Using Switched Delay Lines for Exact Emulation of FIFO Multiplexers with Variable Length Bursts

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*—It has been studied extensively in the literature how one achieves exact emulation of First In First Out (FIFO) multiplexers for fixed size cells (or packets) using optical crossbar Switches and fiber Delay Lines (SDL). In this paper, we take a step further and propose a new architecture that achieves exact emulation of FIFO multiplexers for variable length bursts. Our architecture consists of two blocks: a cell scheduling block and an FIFO multiplexer for fixed size cells. Both blocks are made of SDL units. The objective of the cell scheduling block is to schedule cells in a burst to the right input at the right time so that cells in the same burst depart *contiguously* from the multiplexer for fixed size cells. We show that cell scheduling can be done efficiently by keeping track of a single state variable, called the total virtual waiting time in this paper. Moreover, the delay through the cell scheduling block is bounded above by a constant that only depends on the number of inputs and the maximum number of cells in a burst. Such a delay bound provides a limit on the number of fiber delay lines needed in the cell scheduling block.**

***Index Terms*—conflict resolution, exact emulation, optical multiplexers, multi-stage switches, switched delay lines, variable length bursts**

I. Introduction

One of the key challenges to build high speed packet switches that scale with the transmission speed of fiber optics is to resolve conflicts of packets competing for the same resource. There are two common approaches. The first approach is to use electronic buffers (see e.g., [7], [14], [2], [16]). As the accessing speed of electronic memory is considerably slower than the speed of fiber optics, this approach in general requires a lot of parallel buffers to achieve the needed speedup for fiber optics. The other approach is to resolve conflicts directly by optical Switches and fiber Delay Lines (SDL) (see e.g., [11], [9], [22], [25] and references therein). Unlike electronic memory, fiber delay lines are not capable of providing random memory access. They can only be accessed in a predetermined sequential manner. As such, conflict resolution by SDL is in general much more difficult than that by electronic memory.

The approach of using SDL to resolve conflicts was proposed in the CORD (contention resolution by delay lines) project [4], [5], [6]. The idea is to redistribute packets through delay lines with different delays so that conflicts can be resolved over time and space. As indicated in [11], there are several architectures proposed in the literature that resolve conflicts via SDL. In [20], [8], a genuine SDL 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 relatively easy and only requires local information. However, the number of 2×2 switches in COD is proportional to the buffer size. In [10], a more efficient design, Logarithm Delay-Line Switch, is proposed for the 2×2 buffered switch. 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 more than 2 input/output ports. Such an architecture relies on a special hardware, called a primitive switching element (PSE). The control of the PSEs is much more difficult than the control in COD and the 2×2 Logarithm Delay-Line Switch. To solve the control problem, in [3] we developed mathematical theory for recursive construction of FIFO optical multiplexers with large buffers. Such theory leads to self-routing multiplexers, where the routing path of a packet through the multi-stage SDL units can be determined upon its arrival (a brief review of the self-routing multiplexers will be given in Section II).

Most of the prior works in [20], [8], [10], [12], [3] are targeted for exact emulation of multiplexers (or switches) for fixed size packets (or cells). A natural question is whether one can use the multiplexers for fixed size packets (or cells) for exact emulation of multiplexers with variable length bursts. As in electronic buffers, this requires performing burst *segmentation* and *reassembly*. In this paper, we assume that burst segmenta-

This research is supported in part by the National Science Council, Taiwan, R.O.C., under Contract NSC-91-2213-E007-010, and the program for promoting academic excellence of universities 89-E-FA04-1-4.

tion is feasible. Each variable length burst can be divided into a contiguous sequence of fixed size *cells*, and each cell can then be transmitted within a time slot in the multiplexer for fixed size cells. In doing burst segmentation, we note that there is a granularity problem for choosing the *right* cell size (see e.g., [19], [1]). We will not address the granularity problem in this paper. Our focus is on how one assembles cells back into bursts by using optical switches and fiber delay lines.

There are two natural places for burst reassembly: (i) after the multiplexer for fixed size cells, and (ii) before the multiplexer for fixed size cells. The former approach is much more difficult to realize by SDL as it has to take the multiplexer into account. Moreover, it incurs additional reassembly delay for each burst. As such, exact emulation for multiplexers with variable length bursts cannot be achieved. Only a time shifted version can be achieved. The latter is the approach we use in this paper. Its design objective is to schedule cells in a careful manner so that cells of the same burst depart *contiguously* from the multiplexer for fixed size cells. As such, we propose adding a *cell scheduling block* in front of the multiplexer for fixed size cells. For such an architecture, we show there is an efficient cell scheduling algorithm. Starting from an empty system, we can perform cell scheduling by keeping track of a single state variable, called the total virtual waiting time in this paper. Moreover, the delay through the cell scheduling block is bounded above by a constant that only depends on the number of inputs and the maximum number of cells in a burst. Such a delay bound provides a limit on the number of fiber delay lines needed in the cell scheduling block.

This paper is organized as follows. In Section II, we define the network elements used in this paper, including fiber delay lines, switches, and multiplexers for fixed size cells. In Section III, we address the cell contiguity problem if variable length bursts are put directly into the multiplexer for fixed size cells. We argue that a cell scheduling block is needed to overcome the cell contiguity problem. In Section IV, we propose our architecture for multiplexers with variable length bursts by adding a cell scheduling block in front of the multiplexer for fixed size cells. We also describe the cell scheduling algorithm associated with the architecture. In Section V, we show that the delay through the cell scheduling block is bounded by a constant that only depends on the number of inputs and the maximum number of cells in a burst. As such, our architecture has limited complexity. We conclude the paper in Section VI by addressing possible future research topics.

II. Basic network elements and optical multiplexers for fixed size cells

In this section, we introduce the network elements that will be used in this paper, including fiber delay lines, switches, and multiplexers for fixed size cells. In this paper, we assume that propagation delay is well compensated so that time is synchronized and slotted. By so doing, a (fixed size) cell can be transmitted within a time slot. Since there is at most one cell 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 cell in the link at time t, and it is in state 0 at time t otherwise.

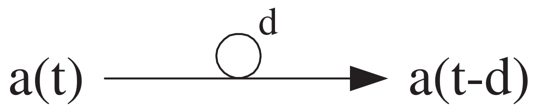

Fig. 1. An optical delay line with delay d

Definition 1 (Delay line) *An (optical) delay line in Figure 1 is a network element that has one input link and one output link. In Figure 1, 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 the construction. Note that at the end of the $t - 1^{th}$ time slot, the cells that arrive at time $t-1, t-2, \ldots, t-d$, are stored in the optical delay line with delay d.

