Computer Communications 25 (2002) 611–622

computer communications

www.elsevier.com/locate/comcom

Load balanced Birkhoff–von Neumann switches, part I: one-stage buffering

Cheng-Shang Chang*, Duan-Shin Lee, Yi-Shean Jou

Institute of Communications Engineering, National Tsing Hua University Hsinchu 300, Taiwan, ROC

Received 8 October 2001; accepted 8 October 2001

Abstract

Motivated by the need for a simple and high performance switch architecture that scales up with the speed of fiber optics, we propose a switch architecture with two-stage switching fabrics and one-stage buffering. The first stage performs load balancing, while the second stage is a Birkhoff–von Neumann input-buffered switch that performs switching for load balanced traffic. Such a switch is called the load balanced Birkhoff–von Neumann switch in this paper. The on-line complexity of the switch is $O(1)$. It is shown that under a mild technical condition on the input traffic, the load balanced Birkhoff–von Neumann switch achieves 100% throughput as an output-buffered switch for both unicast and multicast traffic with fan-out splitting. When input traffic is bursty, we show that load balancing is very effective in reducing delay, and the average delay of the load balanced Birkhoff–von Neumann switch is proven to converge to that of an output-buffered switch under heavy load. Also, by simulations, we demonstrate that load balancing is more effective than the conflict resolution algorithm, *i*-SLIP, in heavy load. When both the load balanced Birkhoff–von Neumann switch and the corresponding output-buffered switch are allocated with the same finite amount of buffer at each port, we also show that the packet loss probability in the load balanced Birkhoff–von Neumann switch is much smaller than that in an output-buffered switch, when the buffer is large. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Input-buffered switches; Load balancing; Scheduling; Stability; Performance analysis

1. Introduction

There is an urgent need to build high speed switches that scale with the transmission speed of fiber optics. As the key limitation of an electronic switch is the memory accessing speed, input-buffered switches, capable of performing parallel read/write, have received a lot of attention recently (see e.g. Refs. [8,10,15,20,21,22,24,25,31,32]). An input-buffered crossbar switch with N input ports and N output ports has a segregated buffer for each input port. In such a switch, time is slotted and synchronized so that packets in different input buffers can be read out simultaneously within a time slot. In a time slot, a crossbar switch sets up a connection pattern corresponding to a permutation matrix. As a permutation matrix is a one-to-one mapping from input ports to output ports, packets destined to the same output ports cannot be transmitted at the same time. As discussed in Ref. [25], such limitation causes two potential problems: low throughput due to head-of-line (HOL) blocking and the difficulty in controlling packet delay. To allow an input-buffered switch to transmit non-HOL packets, the virtual output queueing (VOQ) technique might be used. Instead of having a single FIFO queue at each input port, the VOQ technique maintains a separate (logical) queue for each output port at each input port (see Fig. 1). By scheduling permutation matrices according to a weighted matching algorithm, it is shown in Refs. [24,25] that 100% throughput can be achieved. However, the complexity of the weighted matching algorithm prohibits its practical use and that motivates researchers to consider simpler scheduling policies, such as *i*-SLIP in Ref. [23].

There are several papers that addressed the issue of controlling packet delays in input-buffered switches. The first approach is to mimic the behavior of an output-buffered switch, where packet delays are much easier to control. Exact emulation of an output-buffered switch by a crossbar switch requires buffering at both input and output ports. It is then called a Combined Input–Output Queueing (CIOQ) switch. A CIOQ switch usually requires an internal speedup and a packet-scheduling algorithm with high complexity (see e.g. Refs. [10,32]).

The second approach of controlling packet delays in an input-buffered switch is through bandwidth allocation. This approach does not require internal speedups and usually

* Corresponding author.

E-mail addresses: cschang@ee.nthu.edu.tw (C.-S. Chang), lds@cs.nthu.edu.tw (D.-S. Lee), ysjou@gibbs.ee.nthu.edu.tw (Y.-S. Jou).

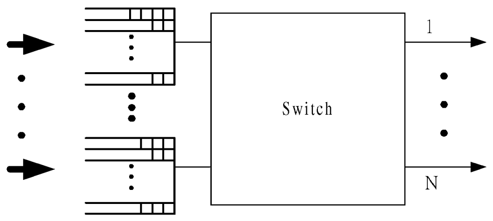


Fig. 1. Virtual output queues in input-buffered switches.

comes with a much simpler scheduling algorithm. In the paper [15], Hung, Kesidis and McKeown used an idling weighted round robin (WRR) algorithm in Ref. [1] to achieve rate guarantee for each input–output pair without internal speedup. Similar approaches are also addressed by Lee and Lam [21] and Li and Ansari [22]. As the usual WRR algorithm, such an approach requires that a frame size be chosen. A large frame size implies a large worst-case packet delay and a large memory requirement for the storage of all the connection patterns in a frame. On the other hand, a small frame size implies large rate granularity (the minimum rate allocated to an input–output pair). As a result, the WRR algorithm fails to provide uniform rate guarantees for all non-uniform traffic due to its framing requirement.

To cope with the granularity problem due to framing, Chang, Chen and Huang [5,6] proposed the Birkhoff–von Neumann input-buffered switch for bandwidth allocation. As in most input-buffered switches, the Birkhoff–von Neumann switch uses the VOQ technique to solve the HOL blocking problem. The main idea of scheduling the connection patterns in the Birkhoff–von Neumann switch is to use the capacity decomposition approach by Birkhoff [3] and von Neumann [35]. To explain the idea, let $\underline{r} = (r_{i,j})$ be the rate matrix with $r_{i,j}$ being the rate allocated to the traffic from input i to output j for an $N \times N$ input-buffered crossbar switch. Then under the following 'no overbooking' conditions

$$\sum_{i=1}^{N} r_{i,j} \le 1, \quad j = 1,2,\dots,N, \tag{1}$$

and

$$\sum_{j=1}^{N} r_{i,j} \le 1, \quad i = 1,2,\dots,N, \tag{2}$$

there exists a set of positive numbers ϕ_k and permutation matrices $P_k, k = 1,\dots,K$ for some $K \le N^2 - 2N + 2$ that satisfies

$$\underline{r} \le \sum_{k=1}^{K} \phi_k p_k, \tag{3}$$

and

$$\sum_{k=1}^{K} \phi_k = 1. \tag{4}$$

The computational complexity of the decomposition is $O(N^{4.5})$. For the details of the decomposition algorithm, we refer to Refs. [5,6].

Once one obtains such a decomposition, one can simply schedule the connection pattern P_k proportional to its weight $\phi_{k,}$ $k = 1,\dots,K$. The on-line scheduling algorithm used in Refs. [5,6] is a simplified version of the Packetized Generalized Processor Sharing (PGPS) algorithm in Parekh and Gallager [28] (or the Weighted Fair Queueing (WFQ) in Demers, Keshav, and Shenkar [13]). In particular, if $\phi_k = 1/K$ for all k, then the algorithm generates a periodic sequence of connection patterns with period K. The complexity of the on-line scheduling algorithm is $O(\log N)$ as one needs to sort the $O(N^2)$ virtual finishing times in the PGPS-like algorithm.

If the allocated bandwidth is larger than the arrival rate for each input-output pair, then it is shown in Refs. [5,6] that the Birkhoff–von Neumann input-buffered switch achieves 100% throughput without framing and internal speedup. This implies that the Birkhoff–von Neumann switch requires the information of the arrival rate of each input–output pair in order to achieve 100% throughput. Such information is gathered by rate estimators in Ref. [6]. Another problem of such a switch is that the number of permutation matrices deduced from the Birkhoff–von Neumann decomposition algorithm is $O(N^2)$, which may not scale for switches with a large number of input/output ports. The scalability problem for the Birkhoff–von Neumann switch was addressed in Ref. [6] by considering two types of multi-stage networks, a two-stage Banyan network and a three-stage rearrangeable network. The two-stage Banyan network is shown to have less than 100% throughput and the three-stage rearrangeable network does achieve 100% throughput at the cost of additional hardware complexity.

