Constructions of Optical FIFO Queues

Cheng-Shang Chang, *Fellow, IEEE*, Yi-Ting Chen, and Duan-Shin Lee, *Senior Member, IEEE*,
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
Email: cschang@ee.nthu.edu.tw
ytchen@gibbs.ee.nthu.edu.tw
lds@cs.nthu.edu.tw

***Abstract*— Discrete-time queues are infinite dimensional switches in time. Ever since Shannon [22] published his paper on the memory requirements in a telephone exchange, there have been tremendous efforts in the search for switches with minimum complexity. Constructing queues with minimum complexity has not received the same amount of attention as queues are relatively cheap to build via electronic memory. Recent advances in optical technologies, however, have spurred interest in building optical queues with minimum complexity.**

In this paper, we develop mathematical theory of constructing discrete-time optical FIFO queues. To our surprise, we find that many classical constructions for switches have their counterparts for constructing queues. Analogous to the three-stage construction of Clos networks, we develop a three-stage construction of optical FIFO queues via Switched Delay lines (SDL). Via recursively expanding the three-stage construction, we show that an optical FIFO queue with buffer 2^n-1 can be constructed by using $2n$ 2×2 switches with the total fiber length $3\cdot 2^{n-1}-2$.

***Index Terms*— FIFO queues, optical switches, optical memory, Clos networks, switched delay lines**

I. Introduction

Discrete-time queues can be viewed as infinite dimensional switches in time. To illustrate this, in Figure 1 we show a typical sample path of a queue with a single input link and a single output link. The first customer arrives at time $t=1$ and departs at time $t=7$, the second customer arrives at time $t=3$ and departs at time $t=5$, the third customer arrives at time $t=5$ and departs at time $t=9$, and so forth. As shown in Figure 1, the queue that realizes this particular sample path can be viewed as a switch that sets up a particular connection pattern between the inputs and the outputs. Unlike traditional switches, the inputs and the outputs in a queue are infinite dimensional as $t\to\infty$. A natural question is then: How does one construct a queue and how complex is it to do so?

Ever since Shannon [22] published his paper on the memory requirements in a telephone exchange, there have been tremendous efforts in the search for switches with minimum complexity (see e.g., the books by V. E. Benes [1], J. Hui [16], M. Schwartz [21], F. K. Hwang [17] and S.-Y. R. Li [19], and references therein). However, constructing queues with minimum complexity has not received the same amount of attention as constructing switches because queues are relatively cheap to build via electronic memory. Due to recent advances in optical technologies, the data transmission speed by photons is now much faster than that by electrons. As it is very costly to store information in electronic memory by converting photons into electrons, building optical queues with minimum complexity has become an important research topic.

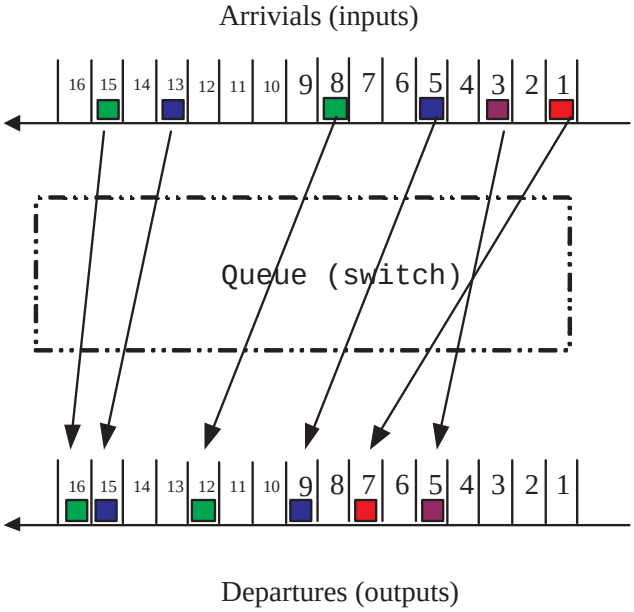

Fig. 1. A typical sample path of a discrete-time queue

The only known way to store photons without converting them into other media is to direct photons via a set of Switches and fiber Delay Lines (SDL) so that the photons come out at the right place and at the right time. The development of optical queues via switched delay lines (SDL) seems to have started in the early 1990s. Early development seems to have focused more on the practical side than developing mathematical theory. It was first demonstrated by M. J. Karol [18] that SDL elements could be used as a buffer for a shared-memory optical packet switch. The buffer in [18] was built by SDL elements with feedbacks (like the optical memory cell in Section II). However, no proofs were given for exact emulation of a shared-memory switch. A huge project (see [4], [5]), called CORD (contention resolution by delay lines), was

This research is supported in part by the National Science Council, Taiwan, R.O.C., under Contract NSC-93-2213-E-007-040, and the Program for Promoting Academic Excellence of Universities NSC 94-2752-E-007-002-PAE.

started by I. Chlamtac et al at Boston University. Once again, no formal proofs for exact emulation of an output-buffered switch (or multiplexer) were given in [4], [5].

