Recursive Construction of FIFO Optical Multiplexers with Switched Delay Lines

Cheng-Shang Chang, Duan-Shin Lee, and Chao-Kai Tu
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
Email: cschang@ee.nthu.edu.tw
lds@cs.nthu.edu.tw
cktu@gibbs.ee.nthu.edu.tw

Abstract

In this paper, we develop mathematical theory for recursive construction of First In First Out (FIFO) optical multiplexers by the combination of (bufferless) crossbar Switches and fiber Delay Lines (SDL). We show that by cascading multi-stage SDL units, 2-to-1 multiplexers with a large buffer can be emulated exactly for both the departure process and the loss process from the multiplexer. Such results are extended to the case of n-to-1 multiplexers by introducing a new class of multiplexers, called delayed-loss multiplexers. A delayed-loss multiplexer has the same departure process as an ordinary multiplexer. However, lost packets due to buffer overflow in a multiplexer might be delayed. A key result from our theory is the self-routing n-to-1 multiplexer, where the routing path of a packet through the multi-stage SDL units can be determined upon its arrival.

Keywords

conflict resolution, optical switches, multi-stage switches, switched delay lines, self routing switches

This research is supported in part by the National Science Council, Taiwan, R.O.C., under Contract NSC-90-2213-E007-034, NSC 90-2219-E007-003, and the program for promoting academic excellence of universities 89-E-FA04-1-4.

I. Introduction

There is an urgent need to build high speed packet switches that scale with the transmission speed of fiber optics. The key challenge to such a problem is to build high speed buffers that resolve conflicts of packets competing for the same resource. Two common approaches are used. The first approach is to use parallel electronic buffers to acquire the needed speedup (see e.g., [7], [14], [5], [16]). The other approach is to use fiber delay lines for buffering in optical packet switches (see e.g., the excellent review papers [11], [9], [21] and references therein).

We will focus on the second approach in this paper. Unlike electronic memory, fiber delay lines are not capable of providing random memory access. They can only be accessed in a pre-determined sequential manner. As such, conflict resolution by fiber delay lines is in general much more difficult than that by electronic memory. As indicated in [11], there are several architectures proposed in the literature that use fiber delay lines as buffers. In [2], [3], [4], SDL (Switched fiber Delay Lines) is proposed in the CORD (contention resolution by delay lines) project. An SDL unit in [3] is an optical fabric that only consists of optical crossbar switches and fiber delay lines. By redistributing packets through delay lines with different delays, it is then possible to resolve conflicts for the same resource over time and space. In Figure 1, we illustrate this idea via a simple example. There are two input links that are multiplexed into an output link. Suppose that two packets arrive at both input links. This cause a conflict for the output link. To resolve such a conflict, we can first put these two packets through a 2×2 switch and distribute one packet to a zero delay transmission line (the upper link) and the other to a fiber delay line with one unit of delay (the lower link). By so doing, these two packets can be multiplexed into the same link sequentially.

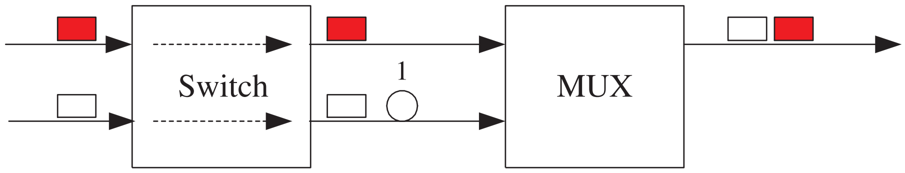

Fig. 1. An illustrating example of using switched delay lines for conflict resolution

To construct a large buffer, one needs to cascade multi-stage SDL units. However, finding efficient control of the switches that distribute packets to resolve conflicts becomes a problem. In [20], [8], a genuine design, named COD (Cascaded Optical Delay-Lines), is proposed for First In First Out (FIFO) buffers by using 2×2 crossbar switches and fiber delay lines. The control of COD is easy and only requires local information. However, the number of 2×2 switches in such an architecture is proportional to the buffer size. In [10], a more efficient design, Logarithm Delay-Line Switch, is proposed for the 2×2 buffered switch. Such an architecture is based on output buffer emulation and the path for a packet to go through the Logarithm Delay-Line Switch is uniquely determined by its virtual delay derived from the output buffer emulation. The number of 2×2 switches needed for such an architecture is only $O(\log B)$, where B is the buffer size. In [12], SLOB (Switch with Large Optical Buffers) is proposed for the extension of optical buffered switches with n input/output ports ($n \geq 2$). Such an architecture also uses output buffer emulation and relies on a special hardware, called a primitive switching element (PSE). Each PSE itself is an $n \times 2n$ output-buffered switch with buffer size $n-1$. Unlike the Logarithm Delay-Line Switch, the routing path of a packet in SLOB cannot be uniquely determined upon its arrival. This makes control of the PSEs much more difficult. In fact, a small control message must be transmitted electronically for each packet, representing the remaining delay over the remaining PSE's.

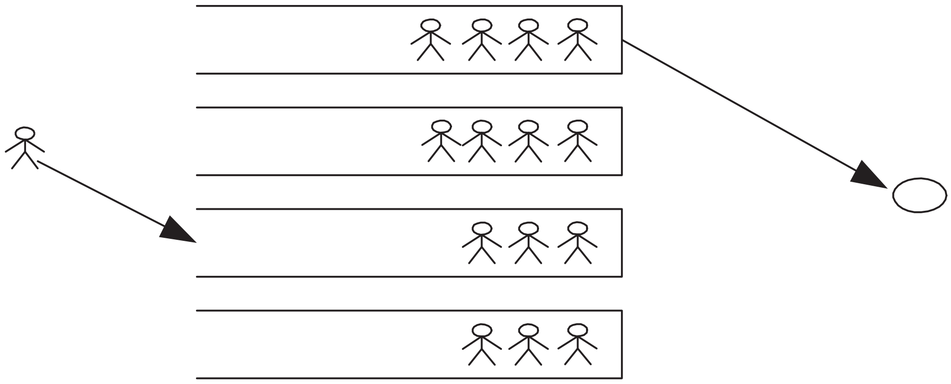

Fig. 2. A system with parallel queues

Inspired by these prior works, in this paper we develop mathematical theory for recursive construction of FIFO optical multiplexers with large buffers by using switched delay lines. Our idea comes from a well known queueing result. Consider a system with parallel queues as shown in Figure 2. Suppose that we operate the system as follows: a customer that arrives at the system joins the shortest queue (the queue with the least number of customers), and the server after completing the service of a customer always chooses a customer from the longest queue (the queue with the largest number of customers). By so doing, these parallel queues are kept in the most balance state, i.e., at any time the difference between the number of customers in the longest queue and the number of customers in the shortest queue is at most 1. If the service time of all the customers are all identical, then the longest queue service policy and the shortest queue dispatching policy are simply the round robin policy. Moreover, the system with parallel queues behaves as if it were a single queue with a shared buffer. Based on this, one can build a multiplexer with a large buffer by time interleaving several multiplexers with small buffers (Surprisingly enough, such an approach was also used in [14], [5], [16] for parallel electronic buffers). Our main work in this paper is then to design an SDL unit associated with an operation rule so that packet arrivals are dispatched in a round robin fashion to the parallel multiplexers with small buffers.

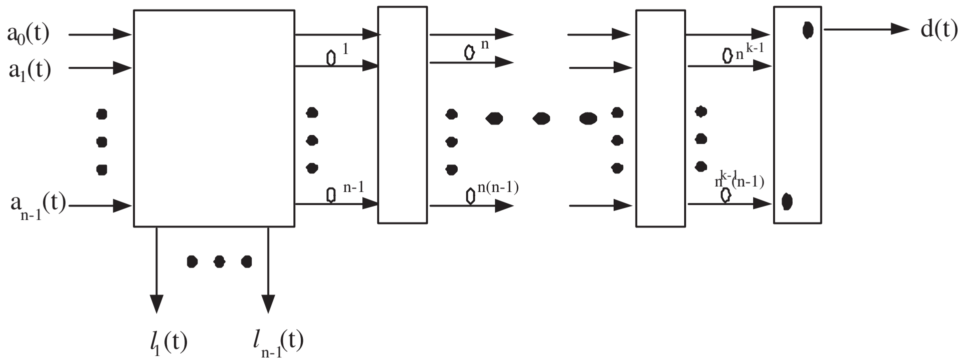

Fig. 3. A self-routing n-to-1 multiplexer with $B = n^k - 1$.

One of the key results that comes from our mathematical theory is the self-routing n-to-1 multiplexer shown in Figure 3. As shown in Figure 3, there are k stages of SDL units (the last one is simply a bufferless multiplexer). Each stage, except the first stage, consists of an $n \times n$ crossbar switch and n fiber delay lines (with delays specified in the figure). Packets are assumed to be of the same size and it takes one unit of delay to transmit a packet. The first stage requires an $n \times (2n-1)$ switch as additional output links are used for dropping packets due to buffer overflow. The buffer size of such a multiplexer is $n^k - 1$. As in [10], [12], we use output buffer emulation for this system. It keeps track of the number of packets stored in the system. If such a number exceeds $n^k - 1$, further arrivals are lost immediately. Specifically, let $q(t)$ be the number of packets stored in the system. Then $q(t)$ is governed by

$$q(t) = \min\Big[\max[0, q(t-1) + \sum_{i=0}^{n-1} a_i(t) - 1], n^k - 1\Big], \tag{1}$$

where $a_i(t)$, $i = 0, 1, \ldots, n-1$, is the number of arrival from the i^{th} input link. Let q be the number of packets stored in the system when a particular packet enters the system. We call this the virtual delay of the packet (as in the queueing theory). Then we have $0 \le q \le n^k - 1$ and there exists a unique vector $r = (r_1, r_2, \ldots, r_k)$ with $0 \le r_j \le n-1$ for all j such that

$$q = \sum_{j=1}^{k} r_j n^{j-1}.$$

The packet can then be self-routed through the network element by taking the ${r_j}^{th}$ output link at the j^{th} $n \times n$ switch. There will not be any conflicts in the self-routing multiplexer, i.e., no more than one packet occupies the same link at any time.

