Time Sliced Optical Burst Switching

Jeyashankher Ramamirtham, Jonathan Turner,
{jai,jst}@arl.wustl.edu,
Computer Science and Engineering Department,
Washington University in St. Louis,
St. Louis, MO-63130

***Abstract*— Time Sliced Optical Burst Switching is a proposed variant of optical burst switching, in which switching is done in the time domain, rather than the wavelength domain. This eliminates the need for wavelength converters, the largest single cost component of systems that switch in the wavelength domain. We examine some of the key design issues for routers that implement time sliced optical packet switching. In particular, we focus on the design of the Optical Time Slot Interchangers (OTSI) needed to effect the required time domain switching. We introduce a novel nonblocking OTSI design and also show how blocking OTSIs can be used to implement the required switching operations. We study the performance of systems using blocking OTSIs and demonstrate that near ideal statistical multiplexing performance can be achieved using even quite inexpensive, blocking OTSI designs. These results suggest that optical technology may one day be able to provide a cost-effective alternative to electronics in packet switching systems.**

***Index Terms*—burst switching, optical networking, time-slot interchangers**

I. Introduction

Wavelength Division Multiplexing (WDM) has made it possible to harness the enormous bandwidth potential of fiber in a cost-effective way and is thus, becoming the method of choice for information transmission in data networks. Systems with hundreds of wavelengths per fiber and transmission rates of 10-40 Gbps per wavelength are becoming available, leading to a serious disparity between electronic switching speeds and optical transmission capacity. All-optical switching seeks to eliminate electronic switching and switch the data in its optical form, thus eliminating the opto-electronic components which contribute a large fraction of the cost of electronic routers. Optical switching has other potential benefits, including bit-rate independence, protocol transparency, and low power consumption.

Optical Burst Switching [1] is an experimental network technology that seeks to use optical switching for the data path, while still retaining the flexibility of electronics for control. To provide good statistical multiplexing performance, burst switching and other forms of optical packet switching [2], [3], [4] require either large optical buffers or optical wavelength converters. The only practical optical buffers today are optical delay lines, which are too expensive to use in the large quantities needed for optical IP routers (IP routers require buffer capacities that are comparable to the product of the link bandwidth and the network round trip delay). Burst switching seeks to eliminate most optical buffering by using wavelength converters to allow dynamic selection from a large number of wavelengths. Unfortunately, optical wavelength conversion remains expensive, and there are no realistic expectations that it will become inexpensive enough to allow optical burst switches to be cost-competitive with electronic routers.

Time Sliced Optical Burst Switching (TSOBS) is a proposed variant of optical burst switching that replaces switching in the wavelength domain with switching in the time domain. While time-domain switching does require the use of optical buffers, the amount of storage needed is less than 1% of that needed for conventional packet switching, greatly changing the cost tradeoffs. Like burst switching, TSOBS separates burst control information from burst data. Specifically, *Burst Header Cells* (BHC) are transmitted on separate control wavelengths on each WDM link. These wavelengths are converted to electronic form at each switch, while all remaining wavelengths are switched through in optical form. The data wavelengths carry information in a Time-Division Multiplexed (TDM) format, consisting of a repeating *frame* structure, which is sub-divided into *time slots* of constant length. A repeating sequence of time slots in successive frames, at a fixed position within the frame is referred to here, as a *channel*. Each BHC "announces" the imminent arrival of a data burst, and includes address information plus the wavelength and channel on which the burst is arriving. It also includes an *offset*, which identifies the frame in which the first timeslot containing data from the burst appears, and a *length*, which identifies the number of timeslots used to transmit the burst. The proposed combination of wavelength and time-division switching has been studied previously in a circuit-switching context [5], [6], [7], but we are not aware of any prior attempts to apply this approach in a packet or burst switching context.