It seems that J. T. Tsai and R. L. Cruz [23], [11] were the first to construct an exact 2-to-1 First In First Out (FIFO) multiplexer with SDL elements. The multiplexer in [23], [11], named COD (Cascaded Optical Delay-Lines), only required local information for the control of the connection patterns of 2×2 switches. However, the number of 2×2 switches in such an architecture is proportional to the buffer size. A more efficient design, called Logarithm Delay-Line Switch, was proposed by D. K. Hunter, M. C. Chia and I. Andonovic in [13]. The 2-to-1 FIFO multiplexer in [13] turned out to be the recursively expanded version of the 2-to-1 FIFO multiplexer present in [9]. As addressed in [9], the number of 2×2 switches needed for such an architecture is only $O(\log B)$, where B is the buffer size. An extension to FIFO multiplexers with variable length bursts was reported in [8]. In [15], SLOB (Switch with Large Optical Buffers) was proposed for the extension of optical buffered switches with N input/output ports ($N \geq 2$). Such an architecture relied on a special hardware, called a primitive switching element (PSE), which was very difficult to control. Finally, we note that a "packing" and "scheduling" optical switch that used the framed Birkhoff-von Neumann decomposition [2], [25] was introduced by E. A. Varvarigos [24]. For additional references of optical packet switches, we refer to the review papers [14], [12], [26].

One of the main contributions of our paper is to develop mathematical theory of constructing discrete-time optical FIFO queues. To our surprise, we find that many classical constructions for switches have their counterparts for constructing queues. Analogous to the three-stage construction of Clos networks [10], we develop a three-stage construction of optical FIFO queues via SDL elements. Via recursively expanding the three-stage construction, we show that an optical FIFO queue with buffer $2^n - 1$ can be constructed by using $2n$ 2×2 switches with the total fiber length $3 \cdot 2^{n-1} - 2$.

The paper is organized as follows. In Section II, we explain the motivation of our research by introducing optical memory cells and SDL elements. Our construction for optical FIFO queues is given in Section III. The paper is then concluded in Section IV.

II. Optical memory cells and SDL elements

There are several well-known approaches for solving the conflicts in high speed packet switches with electronic memory. The key problem of extending these approaches to optical switches is the lack of inexpensive optical random access memory. A memory cell in electronic memory can be easily implemented by a few transistors that store electrical charges. As such, the size of electronic random access memory can be very large, e.g., 512Mbits. Thus, the cost of using electronic random access memory is usually assumed to be independent of the size of memory. Such an assumption is called the *uniform* cost assumption in the literature. However, it is much more difficult to store photons. One way to implement a memory cell for optical memory is to use a 2×2 optical crossbar switch and a fiber delay line (with one unit of delay) as shown in Figure 2. To write the information to the memory cell, set the 2×2 crossbar switch to the "cross" state so that photons can be directed to the fiber delay line. Once the write operation is completed, the crossbar switch is then to set to the "bar" state so that the photons directed into the fiber delay line keep circulating through the fiber delay line. To read out the information from the memory cell, set the crossbar switch to the "cross" state so that the photons in the fiber delay line can be directed to the output link. Unlike transistors, the cost of a 2×2 optical crossbar switch is high in today's technology. Thus, it is important to build an optical queue with a minimum number of 2×2 optical crossbar switches.

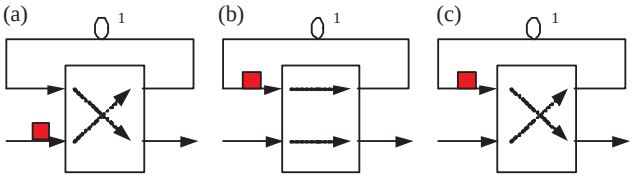

Fig. 2. An optical memory cell: (a) writing information (b) circulating information (c) reading information

A network element that is built by optical crossbar switches and fiber delay lines as described in Figure 2 is called a Switched Delay Line (SDL) element. In this paper, we consider fixed size packets over optical links. Assume that time is slotted and synchronized so that a packet can be transmitted within a time slot. Since there is at most one packet within a time slot, we may use indicator variables to represent the state of a link. A link is in state 1 at time t (for some $t = 0, 1, 2, \ldots$) if there is a packet in the link at time t, and it is in state 0 at time t otherwise. For instance, we show in Figure 3 a delay line with delay d. Let $a(t)$ be the state of the input link. Then the state of the output link is $a(t-d)$. Note that at the end of the t^{th} time slot, the packets that arrive at time $t, t-1, \ldots, t-(d-1)$, are stored in the optical delay line with delay d.

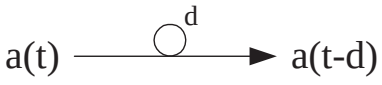

Fig. 3. An optical delay line with delay d

One of the most important properties of SDL elements is the following time interleaving property for scaled SDL elements.

Definition 1 (Scaled SDL element [9]) *A scaled SDL element is said to be with scaling factor m if the delay in every delay line is m times of that in the original (unscaled) SDL element.*

Proposition 2 (Time interleaving property [9]) *A scaled SDL element with scaling factor m can be operated as time interleaving of m SDL elements.*

A formal argument for Proposition 2 can be found in [9]. To understand the intuition of the time interleaving property,

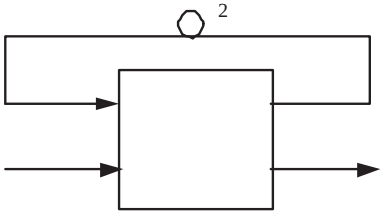

Fig. 4. An optical memory cell with scaling factor 2

consider the memory cell with scaling factor 2 in Figure 4. To see that the scaled memory cell in Figure 4 can be operated as time interleaving of two memory cells, we partition time into even and odd numbered time slots. For the even numbered time slots, we can set the connection patterns of the 2×2 optical crossbar switch in the scaled SDL element according to the read/write operation described in Figure 2 for one memory cell. Similarly, for the odd numbered time slots, we can set the connection patterns of the 2×2 optical crossbar switch in the scaled SDL element according to the read/write operation for another memory cell.