Definition 2 (Switch) *An $N \times M$ (optical) switch has N input links and M output links. Let $a_i(t)$, $i = 0, 1, \ldots, N-1$, be the states of the N inputs at time t and $b_j(t)$, $j = 0, 1, \ldots, M-1$, be the states of the M outputs at time t. Then at any time t, one can specify an $M \times N$ sub-permutation matrix $P(t)$ such that $b(t) = P(t)a(t)$, where $b(t)$ (resp. $a(t)$) is the column vector with elements $b_j(t)$, $t = 0, 1, \ldots, M-1$ (resp. $a_i(t)$, $i = 0, 1, \ldots, N-1$).*

For example, a 2×2 switch is known to have two connection patterns. A 2×2 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 $b_0(t) = a_1(t)$.

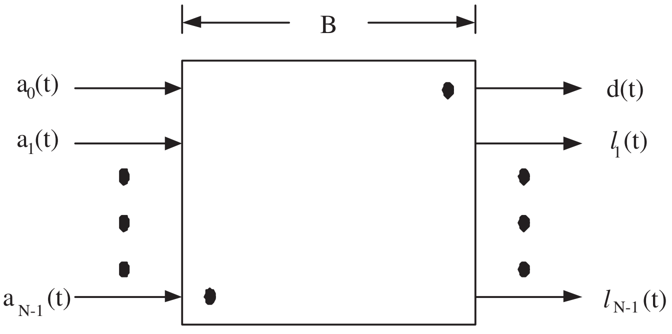

Fig. 2. An N-to-1 multiplexer with buffer B

Definition 3 (Multiplexer for fixed size cells) *An N-to-1 multiplexer with buffer B (see Figure 2) 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 2, 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 cells 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 cells from the N input links are either stored in the buffer or transmitted through the N output links, i.e.,*

$$q(t) = q(t-1) + \sum_{i=0}^{N-1} a_i(t) - d(t) - \sum_{i=0}^{N-1} \ell_i(t) \quad (1)$$

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

$$d(t) = \begin{cases} 0 & \text{if } q(t-1) + \sum_{i=0}^{N-1} a_i(t) = 0 \\ 1 & \text{otherwise} \end{cases} . \quad (2)$$

(P3) *Maximum buffer usage: arriving cells are lost only when buffer is full, i.e., for $i = 1, 2, 3, \ldots, N-1$,*

$$d(t) = \begin{cases} 1 & \text{if } q(t-1) + \sum_{i=0}^{N-1} a_i(t) \geq B + i + 1 \\ 0 & otherwise \end{cases} \quad (3)$$

(P4) *FIFO with prioritized inputs: cells 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 cells at the same time, the cell from the largest input link number is put in the multiplexer first.*

In the queueing context, a multiplexer defined in Definition 3 is simply a FIFO queue with buffer B. Specifically, the $q(t)$ process of an N-to-1 multiplexer satisfies the following recursive equation:

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

where $a(t) = \sum_{i=0}^{N-1} a_i(t)$ is the total number of arrivals at time t, and $x^+ = \max(0, x)$. Moreover, the departure process $d(t)$ from the multiplexer is exactly the same as that from the corresponding FIFO queue with buffer B. In addition to this, the multiplexing order of the N inputs is of particular importance to the later development of this paper. From (P4), we know that *cells depart in the order of their arrival times* and that *cells depart in the descending order of their input link numbers if they arrive (and enter) at the same time.*

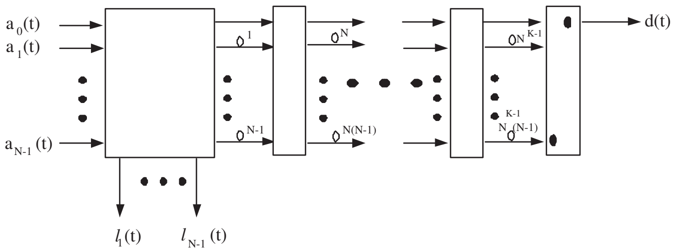

Fig. 3. A self-routing N-to-1 multiplexer with $B = N^k - 1$

As described in Section I, there are several ways proposed in the literature that use switches and fiber delay lines to build multiplexers for fixed size cells in Definition 3. One way to do it is the self-routing multiplexer in [3]. In Figure 3, we show the architecture of the self-routing multiplexer with buffer $B = N^k - 1$ in [3]. In such an architecture, 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). The first stage requires an $N \times (2N-1)$ switch as additional output links are used for dropping cells due to buffer overflow. The buffer size of such a multiplexer is $N^k - 1$. As in [10], [12], such an architecture also uses the output buffer emulation. It keeps track of the number of cells stored in the system. If such a number exceeds $N^k - 1$, further arrivals are dropped immediately. Specifically, let $q(t)$ be the number of cells stored in the system. Then $q(t)$ is governed by

$$q(t) = \min\left[\max[0, q(t-1) + \sum_{i=0}^{N-1} a_i(t) - 1], N^k - 1\right], \quad (5)$$

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 cells stored in the system when a particular cell enters the system. In queueing theory, the number q is known as the virtual delay (or the virtual waiting time) of the cell. Since $0 \leq q \leq N^k - 1$, 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 cell 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 cell destines for any output link in any switch at any time.

There is a natural analogy between the self-routing multiplexer in [3] and the classical Batcher-Banyan self-routing network (see e.g., Schwartz [18] and Hui [13]). One may view the virtual delay (or the virtual waiting time) in the self-routing multiplexer as the "output address" in the Batcher-Banyan self-routing network. By routing packets to different "output addresses," one then resolves conflicts at the multiplexer. Such a concept will also be used in our extension to the multiplexers with variable length bursts.

III. Cell contiguity problem

Now we would like to extend the multiplexer for fixed size cells to cope with variable length bursts. In this paper, we assume that burst segmentation is feasible. Each variable length burst can be divided into a contiguous sequence of fixed size *cells*, and each cell can then be transmitted within a time slot in the multiplexer for fixed size cells. Note that there is a granularity problem in doing burst segmentation (see e.g., [19], [1]). Such a problem will not be addressed here. Our objective is to use SDL units to design a multiplexer that achieves exact emulation of a FIFO finite buffer queue with variable length bursts, i.e., the departure process from the multiplexer is the same as that from a FIFO finite buffer queue with variable length bursts. For this, there are two things we need to do. The first is to schedule these bursts under the FIFO policy. The second is to maintain the contiguity of cells in a burst at the output link. The first thing is rather easy to do. However, maintaining the cell contiguity becomes a problem as shown in the following example in Figure 4.