To see the analogy between our self-routing multiplexer and the classical Batcher-Banyan self-routing network (see e.g., Schwartz [18] and Hui [13]), one may view the virtual delay in our self-routing multiplexer as the "output address" in the Batcher-Banyan self-routing network. By routing packets to different "output addresses," we then resolve conflicts at the multiplexer.

We also note that the self-routing multiplexer in Figure 3 is in fact quite similar to the SLOB in [12]. The difference is that the PSE in the SLOB is now replaced by a simple $n \times n$ (bufferless) crossbar switch. As one can also use n n-to-1 multiplexers to build an $n \times n$ output-buffered switch, the main advantage of using the architecture in Figure 3 is the self-routing property that leads to a much simpler control mechanism than that in the SLOB.

This paper is organized as follows. In Section II, we start from the definitions of basic network elements and develop the mathematical theory for 2-to-1 FIFO optical multiplexers. For the 2-to-1 multiplexers, we can emulate both the departure process and the loss process from a FIFO finite buffer queue. In Section III, we extend the results in Section II to n-to-1 multiplexers. We define a new concept, called a delayed-loss multiplexer. For such a multiplexer, its departure process is the same as that from a FIFO finite buffer queue, but packet losses may be delayed. By discarding packets in advance, we show that the delayed-loss multiplexer can be made into a self-routing multiplexer in Figure 3. The paper is concluded in Section IV, where we address several topics for future research.

II. 2-to-1 multiplexers

A. Definitions of basic network elements

In this paper, we consider multiplexing fixed size packets over optical links. We assume that propagation delay is well compensated so that time is synchronized and slotted. By so doing, 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.

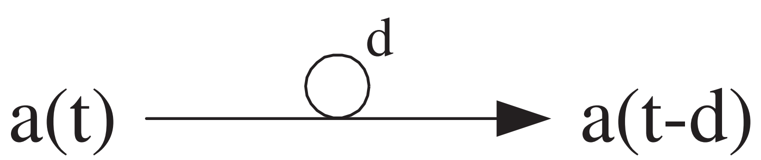

Fig. 4. An optical delay line with delay d

Definition 1 (Delay line) *An (optical) delay line in Figure 4 is a network element that has one input link and one output link. In Figure 4, the delay is d. Let $a(t)$ be the state of the input link. Then the state of the output link is $a(t-d)$.*

An optical delay line acts as a memory element in our construction. Note that at the end of the $t-1^{th}$ time slot, the packets that arrive at time $t-1, t-2, \ldots, t-d$, are stored in the optical delay line with delay d.

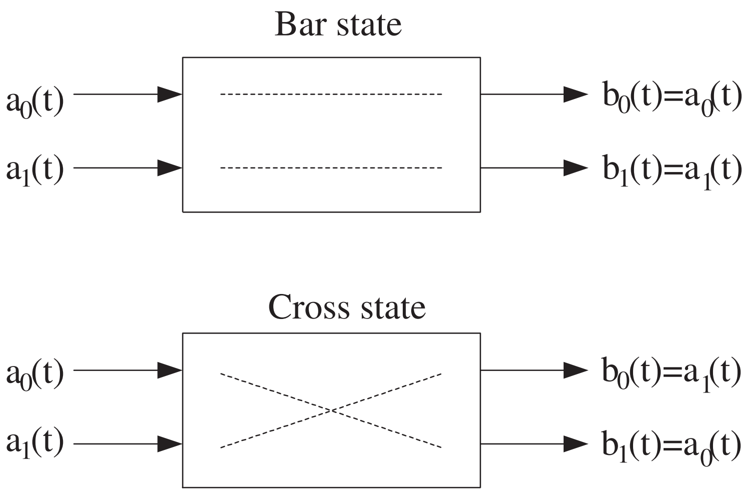

Fig. 5. A 2×2 switch

Definition 2 (Switch) *A 2×2 (optical) switch has two input links and two output links (see Figure 5). Let $a_1(t)$ and $a_0(t)$ be the states of the lower and upper inputs, and $b_1(t)$ and $b_0(t)$ be the states of the lower and upper outputs. The switch is said to be in the "bar" state at time t if $b_1(t) = a_1(t)$ and $b_0(t) = a_0(t)$. It is said to be in the "cross" state at time t if $b_1(t) = a_0(t)$ and $b_0(t) = a_1(t)$.*

Unlike an optical delay line, the 2×2 switch in Definition 2 is a memoryless element. One of the main objectives of this paper is to combine optical delay lines with memoryless optical switches to form buffered multiplexers, which in turn can be used for building buffered switches. One key step in doing this is the prioritized concentrator defined below.

Definition 3 (Concentrator) *A prioritized concentrator in Figure 6 is a 2×2 switch with its connection patterns depending on its two inputs. Let $a_1(t)$ (resp. $a_0(t)$) be the state of the dotted (resp. undotted) input, and $b_0(t)$ (resp. $b_1(t)$) be the state of the dotted (resp. undotted) output. The switch is set to the cross state at time t if $a_1(t) = 1$, i.e.,* there is a packet arrival at the dotted

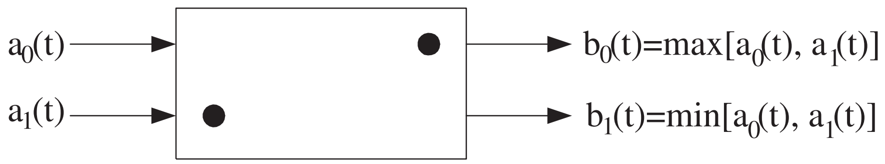

Fig. 6. A prioritized concentrator

input at time t. *Otherwise, the switch is set to the bar state. Thus, if there is a packet at the dotted input at time t, this packet is transmitted to the dotted output. If there is another packet at the undotted input, then the packet at the undotted input is transmitted to the undotted output. When there is no packet at the dotted input and there is a packet at the undotted input, the packet at the undotted input is transmitted to the dotted output. Such an operation rule ensures that*

$$b_0(t) = \max[a_0(t), a_1(t)], \tag{2}$$

and

$$b_1(t) = \min[a_0(t), a_1(t)]. \tag{3}$$

We note that a prioritized concentrator is called a track changer in [20] for 2-to-1 multiplexers. As its main objective is to perform traffic concentration (this will become clear in the general case of n-to-1 multiplexers), we prefer using the name prioritized concentrator as the name reflects its functional objective.

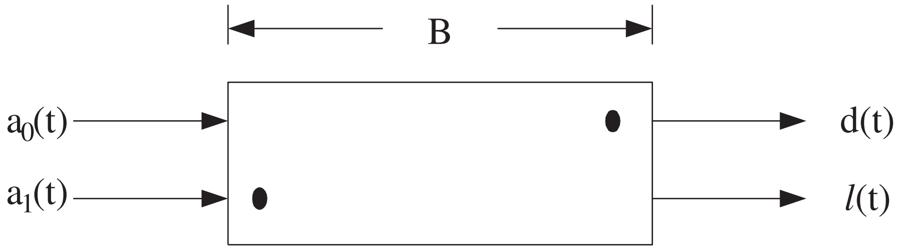

Fig. 7. A 2-to-1 multiplexer with buffer B

Definition 4 (Multiplexer) *A 2-to-1 multiplexer with buffer* B *has two input links and two output links (see Figure 7). One output link is for departing packets and the other is for lost packets. As shown in Figure 7, let* $a_1(t)$ *(resp.* $a_0(t)$*) be the state of the dotted (resp. undotted) input link,* $d(t)$ *(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 multiplexer at time* t *(at the end of the* t^{th} *time slot). Then the 2-to-1 multiplexer with buffer* B *satisfies the following four properties:*

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

$$q(t) = q(t-1) + a_0(t) + a_1(t) - d(t) - \ell(t). \tag{4}$$

(P2) Non-idling: there is always a departing packet if there are packets in the buffer or there are arriving packets, i.e.,

$$d(t) = \begin{cases} 0 & \text{if } q(t-1) = a_0(t) = a_1(t) = 0 \\ 1 & \text{otherwise} \end{cases}. \tag{5}$$

(P3) Maximum buffer usage: arriving packets are lost only when buffer is full, i.e.,

$$\ell(t) = \begin{cases} 1 & \text{if } q(t-1) = B \text{ and } a_0(t) = a_1(t) = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

(P4) FIFO with prioritized inputs: packets depart in the first in first out (FIFO) order. Moreover, if each input has an arriving packet, the packet from the dotted input is put in the multiplexer first. In other word, the virtual delay for the dotted input $a_1(t)$ is $q(t-1)$, the number of packets that is stored in the multiplexer at $t-1$. The virtual delay for the undotted input $a_0(t)$ is then $q(t-1) + a_1(t)$ (if that packet is not lost).

Note that a 2-to-1 multiplexer with buffer B is simply a discrete-time queue with buffer B and two inputs in the queueing theory. As such, the state of a 2-to-1 multiplexer with buffer B is simply $q(t)$, the number of packets stored in the multiplexer at the end of the t^{th} time slot. This is crucial in simplifying the complexity of our analysis as a large number of binary states in optical delay lines now can be summarized by a single number. We also note that a prioritized concentrator in Definition 3 is a 2-to-1 multiplexer with buffer 0. In this case, it is stateless as $q(t) = 0$ for all t.

In addition to the state aggregation property described above, we also need the time interleaving property described below.