III. FIFO QUEUES

FIFO queues are widely used in every one's daily life. A customer arriving at a FIFO queue joins the tail of the queue. When a customer departs at the head of the queue, every one in a FIFO queue moves up one position. If the buffer of a FIFO queue is finite, then an arriving customer to a full queue is lost. The concept of a discrete-time FIFO queue is formalized in the following definition.

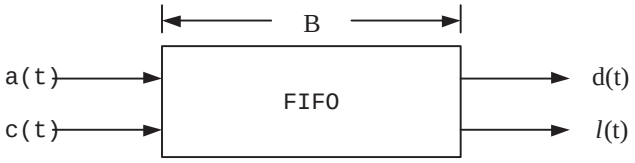

Fig. 5. A FIFO queue with buffer B.

Definition 3 (FIFO queue) *A FIFO queue with buffer B is a network element that has one input link, one control input and two output links (see Figure 5). One output link is for departing packets and the other is for lost packets. As shown in Figure 5, let $a(t)$ be the state of the input link, $c(t)$ be the state of the control input, $d(t)$ (resp. $\ell(t)$) be state of the output link for departing (resp. lost) packets, and $q(t)$ be the number of packets queued at the FIFO queue at time t (at the end of the t^{th} time slot). Then the FIFO queue with buffer B satisfies the following four properties:*

(P1) *Flow conservation: arriving packets from the input link are either stored in the buffer or transmitted through the two output links, i.e.,*

$$q(t) = q(t-1) + a(t) - d(t) - \ell(t). \quad (1)$$

(P2) *Non-idling: if the control input is enabled, i.e., $c(t) = 1$, then there is always a departing packet if there are packets in the buffer or there is an arriving packet, i.e.,*

$$d(t) = \begin{cases} 1 & \text{if } c(t) = 1 \text{ and } q(t-1) + a(t) > 0 \\ 0 & \text{otherwise} \end{cases}. \quad (2)$$

(P3) *Maximum buffer usage: if the control input is not enabled, i.e., $c(t) = 0$, then an arriving packet is lost only when buffer is full, i.e.,*

$$\ell(t) = \begin{cases} 1 & \text{if } c(t) = 0,\ q(t-1) = B \text{ and } a(t) = 1 \\ 0 & \text{otherwise} \end{cases}. \quad (3)$$

(P4) *FIFO: packets depart in the first in first out (FIFO) order.*

One has from (P1) and (P2) that

$$q(t) = (q(t-1) + a(t) - c(t))^+ - \ell(t).$$

In conjunction with (P3), one further has the following Lindley equation

$$q(t) = \min[(q(t-1) + a(t) - c(t))^+, B]. \quad (4)$$

The key difference between a 2-to-1 FIFO multiplexer in [9] and a FIFO queue is that the delay of a packet in a 2-to-1 FIFO multiplexer can be immediately determined upon its arrival. This is not possible in a FIFO queue as the delay of an arriving packet depends on the future of the control input $c(t)$. We note that the control input $c(t)$ is also known as the time varying capacity of the discrete-time FIFO queue in the literature (see e.g., [6] and references therein).

A. Three-stage constructions

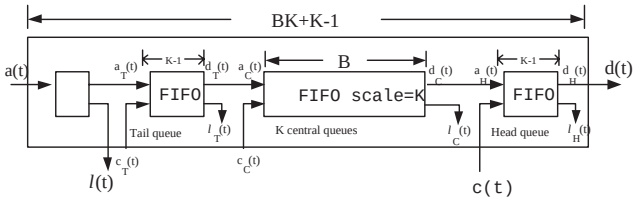

Fig. 6. A three-stage construction of a FIFO queue

In Figure 6, we show a three-stage construction of a FIFO queue with buffer $BK + K - 1$. It is a concatenation of a 1×2 switch, a FIFO queue with buffer $K - 1$ (tail queue), a scaled FIFO queue with buffer B and scaling factor K (K parallel central queues), and a FIFO queue with buffer $K - 1$ (head queue). The 1×2 switch on the left acts as a 1-to-2 demultiplexer. Its objective is for admission control. An arriving packet is admitted only when the total number of packets inside the network element does not exceed $BK + K - 1$ after its admission. Otherwise, an arrival is lost and it is routed to the loss port $\ell(t)$ (as shown in Figure 6). By so doing, the maximum number of packets inside the network element is at most $BK + K - 1$.

To represent the state of the head queue, we let $a_H(t)$ be the state of its input link, $c_H(t)$ be the state of its control input, $d_H(t)$ be the state of its output link for departing packets,

$\ell_H(t)$ be the state of its output link for lost packets, and $q_H(t)$ be the number of packets queued at the head queue at time t. Similarly, we let $a_T(t)$, $c_T(t)$, $d_T(t)$, $\ell_T(t)$, and $q_T(t)$ denote the corresponding states in the tail queue.