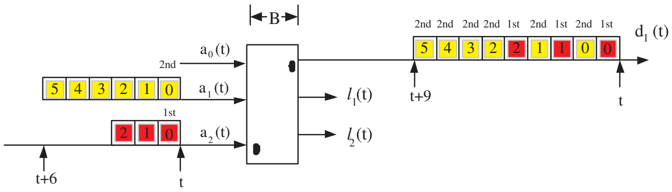

Fig. 4. An illustrating for the contiguity problem

In Figure 4, we show that if we put variable length bursts directly into the multiplexer (for fixed length cells) in Definition 3, there will be the cell contiguity problem. In this figure, we consider a multiplexer with 3 inputs and 3 outputs. Assume that the buffer at time t in the multiplexer is empty, i.e., $q(t) = 0$. There are two bursts that arrive at time t. The cells in a burst are indexed by consecutive integers from zero. We call the burst with length 3 the first burst and the other one the second burst. Assume that the buffer B is so large that no cells are lost (a trivial condition is $B \geq 8$ in this case). According to Definition 3, the order of multiplexing is in the descending order of the input link number for cells that arrive at the same time. Thus, the cell order in the departure port is cell 0 of the first burst, cell 0 of the second burst, cell 1 of the first burst, cell 1 of the second burst, ..., cell 5 of the second burst at last. Clearly, cells in the same bursts do not depart contiguously. As a result, we cannot naively put variable length bursts directly into multiplexers for fixed size cells. To solve the cell contiguity problem, cells need to be scheduled in a careful manner as illustrated in Figure 5.

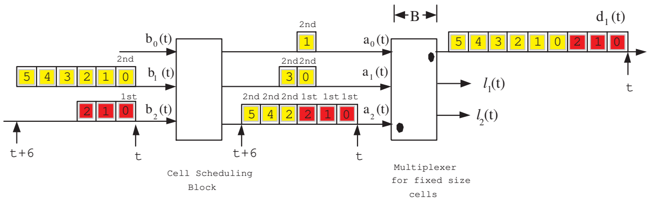

Fig. 5. Basic idea for cell scheduling

In Figure 5, we add a cell scheduling block in front of the multiplexer for fixed size cells. The function of the cell scheduling block is to route each cell to the *right input* of the multiplexer at the *right time* so that we can receive cells from the same burst contiguously. We illustrate this by considering the same traffic in Figure 4. At time t, we can schedule cell 0 of the first burst at input 2 of the multiplexer (for fixed size cells). In order for the cells in the first burst to come out contiguously from the multiplexer, we can not schedule anything at input 1 and input 0 of the multiplexer at time t. At time $t+1$, we then schedule cell 1 of the first burst at input 2. Similarly, we do not schedule anything at input 1 and input 0 at time $t+1$. At time $t+2$, we schedule cell 2 of the first burst at input 2. Clearly, the cells in the first burst come out contiguously this way. Now we can schedule cell 0 of the second burst at input 1 and cell 1 of the second burst at input 0 at time $t+2$. Since the multiplexing order is in the descending order of the input link number for cells that arrive at the same time, cell 0 of the second burst will come out from the multiplexer after cell 2 of the first burst. Similarly, cell 1 of the second burst will be out after cell 0 of the second burst. At time $t+3$, we schedule cell 2 of the second burst at input 2 and cell 3 of the second burst at input 1, respectively. Note that we can not schedule cell 4 of the second burst at input 0 at time $t+3$ as it has not arrived yet at time $t+3$. As such, both cell 4 and cell 5 of the second burst have to be scheduled respectively at time $t+4$ and at time $t+5$ at input 2.

IV. The proposed architecture

A. *The overall multiplexer architecture*

As addressed in the previous section, we need to add a cell scheduling block in order to overcome the cell contiguity problem. In Figure 6, we show our architecture for a variable length burst multiplexer with N inputs. It consists of two blocks. The latter is the N-to-1 multiplexer for fixed size cells described in Section II. The former is the cell scheduling block. The function of the cell scheduling is to route each cell to the right input at the right time so that cells of the same burst come out contiguously. To achieve this, the cell scheduling block consists of two stages. As shown in Figure 6, there are N $1 \times N$ switches at the first stage. The objective of these switches is to route cells to the *right inputs* of the multiplexer for fixed size cells. At the second stage, there are N $N \times M$ switches. The M outputs of each switch are connected to fiber delay lines with

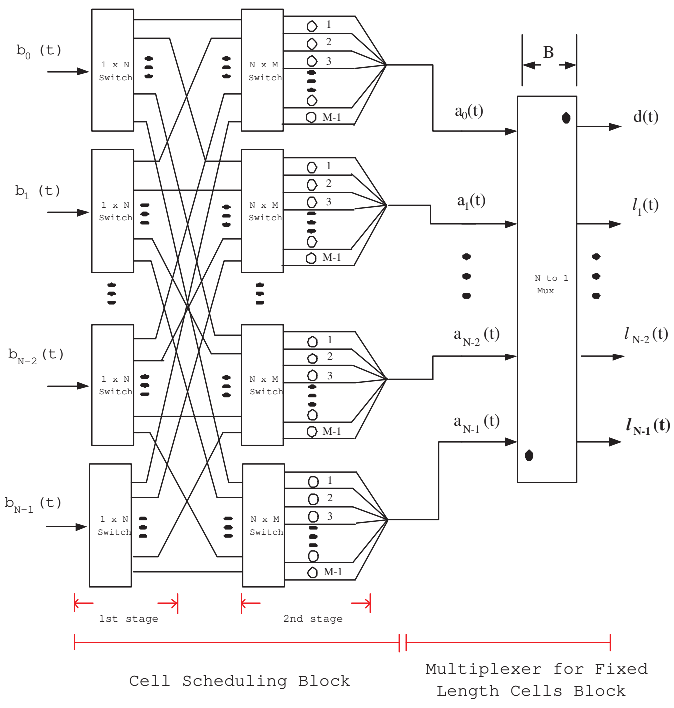

Fig. 6. architecture

delay from 0 to $M-1$. The objective of the second stage is to delay cells so that they arrive at the routed input at the *right time*. The constant $M-1$ is the maximum delay for a cell to go through the cell scheduling block. One key result of this paper is that the delay for a cell in the cell scheduling block never exceeds $\left\lfloor \frac{(2N-2)\ell_{\max}-N+1}{N} \right\rfloor$, where $\ell_{\max}$ is the maximum number of cells in a burst. Thus, we can choose M so that $M \geq \left\lfloor \frac{(2N-2)\ell_{\max}-N+1}{N} \right\rfloor + 1$. One easy choice to satisfy this requirement is $M = 2\ell_{\max}$.