Definition 5 (SDL multiplexer) *A 2-to-1 multiplexer is called a 2-to-1* SDL *multiplexer if the multiplexer is built with delay lines (in Definition 1) and switches (in Definition 2). A 2-to-1* SDL *multiplexer is with scaling factor k if the delay in every delay line is k times of that in the original (unscaled) 2-to-1* SDL *multiplexer.*

Proposition 6 (Time interleaving property) *A 2-to-1 SDL multiplexer with scaling factor k can be operated as time interleaving of k 2-to-1 SDL multiplexers.*

Proof. It suffices to illustrate this for the case with $k = 2$. To perform time interleaving of two 2-to-1 SDL multiplexers, we partition time into even and odd numbered time slots. We then operate these two 2-to-1 SDL multiplexers *alternatively* between even numbered time slots and odd numbered time slots. As such, the states of each of the two 2-to-1 SDL multiplexers are changed every two time slots (and remain unchanged when the multiplexer is not operated). In short, each of the time interleaved SDL multiplexers is operated at the clock rate that is one half of that in the original 2-to-1 SDL multiplexer. In view of this, each of the time interleaved SDL multiplexers can then be implemented by the original 2-to-1 SDL multiplexer by doubling the delay in each delay line and changing the state in each switch every two time slots. Clearly, each of the time interleaved SDL multiplexers can then be implemented by a 2-to-1 SDL multiplexer with scaling factor 2.

Instead of using *two* 2-to-1 SDL multiplexer with scaling factor 2 for time interleaving of two 2-to-1 SDL multiplexers, now we show that we only need *one*. To see this, call the two time interleaved multiplexers *multiplexer A* and *multiplexer B*, and the 2-to-1 SDL multiplexer with scaling factor 2 *multiplexer C*. As described in the last paragraph, note that the states of the switches in multiplexers A and B are changed every two time slots. Thus, for the even numbered time slots, we can set the states of the switches in multiplexer C according to the states of the switches in multiplexer A. Similarly, for the odd numbered time slots, we can set the states of the switches in multiplexer C according to the states of the switches in multiplexer B. By so doing, we can operate a 2-to-1 SDL multiplexer with scaling factor 2 as time interleaving of two 2-to-1 SDL multiplexers.

■

B. Recursive construction

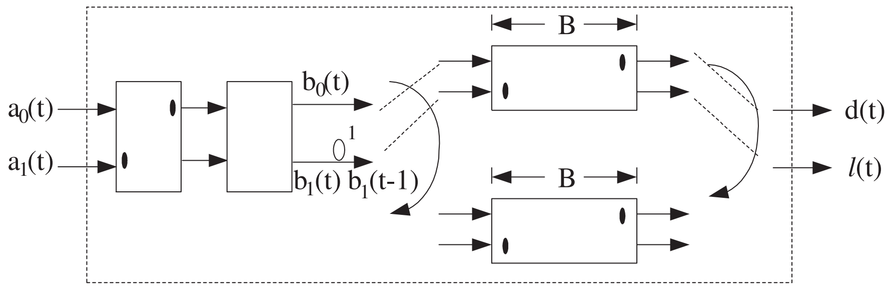

Fig. 8. A 2-to-1 multiplexer with buffer $2B+1$

In this section, we show how one constructs a 2-to-1 multiplexer with a large buffer by time interleaving two 2-to-1 multiplexers with small buffers. In Figure 8, we consider a network element with two inputs and two outputs. It is a concatenation of a prioritized concentrator, a 2×2 switch and two 2-to-1 multiplexers with buffer B. The two outputs of the 2×2 switch are connected to two delay lines with delay 0 and 1, respectively. Partition time into *even* and *odd* numbered time slots. For the even numbered time slots, the two outputs of the delay lines (after the 2×2 switch) and the two outputs of the network element are connected to the two inputs and the two outputs of one multiplexer, respectively. On the other hand, for the odd numbered time slots, the two outputs of the delay lines and the two outputs of the network element are connected to the two inputs and the two outputs of the other multiplexer. The state of each multiplexer, i.e., the number of packets stored in the multiplexer, remains unchanged when the multiplexer is not connected. Thus, every multiplexer changes its state every two time slots.

Define the total number of packets stored in the network element as the sum of the number of packets stored in each multiplexer and the number of packet stored in the delay line with delay 1. The 2×2 switch is set to the cross state if there is an odd number of packets stored in the network element (at the end of the previous time slot), and is set to the bar state otherwise.

Theorem 7 *If the network element in Figure 8 is started from an empty system, then it is a 2-to-1 multiplexer with buffer* $2B + 1$*.*

The proof of Theorem 7 is given in Appendix A.

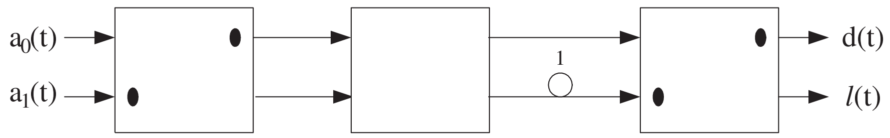

Fig. 9. A 2-to-1 multiplexer with buffer 1

Example 8 Since a prioritized concentrator is a 2-to-1 multiplexer with buffer 0, it follows from Theorem 7 that the network element in Figure 9 is a 2-to-1 multiplexer with buffer 1. The 2×2 switch is set to the cross state if there is a packet stored in the network element and is set to the bar state if the network element is empty.

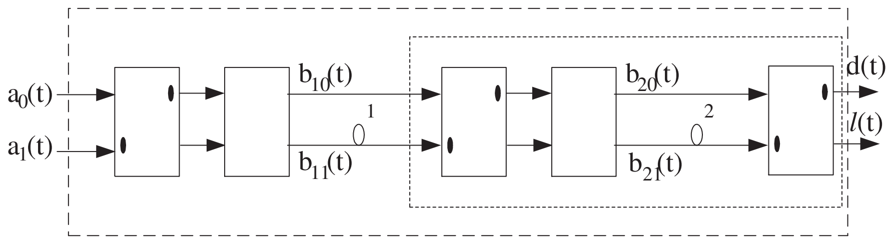

Fig. 10. A 2-to-1 multiplexer with $B = 3$.

Example 9 In Figure 10, we illustrate how one applies Theorem 7 to construct a 2-to-1 multiplexer with buffer 3 from the 2-to-1 multiplexer with buffer 1 in Figure 9. Note that the inner block in Figure 10 is exactly the same as the 2-to-1 multiplexer in Figure 9 except that the delay in the delay line is doubled from 1 to 2. As such, it is a 2-to-1 SDL multiplexer with buffer 1 and scaling factor 2. From the time interleaving property of scaled SDL multiplexers in Proposition 6, this can be viewed as time interleaving two 2-to-1 multiplexers: the even numbered time slots for one multiplexer and the odd numbered time slots for the other. As a direct consequence of Theorem 7, the network element in Figure 10 is indeed a 2-to-1 multiplexer with buffer 3. As shown in Figure 9, let $b_{11}(t)$ (resp. $b_{21}(t)$) be the state of the lower output of the first (resp. second) 2×2 switch. Note that the total number of packets stored in the network element at time $t-1$ is $b_{11}(t-1)+b_{21}(t-1)+b_{21}(t-2)$. Thus, we have from the operation rule in Theorem 7 that the first switch is set to the cross state at time t if $b_{11}(t-1)+b_{21}(t-1)+b_{21}(t-2)$ is an odd number and the bar state otherwise. On the other hand, we have from the operation rule in Example 8 that the second switch is set to the cross state at time t if $b_{21}(t-2)=1$ and the bar state otherwise.

Now one can use Theorem 7 and the time interleaving property in Proposition 6 to recursively construct a 2-to-1 multiplexer with buffer 2^k-1. The m^{th} switch, $m=1,2,\ldots,k$, is set to the cross state at time t if $\sum_{i=m}^{k}\sum_{j=1}^{2^{i-m}} b_{i1}(t-2^{m-1}j)$ is an odd number and the bar state otherwise, where $b_{m1}(t)$ is the state of the lower output of the m^{th} switch. One can further combine the operation of the prioritized concentrator and the 2×2 switch at each stage by a 2×2 switch (see Figure 11). The combined switch is to set to the bar state if both the prioritized concentrator and the original 2×2 switch are set to the same state, and it is set to the cross state otherwise. For such a multiplexer, all its switch patterns are completely determined by the states of the multiplexer.

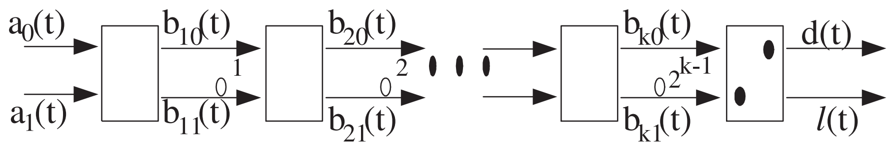

Fig. 11. A 2-to-1 multiplexer with $B=2^k-1$.

III. n-to-1 Multiplexers

In this section, we extend the results for 2-to-1 multiplexers to n-to-1 multiplexers. In Section II, we construct network elements with 2×2 switches and optical delay lines that emulate exact 2-to-1 multiplexers for both the departure process and the loss process. Exact emulation of n-to-1 multiplexers is much more difficult for $n>2$. Instead, we only construct network elements with

$n \times n$ switches and optical delay lines that generate the same departure processes as those from n-to-1 multiplexers. Packet losses at our n-to-1 multiplexers might be delayed. Such a construction is called a delayed-loss multiplexer in this paper.

A. Definitions of basic network elements

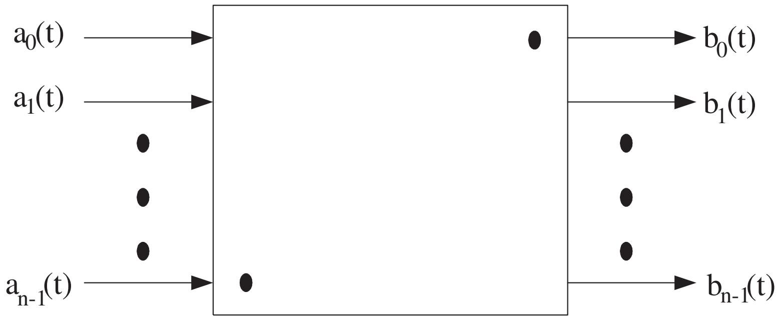

Fig. 12. An n-to-n prioritized concentrator

We first generalize the 2×2 prioritized concentrator in Definition 3.