From the time interleaving property for SDL elements, the scaled FIFO queue with buffer B and scaling factor K can be operated as K parallel queues. These K time interleaved parallel queues are connected to the head queue and the tail queue periodically with period K. An illustrating graph is shown in Figure 7.

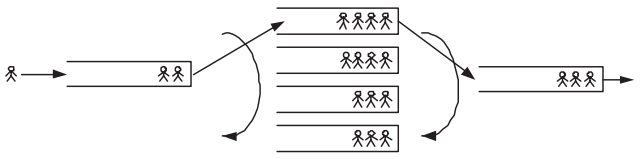

Fig. 7. An illustration of periodic connections in the three-stage construction

To simplify our presentation, we let $a_C(t)$ be the state of input link of the central FIFO queue that is connected by the head queue and the tail queue at time t. Also, let $c_C(t)$ be the state of its control input, $d_C(t)$ be the state of its output link for departing packets, $\ell_C(t)$ be the state of its output link for lost packets, and $q_C(t)$ be the number of packets stored in that central queue. Let $\hat{q}_C(t)$ be the total number of packets stored in the K central queues at time t and

$$q(t) = q_T(t) + \hat{q}_C(t) + q_H(t) \tag{5}$$

be the total number of packets stored in the network element. To summarize, a subscript H (resp. T, C) indicates that its is a state variable of the *head* (resp. *tail, connected central*) queue.

From the three-stage construction in Figure 6, we have $a_T(t) = a(t)$ (if the arrival is not lost), $d_T(t) = a_C(t)$, $d_C(t) = a_H(t)$, and $d(t) = d_H(t)$.

To operate the three-stage construction in Figure 6 as a FIFO queue, we let the control input of the head queue $c_H(t)$ to be the control input of the overall FIFO queue $c(t)$, i.e., $c_H(t) = c(t)$. To complete the operation of the network element, it remains to specify the control input of the tail queue $c_T(t)$ and the control input of the connected central queue $c_C(t)$.

Define a busy period as the period of time that there are packets stored in the central queues, i.e., $\hat{q}_C(t) > 0$. An idle period is the period of time that there are no packets stored in the central queues, i.e., $\hat{q}_C(t) = 0$. Initially, the network element is empty and it is in an idle period.

(R1) (Idle period rule) In an idle period, the tail queue and the central FIFO queues are always enabled as long as the head queue is not full, i.e., $c_T(t) = c_C(t) = 1$ if $\hat{q}_C(t-1) = 0$ and $q_H(t-1) - c(t) < K-1$.

According to the idle period rule, the tail queue and the scaled FIFO queue are transparent during an idle period and the network element is completely determined by the head queue. Thus, during an idle period, the network element is a FIFO queue with buffer $K-1$.

(R2) (Initiation of a busy period) When the head queue is full and there is an arriving packet, the packet has to be stored in one of the central queues and this triggers a busy period. Thus, if $\hat{q}_C(t-1) = 0$ and $q_H(t-1) - c(t) = K-1$, then $c_T(t) = 1$ and $c_C(t) = 0$.

To specify the operation rules in a busy period, we need to keep track of the shortest queue and the longest queue (cf. the argument for a system with parallel queues in [9]). Let $A(t_1, t_2)$ be the number of arrivals to the central queues between $[t_1, t_2 - 1]$, i.e.,

$$A(t_1, t_2) = \sum_{t=t_1}^{t_2-1} a_C(t). \tag{6}$$

Also, let $D(t_1, t_2)$ be the number of departures from the central queues between $[t_1, t_2 - 1]$, i.e.,

$$D(t_1, t_2) = \sum_{t=t_1}^{t_2-1} d_C(t). \tag{7}$$

Suppose that a busy period begins at time τ. As packets are of the same size, the joining-the-shortest-queue policy is simply the round robin assignment of the arriving packets in a busy period. Thus, the connected central queue at time t is the shortest queue if and only if

$$(t - \tau - A(\tau, t)) \bmod K = 0.$$

Similarly, the connected central queue at time t is the longest queue if and only if

$$(t - \tau - D(\tau, t)) \bmod K = 0.$$

(R3) (Serving-the-longest-queue rule) In a busy period, there are two conditions that need to be met in order to enable a packet to depart from the connected central queue to the head queue: (i) there is a buffer space in the head queue, and (ii) the central queue being connected is indeed the longest queue. Specifically, suppose that $\hat{q}_C(t-1) > 0$. Then $c_C(t) = 1$ if and only if $q_H(t-1) - c(t) < K-1$ and $(t - \tau - D(\tau, t)) \bmod K = 0$.

(R4) (Joining-the-shortest-queue rule) In a busy period, there are two conditions that need to be met in order to enable a packet to depart from the tail queue to the connected central queue: (i) there is a buffer space in the connected central queue, and (ii) the central queue being connected is indeed the shortest queue. Specifically, suppose that $\hat{q}_C(t-1) > 0$. Then $c_T(t) = 1$ if and only if $q_C(t-1) - c_C(t) < B$ and $(t - \tau - A(\tau, t)) \bmod K = 0$.

In the following theorem, we prove the main result for the three-stage construction of FIFO queues. Its proof is given in Appendix A.

Theorem 4 *Suppose that the network element in Figure 6 is started from an empty system. Under the operation rules specified in (R1)-(R4), it is a FIFO queue with buffer* $BK + K - 1$.