B. The cell scheduling algorithm

In order for the variable length burst multiplexer with N inputs to work properly, we assume that the burst length of a burst is known when the first cell of a burst arrives. This can be done either by adding the burst length information in the header of the first cell or by transmitting such information through a different channel in advance (see e.g., [23], [26]). The arrival time of the first cell in a burst is called the arrival time of that burst. Bursts that arrive at the same time are scheduled in the descending order of their input link numbers (as in the multiplexer for fixed size cells).

Now we describe our cell scheduling algorithm. Note that there are three natural constraints of the cell scheduling algorithm.

(i) Conflict constraint: no more than one cell can be scheduled at the same input (of the multiplexer for fixed size cells) at the same time.

(ii) Causality constraint: no cell can be scheduled before its arrival.

(iii) Contiguity constraint: cells from the same burst should be scheduled so that they leave the multiplexer for fixed length cells block contiguously.

To satisfy the conflict constraint, we have to keep track of the time slots used in every input. For this, we let $V_j(t)$, $j = 0, 1, \cdots, N-1$, be the number of time slots that cannot be scheduled at input j from t onward. In other words, the next available time slot for input j is at time $t + V_j(t)$. Following the queueing context, we call $V_j(t)$ the virtual waiting time of input j (as a cell that is routed to input j at time t will have to wait $V_j(t)$ time slots). As we will show later, under our cell scheduling algorithm (described below) the virtual waiting times satisfy the following inequalities for all t:

$$\begin{aligned} V_{N-1}(t) &\geq V_{N-2}(t) \geq V_{N-3}(t) \cdots \geq \\ V_1(t) &\geq V_0(t) \geq V_{N-1}(t) - 1 \end{aligned} \tag{6}$$

Initially, we set $V_j(0) = 0$ for all $j = 0, 1, \cdots, N-1$, so that the inequalities in 6 are satisfied. Moreover, if there is no burst arrival at time t, then the next available time slot for input j is still at time $t + V_j(t)$. Thus, we have

$$V_j(t+1) = (V_j(t) - 1)^+, \quad j = 0, 1, \ldots, N-1. \tag{7}$$

In this case, one can also easily verify that $V_j(t+1)$'s satisfy the inequalities in 6.

Let $V(t)$ be the total virtual waiting time at time t, i.e.,

$$V(t) = \sum_{j=0}^{N-1} V_j(t). \tag{8}$$

Using the inequalities in 6, one can relate the total virtual waiting time to the virtual waiting time of input j as follows:

$$V_j(t) = \begin{cases} \left\lfloor \frac{V(t)}{N} \right\rfloor + 1 & \text{for } j = N-1, N-2, \\ & \quad \cdots, N-k \\ \left\lfloor \frac{V(t)}{N} \right\rfloor & \text{for } j = 0, 1, \cdots, N-k-1 \end{cases} \tag{9}$$

where $k = V(t) \mod N$. Thus, the total virtual waiting time $V(t)$ is sufficient for the purpose of cell scheduling.

Now suppose that the $m+1^{th}$ burst arrives at time τ_m with length ℓ_m, $m = 0, 1, \ldots$. Let $V(\tau_m^-)$ be the total virtual waiting time immediately before the arrival of the first cell (cell 0) of that burst. As the multiplexing order of the multiplexer (for fixed size cells) is in the descending order of the input link number and in the ascending order of time, cell 0 should be routed to the input with the smallest virtual waiting time and the largest input link number. From 9, we know it should be routed to input $N - k_0 - 1$ with $k_0 = V(\tau_m^-) \mod N$. Moreover, the delay for cell 0 is simply the virtual waiting time of input $N-k_0-1$, i.e., $\left\lfloor \frac{V(\tau_m^-)}{N} \right\rfloor$. By so doing, the inequalities in 6 are satisfied after cell 0 is scheduled. As the total virtual waiting time is increased by 1 after cell 0 is scheduled, cell 1 should be scheduled at input $N-k_1-1$ with $k_1 = (V(\tau_m^-)+1) \mod N$. The first available time slot at input $N-k_1-1$ is $\tau_m + \left\lfloor \frac{V(\tau_m^-)+1}{N} \right\rfloor$. As cell 1 arrives at time τ_m+1, the delay for cell 1 would be $\left\lfloor \frac{V(\tau_m^-)+1}{N} \right\rfloor - 1$ if cell 1 is scheduled at input $N-k_1-1$. If $\left\lfloor \frac{V(\tau_m^-)+1}{N} \right\rfloor - 1 \geq 0$, then the causality constraint is satisfied and cell 1 can be scheduled this way. We may continue the process to schedule the other cells in the burst until the causality constraint is violated. In general, cell ℓ should be scheduled at input $N-k_\ell-1$ with $k_\ell = (V(\tau_m^-)+\ell) \mod N$ (as the total virtual waiting time has been increased by ℓ after scheduling the first ℓ cells). As cell ℓ arrives at time $\tau_m+\ell$, the delay for cell ℓ would be $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell$ if cell ℓ is scheduled at input $N-k_\ell-1$. If $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell \geq 0$, then the causality constraint is satisfied and cell ℓ can be scheduled this way. If $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell < 0$, then the causality constraint is violated. We have to schedule cell ℓ at its arrival time. In order to satisfy the contiguity constraint, cell ℓ is scheduled at input $N-1$, the highest priority input at its arrival time. It is easy to see that if cell ℓ is the first cell such that $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell < 0$, then all the subsequent cells in the same burst also satisfy the same inequality. As such, all the subsequent cells have to be scheduled at their arrival times at input $N-1$. To illustrate this, we show in Figure 7 how a burst of length 8 is scheduled. Cells 0,1,2,3,4 and 5 can be scheduled without violating the causality constraint. Cells 6 and 7 have to be scheduled at their arrival times.