Definition 10 (Concentrator) *An $n \times n$ prioritized concentrator (see Figure 12) is an $n \times n$ switch with its connection pattern depending on its n inputs. Both the input links and output links are numbered from the top to the bottom. The priority of the input links is* increasing *in the link number and the priority of the output links is* decreasing *in the link number (the dotted input and the dotted output in the diagrammatic representation have the highest priority). The packets that arrive at high priority input links have priority to be switched to high priority output links. Thus, if there is a packet arrival at input link $n-1$, it is switched to output link 0. If there is no arrival at input link $n-1$ and there is an arrival at input link $n-2$, the arrival at input link $n-2$ is switched to output link 0. Mathematically, the state at output link k is*

$$b_k(t) = \sum_{i=k+1}^{n} a_{n-i}(t) 1_{\left\{\sum_{j=1}^{i-1} a_{n-j}(t) = k\right\}},$$

where $1_{\{A\}}$ is 1 if A is true and 0 otherwise.

Note that $b_k(t) = 1$, $k = 0, 1, \ldots, n-1$, if and only if there are at least $k+1$ packet arrivals at time t. Thus, the propose of the $n \times n$ prioritized concentrator is to perform traffic concentration, i.e., to sort inputs $\{a_i(t), i = 0, 1, \ldots, n-1\}$ (according to a pre-assigned priority) so that $b_k(t) = a_i(t)$ for some i and that $b_k(t) \geq b_{k+1}(t)$ for $k = 0, 1, \ldots, n-1$.

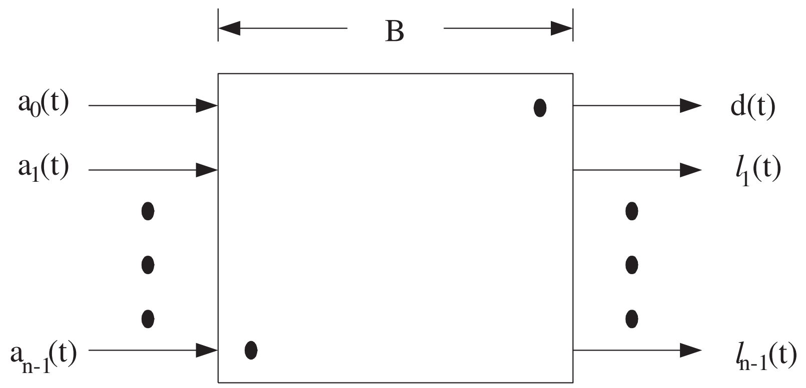

Fig. 13. An n-to-1 multiplexer with buffer B

Definition 11 (Multiplexer) *An n-to-1 multiplexer with buffer B (see Figure 13) is a network element with n input links and n output links. We call the first output link of this multiplexer the departure port and the rest of the output links the loss ports. As shown in Figure 13, let $a_i(t)$, $i = 0, 1, \ldots, n-1$, be the state of the n input links, $d(t)$ be state of the output link for the departure port, $\ell_i(t)$, $i = 1, 2, \ldots, n-1$, be the state of the i^{th} loss port, and $q(t)$ be the number of packets queued at the multiplexer at time t (at the end of the t^{th} time slot). Then the n-to-1 multiplexer with buffer B satisfies the following four properties:*

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

$$q(t) = q(t-1) + \sum_{i=0}^{n-1} a_i(t) - d(t) - \sum_{i=1}^{n-1} \ell_i(t). \tag{7}$$

(P2) Non-idling: there is always a departing packet if there are packets in the buffer or there are arriving packets, i.e.,

$$d(t) = \begin{cases} 0 & \text{if } q(t-1) + \sum_{i=0}^{n-1} a_i(t) = 0 \\ 1 & \text{otherwise} \end{cases}. \tag{8}$$

(P3) Maximum buffer usage: arriving packets are lost only when buffer is full, i.e., for $i = 1, \ldots, n-1$,

$$\ell_i(t) = \begin{cases} 1 & \text{if } q(t-1) + \sum_{i=0}^{n-1} a_i(t) \geq B + i + 1 \\ 0 & \text{otherwise} \end{cases}. \tag{9}$$

(P4) FIFO with prioritized inputs: packets depart in the first in first out (FIFO) order. The priority of the input links is increasing *in the link number. As such, if there are multiple arriving packets, the packet from the largest input link number is put in the multiplexer first. Specifically, the virtual delay for the input $a_i(t)$ is $q(t-1) + \sum_{j=i+1}^{n-1} a_j(t)$, the sum of the number of packets that is stored in the multiplexer at $t-1$ and the number of higher priority packets that arrives at time t.*

From (P1-3), it is well known from the queueing theory that the $q(t)$ process of an n-to-1 multiplexer satisfies the following recursive equation:

$$q(t) = \min[(q(t-1) + a(t) - 1)^+, B], \tag{10}$$

where $a(t) = \sum_{i=0}^{n-1} a_i(t)$ is the total number of arrivals at time t, and $x^+ = \max(0, x)$. In view of (10), if one does not care about the exact match of the loss processes, one can emulate the departure process of an n-to-1 multiplexer by emulating the $q(t)$ process only. This leads to our definition of delayed-loss multiplexers in Definition 12.

Definition 12 (Delayed-loss multiplexer) *An n-to-1 delayed-loss multiplexer with buffer B is a network element with n input links and n output links. As in Definition 11, the first output link of this multiplexer is the departure port and the rest of the output links are the loss ports (we use the same diagrammatic representation in Figure 13). Let $q(t)$ be the number of packets that are queued at the delayed-loss multiplexer at time t (and will be departed from the departure port). Then the n-to-1 delayed-loss multiplexer with buffer B satisfies the recursive equation in (10), (P2) and (P4) of Definition 11.*

As (10) is also the governing equation of the n-to-1 multiplexer, (P2) and (P4) imply that the delayed-loss multiplexer and the multiplexer have identical FIFO departure processes (from the departure ports) if both systems are started from empty systems and subject to identical arrival processes.

We note that an $n \times n$ prioritized concentrator is an n-to-1 multiplexer with buffer 0. It is also an n-to-1 delayed-loss multiplexer with buffer 0.

As in Definition 5, we define scaled SDL multiplexers in Definition 13 below. As explained in Section II-A, scaled SDL multiplexers have the time interleaving property in Proposition 6 .

Definition 13 (SDL multiplexer) *A n-to-1 (delayed-loss) multiplexer is called a n-to-1* SDL *(delayed-loss) multiplexer if the multiplexer is built with delay lines (in Definition 1) and $n \times n$ switches. A n-to-1* SDL *(delayed-loss) multiplexer is with scaling factor k if the delay in every delay line is k times of that in the original n-to-1* SDL *(delayed-loss) multiplexer.*

B. Recursive construction

In this section, we show how one constructs an n-to-1 delayed-loss multiplexer with a large buffer by time interleaving n n-to-1 delayed-loss multiplexers with small buffers. In Figure 14, we consider a network element with n inputs and n outputs. It is a concatenation of a prioritized concentrator, an $n \times n$ switch and n n-to-1 delayed-loss multiplexers with buffer B. The i^{th} output of the $n \times n$

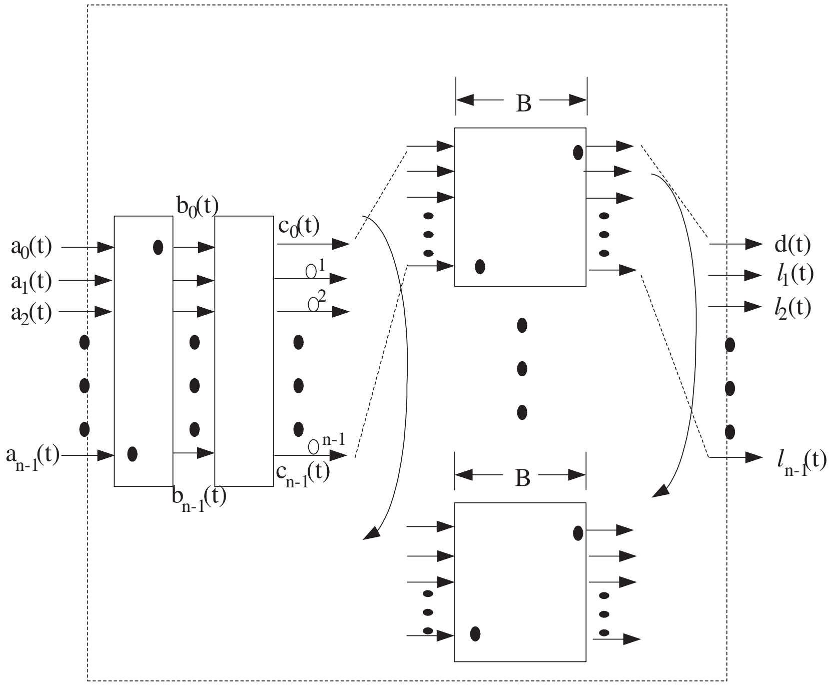

Fig. 14. An n-to-1 delayed-loss multiplexer with buffer $n(B+1)-1$

switch is connected to a delay line with delay i, $i=0,1,\ldots,n-1$. The n outputs of these delay lines are connected to the inputs of the n multiplexers in a round robin fashion. The n outputs of the n multiplexers are connected to the n outputs of the network element in the same order. The state of each multiplexer, i.e., the number of packets stored in the multiplexer, remains unchanged when the multiplexer is not connected. Thus, every multiplexer changes its state every n time slots.

