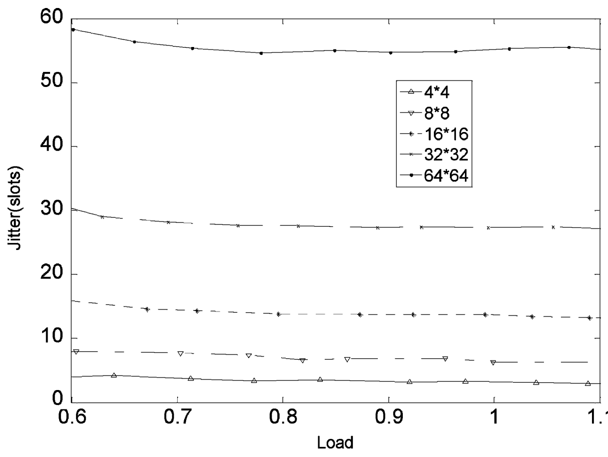


Fig. 2. Jitter as a function of traffic load under uniform traffic scenario for various switch sizes.

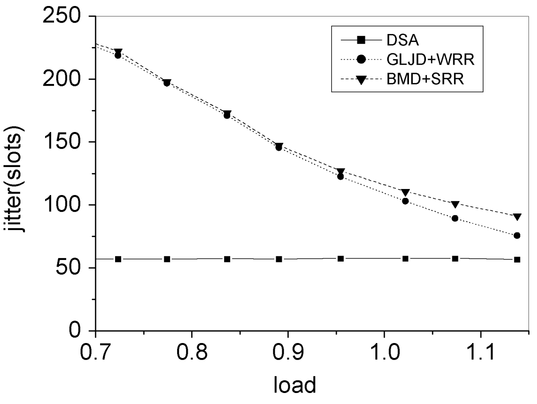


Fig. 4. Comparison of jitter for DSA, $\mathrm{GLJD}+\mathrm{WRR}$, and $\mathrm{BMD}+\mathrm{SRR}$ under uniform traffic scenario.

and $\mathrm{BMD}+\mathrm{SRR}$ [21]. The traffic models considered have destinations with uniform and nonuniform distributions.

A. *Uniform Traffic*

Figs. 1 and 2 show simulation results under uniform traffic scenario. The simulation is done for switches with various sizes, i.e., $4 \times 4, 8 \times 8, 16 \times 16, 32 \times 32, 64 \times 64$. The number of subframes K is set to be 10. The average cell arrival rate λ is randomly generated with even distribution between 0 and u, where u varies from 0 to $2/N$ for different traffic load. *load* is defined as $\max\{\max_k \sum_i \lambda_{i,k}, \max_k \sum_j \lambda_{k,j}\}$. When $\mathrm{load} \leq 1$, the arrival rate is admissible, otherwise cell loss is introduced. In the following graphs, we analyze the relationship between performance and mean load in 100 simulations.

Fig. 1 shows that the cell loss increases with increasing traffic load. For a 64×64 switch, cell loss is less than 0.1% when traffic load approaches 1. Switches with large sizes have lower cell-loss ratio than that of switches with small sizes. This is due to the fact that the increased number of released allocations in every schedule brings more opportunities for surplus traffic to be inserted. Therefore, the cell-loss ratio is reduced when the switch size increases.

Fig. 2 shows the relationship between jitter and traffic load under uniform traffic scenario for various switch sizes. When the load is small, the number of port pairs with deficit traffic is larger than those with surplus traffic. When the load becomes larger, the dominating port pairs change from those with deficit traffic to those with surplus traffic. Theorems 1 and 2 imply that jitter of port pairs with deficit traffic are generally much larger than that of port pairs with surplus traffic, except in some special cases. Therefore, average jitter decreases, though very little, with increasing traffic load (see Fig. 2). In addition, Fig. 2 shows that jitter increases with the switch size, thus conforming to Theorems 1 and 2. It is also shown that the average jitter is around or a little less than N slots, where N is the switch size.

Figs. 3 and 4 compare the performance of DSA with that of $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$. To facilitate proper comparison, we make the following two assumptions of $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$.

a) The speed up of the switch is 1. When the required bandwidth is larger than the frame size, the last allocations are lost.
b) BMD and GLJD may allocate a port pair with more slots than demand. In this case, the traffic is scheduled in the first several slots, and the last unnecessary allocations are released.