The work in Ref. [6] motivates us to propose a much simpler two-stage switch architecture, called the load balanced Birkhoff–von Neumann switch. The first stage performs load balancing, while the second stage is a Birkhoff–von Neumann input-buffered switch that performs switching for load balanced traffic. The switch has the following advantages:

(i) *Scalability*. The on-line complexity of the scheduling algorithm in the switch is $O(1)$.
(ii) *Low hardware complexity*. Only two crossbar switch fabrics and buffers between them are required. Moreover, the two crossbar switch fabrics can be realized by the Banyan networks. Neither internal speedup nor rate estimation is needed in the switch.
(iii) 100% *throughput*. Under a mild technical condition (in Section 3) on the input traffic, the load balanced Birkhoff–von Neumann switch achieves 100% throughput as an output-buffered switch for both unicast and multicast traffic with fan-out splitting.
(iv) *Low average delay in heavy load and bursty traffic*. When input traffic is bursty, load balancing is very

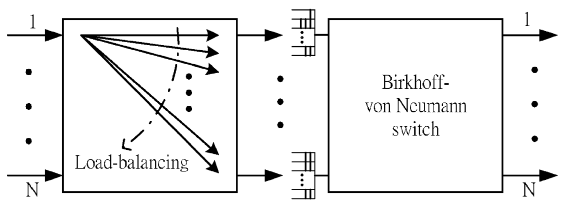


Fig. 2. The switch architecture.

effective in reducing delay, and the average delay of the load balanced Birkhoff–von Neumann switch is proven to converge to that of an output-buffered switch under heavy load. In addition, by simulations, we demonstrate that load balancing is more effective than the conflict resolution algorithm, *i*-SLIP [23] in heavy load.

(v) *Efficient buffer usage*. When both the load balanced Birkhoff–von Neumann switch and the corresponding output-buffered switch are allocated with the same finite amount of buffer at each port, the packet loss probability in the load balanced Birkhoff–von Neumann switch is much smaller than that in an output-buffered switch when the buffer is large.

The main drawback of the switch is that FIFO might be violated for packets from the same input. One quick fix is to add a resequencing buffer at the output port. However, this increases the complexity of the hardware design. How to perform resequencing efficiently will be addressed in the sequel [7].

The paper is organized as follows: in Section 2, we introduce the switch architecture of the load balanced Birkhoff–von Neumann switch. In Section 3, we prove that the load balanced Birkhoff–von Neumann switch indeed achieves 100% throughput as an output-buffered switch under a certain technical condition on the input traffic. We also compare its performance, such as average delay in Section 4 and queue length in Section 5, with an output-buffered switch under uniform independent and identically distributed traffic and uniform bursty traffic. In Section 6, we address the implementation and scalability issues of the switch. We then conclude the paper in Section 7.

2. The switch architecture

The load balanced Birkhoff–von Neumann switch consists of two stages (see Fig. 2). The first stage performs load balancing and the second stage performs switching. The second stage is the Birkhoff–von Neumann input-buffered crossbar switch running with a sequence of periodic connection patterns. The period is equal to the number of input/output ports. To be precise, suppose that the number of input/output ports is N. Let P be any one-cycle $N \times N$ permutation matrix. A typical one-cycle permutation matrix is the circular-shift matrix with $P_{i,j} = 1$ when $j = i + 1 \bmod N$, and $P_{i,j} = 0$ otherwise. Assign $P_k = P^k$ and $\phi_k = 1/N$ for $k = 1, \ldots, N$ in Eq. (3). As P is a one-cycle permutation matrix, P^N is the identity matrix, and the PGPS-like algorithm in the Birkhoff–von Neumann switch is simply periodic with period N. Moreover, each input–output pair is assigned a time slot during every N time slots, and the allocated rate for each input–output pair is $1/N$. This implies that 100% throughput can be achieved if the input traffic to the second stage is *uniform*, which is exactly what we would like to do at the first stage.

The first stage is a unbuffered crossbar switch. Packets arriving at the first stage at time t are switched instantly to the second stage, according to the connection pattern set up at the crossbar switch. To be precise, let $\underline{a}(t) = (a_{i,j}(t))$ be the $N \times N$ traffic matrix at time t, where $a_{i,j}(t)$ is the number of packet arriving at the ith input port and destined to the jth output port at time t. As there is at most one packet arriving at an input port per time slot, $a_{i,j}(t)$s are indicator variables. Also, let $P_1(t)$ be the $N \times N$ permutation matrix assigned at time t at the first stage and $\underline{b}(t) = (b_{i,j}(t))$ be the $N \times N$ traffic matrix entering the second stage, where $b_{i,j}(t)$ is the number of packet arriving at the ith input port of the second stage and destined to the jth output port at time t. Then we have

$$\underline{b}(t) = P_1(t)\underline{a}(t). \tag{5}$$

To perform load balancing, we simply set up the permutation matrices $P_1(t)$ periodically via a one-cycle permutation matrix P as in the second stage.

One of the main advantages of the two-stage load balanced Birkhoff–von Neumann switch is the reduction of complexity. In comparison with the original Birkhoff–von Neumann input-buffered switch, there is no need for rate estimation in the load balanced Birkhoff–von Neumann switch. As a result, there is also no need to perform the Birkhoff–von Neumann capacity decomposition. For the number of permutation matrices needed in the switch, the complexity is reduced from $O(N^2)$ to $O(N)$. In addition, the on-line computational complexity for the scheduling algorithm is reduced from $O(\log N)$ to $O(1)$ as the scheduling policy is now simply periodic. With all the reduction of complexity, we will show in Section 3 that the load balanced Birkhoff–von Neumann switch still has good performance, including 100% throughput.

3. Stability

In this section, we do stability analysis for the load balanced Birkhoff–von Neumann switches. We will show that load balancing at the first stage is able to convert non-uniform traffic into uniform traffic under a mild technical condition on the input traffic so that the load balanced Birkhoff–von Neumann switch achieves the same stability region (100% throughput) as an output-buffered switch.

For our analysis, we assume that the permutation matrices assigned at both stages are started from independent and uniformly distributed phases. To be precise, we consider a periodic sequence of permutation matrices $P(t) = P^{t'}$, where $t' = t \bmod N$, and P is any one-cycle permutation matrix. Let U_1 and U_2 be two uniformly distributed random variables over $\{0, 1, 2, \ldots, N-1\}$ that are independent of each other and everything else. Denote by $P_1(t)$ (resp. $P_2(t)$) the permutation matrix at the first (resp. second) stage at time t. Let

$$P_1(t) = P(t + U_1), \tag{6}$$

$$P_2(t) = P(t + U_2). \tag{7}$$

We make the following assumption on the input:

(**A**1) $\{a(t), t \leq 1\}$ is a stationary and weakly mixing stochastic sequence with the mean rate $\underline{r}$, where $r_{i,j}$ is the mean rate for the traffic from the ith input port to the jth output port.