As shown in Figure 14, let $a_i(t)$, $b_i(t)$ and $c_i(t)$, $i=0,1,\ldots,n-1$, be the inputs of the concentrator, the outputs of the concentrator, and the outputs of the $n\times n$ switch. Let $q_i^0(t-1)$, $i=1,\ldots,n$, be the number of packets stored at time $t-1$ in the n-to-1 delayed-loss multiplexer that is going to be connected to the $n\times n$ switch at time $t+i-1$. As such, $q_1^0(t-1)$ is the number of packets stored at time $t-1$ in the n-to-1 delayed-loss multiplexer that is going to be connected to the $n\times n$ switch at time t. Since the n multiplexers with buffer B are connected to the $n\times n$ switch in a round robin fashion, we have

$$q_i^0(t)=q_{i+1}^0(t-1),\quad i=1,\ldots,n-1. \tag{11}$$

Also, we have from the governing equation for a multiplexer in (10) that

$$q_n^0(t)=\min[(q_1^0(t-1)+\sum_{k=0}^{n-2}c_{1+k}(t-1-k)+c_0(t)-1)^+,B]. \tag{12}$$

We define

$$q_i(t-1) = \min[q_i^0(t-1) + \sum_{k=0}^{n-i-1} c_{i+k}(t-1-k), B+1], \quad i = 1, 2 \ldots, n, \tag{13}$$

with the convention that the sum equals 0 if the upper index is smaller than the lower index. The quantity $q_i(t-1)$ represents the number of packets in the system at time $t-1$ that are eligible to leave the system from the departure port at time $t+i-1$ if the arrivals were blocked from time t onward.

For this network element, we also define

$$q(t-1) = \sum_{i=1}^{n} q_i(t-1). \tag{14}$$

Clearly, $q(t-1)$ is the total number of packets in the system that will depart from the departure link from time t onward if the arrivals to the system were blocked. The connection pattern of the $n \times n$ switch in the middle stage of the network element is set according to the value of $q(t-1)$.

As we shall prove later, the network element is a delayed-loss multiplexer with buffer $n(B+1)-1$ (under the operation rule R_n defined below) and $q(t-1)$ is also the number of packets queued in the delayed-loss multiplexer at time $t-1$.

Rule R_n: If $q(t-1) \bmod n = m$, the switch in the middle stage of the network element in Figure 14 is set to the $n \times n$ permutation matrix P_m, $m = 0, 1, \ldots, n-1$, where the (i,j)-th element of P_m is

$$(P_m)_{ij} = \begin{cases} 1 & \text{if } j = (i+m) \bmod n \\ 0 & \text{otherwise} \end{cases}.$$

Specifically, if output link j of the switch is connected with input link i, then $c_j(t) = b_i(t)$, where $j = (i+m) \bmod n = (i + q(t-1)) \bmod n$.

Intuitively, one may view $q_i(t)$'s as the numbers of customers in the parallel queues in Figure 2 and Rule R_n mimics the join-the-shortest-queue policy and the serve-the-longest-queue policy. By so doing, the parallel queues are kept in the most balanced state, i.e., for all t

$$q_1(t) \geq q_2(t) \geq \ldots \geq q_n(t) \geq q_1(t) - 1.$$

As a result, the parallel queues behaves as if it was a single queue with a shared buffer. This leads to the following theorem and its formal proof is shown in Appendix B.

Theorem 14 *If the network element in Figure 14 is operated under Rule R_n and it is started from an empty system, then it is an n-to-1 delayed-loss multiplexer with buffer $n(B+1)-1$.*

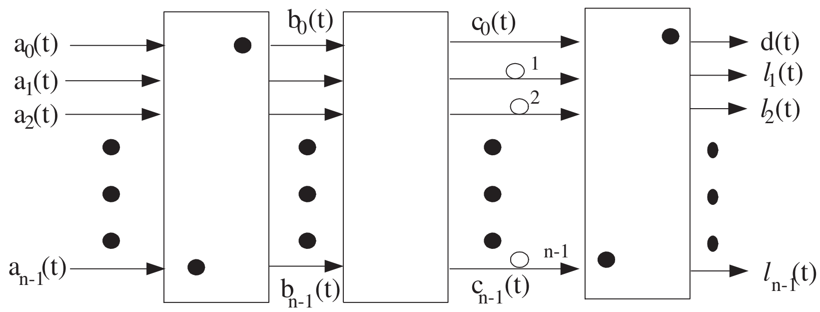

Fig. 15. An n-to-1 delayed-loss multiplexer with buffer $n-1$

Example 15 Since a prioritized concentrator is an n-to-1 multiplexer with buffer 0, it follows from Theorem 14 that the network element in Figure 15 is an n-to-1 delayed-loss multiplexer with buffer $n-1$. For this example, we have from (13) that

$$q_i(t-1) = \min[\sum_{k=0}^{n-i-1} c_{i+k}(t-1-k), 1] = \max_{0 \le k \le n-i-1} [c_{i+k}(t-1-k)], \tag{15}$$

for $i = 1, 2, \ldots, n-1$, and $q_n(t-1) = 0$. Thus,

$$q(t-1) = \sum_{i=1}^{n-1} \max_{0 \le k \le n-i-1} [c_{i+k}(t-k-1)]. \tag{16}$$

The connection pattern of the $n \times n$ switch at time t is then set according to Rule R_n that only depends on $q(t-1)$. We note that the total number of packets that can be stored in the optical delay lines of the network element in Figure 15 is $n(n-1)/2$, which is much larger than the designed buffer size $n-1$. It means that some packets that should have been discarded when they arrive are still stored in the system. In view of (15), one can see that at most one packet can be departed from the departure port at time $t+i-1$ and the rest of packets that are stored in the delay lines will be departed from the loss ports. This is how the delayed losses occur!

Example 16 In Figure 16, we illustrate how one applies Theorem 14 to construct an n-to-1 delayed-loss multiplexer with buffer n^2-1 from the n-to-1 delayed-loss multiplexer with buffer $n-1$ in Figure 15. Note that the inner block in Figure 16 is exactly the same as the n-to-1 delayed-loss multiplexer with buffer $n-1$ in Figure 15 except that the delay in every delay line is scaled n times. As such, it is a n-to-1 delayed-loss SDL multiplexer with buffer 1 and scaling factor n. From the time interleaving property of scaled SDL multiplexers in Proposition 6, this can be viewed as time interleaving n n-to-1 delayed-loss multiplexers with buffer $n-1$. As a direct consequence of

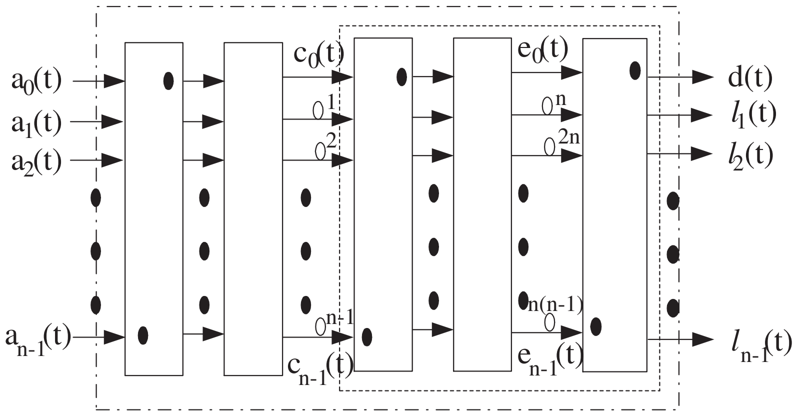

Fig. 16. An n-to-1 delayed loss multiplexer with $B = n^2 - 1$.

Theorem 14, the network element in Figure 16 is indeed an n-to-1 delayed-loss multiplexer with buffer $n^2 - 1$. As shown in (16) in Example 15, $q_j^0(t-1)$, i.e., the number of packets stored in the multiplexer that is going to be connected at time $t + j - 1$, has the following form:

$$q_j^0(t-1) = \sum_{i=1}^{n-1} \max_{0 \le k \le n-i-1} [e_{i+k}(t - (k+1)n + j - 1)], \quad j = 1, 2, \ldots, n. \tag{17}$$

Moreover, the second $n \times n$ switch is set according to $q_1^0(t-1)$. To see the operation of the first $n \times n$ switch, we have from (13) that

$$q_j(t-1) = \min[q_j^0(t-1) + \sum_{k=0}^{n-j-1} c_{j+k}(t-1-k), B+1], \quad j = 1, 2 \ldots, n. \tag{18}$$

Thus, the operation of the first $n \times n$ switch is set according to $q(t-1) = \sum_{j=1}^{n} q_j(t-1)$.

By Theorem 14 and the time interleaving property in Proposition 6, one can then recursively construct a multi-stage multiplexer with a large buffer. One can further combine the operation of the prioritized concentrator and the $n \times n$ switch at each stage by an $n \times n$ switch. In Figure 17, we show the construction of an n-to-1 delayed-loss multiplexer with buffer $n^k - 1$. All its switching patterns are completely determined by the state of the multiplexer.