Optical Time Slot Interchangers (OTSI) are key building blocks of routers in TSOBS networks. Three key factors that affect the cost and performance of an OTSI are (1) the size of its internal crossbar, (2) the amount of fiber required for the delay lines used to reorder the timeslots, and (3) the number of switching operations that bursts may be subjected to when passing through the OTSI. In this paper, we present an overall architecture for a TSOBS router and study how alternative OTSI designs affect its cost and performance. We consider both blocking and non-blocking OTSIs and study how different designs affect

This work is supported by the Advanced Research Projects Agency and Rome Laboratory (contract F30602-97-1-0273)

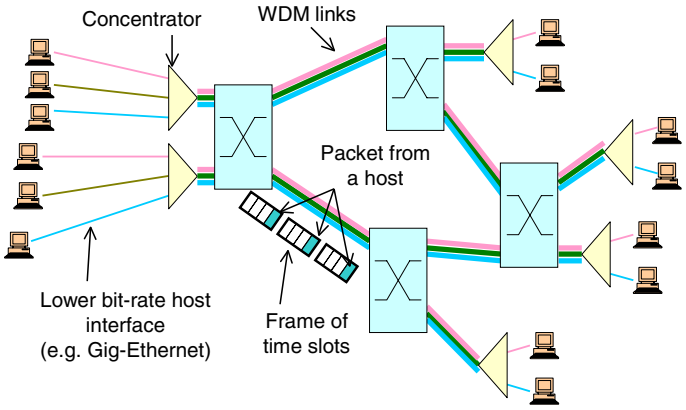


Fig. 1. Time-sliced packet switched network architecture

the system's overall statistical multiplexing performance, using simulation.

The rest of the paper is organized as follows. In Section II, we discuss the overall design issues for TSOBS networks. In Section III, we present an architecture of a TSOBS router and present alternate OTSI designs. We present simulation results for a system using a blocking OTSI design in Section IV. Finally, we present some concluding remarks in Section V.

II. Design Issues for TSOBS Networks

Fig. 1 illustrates the concept of a time-sliced optical burst switched network. Switches are connected with WDM links with multiple wavelength channels carrying data. The information sent on each wavelength is organized into a series of *frames*, each of which is sub-divided into fixed length *timeslots*. Terminals and/or other networks connect to a TSOBS network through concentrators that convert data on lower speed interfaces (e.g. IP-over-Ethernet at 100 Mb/s or 1 Gb/s), to the TSOBS data format. Concentrators transmit user data bursts in time-division channels. The control information needed to switch the data bursts is sent in *Burst Header Cells* (BHC), which are carried on separate control wavelengths. A given fiber optic link may contain multiple control wavelengths. If the ratio of the expected burst length to the BHC length is L, each link will require about one control wavelength for every $L-1$ data wavelengths. Concentrators may switch packets received on low speed interfaces as single bursts in the TSOBS network, or may aggregate packets to form larger bursts. Aggregation increases the average burst length on the TSOBS links, improving efficiency and reducing the amount of control processing required.

The switching of data bursts through a TSOBS network is done entirely in the optical domain. Space-division optical switches are dynamically configured to switch the data from incoming timeslots to timeslots on the appropriate outgoing links. This is done using carefully-timed switching operations to transfer user data bits from input links to output links. Switching a timeslot may involve delaying the data, to shift it from one timeslot position to another. Frames transmitted on different wavelengths are synchronized with one another, allowing the timing of switching operations on the data wavelengths to be determined from the frame timing on the control wavelengths.