Recall that the concepts of ergodicity, weak mixing, and strong mixing are basically measures of how fast a stochastic sequence loses memory (see e.g. Petersen [29] and Nadkarni [27]). For a stochastic sequence $\underline{a} = \{\underline{a}(t), t \geq 1\}$, define the time-shifted sequence $\theta_s \underline{a} = \{\underline{a}(t+s), t \geq 1\}$. The stochastic sequence $\underline{a}$ is *stationary* if for any time shift s, the stochastic sequences $\underline{a}$ and $\theta_s \underline{a}$ have the same joint distribution, i.e.

$$P(\underline{a} \in A) = P(\theta_s \underline{a} \in A)$$

for any $A \in (R^{N \times N})^\infty$. A stationary sequence $\{\underline{a}(t), t \geq 1\}$ is *ergodic* if for all $A, B \in (R^{N \times N})^\infty$

$$\lim_{t \to \infty} \frac{1}{t} \sum_{s=0}^{t-1} P(\theta_s \underline{a} \in A, \underline{a} \in B) = P(\underline{a} \in A) P(\underline{a} \in B).$$

It is *weakly mixing* if for all $A, B \in (R^{N \times N})^\infty$

$$\lim_{t \to 0} \frac{1}{t} \sum_{s=0}^{t-1} |P(\theta_s \underline{a} \in A, \underline{a} \in B) - P(\underline{a} \in A) P(\underline{a} \in B)| = 0.$$

It is *strongly mixing*if for all $A, B \in (R^{N \times N})^\infty$

$$\lim_{t \to 0} P(\theta_t \underline{a} \in A, \underline{a} \in B) = P(\underline{a} \in A) P(\underline{a} \in B).$$

Clearly, strong mixing implies weak mixing, which in turn implies ergodicity. One of the most important properties of a stationary and ergodic sequence $\{\underline{a}(t), t \geq 1\}$ is that the time averages are equal to the ensemble averages, i.e.

$$\lim_{t \to \infty} \frac{1}{t} \sum_{s=1}^{t} \underline{a}(s) = E\underline{a}(1), \qquad a.s. \tag{8}$$

Incidentally, both von Neumann and Birkhoff made important contributions to such a property (see e.g. Ref. [27] as the concept of ergodicity is a generalization of permutation and recurrence. We note that $\{P_1(t), t \geq 1\}$ in Eq. (6) (resp. $\{P_2(t), t \geq 1\}$ in Eq. (7)) is a stationary and ergodic sequence with the mean rate $(1/N)\underline{e}$, where $\underline{e}$ is the $N \times N$ matrix with all its element being 1. From Eq. (8)

$$\lim_{t \to \infty} \frac{1}{t} \sum_{s=0}^{t} P_i(s) = EP_i(1) = \frac{1}{N}\underline{e}, \qquad a.s. \tag{9}$$

Let $\underline{q}(t) = (q_{i,j}(t))$ be the queue length matrix with $q_{i,j}(t)$ being the number of packets that are destined to the jth output port at the ith input buffer of the second stage at time t. Then, we have the following Lindley's recursion (assuming infinite buffer):

$$\underline{q}(t+1) = \max[\underline{q}(t) + \underline{b}(t+1) - P_2(t+1), \underline{o}], \tag{10}$$

where $\underline{o}$ is the zero matrix and the maximum of two matrices is taken component wise. If we start from an empty system, i.e. $\underline{q}(0) = \underline{o}$, then expanding Eq. (10) recursively yields

$$\underline{q}(t) = \max_{0 \leq s \leq t} \left[\sum_{\tau = s+1}^{t} \underline{b}(\tau) - P_2(\tau) \right], \tag{11}$$

with the convention that an empty sum equals 0.

In the following, we present our main stability result. The proof is deferred to the end of this section.

Theorem 1. *Under the assumption in* (*A*1) *and* $\underline{q}(0) = \underline{o}$, $\underline{q}(t)$ *converges in distribution to a steady state random matrix* $\underline{q}(\infty)$ *if the following no overbooking conditions are satisfied*

$$\sum_{i=1}^{N} r_{i,j} < 1, \qquad j = 1, \ldots, N. \tag{12}$$

Note that the other no overbooking conditions in Eq. (2) are not needed as there is at most one packet arrival at each input per time slot. This implies that Theorem 1 still holds for multicast traffic if fan-out splitting is done at the buffer between the two stages. Further discussions along this line will be addressed in the sequel [7].

To compare our stability result with an output-buffered switch subject to the same input, let $q_j^{\mathrm{o}}(t)$ be the number of packets at the jth output buffer at time t. The corresponding Lindley equation is

$$q_j^{\mathrm{o}}(t+1) = \max\left[q_j^{\mathrm{o}}(t) + \sum_{i=1}^{N} a_{i,j}(t+1) - 1, 0 \right]. \tag{13}$$

Let $q^{\mathrm{o}}(t) = (q_1^{\mathrm{o}}(t), \ldots, q_N^{\mathrm{o}}(t))$ and e be the $1 \times N$ row vector with all its elements being 1. Writing Eq. (13) in the vector form yields

$$q^{\mathrm{o}}(t+1) = \max[q^{\mathrm{o}}(t) + e\underline{a}(t) - e, o], \tag{14}$$

where o is a $1 \times N$ row vector with all its elements being 0. It is well-known from the Loynes construction (see e.g. Refs. [2,4] that $q^{\mathrm{o}}(t)$ converges to a steady state random vector $q^{\mathrm{o}}(\infty)$ if $\{a(t), t \geq 1\}$ is stationary and ergodic, and the no overbooking conditions in Eq. (12) are satisfied. In view of this, the load balanced Birkhoff–von Neumann switch achieves the same stability region as that of an output-buffered switch. However, it requires the input process to be weakly mixing, which is a stronger condition than ergodicity needed for an output-buffered switch.

To see the reason that we need the weak mixing condition, consider the case with $N = 2$. In this case, the only one-cycle permutation is

$$P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Consider the periodic sequence $P(t) = P^{t'}$ with $t' = t \bmod 2$. Let $P_1(t) = P(t + U_1)$ and $\underline{a}(t) = z(t)P(t + \tilde{U}_1)$, where U_1 and $\tilde{U}_1$ are two independent Bernoulli random variables with $\mathsf{P}(U_1 = 0) = \mathsf{P}(U_1 = 1) = \mathsf{P}\tilde{U}_1 = 0) = \mathsf{P}(\tilde{U}_1 = 1) = 1/2$, and $\{z(t), t \ge 1\}$ is a sequence of i.i.d. Bernoulli random variables with $\mathsf{P}(z(1) = 1) = 0.9$ and $\mathsf{P}(z(1) = 0) = 0.1$. Though both $\{P_1(t), t \ge 1\}$ and $\{\underline{a}(t), t \ge 1\}$ constructed this way are ergodic, the process $\{\underline{b}(t) = P_1(t)\underline{a}(t), t \ge 1\}$ is not ergodic. This can be easily seen from the fact that it can be decomposed as two ergodic sequences. With an equal probability 1/2, $\underline{b}(t) = z(t)P$ for all t or $\underline{b}(t) = z(t)P^2$ for all t (note that P^2 is simply the identity matrix). In either case, the buffer at each input port of the second stage goes to infinity as $t \to \infty$ when $\{\underline{b}(t), t \ge 1\}$ is fed into second stage. This example shows that for $\{\underline{b}(t), t \ge 1\}$ to be ergodic, one of the two sequences $\{P_1(t), t \ge 1\}$ and $\{\underline{a}(t), t \ge 1\}$ has to be weakly mixing (for more details, see e.g. Ref. [29]). Certainly, if we choose $P_1(t)$ randomly from $P^1, P^2, ..., P^N$ for every t, then $\{P_1(t), t \ge 1\}$ is strongly mixing and hence weakly mixing. In this case, we only need to assume that $\{\underline{a}(t), t \ge 1\}$ is ergodic. However, the drawback of doing load balancing randomly (randomization in Refs. [26,24]) is the degradation of performance (see e.g. Ref. [33]).

Now we prove Theorem 1.

Proof. (Theorem 1) We first show that $\{\underline{b}(t), t \ge 1\}$ is stationary and ergodic with the mean rate $(1/N)\underline{er}$. As $\{\underline{a}(t), t \ge 1\}$ and $\{P_1(t), t \ge 1\}$ are stationary and independent, $\{\underline{b}(t), t \ge 1\}$ is also stationary. Moreover

$$E\underline{b}(t) = EP_1(t)\underline{a}(t) = EP_1(t)E\underline{a}(t) = \frac{1}{N}\underline{er}.$$

Since $\{\underline{a}(t), t \ge 1\}$ is weakly mixing and $\{P_1(t), t \ge 1\}$ is ergodic, $\{\underline{b}(t), t \ge 1\}$ is ergodic ([29], Theorem 2.6.1). Thus, $\{\underline{b}(t), t \ge 1\}$ is stationary and ergodic with the mean rate $(1/N)\underline{er}$.