Now we have shown from Theorem 14 a way to control the switching patterns in the n-to-1 delayed-loss multiplexer with buffer $n^k - 1$ in Figure 17. One key observation from this is that those packets departing from the departure port have the same delays as the (ideal) n-to-1 multiplexer with buffer $n^k - 1$ since the two departure processes are identical and both of them are FIFO. Moreover, for any packet that departs from the departure port and experiences delay $0 \le d \le n^k - 1$, there is a *unique* path through the network element. The path can be determined by

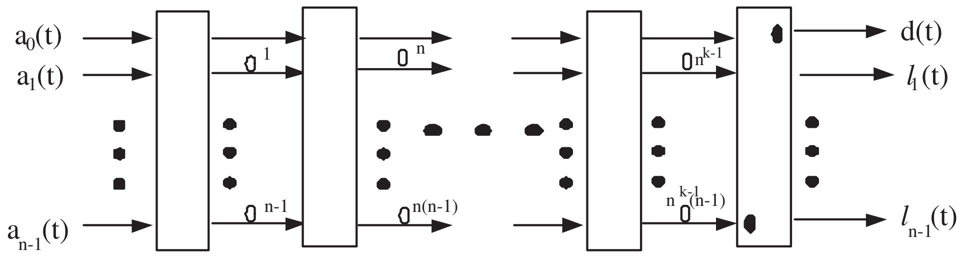

Fig. 17. An n-to-1 delayed-loss multiplexer with $B = n^k - 1$.

the unique decomposition of $d = \sum_{j=1}^{k} r_j n^{j-1}$ and the packet is sent through the network element by taking the ${r_j}^{th}$ output link at the j^{th} $n \times n$ switch. However, the path of a packet that departs from a loss port cannot be determined this way. Thus, if we discard all the packets that depart from the loss ports before entering the network element, then we are left with the packets that depart from the departure port and the paths of those packets can be determined upon their arrivals. As these paths are identical to those from the n-to-1 delayed-loss multiplexer, we conclude that these paths do not conflict with each other, i.e., no more than one packet occupies the same link at any time. This leads to the *self-routing* multiplexer in Figure 3.

In Figure 3, we replace the first $n \times n$ switch by an $n \times (2n-1)$ switch in Figure 17. The network element in Figure 3 keeps track of the number of packets stored in it. If such a number exceeds $n^k - 1$, further arrivals are lost immediately. Specifically, let $q(t)$ be the number of packets stored in the network element. Then $q(t)$ is governed by

$$q(t) = \min[(q(t-1) + \sum_{i=0}^{n-1} a_i(t) - 1)^+, n^k - 1], \tag{19}$$

and

$$\ell_i(t) = \begin{cases} 1 & \text{if } q(t-1) + \sum_{i=0}^{n-1} a_i(t) \geq n^k + i \\ 0 & \text{otherwise} \end{cases}, \tag{20}$$

for $i = 1, \ldots, n-1$. Let q be the number of packets stored in the network element when a particular packet enters the network element. Then we have $0 \leq q \leq n^k - 1$ and there exists a unique vector $r = (r_1, r_2, \ldots, r_k)$ with $0 \leq r_j \leq n-1$ for all j such that

$$q = \sum_{j=1}^{k} r_j n^{j-1}.$$

The packet is then routed through the network element by taking the ${r_j}^{th}$ output link at the j^{th} $n \times n$ switch. Note that we now not only match the departure process but also the loss processes. Thus, the network element in Figure 3 is an n-to-1 multiplexer with buffer $n^k - 1$.

IV. Conclusions

In this paper, we developed mathematical theory for recursive construction of FIFO optical multiplexers with switched delay lines. In Section II, we showed that exact emulation of 2-to-1 FIFO optical multiplexers can be achieved for both the departure process and the loss process. In Section III, we extended the results for 2-to-1 multiplexers to n-to-1 delayed-loss multiplexers by emulating the departure process only. By discarding packets in advance, we proposed a self-routing n-to-1 multiplexer that has a very simple control mechanism.

There are several research topics that need further investigation.

(i) Multiplexers for variable length packets: in this paper, we only considered packets of the same size. In the current Internet, packets are of variable lengths and the problem of supporting variable length packets in optical networks becomes important (see.e.g., [22], [19], [1]). In order to use the fixed length multiplexers developed in our paper, segmentation and reassembly might be needed. It might be of interest to develop an SDL unit that does segmentation and reassembly. A follow-up paper is in [6].

(ii) Scheduling policies other than FIFO: in order to achieve quality of service (QoS), more sophisticated scheduling policies, such as priority queues [15], [23], [17] and the earliest deadline first policy, might be needed. Implementation of scheduling policies other than FIFO via switched delay lines appears to be difficult.

Appendix

I. Appendix A

In this section, we prove Theorem 7.

As shown in Figure 8, let $a_1(t)$ (resp. $a_0(t)$) be the state of the dotted (resp. undotted) input of the prioritized concentrator at time t, $b_1(t)$ (resp. $b_0(t)$) be the state of the lower (resp. upper) output of the 2×2 switch at time t, and $d(t)$ (resp. $\ell(t)$) be the state of the link for departing (resp. lost) packets at time t. Also, let $q_1(t-1)$ be the number of packet stored in the 2-to-1 multiplexer at time $t-1$ that is going to be *connected* to the 2×2 switch at time t and $q_2(t-1)$ be the number of packet stored in the other 2-to-1 multiplexer at time $t-1$. As there is one unit delay line after the 2×2 switch in Figure 8, the state of the network element at time $t-1$ is the 3-vector $(b_1(t-1), q_1(t-1), q_2(t-1))$.

Now we write down the governing equations for the network element. As the 2×2 switch is connected to the two multiplexers alternatively, we have from (4)-(6) that

$$q_2(t) = q_1(t-1) + b_1(t-1) + b_0(t) - d(t) - \ell(t), \tag{21}$$

$$q_1(t) = q_2(t-1), \tag{22}$$

$$d(t) = \begin{cases} 0 & \text{if } q_1(t-1) = b_1(t-1) = b_0(t) = 0 \\ 1 & \text{otherwise} \end{cases}, \tag{23}$$

$$\ell(t) = \begin{cases} 1 & \text{if } q_1(t-1) = B \text{ and } b_1(t-1) = b_0(t) = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{24}$$

Let

$$q(t-1) = q_1(t-1) + q_2(t-1) + b_1(t-1) \tag{25}$$

be the total number of packets stored in the network element at the end of the $t-1^{th}$ time slot. According to the operation rule of the 2×2 switch, the switch is set to the cross state if $q(t-1)$ is an odd number and is set to the bar state otherwise. From the operation rule of a prioritized concentrator in Definition 3, the state of the upper input of the 2×2 switch is $\max[a_0(t), a_1(t)]$ and the state of the lower input of the 2×2 switch is $\min[a_0(t), a_1(t)]$. Thus,

$$b_1(t) = \begin{cases} \max[a_0(t), a_1(t)] & \text{if } q(t-1) \text{ is odd} \\ \min[a_0(t), a_1(t)] & \text{otherwise} \end{cases}. \tag{26}$$

Similarly,

$$b_0(t) = \begin{cases} \min[a_0(t), a_1(t)] & \text{if } q(t-1) \text{ is odd} \\ \max[a_0(t), a_1(t)] & \text{otherwise} \end{cases}. \tag{27}$$

Now we show that the network element is a 2-to-1 multiplexer with buffer $2B+1$ by verifying the four properties in Definition 4. Since arriving packets from the two input links are either stored in the network element or transmitted through the two output links, flow conservation of the network element is obviously satisfied. We will verify the other three properties by induction on t with the following additional induction hypothesis.

(P5) If we start from an empty system, i.e., $b_1(0) = q_1(0) = q_2(0)$, then

$$q_2(t) \le b_1(t) + q_1(t) \le q_2(t) + 1, \quad \forall\, t. \tag{28}$$

In view of the induction hypothesis in (28), there are four possible cases as described below.

Case 1. $(b_1(t-1), q_1(t-1), q_2(t-1)) = (0,0,0)$:

In this case, we have from (21)-(24) that

$$d(t) = b_0(t), \tag{29}$$

$$\ell(t) = 0, \tag{30}$$

$$q_2(t) = 0, \tag{31}$$

$$q_1(t) = 0. \tag{32}$$

In this case, $q(t-1)=0$. It then follows from (26) and (27) that

$$b_0(t) = \max[a_0(t), a_1(t)], \tag{33}$$

$$b_1(t) = \min[a_0(t), a_1(t)], \tag{34}$$

Observe that $0 \leq b_1(t) \leq 1$. Thus, the induction hypothesis in (28) follows from (31) and (32).

Since $q(t-1)=0$, there should be no packet loss at time t for a 2-to-1 multiplexer with buffer $2B+1$. Equation (30) verifies this. Also, there should a packet departure at time t if there is at least one packet arrival at time t. This is shown by (29) and (33).

As $q(t-1)=0$, the 2×2 switch is set to the bar state. It is easy to see that the virtual delay for $a_1(t)$ is 0 and the virtual delay for $a_0(t)$ is $a_1(t)$ so that the FIFO order is maintained.

Case 2. $(b_1(t-1), q_1(t-1), q_2(t-1)) = (1, q-1, q)$ or $(b_1(t-1), q_1(t-1), q_2(t-1)) = (0, q, q)$ for some $0 < q \leq B$:

In this case, we have from (21)-(24) that

$$d(t) = 1, \tag{35}$$

$$\ell(t) = 0, \tag{36}$$

$$q_2(t) = q + b_0(t) - 1, \tag{37}$$

$$q_1(t) = q. \tag{38}$$

In this case, $q(t-1) = 2q$ and the 2×2 switch is set to the bar state. It then follows from (26) and (27) that

$$b_0(t) = \max[a_0(t), a_1(t)], \tag{39}$$

$$b_1(t) = \min[a_0(t), a_1(t)], \tag{40}$$

Thus,

$$b_0(t) - 1 \leq b_1(t) \leq b_0(t). \tag{41}$$

The induction hypothesis in (28) then follows from (41), (37) and (38).

Since $0 < q(t-1) = 2q < 2B+1$, there should a packet departure at time t and there should be no packet loss at time t for a 2-to-1 multiplexer with buffer $2B+1$. Equations (35) and (36) verify these for this case.