Note that the first 1×2 switch in the three-stage construction in Figure 6 is only to make sure that the total number of

packets inside the network element does not exceed $BK + K - 1$. In other words, this 1×2 switch can be omitted in the construction as long as the queue never exceeds its buffer size. For this, we call a network element a pre-FIFO queue with buffer B if it behaves exactly the same as a FIFO queue with buffer B as long as the queue never exceeds its buffer size, i.e., it can realize all the sample paths that do not lead to a buffer overflow. For instance, an optical memory cell is a pre-FIFO queue with buffer 1. As there is no internal loss in the three-stage construction in Figure 6, the FIFO queues there can be replaced by pre-FIFO queues. As such, one can build a pre-FIFO queue with buffer $BK + K - 1$ by two pre-FIFO queues with buffer $K - 1$ and a scaled pre-FIFO queue with buffer B and scaling factor K. Let $H(K)$ be the number of 2×2 switches needed for constructing a pre-FIFO queue with buffer K. From the three-stage construction, it follows that

$$H(BK + K - 1) = 2H(K - 1) + H(B). \tag{8}$$

As an optical memory cell can be used for a pre-FIFO queue with buffer 1, we have $H(1) = 1$. Letting $K = 2$ in (8) yields

$$H(2B + 1) = 2 + H(B). \tag{9}$$

Solving this yields

$$H(2^n - 1) = 2n - 1. \tag{10}$$

In fact, the recursive expansion using $K = 2$ can be used for building a pre-FIFO queue with buffer $2^n - 1$ (see Figure 8 for an implementation of a pre-FIFO queue with buffer 7). As one can add 1×2 switch in front of a pre-FIFO queue for dropping overflowed packets, a FIFO queue with buffer $2^n - 1$ can be constructed by using $2n$ 2×2 switches with the total fiber length $3 \cdot 2^{n-1} - 2$.

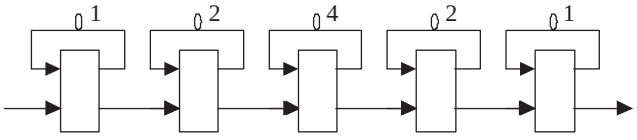

Fig. 8. An implementation of a pre-FIFO queue with buffer 7 via a concatenation of scaled optical memory cells

Even though the total number of buffers in the three-stage construction is $BK+2(K-1)$, it is not possible to admit more than $BK+K-1$ packets without violating the properties of a FIFO queue. To see this, suppose that we relax the admission control rule by admitting at most $BK + K$ packets. Consider the following scenario: the head queue and the central queues are all full at time $t-1$, i.e., $q_H(t-1) = K-1$ and $\hat{q}_C(t-1) = BK$, and the connected central queue at time t is both the shortest queue and the longest queue. Suppose that $c(t) = 0$, $a(t) = 1$ and that $a(s) = c(s) = 1$ for $s = t+1, \ldots, t+K-1$. According to the relaxed admission control rule, the packet that arrives at time t is admitted to the tail queue and we have $q_T(t) = 1$, $q_H(t) = K - 1$ and $\hat{q}_C(t) = BK$. As the connected central queue at time t is both the shortest queue and the longest queue, the connected central queue and the tail queue are not enabled for $s = t + 1, \ldots, t + K - 1$. Thus, all the subsequent arrivals have to be store at the tail queue. Moreover, the departures from the head queue are not replenished by the packets from the central queues. At time $t+K-2$, we then have $q_H(t+K-2) = 1$, $\hat{q}_C(t+K-2) = BK$ and $q_T(t + K - 2) = K - 1$. Even though the last packet in the head queue will depart at $t + K - 1$, we are not able to admit the arriving packet at $t + K - 1$ as the tail queue is full. This violates the maximum buffer usage property for a FIFO queue with buffer $BK + K$.

IV. Conclusions

In this paper, we developed a three-stage construction for FIFO queues. Recursive expansion of the three-stage construction yields an optical FIFO queue that is a concatenation of scaled optical memory cells. In our implementation of the three-stage construction of a FIFO queue, we need to keep track of the longest queue and the shortest queue. It becomes troublesome when one recursively expands the three-stage construction. It would be of interest to look for simple control mechanisms for FIFO queues.

We note that many classical constructions of switches can also be used for the constructions of optical queues, including linear compressors, non-overtaking delay lines, and flexible delay lines. Results along this line can be found in [7]. In addition to FIFO queues, we also note that there is a construction of priority queues in [20] using a feedback architecture.

Appendix

Appendix A

In this appendix, we prove Theorem 4.

To prove the three-stage construction in Figure 6 is indeed a FIFO queue with buffer $BK + K - 1$, we need to verify the four properties in Definition 3. From (R1), the construction is the same as a FIFO queue with buffer $K-1$ in an idle period. It suffices to verify these four properties in a busy period.

(P1) Flow conservation: it is easy to see that under the serving-the-longest-queue rule in (R3), $q_H(t) \leq K - 1$ for all t and there is no loss in the head queue, i.e., $\ell_H(t) = 0$. Similarly, under the joining-the-shortest-queue rule in (R4), $q_C(t) \leq B$ for all t and we have $\ell_C(t) = 0$. We will show in (P3) that there is no loss in the tail queue, i.e., $\ell_T(t) = 0$, as long as the total number of packets inside the network element is not greater than $BK + K - 1$. As such, flow conservation can be preserved.