From the standard Loynes construction (see e.g. Refs. [2,4]), it then suffices to show that

$$\lim_{t\to\infty} \frac{1}{t} \sum_{s=1}^{t} \underline{b}(s) - P_2(s) < \underline{o}, \qquad a.s.$$

As both $\{\underline{b}(t), t \ge 1\}$ and $\{P_2(t), t \ge 1\}$ are stationary and ergodic, it then follows from the ergodic property in Eqs. (8) and (9) and the no overbooking conditions in Eq. (12) that

$$\lim_{t\to\infty} \frac{1}{t} \sum_{s=1}^{t} \underline{b}(s) - P_2(s) = \frac{1}{N}\underline{er} - \frac{1}{N}\underline{e} < \underline{o}, \qquad a.s.$$

□

4. Delay

In this section, we do delay analysis for the load balanced Birkhoff–von Neumann switches. In addition to the effect of converting non-uniform traffic into uniform traffic, load balancing at the first stage also achieves burst reduction. The effect of burst reduction greatly reduces average delay as shown in this section. In Sections 4.1 and 4.2, we consider two different traffic models: uniform i.i.d. traffic model and uniform bursty traffic model. For the uniform i.i.d. traffic model, load balancing has no effect on the input, and the performance is poor when compared with an output-buffered switch. In contrast to the uniform i.i.d. model, load balancing achieves perfect burst reduction in the uniform bursty traffic model. In this case, its performance is good.

4.1. Uniform i.i.d. traffic model

To carry out the analysis for more specific performance measures, such as average queue length and average delay, we need to have a more specific model for the input. In this section, we consider a uniform i.i.d. traffic model. With probability ρ, a packet arrives at each input (of the first stage) for every time slot. This is independent of everything else. The destination of an arriving packet is chosen uniformly among the N output ports. This is also independent of everything else. Based on this traffic model, we make the following two observations:

(i) Load balancing at the first stage has no effect at all (as the traffic is already balanced). To be precise, $\{\underline{b}(t), t \ge 1\}$ has the same joint distribution as $\{\underline{a}(t), t \ge 1\}$. Moreover, $\{b_{i,j}(t), t \ge 1\}$ and $\{a_{i,j}(t), t \ge 1\}$ for all i and j are sequences of i.i.d. Bernoulli random variables with mean ρ/N.
(ii) As the traffic is uniform, $q_{i,j}(t)$s are all identically distributed.

Without loss of generality, let us look at the recursive equation for $q_{1,1}(t)$. Note from (i) that the arrival sequence to $q_{1,1}$ is simply a sequence of i.i.d. Bernoulli random variables with mean ρ/N. Let T be a time that $T + U_2$ is an integer multiple of N. Note from Eq. (7) that $P_2(T)$ is the identity matrix. As $\{P_2(t), t \ge 1\}$ is generated from a

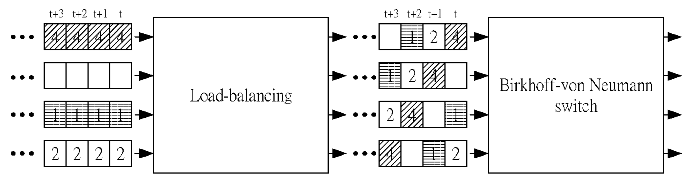


Fig. 3. Burst reduction in the uniform bursty traffic model.

one-cycle permutation matrix with period N, we then have

$$q_{1,1}(T+s) = q_{1,1}(T) + \sum_{k=1}^{s} b_{1,1}(T+k); \quad s = 1,\ldots,N-1, \tag{15}$$

$$q_{1,1}(T+N) = \max\left[q_{1,1}(T) + \sum_{k=1}^{N} b_{1,1}(T+k) - 1, 0 \right]. \tag{16}$$

In the steady state, $q_{1,1}(T)$ and $q_{1,1}(T+N)$ have the same distribution. The Lindley recursion in Eq. (16) has the following well-known solution (see e.g. [30], Eqs. (5)–(41))

$$Eq_{1,1}(T) = \frac{N-1}{N}\frac{\rho^2}{2(1-\rho)}. \tag{17}$$

From Eq. (15), it follows that

$$Eq_{1,1}(T+s) = Eq_{1,1}(T) + s\frac{\rho}{N}, \quad s = 1,\ldots,N-1. \tag{18}$$

This then implies that in the steady state

$$Eq_{1,1}(\infty) = \frac{1}{N}\sum_{s=0}^{N-1} Eq_{1,1}(T+s) = \frac{N-1}{N}\frac{\rho}{2(1-\rho)}. \tag{19}$$

Let $\bar{d}_1$ be the average delay for a packet. As the average arrival rate to $q_{1,1}$ is ρ/N, we have from Little's formula that

$$\bar{d}_1 = \frac{N-1}{2(1-\rho)}. \tag{20}$$

To compare the performance with the corresponding output-buffered switch, we note that Eqs. (13) and (16) are stochastically identical as both $\{b_{1,1}(t), t \geq 1\}$ and $\{a_{1,1}(t), t \geq 1\}$ are sequences of i.i.d. Bernoulli random variables with mean ρ/N. This then implies that

$$Eq_1^{\mathrm{o}}(\infty) = Eq_{1,1}(T).$$

Let $\bar{d}_1^{\mathrm{o}}$ be the average delay for a packet in the corresponding output-buffered switch. As the arrival rate to an output port in the output-buffered switch is ρ, once again we have from Little's formula that

$$\bar{d}_1^{\mathrm{o}} = \frac{N-1}{N}\frac{\rho}{2(1-\rho)}. \tag{21}$$

This shows that

$$\frac{\bar{d}_1^{\mathrm{o}}}{\bar{d}_1} = \frac{\rho}{N}, \tag{22}$$

and the performance of the load balanced Birkhoff–von Neumann switch is poor compared with that of an output-buffered switch. This is not surprising as load balancing has no effect at all for this traffic model.

4.2. Uniform bursty traffic model

In this section, we consider the following uniform bursty traffic model. Packets come as a burst of length N, which is exactly the same as the number of input/output ports. Packets within the same burst are destined to the same output. For every N time slots, the probability that there is a burst arriving at a particular input port (of the first stage) is ρ, and the probability that there are no packet arrivals in these N slots is $1-\rho$. This is independent of everything else. The destination of N packets within that burst is chosen uniformly among the N output ports. This is also independent of everything else. Based on this traffic model, we also make the following two observations:

(i) In contrast to the uniform i.i.d. traffic model in Section 4.1, load balancing achieves perfect burst reduction in this model (see Fig. 3). The N packets within a burst are distributed *evenly* to the input ports at the second stage. In this model, $\{b_{i,j}(t), t \geq 1\}$ for all i and j are still sequences of i.i.d. Bernoulli random variables with mean ρ/N.
(ii) As the traffic is uniform, $q_{i,j}(t)$s are still identically distributed.