In $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$, the allocations in the last several slots are lost when its required bandwidth exceeds the frame length. So, they produce larger cell loss than DSA as

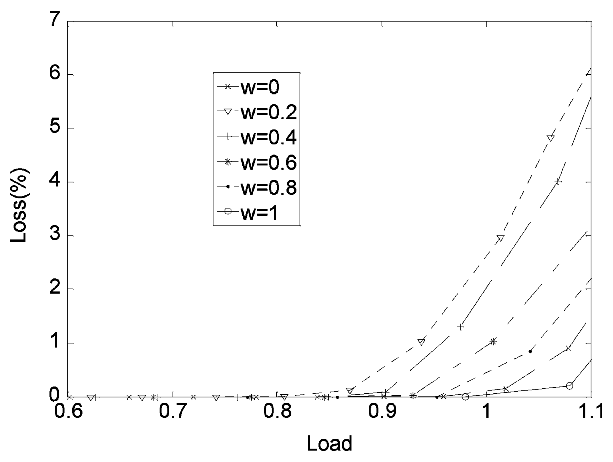


Fig. 5. Cell-loss ratio as a function of traffic load for unbalanced traffic scenario with different w.

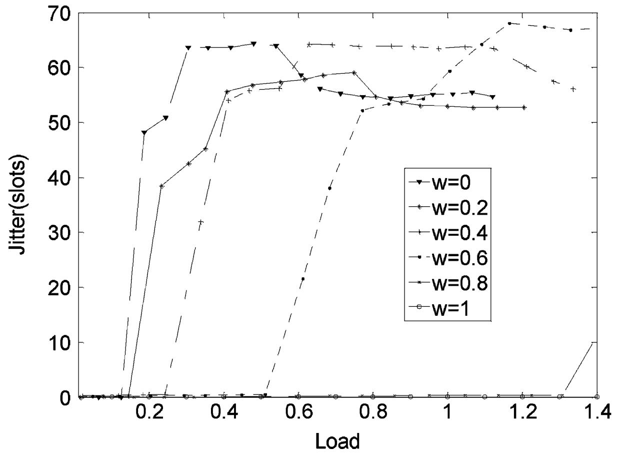


Fig. 6. Jitter as a function of traffic load for unbalanced traffic scenario with different w.

shown in Fig. 3. Although $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$ require the same bandwidth, they produce different loss ratios because of the different scheduling schemes for computing permutation matrices. It is shown that the loss ratios of $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD} + \mathrm{SRR}$ are larger than that of DSA when the load is greater than 0.9.

Fig. 4 shows that the average jitter of DSA remains almost constant with the increase of the switch load. Because of our assumptions b) of $\mathrm{GLJD} + \mathrm{WRR}$ and $\mathrm{BMD} + \mathrm{SRR}$, the allocations of some port pairs may be concentrated in the front of the schedule table, thus producing a large jitter. This phenomenon is especially evident with the small switch load. Even in the best case, $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$ cannot achieve the same performance as DSA. In fact, their jitter can be reduced by selecting allocations which introduce the least jitter. However, the complexity of the required selection scheme is about $O(N^2 * L)$. Then, the complexity is increased if $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD} + \mathrm{SRR}$ attempt to Fig. 5 shows the loss ratio for unbalanced traffic with different w. It is seen that cell loss decreases with the increase of w. Directional traffic with $w = 1$ has the smallest cell-loss ratio while uniform traffic has the largest cell-loss ratio. For directional traffic scenario, much of the traffic is directed to the directional port pairs, making other port pairs lightly loaded. Generally, directional port pairs have surplus traffic, while the other port pairs have deficit traffic. In this case, the insertion process is mainly performed for directional port pairs. For the other port pairs, the insertion process is implemented with small probability. The directional port pairs do not constitute input or output conflict. In other words, during re-allocation, re-allocated slots for one port pair will not reduce other port pairs' opportunity of being re-allocated. So, cell loss is small. achieve better jitter performance, which is beyond the discussion of this paper.

B. Nonuniform Traffic

1) Unbalanced Traffic: The unbalanced traffic model uses a probability, w, as the fraction of input load directed to a single predetermined output, while the rest of the input load is directed to all outputs with uniform distribution. Let $p_{i,j}$ be probability that the total load for input port i is directed to output port j

$$p_{i,j} = \begin{cases} \left(w + \frac{1-w}{N}\right), \text{if } i = j \\ \frac{1-w}{N}, \text{otherwise} \end{cases} \tag{11}$$

When $w = 0$, the traffic is uniform, while it is directional when $w = 1$.

In the simulation, the switch size is 64×64, the average cell arrival rate λ for port pair $\{(i,j)\,|\,i \neq j\}$ is randomly generated with even distribution between 0 and $(1-w)*u$, and the cell arrival rate λ for port pair $\{(i,j)\,|\,i = j\}$ is randomly generated with even distribution between 0 and $(N*w+1-w)*u$. u is varied from 0 to $2/N$ to obtain different network load.