(P2) Non-idling: we will prove this by contradiction. Suppose that a busy period begins at time τ and the non-idling property is violated for the first time at time $t_0 > \tau$. From the non-idling property of the head queue, the head queue must be empty at $t_0 - 1$. Moreover, a packet p in the longest queue (called queue q) cannot be dequeued to the empty head queue as queue q is not connected at t_0. Mathematically, we have (i) the (external) control input is enabled, i.e., $c(t_0) = 1$, (ii) the head queue is empty, i.e., $q_H(t_0 - 1) = 0$, and (iii) the central queues are not empty, i.e., $\hat{q}_C(t_0 - 1) > 0$, and (iv) the connected central queue is not the longest queue, i.e.,

$$(t_0 - \tau - D(\tau, t_0)) \bmod K \neq 0. \tag{11}$$

Let t_1 be the time that the last packet is dequeued to the head queue from the central queues. If no packet is dequeued

to the head queue, let $t_1 = \tau - 1$. Clearly, $t_1 + 1 \leq t_0$ as it takes at least one time slot to dequeue a packet. Moreover, queue q is the longest queue since $t_1 + 1$. We first claim that the longest queue (queue q) is connected at $t_1 + 1$, i.e.,

$$(t_1 + 1 - \tau - D(\tau, t_1 + 1)) \bmod K = 0. \tag{12}$$

As such, $t_1 + 1$ is the *first* time that queue q is connected after the last packet is dequeued to the head queue from the central queues (if there is one). Note that (12) holds trivially if no packet is dequeued to the head queue since the beginning of the busy period. On the other hand, according to the serving-the-longest-queue rule in (R3), we have

$$(t_1 - \tau - D(\tau, t_1)) \bmod K = 0. \tag{13}$$

As t_1 is the time that the last packet is dequeued to the head queue, we also have $D(\tau, t_1 + 1) = D(\tau, t_1) + 1$. Replacing this in (13) yields (12).

Secondly, we claim that there exists t_2 with $t_0 - K < t_2 < t_0$ such that queue q is connected at t_2, i.e.,

$$(t_2 - \tau - D(\tau, t_2)) \bmod K = 0. \tag{14}$$

As such, t_2 is the *last* time (before t_0) that queue q is connected after the last packet is dequeued to the head queue from the central queues (if there is one). Since there is no packet that is dequeued from the central queues to the head queue from $t_1 + 1$ to t_0, we have

$$D(\tau, t_1 + 1) = D(\tau, t) = D(\tau, t_0) \tag{15}$$

for all $t_1 + 1 \leq t \leq t_0$. Let $m = \lfloor (t_0 - t_1 - 1)/K \rfloor$, where $\lfloor x \rfloor$ is the floor function that returns the largest integer not greater than x. Let

$$t_2 = t_1 + 1 + mK. \tag{16}$$

Since $x - 1 < \lfloor x \rfloor \leq x$, we have

$$t_0 - K < t_2 \leq t_0. \tag{17}$$

That (14) holds then follows from (16), (15) and (12). Furthermore, as $D(\tau, t_2) = D(\tau, t_0)$, replacing this in (14) yields

$$(t_2 - \tau - D(\tau, t_0)) \bmod K = 0.$$

In view of (11), we know that $t_2 \neq t_0$. Thus, we have $t_0 - K < t_2 < t_0$.

Now we claim that the head queue must be full at t_2, i.e.,

$$q_H(t_2) = K - 1. \tag{18}$$

According to the serving-the-longest-queue rule in (R3), the condition in (14) indicates that queue q would be enabled at time t_2 if the other condition $q_H(t_2 - 1) - c(t_2) < K - 1$ were satisfied. As packet p is still in queue q at time t_0, this implies that

$$q_H(t_2 - 1) - c(t_2) = K - 1. \tag{19}$$

As $q_H(t) \leq K - 1$ for all t and $c(t)$ is nonnegative, we have $q_H(t_2 - 1) = K - 1$ and $c(t_2) = 0$. Thus, the head queue remains unchanged at t_2 and we have $q_H(t_2) = q_H(t_2 - 1) = K - 1$.

Finally we show there is a contradiction to the empty head queue condition $q_H(t_0 - 1) = 0$. As $t_0 - K < t_2 < t_0$, we have $t_0 - 1 - t_2 < K - 1$. Since the head queue can be decreased by at most 1 in a time slot, we also have from (18) that

$$q_H(t_0 - 1) \geq q_H(t_2) - (t_0 - 1 - t_2) > 0. \tag{20}$$

This leads to a contradiction to the empty head queue condition $q_H(t_0 - 1) = 0$ when the non-idling property is violated.

(P3) Maximum buffer usage: once again, we will show by contradiction that there is no loss in the tail queue as long as the total number of packets inside the network element is not greater than $BK + K - 1$, i.e., for all t

$$q_H(t) + \hat{q}_C(t) + q_T(t) \leq BK + K - 1. \tag{21}$$

Suppose that the maximum buffer usage property is violated for the first time at time t_0, i.e., a packet arriving at the tail queue is lost at time t_0. When this happens, we have from the maximum buffer usage property of the tail queue that (i) the tail queue must be full, i.e., $q_T(t_0 - 1) = K - 1$ and (ii) the control of the tail queue is not enabled, i.e., $c_T(t_0) = 0$.