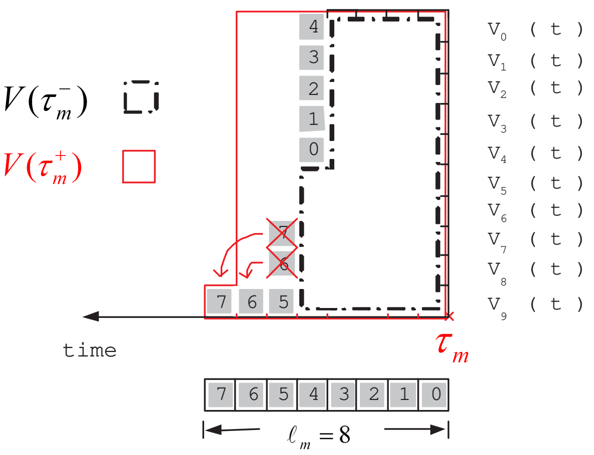

Fig. 7. An illustrating example of the cell scheduling algorithm

To summarize, we let

$$I_0(m,\ell) = N-1-((V(\tau_m^-)+\ell) \mod N). \quad (10)$$

Cell ℓ of the $m+1^{th}$ burst is routed to the $I_0(m,\ell)^{th}$ input of the multiplexer for fixed size cells if $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor \geq \ell$. In this case, its delay is $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell$. On the other hand, if $\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor < \ell$, cell ℓ of the $m+1^{th}$ burst is routed to input $N-1$ of the multiplexer for fixed size cells. In this case, its delay is zero. This is formalized in the following algorithm.

Algorithm 4 (Cell scheduling algorithm) *Let $I(m,\ell)$ and $D(m,\ell)$ be the routed input (of the multiplexer for fixed size cells) and the delay of cell ℓ of the $m+1^{th}$ burst, $\ell = 0,1,\ldots,\ell_m-1$, and $m = 0,1,2,\ldots$. Then*

$$I(m,\ell) = \begin{cases} I_0(m,\ell) & \text{if } \left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor \geq \ell \\ N-1 & \text{otherwise} \end{cases}, \quad (11)$$

and

$$D(m,\ell) = (\left\lfloor \frac{V(\tau_m^-)+\ell}{N} \right\rfloor - \ell)^+. \quad (12)$$

Now we discuss how we update the total virtual waiting time. Let $V(\tau_m^+)$ be the total virtual waiting time immediately after the cells in the $m+1^{th}$ burst are scheduled. As described in our cell scheduling algorithm, there are two cases that need to be considered. The first case is that all the cells in that burst are scheduled using the rule specified by $I_0(m,\ell)$. In this case, after the last cell in the burst, i.e., cell ℓ_m-1, is scheduled, we have

$$V(\tau_m^+) = V(\tau_m^-) + \ell_m, \quad (13)$$

and all the inequalities in 6 are still satisfied. The second case is that there exists a cell that does not follow the rule specified by $I_0(m,\ell)$. When this happens, cell ℓ_m-1 is scheduled at input $N-1$ at time $\tau_m+\ell_m-1$. In order to satisfy the contiguity constraint, no cells (from other bursts) can be scheduled before $\tau_m+\ell_m-1$. Thus, the first available time slot for input j, $j = 0,1,\ldots,N-2$, is $\tau_m+\ell_m-1$ and the first available time slot for input $N-1$ is $\tau_m+\ell_m$ (see Figure 7 for an illustrating example). Clearly, the inequalities in 6 are still satisfied and we have

$$V(\tau_m^+) = (N-1)(\ell_m-1) + \ell_m = N\ell_m - N + 1. \quad (14)$$

These two cases can be combined as follows:

$$V(\tau_m^+) = \max[V(\tau_m^-)+\ell_m, N\ell_m - N + 1]. \quad (15)$$

To see this, note that the condition for the first case is equivalent to that cell ℓ_m-1 is routed to the $I_0(m,\ell_m-1)^{th}$ input, i.e.,

$$\left\lfloor \frac{V(\tau_m^-)+\ell_m-1}{N} \right\rfloor \geq \ell_m - 1. \quad (16)$$

As ℓ_m-1 is an integer, this is equivalent to

$$\frac{V(\tau_m^-)+\ell_m-1}{N} \geq \ell_m - 1. \quad (17)$$

Thus, in the first case, we have from (13) that

$$V(\tau_m^+) = V(\tau_m^-) + \ell_m = \max[V(\tau_m^-) + \ell_m, N\ell_m - N + 1]. \quad (18)$$

On the other hand, the inequality in (17) is reversed in the second case and we have from (14) that

$$V(\tau_m^+) = N\ell_m - N + 1 = \max[V(\tau_m^-) + \ell_m, N\ell_m - N + 1]. \quad (19)$$

Now we describe how we update the total virtual waiting time between two successive bursts. Suppose that the $m+2^{th}$ burst arrives at τ_{m+1}. There are two cases that need to be considered:

Case 1. $V(\tau_m^+) - N(\tau_{m+1} - \tau_m) > 0$: in this case, we have from 9 that $V_{N-1}(\tau_m^+) > \tau_{m+1} - \tau_m$. Note from 6 that $V_j(\tau_m^+) \geq V_{N-1}(\tau_m^+) - 1$ for all j. Thus, we have $\tau_m + V_j(\tau_m^+) - \tau_{m+1} \geq 0$ for all $j = 0, 1 \ldots, N-1$. Since the next available time slot of input j is $\tau_m + V_j(\tau_m^+)$, we then have

$$V_j(\tau_{m+1}^-) = \tau_m + V_j(\tau_m^+) - \tau_{m+1}.$$

Summing up over j yields

$$V(\tau_{m+1}^-) = V(\tau_m^+) - N(\tau_{m+1} - \tau_m).$$

Case 2. $V(\tau_m^+) - N(\tau_{m+1} - \tau_m) \leq 0$: in this case, we have from 9 that $V_{N-1}(\tau_m^+) \leq \tau_{m+1} - \tau_m$. Note from 6 that $V_j(\tau_m^+) \leq V_{N-1}(\tau_m^+)$ for all j. Thus, we have $\tau_m + V_j(\tau_m^+) - \tau_{m+1} \leq 0$ for all $j = 0, 1 \ldots, N-1$. As the next available time slot of input j is $\tau_m + V_j(\tau_m^+) \leq \tau_{m+1}$, we then have $V_j(\tau_{m+1}^-) = 0$ for all $j = 0, 1 \ldots, N-1$. Thus, $V(\tau_{m+1}^-) = 0$.

From these two cases, we then have $V(\tau_{m+1}^-) = (V(\tau_m^+) - N(\tau_{m+1} - \tau_m))^+$. In the following, we summarize the algorithms for updating the total virtual waiting time.