Now we show that the virtual delay for $a_1(t)$ is $q(t-1)$. Since the 2×2 switch is set to the bar state, $a_1(t)$ is routed to the multiplexer with buffer B that is going to be connected at time t. As this multiplexer with buffer B is operated under the FIFO policy, the number of packets that should depart before $a_1(t)$ is $q_1(t-1) + b_1(t-1) = q$. Note that this multiplexer with buffer B

is connected to the outputs every two time slots. Thus, the virtual delay of $a_1(t)$ is $2q$, which is exactly $q(t-1)$. On the other hand, if $a_1(t)=1$, then $a_0(t)$ is routed to the multiplexer that is going to be connected at time $t+1$. As $q_2(t-1)=q$, the virtual delay for $a_0(t)=2q+1$. Thus, the FIFO order is maintained.

Case 3. $(b_1(t-1), q_1(t-1), q_2(t-1)) = (0, q+1, q)$ or $(b_1(t-1), q_1(t-1), q_2(t-1)) = (1, q, q)$ for some $0 < q < B$:

In this case, we have from (21)-(24) that

$$d(t) = 1, \tag{42}$$

$$\ell(t) = 0, \tag{43}$$

$$q_2(t) = q + b_0(t), \tag{44}$$

$$q_1(t) = q. \tag{45}$$

In this case, $q(t-1) = 2q+1$ and the 2×2 switch is set to the cross state. It then follows from (26) and (27) that

$$b_0(t) = \min[a_0(t), a_1(t)], \tag{46}$$

$$b_1(t) = \max[a_0(t), a_1(t)], \tag{47}$$

Thus,

$$b_0(t) \le b_1(t) \le b_0(t) + 1. \tag{48}$$

The induction hypothesis in (28) then follows from (48), (44) and (45).

Since $0 < q(t-1) = 2q+1 < 2B+1$, there should a packet departure at time t and there should be no packet loss at time t for a 2-to-1 multiplexer with buffer $2B+1$. Equations (42) and (43) verify these.

Now we show that the virtual delay for $a_1(t)$ is $q(t-1)$. Since the 2×2 switch is set to the cross state, $a_1(t)$ is routed to the multiplexer with buffer B that is going to be connected at time $t+1$. As the number of packets in this multiplexer is $q_2(t-1) = q$, the virtual delay of $a_1(t)$ is $2q+1$, which is exactly $q(t-1)$. On the other hand, if $a_1(t) = 1$, then $a_0(t)$ is routed to the multiplexer that is going to be connected at time t. As $q_1(t-1) + b_1(t-1) = q+1$, the virtual delay for $a_0(t) = 2q+2$. Thus, the FIFO order is maintained.

Case 4. $(b_1(t-1), q_1(t-1), q_2(t-1)) = (1, B, B)$:

In this case, we have from (21)-(24) that

$$d(t) = 1, \tag{49}$$

$$\ell(t) = b_0(t), \tag{50}$$

$$q_2(t) = B, \tag{51}$$

$$q_1(t) = B. \tag{52}$$

In this case, $q(t-1) = 2B+1$ and the 2×2 switch is set to the cross state. Thus, (46) and (47) still hold. Since $0 \leq b_1(t) \leq 1$, the induction hypothesis in (28) then follows from (51) and (52).

Since $q(t-1) = 2B+1$, there should a packet departure at time t. This is shown in (49). On the other hand, since the buffer is full, there should be a packet loss at time t if two packets arrive at time t. Equations (50) and (46) verify this.

Verification of the virtual delay is the same as that in Case 3.

II. Appendix B

In this section, we prove Theorem 14. The proof of Theorem 14 requires the following lemmas. In Lemma 17, we first derive the governing equations for $q_i(t)$, $i = 1, \ldots, n$. To gain the intuition of our proof, one may view $q_i(t)$'s as the numbers of customers in the parallel queues in Figure 2.

Lemma 17 *The quantities $q_i(t)$, $i = 1, 2, \ldots, n$, satisfy the following recursive equations:*

$$q_i(t) = \min[q_{i+1}(t-1) + c_i(t), B+1], \quad i = 1, 2, \ldots, n-1, \tag{53}$$

and

$$q_n(t) = \min[(q_1(t-1) + c_0(t) - 1)^+, B]. \tag{54}$$

Proof. We have from (11) and (13) that for $i = 1, \ldots, n-1$,

$$\begin{aligned}
q_i(t) &= \min[q_i^0(t) + \sum_{k=0}^{n-i-1} c_{i+k}(t-k), B+1] \\
&= \min[q_{i+1}^0(t-1) + \sum_{k=0}^{n-i-2} c_{i+1+k}(t-1-k) + c_i(t), B+1] \\
&= \min[q_{i+1}^0(t-1) + \sum_{k=0}^{n-i-2} c_{i+1+k}(t-1-k) + c_i(t), B+1, B+1+c_i(t)] \\
&= \min\Big[\min[q_{i+1}^0(t-1) + \sum_{k=0}^{n-i-2} c_{i+1+k}(t-1-k), B+1] + c_i(t), B+1\Big] \\
&= \min[q_{i+1}(t-1) + c_i(t), B+1].
\end{aligned}$$

Moreover, it follows from (12) and (13) that

$$q_n(t) = \min[q_n^0(t), B+1] = q_n^0(t)$$

$$
\begin{aligned}
&= \min[(q_1^0(t-1) + \sum_{k=0}^{n-2} c_{1+k}(t-1-k) + c_0(t) - 1)^+, B] \\
&= \min[(q_1^0(t-1) + \sum_{k=0}^{n-2} c_{1+k}(t-1-k) + c_0(t) - 1)^+, B, B + c_0(t)] \\
&= \min\left[(\min[q_1^0(t-1) + \sum_{k=0}^{n-2} c_{1+k}(t-1-k), B+1] + c_0(t) - 1)^+, B\right] \\
&= \min[(q_1(t-1) + c_0(t) - 1)^+, B]
\end{aligned}
$$

■

Lemma 18 shows that if the vector $(q_1(t-1), q_2(t-1), \ldots, q_n(t-1))$ is in the most balanced state, i.e., the difference between the largest element and the smallest element is at most 1, then Rule R_n behaves as if it is the join-the-shortest-queue policy and the state is still kept in the most balanced state.

Lemma 18 *If*

$$q_1(t-1) \geq q_2(t-1) \geq \ldots \geq q_n(t-1) \geq q_1(t-1) - 1, \tag{55}$$

then under Rule R_n

$$
\begin{aligned}
& q_1(t-1) + c_0(t) \geq q_2(t-1) + c_1(t) \geq \ldots \\
& \qquad \geq q_n(t-1) + c_{n-1}(t) \geq q_1(t-1) + c_0(t) - 1.
\end{aligned} \tag{56}
$$

Proof. We define the function $h(k, q(t-1)) = k - q(t-1) \bmod n$. Then under Rule R_n, we have

$$c_i(t) = b_{h(i,q(t-1))}(t), \quad i = 0, 1, \ldots, n-1. \tag{57}$$

Note that for any fixed $q(t-1)$, $h(k, q(t-1))$ is increasing with k, except when $k = q(t-1) - 1 \bmod n$ at which $h(k, q(t-1)) = n-1$ and $h(k+1, q(t-1)) = 0$.

We first show that

$$q_{i+1}(t-1) + c_i(t) \geq q_{i+2}(t-1) + c_{i+1}(t), \quad i = 0, 1, \ldots, n-2. \tag{58}$$

If $i \neq (q(t-1) - 1) \bmod n$, then

$$h(i+1, q(t-1)) \geq h(i, q(t-1)).$$

Since $b_i(t)$ is decreasing in i ($b_i(t)$'s are the outputs from a prioritized concentrator), it then follows that

$$c_i(t) = b_{h(i,q(t-1))}(t) \geq b_{h(i+1,q(t-1))}(t) = c_{i+1}(t). \tag{59}$$

Hence, (58) follows from (55) and (59).

On the other hand, if $i = (q(t-1)-1) \bmod n$, then we have from (55) that $q_{i+1}(t-1) = q_{i+2}(t-1)+1$ for this case. This implies that

$$\begin{aligned} & q_{i+1}(t-1) + c_i(t) = q_{i+2}(t-1) + 1 + c_i(t) \\ & \geq q_{i+2}(t-1) + c_{i+1}(t) \end{aligned}$$

Now we show that

$$q_n(t-1) + c_{n-1}(t) \geq q_1(t-1) + c_0(t) - 1. \tag{60}$$

If $q(t-1) \bmod n \neq 0$, then $h(n-1, q(t-1)) \leq h(0, q(t-1))$. Since $b_i(t)$ is decreasing in i, it then follows that

$$c_{n-1}(t) = b_{h(n-1,q(t-1))}(t) \geq b_{h(0,q(t-1))}(t) = c_0(t).$$

That (60) holds then follows from the last inequality in (55). On the other hand, if $q(t-1) \bmod n = 0$, then we have from (55) that $q_1(t-1) = q_2(t-1) = \ldots = q_n(t-1)$. The inequality in (60) holds trivially as $0 \leq c_j(t) \leq 1$ for all j. ■

Lemma 19 shows that Rule R_n behaves as if it is the serve-the-longest-queue policy and the state $(q_1(t), q_2(t), \ldots, q_n(t))$ is always kept in the most balanced state if the state is started from an empty system.

Lemma 19 *If the network element in Figure 14 is operated under Rule R_n and it is started from an empty system, then for all $t \geq 0$*

$$q_{i+1}(t) \leq q_i(t), \quad i = 1, 2, \ldots, n-1, \tag{61}$$

and

$$q_1(t) \leq q_n(t) + 1. \tag{62}$$

Proof. Since we assume that the network element starts from an empty system, Eq. (61) and (62) hold for $t = 0$. Now we assume that they hold for some $t-1$.