Denote by packet p the head-of-line packet of the tail queue at time t_0. Let t_1 be the time that the last packet is dequeued to the central queues from the tail queue. If t_1 is in a busy period, let τ be the beginning of that busy period. Otherwise, let $\tau = t_1$. We first claim that

$$(t_1 + 1 - \tau - A(\tau, t_1 + 1)) \bmod K = 0. \tag{22}$$

Note that if $\tau = t_1$, then $A(\tau, t_1 + 1) = 1$ and (22) holds trivially. On the other hand, if τ is the beginning of a busy period, it then follows from the joining-the-shortest-queue rule in (R4) that

$$(t_1 - \tau - A(\tau, t_1)) \bmod K = 0. \tag{23}$$

As t_1 is the time that the last packet is dequeued to the central queues from the tail queue, we also have $A(\tau, t_1 + 1) = A(\tau, t_1) + 1$. Replacing this in (23) yields (22).

Secondly, we claim that there exists t_2 with $t_0 - K < t_2 \leq t_0$ such that

$$(t_2 - \tau - A(\tau, t_2)) \bmod K = 0. \tag{24}$$

The argument for (24) is similar to that in the proof of the non-idling property. Since there is no packet that is dequeued to the central queues from the tail queue from $t_1 + 1$ to t_0, we have

$$A(\tau, t_1 + 1) = A(\tau, t) = A(\tau, t_0) \tag{25}$$

for all $t_1 + 1 \leq t \leq t_0$. Let

$$t_2 = t_1 + 1 + \lfloor (t_0 - t_1 - 1)/K \rfloor K. \tag{26}$$

Analogous to the argument for the non-idling property, we have

$$t_0 - K < t_2 \leq t_0. \tag{27}$$

That (24) holds then follows from (26), (25) and (22).

Now we claim that packet p is in the tail queue at t_2. If packet p has not arrived at the tail queue by t_2, then the tail queue must be empty at t_2, i.e., $q_T(t_2) = 0$, as the last packet departs at $t_1 < t_2$. Since there is at most one packet arrival per time slot, we have from (27) that

$$q_T(t_0 - 1) \leq q_T(t_2) + (t_0 - 1 - t_2) < K - 1. \tag{28}$$

This leads to a contradiction that $q_T(t_0 - 1) = K - 1$.

Finally we show there is a contradiction to the maximum number of packets inside the network element. In view of (24) and the joining-the-shortest-queue rule in (R4), the only reason that packet p did not depart from the tail queue at t_2 is that

$$q_C(t_2 - 1) - c_C(t_2) = B.$$

This implies that the connected central queue is full, i.e., $q_C(t_2 - 1) = B$ and the control input of the connected central queue is not enabled, i.e., $c_C(t_2) = 0$. As the connected central queue is the shortest queue, all the central queues are full, i.e., $\hat{q}_C(t_2 - 1) = BK$. As τ is the beginning of a busy period, we have

$$A(\tau, t_2) = D(\tau, t_2) + \hat{q}_C(t_2 - 1) = D(\tau, t_2) + BK. \quad (29)$$

From (24), it then follows that

$$(t_2 - \tau - D(\tau, t_2)) \bmod K = 0. \quad (30)$$

Thus, the connected central queue is also the longest queue. According to the serving-the-longest-queue rule in (R3), the condition in (30) indicates that the connected central queue would be enabled at time t_2 if the other condition $q_H(t_2 - 1) - c(t_2) < K - 1$ were satisfied. This in turn implies that the head queue is also full, i.e., $q_H(t_2 - 1) = K - 1$ and the head queue is not enabled at t_2, i.e., $c(t_2) = 0$. Since both the head queue and the connected central queue are not enabled at t_2, they remain unchanged at t_2. Thus, we have $q_H(t_2) = K - 1$ and $\hat{q}_C(t_2) = BK$. Adding packet p in the tail queue at t_2, the total number of packets inside the network element at t_2 is at least $BK + K$, which contradicts to (21).

(P4) FIFO: since both the tail queue and the head queue are FIFO queues, we only need to consider the order in the central queues. The FIFO property in the central queues is trivially preserved from the joining-the-shortest-queue rule and the serving-the-longest-queue rule.

REFERENCES

[1] V. E. Benes. *Mathematical Theory of Connecting Networks and Telephone Traffic*. New York: Academic Press, 1965.

[2] G. Birkhoff, "Tres observaciones sobre el algebra lineal," *Univ. Nac. Tucumán Rev. Ser. A*, Vol. 5, pp. 147-151, 1946.

[3] I. Chlamtac and A. Fumagalli, "QUADRO-star: High performane optical WDM star networks," *Proceedings of IEEE GLOBAECOM'91*, Phoenix, AZ, Dec. 1991.

[4] I. Chlamtac, A. Fumagalli, L. G. Kazovsky, P. Melman, W. H. Nelson, P. Poggiolini, M. Cerisola, A. N. M. M. Choudhury, T. K. Fong, R. T. Hofmeister, C. L. Lu, A. Mekkittikul, D. J. M. Sabido IX, C. J. Suh and E. W. M. Wong, "Cord: contention resolution by delay lines," *IEEE Journal on Selected Areas in Communications*, Vol. 14, pp. 1014-1029, 1996.