Algorithm 5 (Algorithms for updating the total virtual waiting time)

(i) *The total virtual waiting time after a burst is scheduled is updated as follows:*

$$V(\tau_m^+) = \max[V(\tau_m^-) + \ell_m, N\ell_m - N + 1]. \quad (20)$$

(ii) *The total virtual waiting time between two successive bursts is updated as follows:*

$$V(\tau_{m+1}^-) = (V(\tau_m^+) - N(\tau_{m+1} - \tau_m))^+. \quad (21)$$

We illustrate how we use the cell scheduling algorithm in the following example.

Example 6 In Figure 8, we consider multiplexing variable length bursts over 3 links, i.e., $N = 3$. As shown in Figure 8, there are two bursts coming at time t_0 and four bursts coming respectively at time $t_0+1, t_0+4, t_0+9, t_0+10$. To break the tie, we choose the burst with length 5 at time t_0 to be the first burst. As shown in the figure, now we have $(\tau_0, \ell_0) = (t_0, 5)$,

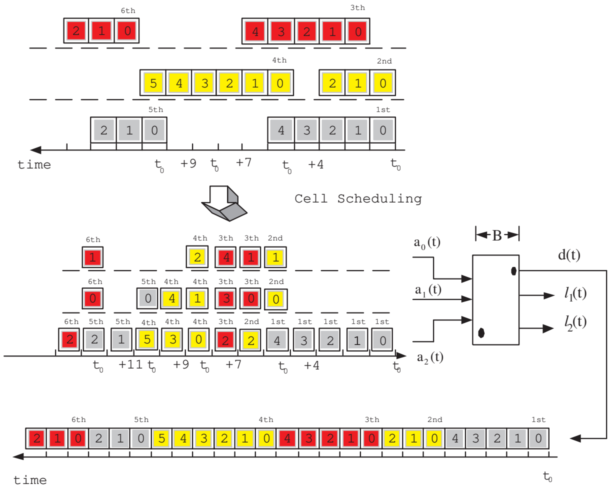

Fig. 8. An illustrating example for N=3

$(\tau_1, \ell_1) = (t_0, 3)$, $(\tau_2, \ell_2) = (t_0+1, 5)$, $(\tau_3, \ell_3) = (t_0+4, 6)$, $(\tau_4, \ell_4) = (t_0+9, 3)$, and $(\tau_5, \ell_5) = (t_0+10, 3)$. Assume that the buffer in the multiplexer is empty at t_0 and the buffer size B is so large that no cells of the six bursts are lost. Initially, set $V_0(t_0) = V_1(t_0) = V_2(t_0) = 0$. Thus, $V(\tau_0^-) = V(t_0) = 0$ and the cells in the first burst are all scheduled at their arrival times at input 2. Using the algorithms for updating the total virtual waiting time, we then have $V(\tau_0^+) = 13$ and $V(\tau_1^-) = V(\tau_0^+) = 13$. According to the cell scheduling algorithm, cell 0 of the second burst is scheduled at input 1 at t_0+4, cell 1 of the second burst is scheduled at input 0 at t_0+4, and cell 2 of the second burst is scheduled at input 2 at t_0+5. Thus, $V(\tau_1^+) = 16$ and $V(\tau_2^-) = 13$. All the cells in the third burst are scheduled using the rule by $I_0(m, \ell)$. As a result, $V(\tau_2^+) = 18$ and $V(\tau_3^-) = 9$. Note that the last cell (cell 5) of the fourth burst does not follow the $I_0(m, \ell)$ rule and it is scheduled at input 2 at its arrival time. Thus, $V(\tau_3^+) = 16$ and $V(\tau_4^-) = 1$. Cell 0 and cell 1 of the fifth burst still follow the $I_0(m, \ell)$ rule. However, cell 2 of the fifth burst does not follow the same rule and it is scheduled at its arrival time at input 2. Thus, $V(\tau_4^+) = 7$ and $V(\tau_5^-) = 4$. The cells in the last burst all follow the $I_0(m, \ell)$ rule.

V. Delay Bound

We have introduced the architecture and the associated cell scheduling algorithm for the variable length burst multiplexer with N inputs. In this section, we will further show that both the total virtual waiting time and the delay through the cell scheduling block are bounded by constants that only depend on the number of inputs and the maximum number of cells in a burst.

Theorem 7 *Let $\ell_{max} = \sup_{m \geq 0} \ell_m$ be the maximum number of cells in a burst.*

(i) *The total virtual waiting time is bounded by* $(2N-1)\ell_{\max}-N+1$ *for all t, , i.e.,*

$$V(t) \leq (2N-1)\ell_{\max}-N+1. \tag{22}$$

(ii) *The delay for every cell through the cell scheduling block is bounded by* $\left\lfloor \frac{(2N-2)\ell_{\max}-N+1}{N} \right\rfloor$, *i.e., for all* $m=0,1,2,\ldots$, $\ell=0,1,\ldots,\ell_m-1$,

$$D(m,\ell) \leq \left\lfloor \frac{(2N-2)\ell_{\max}-N+1}{N} \right\rfloor. \tag{23}$$

Note that the bounds in Theorem 7 can be achieved in the worst case. We now construct a worst case to achieve these bounds. To see this, consider the scenario that there are N bursts arriving at time 0 at the N inputs of the variable length burst multiplexer. The burst lengths of these N bursts are all ℓ_{max}. Thus, we have $\tau_m=0$, $\ell_m=\ell_{max}$, for $m=0,1,\ldots,N-1$. Under our cell scheduling algorithm, we then have

$$V(\tau_0^-)=V(0)=0,$$

$$V(\tau_m^+)=V(\tau_{m+1}^-)=N(\ell_{\max}-1)+1+m\ell_{\max},$$

$$m=0,1,\ldots,N-2,$$

and

$$V(\tau_{N-1}^+)=N(\ell_{\max}-1)+1+(N-1)\ell_{\max}.$$

Thus, $V(\tau_{N-1}^+)$ in this scenario achieves the upper bound in (22). Moreover, the delay of cell 0 of the N^{th} burst is $\lfloor \frac{V(\tau_{N-1}^-)}{N} \rfloor$, which is exactly the maximum delay in (23).

One important consequence of Theorem 7 is that the number of the delay lines, M, used in each switch at the second stage of the cell scheduling block is bounded by $\left\lfloor \frac{(2N-2)\ell_{\max}-N+1}{N} \right\rfloor + 1$. A simple choice is $M=2\ell_{\max}$. Such a choice is independent of the number of inputs N.

The rest of this section is devoted to the proof of Theorem 7. We will need Lemma 8 and Lemma 9 below for the proof of Theorem 7. In Lemma 8, we first expand the recursive equations in (20) and (21) to derive closed form expressions of $V(\tau_m^+)$ and $V(\tau_m^-)$.