Without loss of generality, let us also look at the recursive equation for $q_{1,1}(t)$. As the arrival sequence to $q_{1,1}$ is still a sequence of i.i.d., Bernoulli random variables with mean ρ/N, the whole analysis is the same as that in the uniform i.i.d. traffic model, and we conclude that the average delay for a packet in this model, denoted by $\bar{d}_2$, is the same as that in the uniform i.i.d. traffic model, i.e.

$$\bar{d}_2 = \frac{N-1}{2(1-\rho)}. \tag{23}$$

Now we do the performance analysis for the corresponding output-buffered switch. As packets come as a burst of length N, we have $\underline{a}(Nt+1) = \underline{a}(Nt+2) = \cdots = \underline{a}(Nt+N)$ for all t. Note from the Lindley recursion in Eq. (13) that for $s = 1,\ldots,N$,

$$q_1^{\mathrm{o}}(Nt+s) = \max\left[q_1^{\mathrm{o}}(Nt) + s\sum_{i=1}^{N} a_{i,1}(Nt+1) - s, 0 \right]. \tag{24}$$

Table 1
Average delay for output-buffered switches and load balanced Birkohff–von Neumann switches

Delay	Output-buffered	Load balanced
i.i.d.	$(N-1/N)(\rho/2(1-\rho))$	$(N-1)/2(1-\rho)$
Bursty	$(N-1)\rho/2(1-\rho)$	$(N-1)/2(1-\rho)$

In particular, for $s = N$, we have

$$\frac{q_1^o(N(t+1))}{N} = \max\left[\frac{q_1^o(Nt)}{N} + \sum_{i=1}^{N} a_{i,1}(Nt+1) - 1, 0\right]. \tag{25}$$

This recursion is stochastically identical to that in Eq. (16). It then follows from Eq. (17) that (in the steady state)

$$Eq_1^o(Nt) = \frac{(N-1)\rho^2}{2(1-\rho)}. \tag{26}$$

Now we show that $Eq_1^o(Nt+s) = Eq_1^o(Nt)$ for $s = 1, \ldots, N-1$. To simplify the notation, let $Z = \sum_{i=1}^{N} a_{i,1}(Nt+1)$. As packets come as a burst of length N, the random variable $q_1^o(Nt)$ only takes values on integer multiples of N. This implies that for $s = 1, \ldots, N$, $q_1^o(Nt+s) = q_1^o(Nt) + sZ - s$ if $q_1^o(Nt) > 0$ or $Z > 0$, and $q_1^o(Nt+s) = 0$ otherwise. Let $\mathbf{1}_A$ be the indicator random variable for an event A. Then we can rewrite this as follows:

$$q_1^o(Nt+s) = \left(q_1^o(Nt) + sZ - s\right)\left(1 - \mathbf{1}_{\{q_1^o(Nt)=0, Z=0\}}\right). \tag{27}$$

Taking expectations on both sides of Eq. (27) yields

$$Eq_1^o(Nt+s) = Eq_1^o(Nt) + sEZ - s + sP\left(q_1^o(Nt) = 0, Z = 0\right). \tag{28}$$

When $s = N$, we have from $Eq_1^o(Nt+N) = Eq^o(Nt)$ and Eq. (28) that

$$EZ = 1 - P\left(q_1^o(Nt) = 0, Z = 0\right). \tag{29}$$

Replacing Eq. (29) in Eq. (28) yields $Eq_1^o(Nt+s) = Eq_1^o(Nt)$ for all $s = 1, \ldots, N-1$. This then implies that in the steady state

$$Eq_1^o(\infty) = Eq_1^o(Nt) = \frac{(N-1)\rho^2}{2(1-\rho)}. \tag{30}$$

Let $\bar{d}_2^o$ be the average delay for a packet in the corresponding output-buffered switch. Once again, we have from Little's formula that

$$\bar{d}_2^o = \frac{(N-1)\rho}{2(1-\rho)}. \tag{31}$$

For this traffic model, we have that

$$\frac{\bar{d}_2^o}{\bar{d}_2} = \rho. \tag{32}$$

This shows that the delay in this traffic model converges to that of an output-buffered switch when $\rho \to 1$.

We summarize our results for the average delay of these two traffic models in Table 1.

4.3. Simulation

In this section, we perform various simulations to verify our observations and conclusions in the previous section. Since the real traffic is bursty (see e.g. Ref. [17]), we focus our simulations on bursty traffic. In all the simulations, the switch size is 16×16, i.e. $N = 16$. In our first experiment, we consider the uniform bursty traffic model in Section 4.2. In Fig. 4, we report the simulation results for the average delay under the uniform bursty traffic model for the load balanced Birkhoff–von Neumann switch, the output-buffered switch, the Birkhoff–von Neumann switch [6], and the 4-SLIP [23], respectively. In the simulations for the Birkhoff–von Neumann switch [6], we assume that the arrival rates are known and no dynamic rate estimation and adjustment is performed. These simulation results are obtained with 99% confidence intervals (for the clarity of the results, the confidence intervals are not shown in the figure). As expected, the simulations results of the

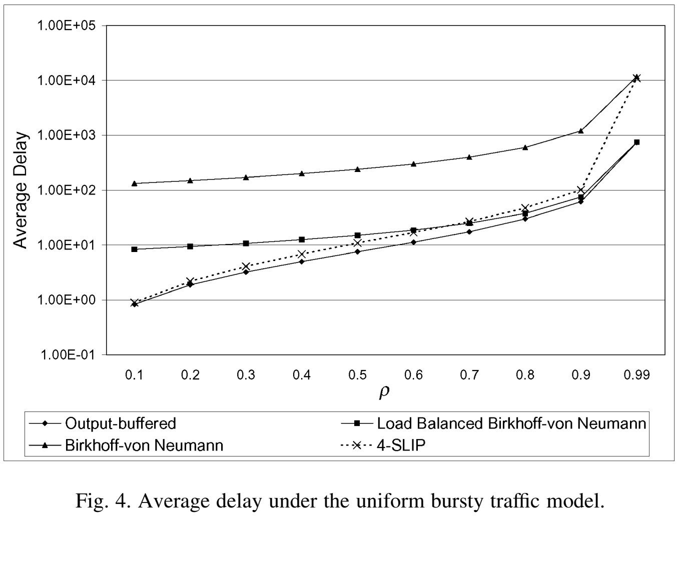


Fig. 4. Average delay under the uniform bursty traffic model.

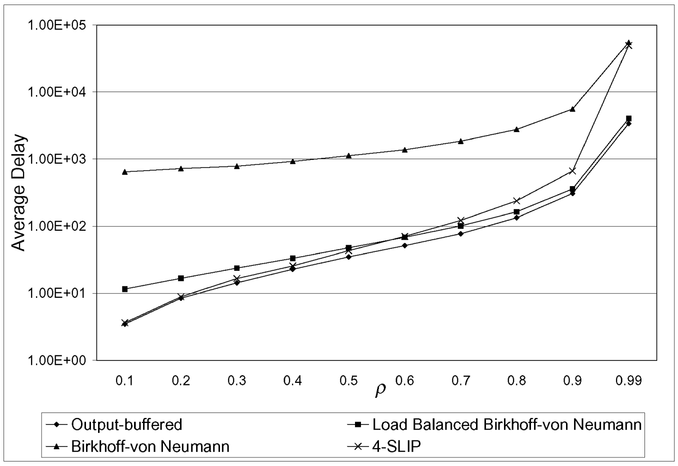


Fig. 5. Average delay under the uniform Pareto traffic model.

output-buffered switch and the load balanced Birkhoff–von Neumann switch match perfectly with the theoretical results in Eqs. (23) and (31). In comparison with the original Birkhoff–von Neumann switch, load balancing is very effective in reducing the average delay. In light load ($\rho \leq 0.4$), conflict resolution is very effective and the 4-SLIP performs much better than the load balanced Birkhoff–von Neumann switches. However, as load increases, load balancing is much more effective than conflict resolution. From our simulations, the load balanced Birkhoff–von Neumann switches performs much better than the 4-SLIP in heavy load ($\rho \geq 0.7$). To verify this observation, in our second experiment, we run simulations with random burst length instead. As in the uniform bursty traffic model, packets come as a burst. However, the burst lengths are chosen independently according to the following (truncated) Pareto distribution:

$$P(\text{A burst has length } i) = \frac{c}{i^{2.5}}, \quad i = 1, \ldots, 10,000,$$

where $c = (\sum_{i=1}^{10,000} 1/i^{2.5})^{-1}$ the normalization constant. In this experiment, the average burst length is 1.932, which is considerably smaller than 16, the fixed burst length in the first experiment. However, we still see the same effect in Fig. 5. The intuition behind this is that the dominating effect on the average delay is the heavy tail of the burst length distribution (see e.g. Refs. [9,16]). For a large burst, load balancing is quite effective in burst reduction and thus yields better performance.