Fig. 6 shows the jitter performance under unbalanced traffic scenario with different w. In every traffic scenario, jitter is nearly zero under small load, and then increases to a high value and almost plateaus at that value. Based on Theorem 1, jitter is 0 or N slots when traffic between a port pair is 0, 1, 2 cells. This is the reason why jitter is almost zero with small load. Also, Theorem 1 indicates that jitter suddenly increases when traffic is increased beyond three cells. Therefore, the average jitter will be suddenly increased when the load is increased to a certain number, referred to as the bound. As shown in Fig. 6, the bound depends on w. The bound for switches with large w is larger than those with small w. The reason is as follows. When the traffic is approaching directionally, more loads will be exerted on the directional port pair, leaving the other port pairs lightly loaded. The majority of the port pairs will have low jitter, thus lowering the average jitter.

2) Chang's and Asymmetric Traffic: Chang's traffic model can be defined as $\lambda_{i,j} = 0$ for $i = j$, and $\lambda_{i,j} = 1/(N-1)$ otherwise. The asymmetric traffic model has different load for each port pair, such as $p_{i,(i+j)\bmod N} = a_j$, where $a_0 = 0$, $a_1 = (r-1)/(r^N - 1)$, $a_j = a_1 r^{j-1}$, $r = (100:1)^{-1/(N-2)}$, $p_{i,j}$ is the probability that the total load for input port i is directed to output port j.

Consider a 64×64 switch. The average cell arrival rate λ for port pair (i,j) is randomly generated with even distribution

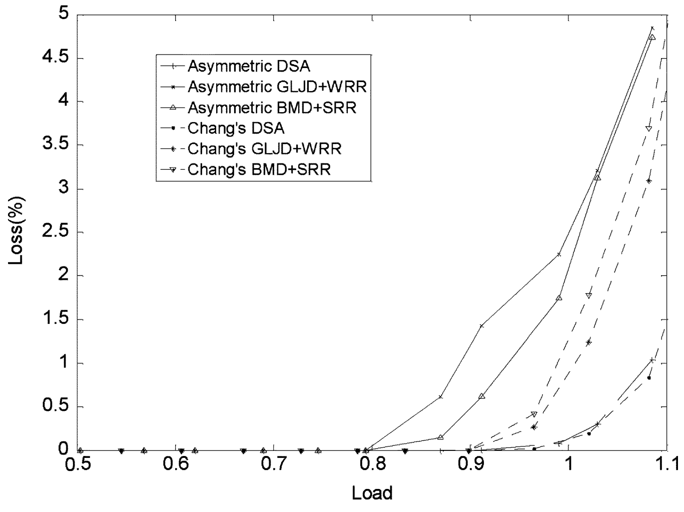


Fig. 7. Comparison of cell-loss ratio for DSA, $\mathrm{GLJD}+\mathrm{WRR}$, and $\mathrm{BMD}+\mathrm{SRR}$ under Chang's and Asymmetric traffic scenario.

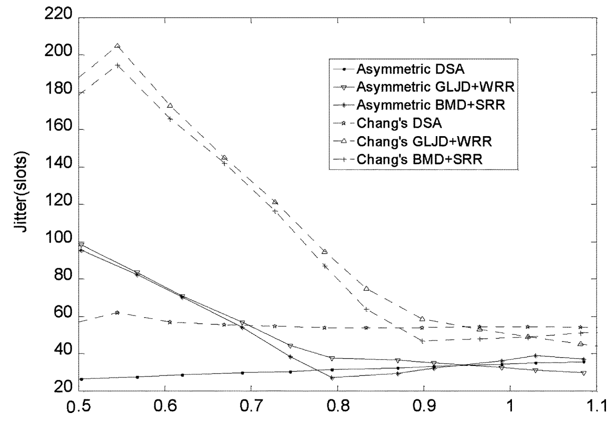


Fig. 8. Comparison of jitter for DSA, $\mathrm{GLJD}+\mathrm{WRR}$, and $\mathrm{BMD}+\mathrm{SRR}$ under Chang's and Asymmetric traffic scenario.

between 0 and a_j, and u varies from 0 to 2 to obtain different network load.

Fig. 7 shows that the cell-loss ratio of DSA is nearly zero when the load approaches 1 under both Chang's and Asymmetric traffic scenarios. The cell loss of DSA is always less than that of $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$.