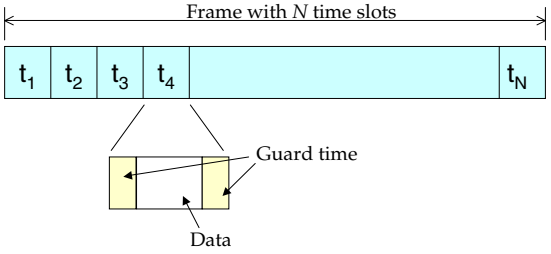


Fig. 2. Format of a frame and a time slot within it

Solid-state optical switches can perform switching operations with a precision of 10 ns or less. To allow for timing uncertainties, timeslots must be separated by a *guard time* of at least 10 ns and possibly as large as 100 ns. To achieve reasonable data transmission efficiencies, timeslot durations should be at least ten times the guard time, or 100 ns to 1 μs. A 1 μs time slot would allow roughly 1100 bytes of user data to be sent in a single time slot, assuming a transmission rate of 10 Gb/s per wavelength, or 4400 bytes of user data, assuming a transmission rate of 40 Gb/s. With a 1 μs timeslot duration and 40 Gb/s transmission speeds, a system with 350 timeslots per frame would support an individual channel rate of about 100 Mb/s. This would correspond to a frame duration of 350 μs. This is the maximum period that a timeslot would have to be delayed when passing through a TSOBS router. By contrast, a conventional router may have to delay data by hundreds of milliseconds in order to provide acceptable performance, requiring very large amounts of data storage. Of course, a shorter timeslot duration or a smaller number of timeslots per frame would allow this maximum delay for a TSOBS router to be reduced proportionally.

In a large TSOBS network, bursts may pass through many hops before reaching their destination, leading to excessive degradation of the optical signal [8]. To avoid the need for strict limits on the number of hops and/or the distance traveled by bursts, we provide for periodic regeneration of bursts. This is done by including fields in the BHC, which record the distance traveled by a burst since its last regeneration and the number of optical switching operations that the burst has been subjected to since its last regeneration. This information can be used to regenerate bursts *as necessary*, as they pass through a large network. Each switch includes some small number of either all optical or opto-electronic regeneration circuits that bursts can be passed through when regeneration is needed. If bursts can travel through an average of ten or more routers before requiring regeneration, TSOBS could achieve a decisive cost advantage over electronic routers. Since each switch operation that a burst is subjected to attenuates the signal and adds noise, it is important to minimize the number of switching operations required by each router. The number of switching operations is hence a key metric for TSOBS router designs.

TSOBS has been designed to make wavelength conversion unnecessary. At the same time, the overall architecture does not preclude the switching of bursts to different wavelengths, should

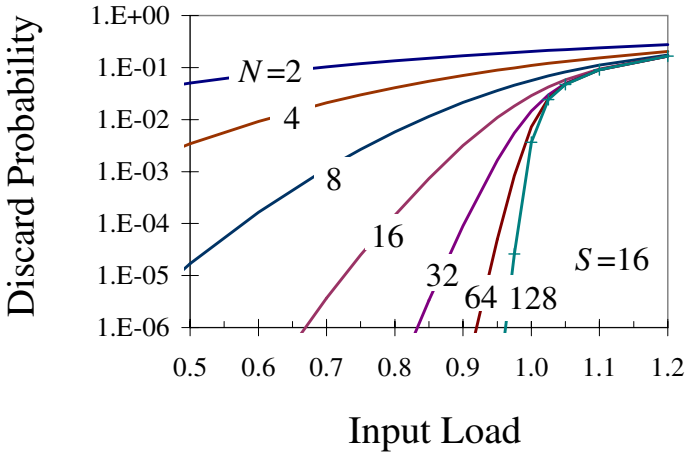


Fig. 3. Packet discard probability for a system with 16 sources and different frame times (N)

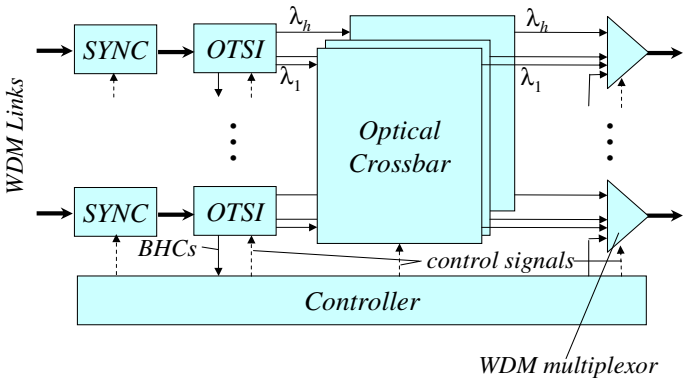


Fig. 4. The overall Time-Sliced Optical Burst Switch design

wavelength conversion become inexpensive enough to make this practical.