Using the induction hypotheses and Lemma 18, we have from (53) that $q_i(t) \geq q_{i+1}(t)$ for $i = 1, \ldots, n-2$. To see that $q_n(t) + 1 \geq q_1(t)$, observe from (54) that

$$\begin{aligned} q_n(t) + 1 &= \min[(q_1(t-1) + c_0(t) - 1)^+, B] + 1 \\ &= \min[(q_1(t-1) + c_0(t) - 1)^+ + 1, B + 1] \\ &\geq \min[q_1(t-1) + c_0(t), B + 1]. \end{aligned}$$

Once again, using the induction hypotheses, Lemma 18 and (53), we have that

$$\begin{aligned} q_n(t)+1 &\geq \min[q_1(t-1)+c_0(t), B+1] \\ &\geq \min[q_2(t-1)+c_1(t), B+1] = q_1(t). \end{aligned}$$

It remains to show that $q_{n-1}(t) \geq q_n(t)$. Note from (54), the induction hypotheses, the last inequality in (56), and (53) that

$$\begin{aligned} q_n(t) &= \min[(q_1(t-1)+c_0(t)-1)^+, B] \\ &\leq \min[(q_n(t-1)+c_{n-1}(t))^+, B] \\ &= \min[q_n(t-1)+c_{n-1}(t), B] \\ &\leq \min[q_n(t-1)+c_{n-1}(t), B+1] = q_{n-1}(t). \end{aligned}$$

■

(Proof of Theorem 14) We first show that $q(t)$ satisfies the following recursive equation:

$$q(t) = \min\left[\left(q(t-1)+\sum_{i=0}^{n-1} a_i(t)-1\right)^+, n(B+1)-1\right]. \tag{63}$$

Since the network element in Figure 14 is started from an empty system, we first consider the case that $q(t-1)=0$. If $q(t-1)=0$, then $q_i(t-1)=0$ for all i and $c_i(t)=b_i(t)$, $i=0,1,2\ldots,n-1$ (according to Rule R_n). Therefore, we have from (53) and (54) that

$$\begin{aligned} \sum_{i=1}^{n} q_i(t) &= \sum_{i=1}^{n-1} \min[q_{i+1}(t-1)+c_i(t), B+1] + \min[(q_1(t-1)+c_0(t)-1)^+, B] \\ &= \sum_{i=1}^{n-1} b_i(t) \end{aligned}$$

We argue that

$$\sum_{i=1}^{n-1} b_i(t) = (\sum_{i=0}^{n-1} b_i(t)-1)^+. \tag{64}$$

To see this, note that (64) holds trivially if $b_0(t)=1$. On the other hand, if $b_0(t)=0$, then $b_i(t)=0$, $i=1,2,\ldots,n-1$, as $b_0(t)$ is the largest element in $b_i(t)$. Both sides of (64) equal 0. Using (64) yields

$$q(t) = \sum_{i=1}^{n} q_i(t) = (\sum_{i=0}^{n-1} b_i(t)-1)^+$$

$$
\begin{aligned}
&= (\sum_{i=0}^{n-1} a_i(t) - 1)^+ \\
&= (a(t) - 1)^+ \\
&= \min[(a(t) - 1)^+, n(B+1) - 1].
\end{aligned}
$$

Thus, (63) holds for the case $q(t-1) = 0$.

Now consider the case that $q(t-1) > 0$. As $q_1(t-1)$ is the largest element in $q_i(t-1)$, $i = 1, 2, \ldots, n$ (Lemma 19), it follows that $q_1(t-1) > 0$. In conjunction with (54), we have

$$
q_n(t) = \min[q_1(t-1) + c_0(t) - 1, B] = \min[q_1(t-1) + c_0(t), B+1] - 1. \tag{65}
$$

Thus, we have from (53) and (65) that

$$
q(t) = \sum_{i=1}^{n} q_i(t) = \sum_{i=0}^{n-1} \min[q_{i+1}(t-1) + c_i(t), B+1] - 1. \tag{66}
$$

We argue that

$$
\sum_{i=0}^{n-1} \min[q_{i+1}(t-1) + c_i(t), B+1] = \min[\sum_{i=0}^{n-1} q_{i+1}(t-1) + c_i(t), n(B+1)]. \tag{67}
$$

If $q_1(t-1)+c_0(t) \leq B+1$, then it follows from Lemma 19 and Lemma 18 that $q_{i+1}(t-1)+c_i(t) \leq B+1$ for all $i = 0, 1, \ldots, n-1$. Thus,

$$
\begin{aligned}
&\sum_{i=0}^{n-1} \min[q_{i+1}(t-1) + c_i(t), B+1] \\
&= \sum_{i=0}^{n-1} q_{i+1}(t-1) + c_i(t) \\
&= \min[\sum_{i=0}^{n-1} q_{i+1}(t-1) + c_i(t), n(B+1)].
\end{aligned}
$$

On the other hand, if $q_1(t-1) + c_0(t) \geq B+2$, then it follows from Lemma 19 and Lemma 18 that $q_{i+1}(t-1) + c_i(t) \geq B+1$ for all $i = 0, 1, \ldots, n-1$. Thus,

$$
\begin{aligned}
&\sum_{i=0}^{n-1} \min[q_{i+1}(t-1) + c_i(t), B+1] \\
&= \sum_{i=0}^{n-1} B + 1 = n(B+1) \\
&= \min[\sum_{i=0}^{n-1} q_{i+1}(t-1) + c_i(t), n(B+1)].
\end{aligned}
$$

Using (67) in (66) yields

$$\begin{aligned} q(t) &= \sum_{i=1}^{n} q_i(t) \\ &= \min[\sum_{i=0}^{n-1} q_{i+1}(t-1) + c_i(t), n(B+1)] - 1 \\ &= \min[\sum_{i=1}^{n} q_i(t-1) + \sum_{i=0}^{n-1} a_i(t), n(B+1)] - 1 \\ &= \min[q(t-1) + a(t), n(B+1)] - 1 \\ &= \min[q(t-1) + a(t) - 1, n(B+1) - 1]. \end{aligned}$$

Thus, we have shown that $q(t)$ satisfies (63).

Now we show the non-idling property in (P2), i.e.,

$$d(t) = 1_{\{q(t-1)+a(t)>0\}}. \tag{68}$$

From the non-idling property for the delayed-loss multiplexer connected to the outputs at time t, it follows that

$$d(t) = 1_{\{q_1(t-1)+c_0(t)>0\}}.$$

Note that $q_1(t-1) > 0$ if and only if $q(t-1) > 0$. Therefore, if $q_1(t-1) > 0$,

$$d(t) = 1 = 1_{\{q(t-1)+a(t)>0\}}.$$

On the other hand, if $q_1(t-1) = q(t-1) = 0$, then $c_0(t) = b_0(t)$ from Rule R_n. As $b_0(t)$ is the largest element in $b_i(t)$, $i = 0, 1, \ldots, n-1$, $c_0(t) > 0$ if and only if $a(t) > 0$. Thus,

$$d(t) = 1_{\{c_0(t)>0\}} = 1_{\{a(t)>0\}}.$$

Finally, we verify the departure order is FIFO in (P4). Without loss of generality, assume that $q(t-1) = mn + k$ for some $m \geq 0$ and $0 \leq k \leq n-1$. From Lemma 19, it follows that

$$q_i(t-1) = \begin{cases} m+1 & 1 \leq i \leq k \\ m & k+1 \leq i \leq n \end{cases}.$$

Also, we have $b_0(t) = c_k(t)$ from Rule R_n. From (53) and (54) in Lemma 17, $c_k(t)$ is added to the end of $q_{k+1}(t-1)$. Thus, the virtual delay for $b_0(t)$ is $nq_{k+1}(t-1) + k = mn + k = q(t-1)$ as the n multiplexers are served in a round robin fashion. Similarly, if $b_0(t) = 1$, then $b_1(t)$ is added to the end of $q_{k+2 \bmod n}(t-1)$. One can easily verify that the virtual delay for $b_1(t)$ is $q(t-1)+1$. Continuing the same argument shows that the virtual delay for $b_i(t)$ is $q(t-1)+i$ if $b_0(t) = b_1(t) = \ldots = b_{i-1}(t) = 1$. Thus, the FIFO order is maintained. ■

References

[1] F. Callegati, "Approximate modeling of optical buffers for variable length packets," *Photonic Network Communications*, Vol. 3, pp. 383-390, 2001.

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

[3] 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.

[4] 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.

[5] C.-S. Chang, D.-S. Lee and C.-M. Lien, "Load Balanced Birkhoff-von Neumann Switches, Part II: Multi-stage Buffering," *Computer Communications*, Vol. 25, pp. 623-634, 2002.

[6] C.-S. Chang, D.-S. Lee and C.-K. Tu, "Using switched delay lines for exact emulation of FIFO multiplexers with variable length bursts," *Proceedings of IEEE INFOCOM*, 2003.

[7] S.-T. Chuang, A. Goel, N. McKeown and B. Prabhkar, "Matching output queueing with a combined input output queued switch," *IEEE INFOCOM'99*, pp. 1169-1178, New York, 1999.

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

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

[14] S. Iyer and N. McKeown, "Making parallel packet switches practical." *IEEE INFOCOM 2001*.

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

[16] I. Keslassy and N. McKeown, "Maintaining packet order in two-stage switches," *preprint*, 2001.

[17] M.R.N. Ribeiro and M.J. O'Mahony "Traffic management in photonic packet switching nodes by priority assignment and selective discarding," *Computer Communications*, Vol 24, pp. 1689-1701, 2001.

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

[19] L. Tancevski, S. Yegnanarayanan, G. Castanon, et al. "Optical routing of asynchronous, variable length packets" *Journal on Selected Areas in Communications*, Vol. 18, pp. 2084-2093, 2000.

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

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

[22] M. Yoo, C. Qiao and S. Dixit, "QoS performance of optical burst switching in IP-over-ATM networks," *IEEE Journal on Selected Areas in Communications*, Vol. 18, pp. 2062-2071, 2000.

[23] K.Y. Yun, K.W. James, R.H. Fairlie-Cuninghame, S. Chakraborty, and R.L. Cruz, "A self-timed real-time sorting network," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 8, pp. 356-363, 2000.