Fig. 8 shows that jitter of asymmetric traffic is less than that of Chang's traffic. This is because that asymmetric traffic, as compared to Chang's traffic, is rather unbalanced with the property that some port pairs are heavily loaded while some others are lightly loaded. As shown in Fig. 6, unbalanced traffic with larger w has lower jitter performance. So, asymmetric traffic has less jitter than Chang's traffic. Fig. 8 also shows that DSA almost produces constant jitter regardless of the variation of the load while jitter of $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$ decreases with the traffic load. The three algorithms yield almost the same jitter under heavy loaded cases.

VI. Conclusion

In this paper, we have proposed a low-jitter scheduling scheme referred to as DSA. DSA first allocates slots without jitter to each port pair assuming the traffic is uniform. To accommodate nonuniform traffic, DSA adjusts the jitter-free allocation. The adjustment consists of two operations: select allocations and release the unselected ones for port pairs which need less; re-allocate released allocations to those which require more. Specifically, Dichotomy Order has been proposed to achieve low jitter and small cell loss in the adjustment process. Finally, we compare our algorithm with the two existing smooth scheduling algorithms $\mathrm{GLJD}+\mathrm{WRR}$ and $\mathrm{BMD}+\mathrm{SRR}$. Simulation results show that DSA can achieve good performance in terms of throughput and jitter.

References

[1] J. Zhang, Y. Jin, N. Ansari, and W. Hu, "Dichotomy slot allocation: A low-jitter scheduling scheme for input-queued switches," in *Proc. IEEE High Performance Switching and Routing (HPSR 2007)*, Brooklyn, NY, May 30–Jun. 1 2007.

[2] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space-division packet switch," *IEEE Trans. Commun.*, vol. 35, no. 12, pp. 1347–1356, Dec. 1987.

[3] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Trans. Commun.*, vol. 47, no. 8, pp. 1260–1267, Aug. 1999.

[4] S. Li and N. Ansari, "Input-queued switching with QoS guarantees," in *Proc. IEEE INFOCOM*, New York, Mar. 21–25, 1999, vol. 3, pp. 1152–1159.

[5] H. J. Chao and X. Guo, *Quality of Service Control in High-Speed Networks*. New York: Wiley, 2004.

[6] J. C. R. Bennett, K. Benson, A. Charny, W. F. Courtney, and J.-Y. Le Boudec, "Delay jitter bounds and packet scale rate guarantee for expedited forwarding," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 529–540, Aug. 2002.

[7] Y. Mansour and B. Patt-Shamir, "Jitter control in QoS networks," *IEEE/ACM Trans. Netw.*, vol. 9, no. 4, pp. 492–502, Aug. 2001.

[8] I. Keslassy, M. Kodialam, T. V. Lakshman, and D. Stiliadis, "On guaranteed smooth scheduling for input-queued switches," *IEEE/ACM Trans. Netw.*, vol. 13, no. 6, pp. 1364–1375, Dec. 2005.

[9] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. ACM SIGCOMM*, Austin, TX, Sep. 19–22, 1989.

[10] J. Bennett and H. Zhang, "WF^2 Q: Worst-case fair weighted fair queuing," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 24–28, 1996, pp. 120–128.

[11] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round robin," *IEEE/ACM Trans. Netw.*, vol. 4, no. 3, pp. 375–385, Jun. 1996.

[12] C. Guo, "SRR: An O(1) time-complexity packet scheduler for flows in multiservice packet networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 6, pp. 1144–1155, Dec. 2004.

[13] M. Choy and T. T. Lee, "Huffman fair queuing: A scheduling algorithm providing smooth output traffic," in *Proc. IEEE Int. Conf. Communications (ICC,2006)*, Istanbul, Turkey, Jun. 11–15, 2006, pp. 796–799.

[14] D. Wei, Y. Jie, N. Ansari, and S. Papavassiliou, "Guaranteeing service rates for cell-based schedulers with a grouping architecture," in *Proc. IEE Commun.*, 2003, vol. 150.

[15] J. Fonda, M. Zawodniok, S. Jagannathan, and S. Watkins, "Adaptive distributed fair scheduling and its implementation in wireless sensor networks," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, Oct. 2006, pp. 3382–3387.

[16] T. T. Lee, "The Kraft's inequality of scheduling for packet-switched clos networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 2002–2010.

[17] S. He, S. Sun, H. Guan, Q. Zheng, Y. Zhao, and W. Gao, "On guaranteed smooth switching for buffered crossbar switches," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 718–731, Jun. 2008.

[18] T. H. Szymanski, "Throughput and QoS optimization in nonuniform multichannel wireless mesh networks," in *Proc. 4th ACM Symp. QoS and Security For Wireless and Mobile Networks (Q2SWinet'08)*, Vancouver, BC, Canada, Oct. 2008.