Lemma 8 *Let* $L(n)=\sum_{m=0}^{n-1}\ell_m$ *be the total number of cells in the first n bursts. Suppose* $V(\tau_0^-)=0$.

Then $V(\tau_m^+)=$

$$\max_{0\leq n\leq m}\big(\, N\ell_n-N+1-N(\tau_m-\tau_n)+L(m+1)-L(n+1)\,\big), \tag{24}$$

and $V(\tau_m^-)=$

$$\max_{0\leq n\leq m-1}((N-1)\ell_n-N+1-N(\tau_m-\tau_n)+L(m)-L(n))^+. \tag{25}$$

Proof. We prove (24) by induction. Since we assume that $V(\tau_0^-)=0$, we have from (20) that

$$\begin{aligned} V(\tau_0^+) &= \max(\,V(\tau_0^-)+\ell_0, N\ell_0-N+1) \\ &= N\ell_0-N+1 \end{aligned}$$

and (24) is satisfied trivially for $m=0$.

Now suppose it holds for some $m\geq 0$ as the induction hypothesis. It follows from (20) and the induction hypothesis that

$$\begin{aligned} V(\tau_{m+1}^+) &= \max(N\ell_{m+1}-N+1, \\ &\quad (V(\tau_m^+)-N(\tau_{m+1}-\tau_m))^+ + \ell_{m+1}) \\ &= \max(N\ell_{m+1}-N+1, \\ &\quad V(\tau_m^+)-N(\tau_{m+1}-\tau_m)+\ell_{m+1}, \ell_{m+1}) \\ &= \max(N\ell_{m+1}-N+1, \\ &\quad \max_{0\leq n\leq m}(N\ell_n-N+1-N(\tau_m-\tau_n)+ \\ &\quad L(m+1)-L(n+1)-N(\tau_{m+1}-\tau_m) \\ &\quad +\ell_{m+1})) \\ &= \max(N\ell_{m+1}-N+1, \\ &\quad \max_{0\leq n\leq m}(N\ell_n-N+1-N(\tau_{m+1}-\tau_n) \\ &\quad +L(m+1)-L(n+1)+\ell_{m+1})) \\ &= \max_{0\leq n\leq m+1}(N\ell_n-N+1-N(\tau_{m+1}-\tau_n) \\ &\quad +L(m+2)-L(n+1)). \end{aligned}$$

This completes the inductive argument for (24).

Since $V(\tau_m^-)=(\,V(\tau_{m-1}^+)-N(\tau_m-\tau_{m-1})\,)^+$ in (21), using (24) yields

$$\begin{aligned} V(\tau_m^-) &= (\max_{0\leq n\leq m-1}(N\ell_n-N+1-N(\tau_{m-1}-\tau_n) \\ &\quad +L(m)-L(n+1)\,)-N(\tau_m-\tau_{m-1})\,)^+ \\ &= \max_{0\leq n\leq m-1}(N\ell_n-N+1-N(\tau_m-\tau_n) \\ &\quad +L(m)-L(n+1))^+ \\ &= \max_{0\leq n\leq m-1}((N-1)\ell_n-N+1 \\ &\quad -N(\tau_m-\tau_n)+L(m)-(L(n+1)-\ell_n))^+ \\ &= \max_{0\leq n\leq m-1}((N-1)\ell_n-N+1 \\ &\quad -N(\tau_m-\tau_n)+L(m)-L(n))^+. \end{aligned}$$

■

In Lemma 9, we establish a bound for the multiplexed traffic of N inputs. This bound will be used for the proof of Theorem 7.

Lemma 9 *For all* $n\leq m$,

$$L(m)-L(n)-N(\tau_m-\tau_n)\leq (N-1)\ell_{\max}. \tag{26}$$

Proof. Let $\tau_k(n)$ and $\ell_k(n)$, $k = 0, 1, \ldots, N-1$, $n = 0, 1, 2, \ldots$, be the arrival time and the burst length of the $n+1^{th}$ burst at the k^{th} input of the multiplexer with variable length bursts. Also, let $L_k(n) = \sum_{m=0}^{n-1} \ell_k(m)$ be the total number of cells in the first n bursts from the k^{th} input.

Without loss of generality, suppose that in the first n (resp. m) bursts, there are n_k (resp. m_k) bursts from the k^{th} input, $k = 0, 1, \ldots, N-1$. Moreover, suppose that the $m+1^{th}$ burst is from the j^{th} input for some j. Thus, we have

$$L(m) - L(n) = \sum_{k=0}^{N-1} (L_k(m_k) - L_k(n_k)). \tag{27}$$

Note that $L_k(m_k) - L_k(n_k) = \sum_{i=n_k}^{m_k-1} \ell_k(i)$ is the total number of cells from the $n_k + 1^{th}$ burst to the m_k^{th} burst at input k. These cells must arrive during the time interval $[\tau_k(n_k), \tau_k(m_k - 1) + \ell_{\max} - 1]$. Since there is at most one cell arrival within a time slot, we then have

$$L_k(m_k) - L_k(n_k) \le \tau_k(m_k - 1) + \ell_{\max} - \tau_k(n_k). \tag{28}$$

As there are exactly n_k bursts from the k^{th} input in the first n bursts, the arrival time of the $n_k + 1^{th}$ burst from the k^{th} input cannot be earlier than the arrival time of the $n+1^{th}$ burst, i.e.,

$$\tau_k(n_k) \ge \tau_n. \tag{29}$$

Similarly, as there are exactly m_k bursts from the k^{th} input in the first m bursts, the arrival time of the m_k^{th} burst from the k^{th} input cannot be later than the arrival time of the $m+1^{th}$ burst, i.e.,

$$\tau_k(m_k - 1) \le \tau_m. \tag{30}$$

Using (29) and (30) in (28) yields

$$L_k(m_k) - L_k(n_k) \le \tau_m - \tau_n + \ell_{\max}. \tag{31}$$

Now we refine the bound in (31) for the j^{th} input. As the $m+1^{th}$ burst is from the j^{th} input, the cells in $L_j(m_j) - L_j(n_j)$ must arrive during the time interval $[\tau_j(n_j), \tau_m - 1]$. Thus,

$$L_j(m_j) - L_j(n_j) \le \tau_m - \tau_j(n_j) \le \tau_m - \tau_n. \tag{32}$$

It then follows from (27), (31) and (32) that

$$\begin{aligned} & L(m) - L(n) = L_j(m_j) - L_j(n_j) \\ & + \sum_{k \ne j} (L_k(m_k) - L_k(n_k)) \\ & \le N(\tau_m - \tau_n) + (N-1)\ell_{\max}. \end{aligned}$$