[5] I. Chlamtac and A. Fumagalli, and C.-J. Suh, "Multibuffer delay line architectures for efficient contention resolution in optical switching nodes," *IEEE Transactions on Communications*, Vol. 48, pp. 2089-2098, 2000.

[6] C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer-verlag: London, 2000.

[7] C.-S. Chang, Y.-T. Chen, J. Cheng, and D.-S. Lee, "Multistage constructions of linear compressors, non-overtaking delay lines, and flexible delay lines," accepted by *IEEE INFOCOM 2006*.

[8] C.-S. Chang, D.-S. Lee and C.-K. Tu, "Using switched delay lines for exact emulation of FIFO multiplexers with variable length bursts," to appear in *IEEE Journal on Selected Areas in Communications*. Conference version in *Proceedings of IEEE INFOCOM*, 2003.

[9] C.-S. Chang, D.-S. Lee and C.-K. Tu, "Recursive construction of FIFO optical multiplexers with switched delay lines," *IEEE Transactions on Information Theory*, Vol. 50, pp. 3221-3233, 2004.

[10] C. Clos, "A study of nonblocking switching networks," *BSTJ*, Vol. 32, pp. 406-424, 1953.

[11] R. L. Cruz and J. T. Tsai, "COD: alternative architectures for high speed packet switching," *IEEE/ACM Transactions on Networking*, Vol. 4, pp. 11-20, February 1996.

[12] D. K. Hunter and I. Andonovic, "Approaches to optical Internet packet switching," *IEEE Communication Magazine*, Vol. 38, pp. 116-122, 2000.

[13] D. K. Hunter, D. Cotter, R. B. Ahmad, D. Cornwell, T. H. Gilfedder, P. J. Legg and I. Andonovic, "2×2 buffered switch fabrics for traffic routing, merging and shaping in photonic cell networks," *IEEE Journal of Lightwave Technology*, Vol. 15, pp. 86-101, 1997.

[14] D. K. Hunter, M. C. Chia and I. Andonovic, "Buffering in optical packet switches," *IEEE Journal of Lightwave Technology*, Vol. 16, pp. 2081-2094, 1998.

[15] D. K. Hunter, W. D. Cornwell, T. H. Gilfedder, A. Franzen and I. Andonovic, "SLOB: a switch with large optical buffers for packet switching," *IEEE Journal of Lightwave Technology*, Vol. 16, pp. 1725-1736, 1998.

[16] J. Hui, *Switching and Traffic Theory for Integrated Broadband Networks.* Boston: Kluwer Academic Publishers, 1990.

[17] F. K. Hwang. *The Mathematical Theory of Nonblocking Switching Networks*, Singapore: World Scientific Publishing Co., 1998.

[18] M. J. Karol, "Shared-memory optical packet (ATM) switch," SPIE Vol. 2024 Multigigabit Fiber Communications Systems, pp. 212-222, 1993.

[19] S.-Y. R. Li. *Algebraic Switching Theory and Broadband Applications*. Academic Press, 2001.

[20] A. D. Sarwate and V. Anantharam, "Exact emulation of a priority queue with a switch and delay lines," *submitted to Queueing Systems Theory and Applications.*

[21] M. Schwartz, *Broadband Integrated Networks*. New Jersey: Prentice Hall, 1996.

[22] C. E. Shannon, "Memory requirements in a telephone exchange," *Bell System Technical Journal*, Vol. 29, pp. 343-349, 1950.

[23] J. T. Tsai, "COD: architectures for high speed time-based multiplexers and buffered packet switches," Ph.D. Dissertation, University of California, San Diego, 1995.

[24] E. A. Varvarigos, "The 'Packing' and 'Scheduling' switch architectures for almost-all optical lossless networks," *IEEE Journal of Lightwave Technologies*, vol. 16 (no. 10), pp. 1757-67, Oct. 1998.

[25] J. von Neumann, "A certain zero-sum two-person game equivalent to the optimal assignment problem, " *Contributions to the Theory of Games*, Vol. 2, pp. 5-12, Princeton University Press, Princeton, New Jersey, 1953.

[26] S. Yao, B. Mukherjee, and S. Dixit, "Advances in photonic packet switching: An overview," *IEEE Communication Magazine*, Vol. 38, pp. 84-94, 2000.

Cheng-Shang Chang (S'85-M'86-M'89-SM'93-F'04) 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. Since 1993, he has been with 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 high speed switching, communication network theory, and mathematical modeling of the Internet. Dr. Chang received an IBM Outstanding Innovation Award in 1992, an IBM Faculty Partnership Award in 2001, and Outstanding Research Awards from the National Science Council, Taiwan, in 1998, 2000 and 2002, respectively. He also received Outstanding Teaching Awards from both the college of EECS and the university itself in 2003. He was appointed as the first Y. Z. Hsu Scientific Chair Professor in 2002. 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. Dr. Chang is a member of IFIP Working Group 7.3.

Yi-Ting Chen received the B.S. degree in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 2004, and he is currently a graduate student in communications engineering in the National Tsing Hua University, Hsinchu, Taiwan.

Duan-Shin Lee received the B.S. degree from National Tsing Hua University, Taiwan, in 1983, and the MS 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. He joined the Department of Computer Science of National Tsing Hua University in Hsinchu, Taiwan, in 1998. Since August 2003, he has been a professor. His research interests are high-speed switch and router design, wireless networks, performance analysis of communication networks and queueing theory. He is a senior IEEE member.