The intuition for the 4-SLIP not performing well when the traffic is heavy and bursty is that SLIP might get trapped in 'bad modes'. To see this, consider a 3×3 SLIP switch with the periodic input traffic shown in Fig. 6. At time $t-1$, the first (resp. second, third) input buffer has two packets destined to output ports 2 and 3 (resp. 1 and 3, 2 and 1). At time t, a packet destined to output port 2 (resp. 3, 1) arrives at the first (resp. second, third) buffer. At time $t+1$, another packet destined to output port 3 (resp. 1, 2) arrives at the first (resp. second, third) buffer. The input pattern then repeats itself from time $t+2$ onward as shown in Fig. 6. For this input traffic, the SLIP algorithm (with as many iterations as possible) Ref. [23] produces the connection patterns as shown in Fig. 7. Note that all the pointers at $t+2$ and $t+5$ are the same and they yield the same connection pattern. As a result, SLIP is trapped in a periodical sequence of connection patterns and all of these connection patterns can send two packets per time slot. Thus, the throughput in this example is only 66.667%, instead of 100% in an output-buffered switch. For SLIP to get out of the trap, the traffic needs to be changed, and this might take a long time when the traffic is heavy and bursty.

5. Buffer usage

Now we turn to the comparison for queue length distributions. Consider the uniform bursty traffic model in Section 4.2. We will show in this section that the buffer usage in the load-balanced Birkhoff–von Neumann switch is more efficient than that of the corresponding output-buffered switch

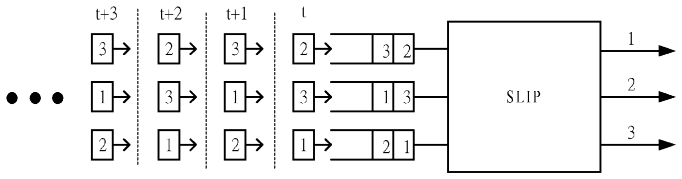


Fig. 6. The input traffic to a 3×3 SLIP switch.

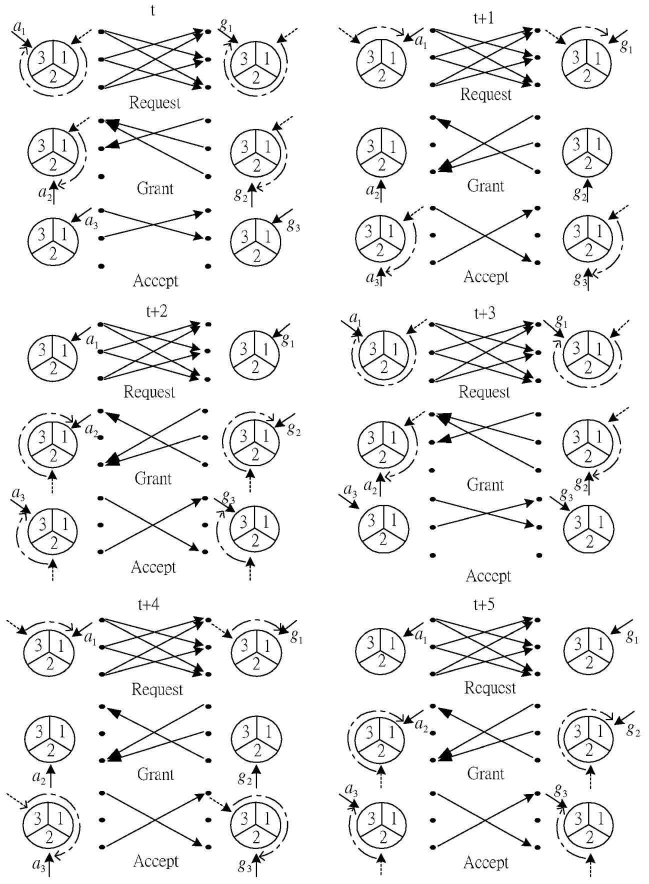


Fig. 7. An illustrating example of a bad mode in a SLIP switch.

(without sharing) when the buffer is large. Let

$$\Lambda(\theta) = \log E \exp\left(\theta \sum_{i=1}^{N} \alpha_{i,1}(Nt+1)\right). \tag{33}$$

As $a_{i,1}(Nt+1)$, $i = 1,\ldots,N$ are i.i.d. Bernoulli random variables with mean ρ/N, we have

$$\Lambda(\theta) = N \log\left(\frac{\rho}{N}e^{\theta} + \left(1 - \frac{\rho}{N}\right)\right). \tag{34}$$

In view of the Lindley recursion in Eq. (25), it follows from the theory of effective bandwidth (see e.g. Ref. [4], Chapter 9) that

$$\lim_{x\to\infty} \frac{1}{x} \log P\left(\frac{q_1^{o}(Nt)}{N} \geq x\right) = -\theta^*, \tag{35}$$

where θ^* is the unique non-zero solution of the following equation:

$$\frac{\Lambda(\theta)}{\theta} = 1. \tag{36}$$

Also, note from Eq. (24) that for $s = 1,\ldots,N-1$,

$$q_1^{o}(Nt) - N \leq q_1^{o}(Nt+s) \leq q_1^{o}(Nt) + N^2.$$

In conjunction with Eq. (35), we then have

$$\lim_{x\to\infty} \frac{1}{x} \log P\left(\frac{q_1^{o}(\infty)}{N} \geq x\right) = -\theta^*. \tag{37}$$

This shows that

$$P(q_1^{o}(\infty) \geq x) \approx \exp\left(-\frac{\theta^*}{N}x\right). \tag{38}$$

Similarly, for the load balanced Birkhoff–von Neumann switch, we have from Eq. (16) that

$$\lim_{x\to\infty} \frac{1}{x} \log P(q_{1,1}(T) \geq x) = -\theta^*. \tag{39}$$

Also, note from Eq. (15) that

$$q_{1,1}(T) \leq q_{1,1}(T+s) \leq q_{1,1}(T) + N, \quad s = 1,\ldots,N-1.$$

In conjunction with Eq. (39), it follows that

$$\lim_{x\to\infty} \frac{1}{x} \log P(q_{1,1}(\infty) \geq x) = -\theta^*. \tag{40}$$

Let $q_1(\infty) = \sum_{j=1}^{N} q_{1,j}(\infty)$ be the total number of packets at the first input port of the second stage in the steady state. Though $q_{1,j}(\infty), j = 1,\ldots,N$ are identically distributed, they are not independent as their arrival sequences come from splitting sequences of i.i.d. Bernoulli random variables with mean ρ. However, when $N \to \infty$, they become independent

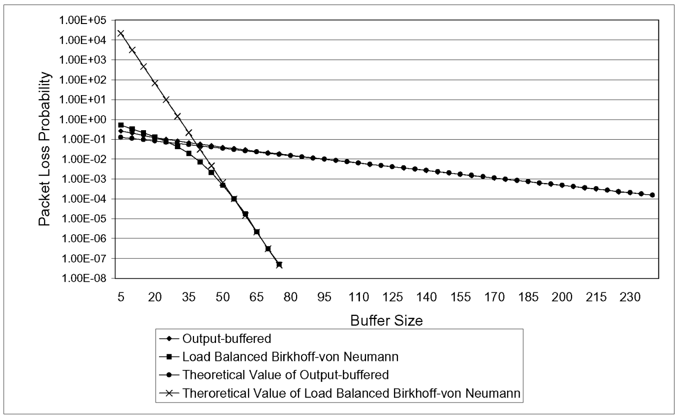


Fig. 8. Packet lost probability under uniform bursty traffic (switches size: 16 × 16, arrival rate $\rho = 0.8$).