■

Proof. (Proof of Theorem 7) (i) Note that $V(t)$ is decreasing between the interarrival time of two successive bursts and that $V(\tau_m^-) \le V(\tau_m^+)$ for all m. Thus,

$$V(t) \le \sup_{m \ge 0} V(\tau_m^+).$$

It suffices to show that $V(\tau_m^+) \le (2N-1)\ell_{\max} - N + 1$ for all $m = 0, 1, 2, \ldots$. Note from (24) in Lemma 8 and (26) in Lemma 9 that

$$\begin{aligned} V(\tau_m^+) &= \max_{0 \le n \le m} (\, N\ell_n - N + 1 - N(\tau_m - \tau_n) \\ &\quad + L(m+1) - L(n+1) \,) \\ &= \max_{0 \le n \le m} \Big(\, (N-1)\ell_n - N + 1 - N(\tau_m - \tau_n) \\ &\quad + L(m) + \ell_m - (L(n+1) - \ell_n) \Big) \\ &\le \max_{0 \le n \le m} (\, (N-1)\ell_n - N + 1 - N(\tau_m - \tau_n) \\ &\quad + L(m) - L(n) + \ell_{max}) \\ &\le \max_{0 \le n \le m} (\, (N-1)\ell_n - N + 1 + N\ell_{\max}) \\ &\le (2N-1)\ell_{\max} - N + 1. \end{aligned}$$

(ii) Note from (25) in Lemma 8 and (26) in Lemma 9 that

$$\begin{aligned} V(\tau_m^-) &= \max_{0 \le n \le m-1} ((N-1)\ell_n - N + 1 - N(\tau_m - \tau_n) \\ &\quad + L(m) - L(n))^+ \\ &\le \max_{0 \le n \le m-1} ((N-1)\ell_n - N + 1 + (N-1)\ell_{\max}) \\ &\le (2N-2)\ell_{\max} - N + 1 \end{aligned}$$

Thus, we have from (12) that

$$\begin{aligned} D(m, \ell) &= (\left\lfloor \frac{V(\tau_m^-) + \ell}{N} \right\rfloor - \ell)^+ \\ &\le \left\lfloor \frac{V(\tau_m^-)}{N} \right\rfloor \\ &= \left\lfloor \frac{(2N-2)\ell_{\max} - N + 1}{N} \right\rfloor. \end{aligned}$$

■

VI. Conclusions

In this paper, we proposed an architecture that achieves exact emulation of FIFO multiplexers for variable length bursts. Our architecture consists of two blocks: a cell scheduling block and an FIFO multiplexer for fixed size cells. Both blocks are made of SDL units. The cell scheduling block schedules cells in a burst to the right input at the right time so that cells in the same burst depart *contiguously* from the multiplexer for fixed size cells. We showed that cell scheduling can be done efficiently by keeping track of the total virtual waiting time. Moreover, the delay through the cell scheduling block is bounded above by a constant that only depends on the number of inputs and the maximum number of cells in a burst. Such a delay bound provides a limit on the number of fiber delay lines needed in the cell scheduling block.

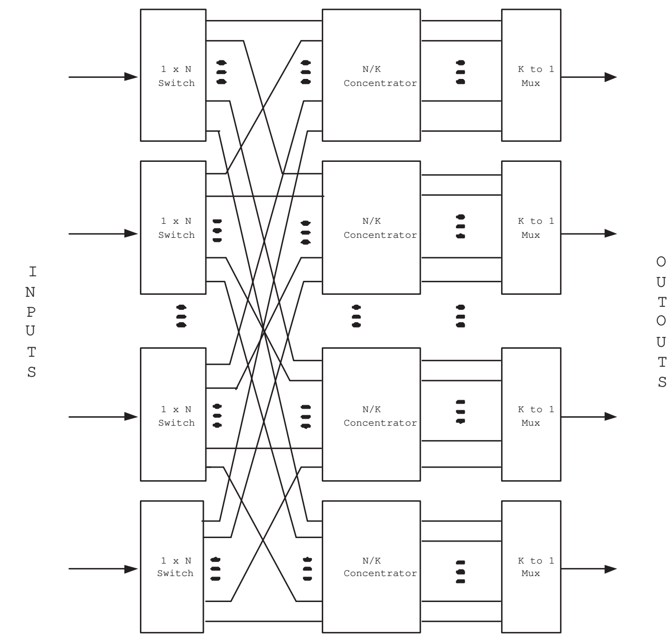

Fig. 9. An $N \times N$ Knockout switch

One may use the multiplexer developed in this paper to build all optical Knockout switches. In Figure 9, we show the architecture of an $N \times N$ Knockout switch in [21]. These are three basic network elements in Figure 9: $1 \times N$ switches, N inputs/K outputs concentrators and K-to-1 multiplexers. As shown in Fig. 8 of [21], it is possible to realize the N inputs/K outputs concentrators using SDL units. As we have shown in this paper that the K-to-1 multiplexers with variable length bursts can be realized by SDL units, Knockout switches (with variable length bursts) can also be realized by SDL units.

In this paper and our previous paper in [3], we only focused on designing FIFO multiplexers using SDL units. In order to achieve quality of service (QoS), more sophisticated scheduling policies, such as priority queues [15], [24], [17] and the earliest deadline first policy, might be needed. It would be of interest to investigate the complexity of implementing these sophisticated scheduling policies using SDL units.

REFERENCES

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

[2] C.S. Chang, D.S. Lee and C.M. Lien, “Load Balanced Birkhoff-von Neumann Switches, Part II: Multi-stage Buffering,” to appear in the special issue of *Computer Communications* on “Current Issues in Terabit Switching,” 2002.

[3] C.S. Chang, D.S. Lee and C.K. Tu, “Recursive construction of FIFO optical multiplexers with switched delay lines,” *preprint* 2002.

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

[5] 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.

[6] 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.

[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] Y.S. Yeh, M.G. Hluchyj, and A.S. Acampora, “The Knockout switch: a simple, modular architecture for high-performance packet switching,” *IEEE Journal of Selected Areas in Communications*, Vol. SAC-5, pp. 1274-1283, 1987.

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

[23] 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.

[24] 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.

[25] 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.

[26] E.A. Varvarigos and V. Sharma, “An efficient reservation connection control protocol for gigabit networks, ” *Computer Networks and ISDN Systems* ,vol. 30, (no. 12), 13 July 1998, pp. 1135-1156.