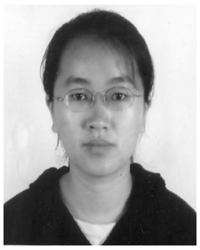
[19] C. S. Chang, W. J. Chen, and H. Y. Huang, "Birkhoff-von Neumann input buffered crossbar switches," in *Proc. IEEE INFOCOM*, Tel Aviv, Israel, Mar. 26–30, 2000, vol. 3, pp. 1614–1623.

[20] P. A. Baziana and I. E. Pountourakis, "An efficient metropolitan WDM ring architecture for a slotted transmission technique," *IEEE/OSA J. Lightwave Technol.*, vol. 26, no. 19, pp. 3307–3317, Oct. 1, 2008.

[21] S. R. Mohanty and L. N. Bhuyan, "Guaranteed smooth switch scheduling with low complexity," in *Proc. IEEE GLOBECOM*, St. Louis, MO, Nov. 28–Dec.2 2005.

[22] S. Mneimneh and K.-Y. Siu, "On achieving throughput in an input-queued switch," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 858–867, Oct. 2003.

[23] T. T. Lee and C. H. Lam, "Path switching—A quasi-static routing scheme for large-scale ATM packet switches," *IEEE J. Select. Areas Commun.*, vol. 15, no. 5, pp. 914–924, Jun. 1997.



Jingjing Zhang (S'09) received the B.E. degree from Xi'an Institute of Posts and Telecommunications, Xi'an, China, in 2003, and the M.E. degree from Shanghai Jiao Tong University, Shanghai, China, in 2006, both in electrical engineering. She is currently pursuing the Ph.D. degree in electrical engineering at the New Jersey Institute of Technology (NJIT), Newark.

Her research interests are broadly in optimization and broadband networks, with current focuses mainly on capacity analysis, planning, design, and resource allocation in next-generation passive optical networks, handover and resource allocation in integrated wireless and optical access networks, and QoE provisioning in next-generation networks.

Nirwan Ansari (S'78–M'83–SM'94–F'09) received the B.S.E.E. (summa cum laude) from the New Jersey Institute of Technology (NJIT), Newark, in 1982, the M.S.E.E. degree from the University of Michigan, Ann Arbor, in 1983, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 1988, respectively.

He is a Professor of Electrical and Computer Engineering at NJIT. His current research focuses on various aspects of broadband networks and multimedia communications.

Dr. Ansari is a Senior Technical Editor of the IEEE Communications Magazine, and was/is serving on the Advisory/Editorial Board of six other journals. He received the IEEE Leadership Award in 2007 (Central Jersey/Princeton Section), the NJIT Excellence in Teaching in Outstanding Professional Development (2008), the IEEE MGA Leadership Award (2008), the NCE Excellence in Teaching Award (2009), and designation as an IEEE Communications Society Distinguished Lecturer.

Yaohui Jin (A'05) received the B.S. degree in applied physics from Anhui University, Hefei, China, in 1992, the M.S. degree in condensed matter physics from the University of Science and Technology of China, Hefei, in 1995, and the Ph.D. degree in electromagnetic field and microwave technology from Shanghai Jiao Tong University, Shanghai, China, in 2000.

He is a Professor of State Key Laboratory of Advanced Optical Communication Systems and Networks, Shanghai Jiao Tong University. Previously, he was an MTS at Bell Labs Research China. His research interests include high-performance networks and applications, optimal control of multilayer multidomain optical networks, and photonic networks on chip. He has published more than 100 technical papers in leading conferences and journals.

Dr. Jin served as Public Co-Chair of IEEE Advanced Networks and Telecommunication Systems (ANTS) in 2007, Technical Program Committee (TPC) Co-Chair of IEEE High Performance Switching and Routing (HPSR) in 2008, Local Co-Chair of OSA/IEEE/SPIE Asia Communications and Photonics Conference (ACP) in 2009, and TPC member of OFC and ECOC.

Weisheng Hu (M'07) received the B.Sc. degree from Tsinghua University, Beijing, China, in 1986, the M.Eng. degree from Beijing University of Science and Technology, Beijing, in 1989, and the Ph.D. degree from Nanjing University, Nanjing, China, in 1997.

He joined Shanghai Jiao Tong University as a Postdoctorate Fellow in 1997 and has been a Professor since 1999. He is the Deputy Director of the State Key Laboratory of Advanced Optical Communication Systems and Networks. His research activities are mainly on photonic switching and optical networks. He has published ~ 200 peering journal and conference papers.

Dr. Hu serves on the Editorial Board for several journals and conferences, including JLT, OFC, APOC, ACP, and OpticsEast. He is a member of OSA.