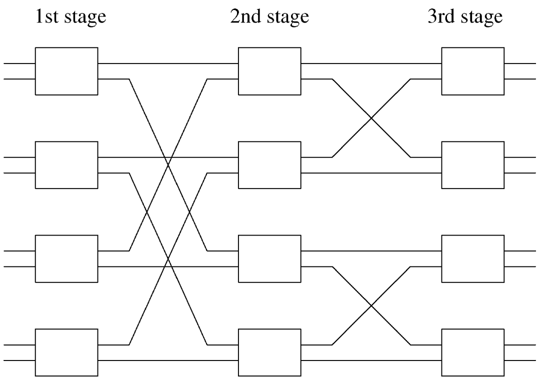


Fig. 9. An illustrating example for implementing the 8×8 crossbars in the load balanced Birkhoff–von Neumann switch.

as they behave as if they were split from Poisson processes. Thus, when N is large and $x \gg N$, we expect to have the following approximation

$$P(q_1(\infty) \geq x) \approx e^{-\theta^* x}. \tag{41}$$

Comparing this with Eq. (38), we conclude that the decay rate of the tail distribution of queue length in the load balanced Birkhoff–von Neumann switch is much smaller than that in the corresponding output-buffered switch. This implies that if we allocate the same finite amount of buffer in each port of both switches (without sharing with other ports), the load balanced Birkhoff–von Neumann switch has much smaller packet loss probability than that in the output-buffered switch.

We verify our observation by simulation. In this experiment, we allocate the same amount of buffer to each port in the load balanced Birkhoff–von Neumann switch and the output-buffered switch (without sharing with other ports). We run the simulations under the uniform bursty traffic model with the arrival rate $\rho = 0.8$ in both switches. The simulation results for packet loss probabilities are shown in Fig. 8. By solving Eq. (36), we find $\theta^* \approx 0.4575$. The results in Eqs. (38) and (41) match well with the slopes in both curves in Fig. 8. This experiment further verifies our observation that the load balanced Birkhoff–von Neumann switch has a much smaller packet loss probability than that in the corresponding output-buffered switch when the buffer is large.

6. Implementation and scalability issues

The load balanced Birkhoff–von Neumann switch requires two $N \times N$ crossbar switches. This may not be scalable for large N. To build large crossbar switches, it is well known that one can reduce complexity by using the three-stage Clos networks [12] (for additional information on multi-stage networks, we refer to Refs. [14,30]). By recursively expanding the three-stage Clos networks, one can then build an $N \times N$ crossbar switch by using 2×2 switches. This is known as the Benes network and the number of 2×2 switches needed is $(2 \log_2 N - 1)N/2$.

Now we show that the complexity of building the crossbars in the load balanced Birkhoff–von Neumann switch can be further reduced by using the Banyan network. The key observation is that we do not need to realize all the permutation matrices in the crossbar. The connection patterns in the load balanced Birkhoff–von Neumann switch are periodically generated via a one-cycle permutation matrix. Thus, we only need to realize the N permutation matrices generated via a one-cycle permutation matrix. In Fig. 9, we illustrate how one implements an 8×8 crossbar in the load balanced Birkhoff–von Neumann switch by the Banyan network with 2×2 switches. Note that there are only two connection patterns in a 2×2 switch. In Fig. 9, we set the connection patterns at the first stage to toggle every time slot, the connection patterns at the second stage to toggle every two time slots, and the connection patterns at the third stage to toggle every four time slots. (In the general case, the connection patterns at the nth stage are set to toggle every 2^{n-1} time slots.) By doing so, the connection patterns repeat themselves every 8 time slots and we have all the connection patterns needed for the load balanced Birkhoff–von Neumann switch. Note that the number of 2×2 switches needed for the $N \times N$ Banyan network is only $(N \log_2 N)/2$, which is much smaller than that for the Benes network.

The Banyan network was previously used for load balancing via a randomization technique in Refs. [34,26]. Instead of using the deterministic connection patterns as ours, the randomization technique uses the self-routing property of the Banyan network. In the randomization technique, every packet, upon its arrival at the first stage, randomly selects an output port at the first stage. The packet is then routed through the Banyan network at the first stage via the self-routing property of the Banyan network. The problem of the randomization technique is internal blocking. There might be two or more packets that share a common internal link in the Banyan network. As a result, packets might be lost inside the Banyan network, and this leads to throughput degradation. The internal blocking problem can be solved by adding a sorting network in front of the Banyan network. This results in the Batcher–Banyan network (see e.g. Ref. [14]). However, even the Batcher–Banyan cannot solve the external blocking problem when two or more packets are destined for the same output port. To solve the external blocking problem, an additional conflict resolution phase is added in the three phase switching network described in Ref. [14]. To summarize, traditional multi-stage networks aim to realize *all* the permutation matrices and thus have much higher implementation complexity than the load balanced Birkhoff–von Neumann switch. The VOQs in front of the inputs of the second stage can be viewed as ‘share memory’ switches. It is not necessary to have N of them for the switch to work. To see this, suppose there is exactly one of them, say the first one. Then one can disable packet transmissions from the first stage to any output ports but the first one. In this case, the first stage acts as a concentrator and the load balanced Birkhoff–von Neumann switch is reduced to the standard share memory switch (see e.g. Ref. [30] for more details of share memory switches). Certainly, the line rate for this case should be reduced to 1/N of the rate of the crossbar switch. This implies that the number of share memory switches (VOQs) in front of the inputs of the second stage can be gradually added to the maximum number N as the line rates are gradually upgraded. Thus, the load balanced Birkhoff–von Neumann switch can be an expandable switch.

The load balanced Birkhoff–von Neumann switch can also be viewed as a three-stage switch if one considers the VOQs in front of the inputs of the second stage as share memory switches. These three stages then include a crossbar at the first stage (Space switch), share memory switches in the middle stage (Time switch), and another crossbar at the last stage (Space switch). Such an S–T–S arrangement is quite different from the traditional T–S–T arrangement for the three-stage Clos network, where the time switches require a fixed frame size. The frame size in the time switches determines the number of switching patterns that can be realized by the T–S–T switch. In our S–T–S arrangement, there is no need to choose a frame size for the share memory (time) switches. In fact, its buffer usage depends on the traffic load. When the load increases, the buffer usage also increases. As a result, the number of switching patterns that can be realized by the S–T–S arrangement also increases. In view of this, the S–T–S arrangement is more flexible than the T–S–T arrangement and more suitable for packet switching.

Finally, as commented in the recent paper by Keslassy and McKeown [19], the two-stage switching fabrics can be implemented by a single optical switch fabric with micromirrors. This is because light is bi-directional and the connection patterns at the two stages are simply the permutation matrices generated by a one-cycle permutation matrix.

7. Conclusions

Motivated by the need for a simple and high performance switch architecture that scales up with the speed of fiber optics, we proposed the two-stage load balanced Birkhoff–von Neumann switch. The first stage performs load balancing, while the second stage is a Birkhoff–von Neumann input-buffered switch that performs switching for load balanced traffic. We showed that the switch is scalable, with low hardware complexity, and with 100% throughput if the traffic is weakly mixing.

Since load balancing not only converts non-uniform traffic into uniform traffic, but also performs burst reduction for incoming traffic, we showed that load balancing is quite effective in reducing delay when the traffic is heavy and bursty. In the uniform bursty traffic model, the average delay of the load balanced Birkhoff–von Neumann switch is proven to converge to that of an output-buffered switch under heavy load. We also demonstrated that load balancing is more effective than the conflict resolution algorithm, *i*-SLIP [23], in heavy load. This is because the *i*-SLIP algorithm might be trapped in bad modes when the traffic is heavy and bursty.

Burst reduction also yields much more efficient buffer usage. When both the load balanced Birkhoff–von Neumann switch and the corresponding output-buffered switch are allocated with the same finite amount of buffer at each port, we showed from both the theory of effective bandwidth and simulations that the packet loss probability in the load balanced Birkhoff–von Neumann switch is much smaller than that in an output-buffered switch when the buffer is large. This implies that exact emulation of an output-buffered switch by a CIOQ switch [10,32] may not perform well when the traffic is bursty.

The obvious drawback of the switch is that FIFO might be violated for packets from the same input. This might be fixed by adding a resequencing buffer at the output port. However, this increases the complexity of the hardware design and whether resequencing should be performed at the core routers might need further investigation [11]. In the sequel [7], we provide some solutions for solving the resequencing problem in the load balanced Birkhoff–von Neumann switch.

Acknowledgements

This research is supported in part by the National Science Council, Taiwan, ROC, under Contract NSC-88-2213-E007-004 and the program for promoting academic excellence of universities 89-E-FA04-1-4.

References

[1] T. Anderson, S. Owicki, J. Saxes, C. Thacker, High speed switch scheduling for local area networks, ACM Trans. Comput. Syst. 11 (1993) 319–352.

[2] F. Baccelli, P. Bremaud, Elements of Queueing Theory, Springer, New York, 1994.

[3] G. Birkhoff, Tres observaciones sobre el algebra lineal, Univ. Nac. Tucumán Rev. Ser. A 5 (1946) 147–151.

[4] C.S. Chang, Performance Guarantees in Communication Networks, Springer, London, 2000.

[5] C.S. Chang, W.J. Chen, H.Y. Huang, On service guarantees for input buffered crossbar switches: a capacity decomposition approach by Birkhoff and von Neumann, IEEE IWQoS'99, London, UK, 1999 (US patent pending), pp. 79–86.

[6] C.S. Chang, W.J. Chen, H.Y. Huang, Birkhoff–von Neumann input buffered crossbar switches, IEEE INFOCOM2000, Tel Aviv, Israel, 2000, pp. 1614–1623.

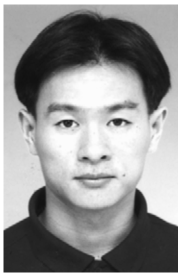
[7] C.S. Chang, D.S. Lee, C.M. Lien, Load balanced Birkhoff–von Neumann Switches, Part II: multi-stage buffering, Comput. Commun. (2001) Current Issues in Terabit Switching, in press.

[8] A. Charny, P. Krishna, N. Patel, R. Simcoe, Algorithms for providing

bandwidth and delay guarantees in input-buffered crossbars with speedup, IEEE IWQoS'98, Napa, CA, 1998, pp. 235–244.
[9] G.L. Choudhury, W. Whitt, Long-tail buffer-content distributions in broadband networks, Performance Evaluation 30 (1997) 177–190.
[10] S.-T. Chuang, A. Goel, N. McKeown, B. Prabhkar, Matching output queueing with a combined input output queued switch, IEEE INFOCOM'99, New York, 1999, pp. 1169–1178.
[11] C. Diot, personal communication.
[12] C. Clos, A study of nonblocking switching networks, BSTJ 32 (1953) 406 see also p. 424.
[13] A. Demers, S. Keshav, S. Shenkar, Analysis and simulation of a fair queueing algorithm, Proceedings of SIGCOMM'89, Austin, TX, September 1989, pp. 1–12.
[14] J. Hui, Switching and Traffic Theory for Integrated Broadband Networks, Kluwer Academic Publishers, Boston, 1990.
[15] A. Hung, G. Kesidis, N. McKeown, ATM input-buffered switches with guaranteed-rate property, Proceedings of IEEE ISCC'98, Athens, 1998, pp. 331–335.
[16] P.R. Jelenković, A.A. Lazar, Subexponential asymptotics of a Markov-modulated random walk with queueing applications, J. Appl. Prob. (1998).
[17] W.E. Leland, M.S. Taqqu, W. Willinger, D.V. Wilson, On the self-similar nature of ethernet traffic, IEEE/ACM Trans. Networking 2 (1994) 1–15.
[19] I. Keslassy, N. McKeown, Maintaining packet order in two-stage switches, preprint, 2001.
[20] P. Krishna, N.S. Patel, A. Charny, R. Simcoe, On the speedup required for work-conserving crossbar switches, IEEE IWQoS'98, Napa, CA, 1998, pp. 225–234.
[21] T.T. Lee, C.H. Lam, Path switching-a quasi-static routing scheme for large scale ATM packet switches, IEEE J. Sel. Areas Commun. 15 (1997) 914–924.
[22] S. Li, N. Ansari, Input-queued switching with QoS guarantees, IEEE INFOCOM'99, New York, 1999, pp. 1152–1159.
[23] N. McKeown, Scheduling algorithms for input-queued cell switches, PhD Thesis, University of California at Berkeley, 1995.
[24] N. McKeown, V. Anantharam, J. Walrand, Achieving 100% throughput in an input-queued switch, Proceedings of IEEE INFOCOM'96, 1996, pp. 296–302.
[25] A. Mekkittikul, N. McKeown, A practical scheduling algorithm to achieve 100% throughput in input-queued switches, Proceedings of IEEE INFOCOM'98.
[26] D. Mitra, R.A. Cieslak, Randomized parallel communications on an extension of the omega network, J. Assoc. Comput. Machinery 34 (4) (1987) 802–824.
[27] M.G. Nadkarni, Basic Ergoidc Theory, Birkhaüser, Berlin, 1998.
[28] A.K. Parekh, R.G. Gallager, A generalized processor sharing approach to flow control in integrated service networks: the single-node case, IEEE/ACM Trans. Networking 1 (1993) 344–357.
[29] K. Petersen, Ergodic Theory, Cambridge University Press, Cambridge, 1983.
[30] M. Schwartz, Broadband Integrated Networks, Prentice-Hall, Englewood Cliffs, NJ, 1996.
[31] D. Stiliadis, A. Varma, Providing bandwidth guarantees in an input-buffered crossbar switch, Proceedings of IEEE INFOCOM'95, 1995, pp. 960–968.
[32] I. Stoica, H. Zhang, Exact emulation of an output queueing switch by a combined input output queueing switch, IEEE IWQoS'98, Napa, CA, 1998, pp. 218–224.
[33] D. Stoyan, Comparison Methods for Queues and Other Stochastic Models, Wiely, Berlin, 1983.
[34] L.G. Valiant, A scheme for fast parallel communication, SIAM J. Comput. 11 (2) (1982) 350–361.
[35] J. von Neumann, A certain zero-sum two-person game equivalent to the optimal assignment problem, Contributions to the Theory of Games, vol. 2, Princeton University Press, Princeton, NJ, 1953 pp. 5–12.

__Cheng-Shang Chang__ received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1983, and the M.S. and Ph.D. degrees from Columbia University, New York, NY, in 1986 and 1989, respectively, all in Electrical Engineering. From 1989 to 1993 he was employed as a Research Staff Member at the IBM Thomas J. Watson Research Center, Yorktown Heights, N.Y. In 1993, he joined the Department of Electrical Engineering at National Tsing Hua University, Taiwan, R.O.C., where he is a Professor. His current research interests are concerned with queueing theory, stochastic scheduling and performance evaluation of telecommunication networks and parallel processing systems. Dr. Chang received the IBM Outstanding Innovation Award in 1992, and the Outstanding Research Award from the National Science Council, Taiwan, in 1999 and 2001. He is the author of the book "Performance Guarantees in Communication Networks," and he served as an editor for Operations Research from 1992 to 1999.

__Duan-Shin Lee__ was born in Taiwan in 1961. He received the B.S. degree from National Tsing Hwa University, Taiwan, in 1983, and the M.S. and Ph.D. degrees from Columbia University, New York, in 1987 and 1990, all in electrical engineering. He worked as a research staff member at the C&C Research Laboratory of NEC USA, Inc. in Princeton, New Jersey from 1990 to 1998. Since 1998, he has been an associate professor in the Department of Computer Science of National Tsing Hwa University in Hsinchu, Taiwan. His research interests are switch and router design, personal communication systems, performance analysis computer networks and queueing theory. Dr. Lee is a senior member of IEEE.



__Yi-Shean Jou__ received the B.S. and M.S. degrees in electrical engineering from National Tsing Hua University, Taiwan, in 1998 and 2000, respectively. Since 2000, He has been with TECOM Co., LTD, Taiwan as a R&D engineer.