The statistical multiplexing performance of a TSOBS network is determined primarily by the number of timeslots per frame. For simplicity, we focus here on the case of simple multiplexor that corresponds to a $S \times 1$ switch receiving bursts from S input channels that can be accommodated by a single timeslot. If the multiplexor assigns each arriving burst to the first available outgoing timeslot, the multiplexor operates like an M/D/1 queueing system with a buffer capacity of N, where N is the number of timeslots per frame.

Fig. 3 shows the burst discard probabilities for a multiplexor with various values of N. Notice that the discard probability drops very quickly with N, but increases beyond a certain point yielding diminishing returns. 32 timeslots is sufficient to maintain a discard probability of 10^{-6} at an offered load of about 83%. Doubling N increases the load at which this target discard probability is reached to about 92%. Variable length bursts lead to higher discard probabilities, but the number of timeslots per frame remains the key determinant of performance. This illustrates the central trade-off for TSOBS networks. While increasing the number of timeslots per frame improves the statistical multiplexing performance, this improvement comes at the cost of longer frame durations, which translates to larger optical buffering requirements. In Section IV we look in more detail at the performance of a TSOBS router, using simulation.

III. Switch Architecture

A. Overview

Fig. 4 shows the overall design for a TSOBS router. Each incoming WDM link terminates on a *Synchronizer* (SYNC) which synchronizes the incoming frame boundaries to the local timing reference. This is done using variable delay lines, with feedback control of the delays being provided through the system controller. The synchronizers are followed by Optical Time Slot Interchangers (OTSI), which provide the required time domain switching for all wavelengths. The OTSIs also separate the control wavelengths carrying the BHCs and forward those to the system controller. In addition the input OTSIs separate the data wavelengths and forward these on separate fibers to each of a set of Optical Crossbars at the center of the diagram. The crossbars perform the required space division switching operation. These are followed by a set of passive optical multiplexors, which combine the data wavelengths with the control wavelengths (carrying the outgoing BHCs) on the output fibers. The controller uses the information in the BHCs to make switching decisions and generates electronic control signals which are used to control the operation of the OTSIs and the crossbars.

Fig. 5 shows a high level design for one of the OTSIs. Each OTSI contains a set of optical crossbars for switching timeslots among the inputs, outputs and a set of delay lines. The signals are demultiplexed to perform the switching operations and remultiplexed onto the delay lines, allowing the cost of the delay lines to be shared by the different wavelengths. The number of delay lines and the choice of delay line values are key design parameters, significantly affecting both the cost and performance of the OTSI.

B. Nonblocking OTSIs

We can classify OTSI designs as either blocking or nonblocking. While nonblocking designs provide the best performance, they are significantly more expensive than blocking designs. We start with the conceptually simplest nonblocking design, which has N delay lines with a delay value equal to the duration of one time slot. With this design, each incoming timeslot i can be delayed by d timeslot intervals by recirculating it through the ith delay line d times. Since each timeslot is assigned to a separate delay line, there are no conflicts, hence the design is nonblocking. It also uses the smallest possible total delay line length (N, where the unit is the distance light propagates in one timeslot interval). Unfortunately, it requires a large number of separate delay lines (N) and large optical crossbars ($(N+1) \times (N+1)$). The optical crossbars are a particular concern since their cost grows as the product of the number of inputs and outputs. Finally, the design can subject a signal to up to N optical switching operations, causing excessive degradation to the optical signal quality, when N is large. This last fault can be corrected by

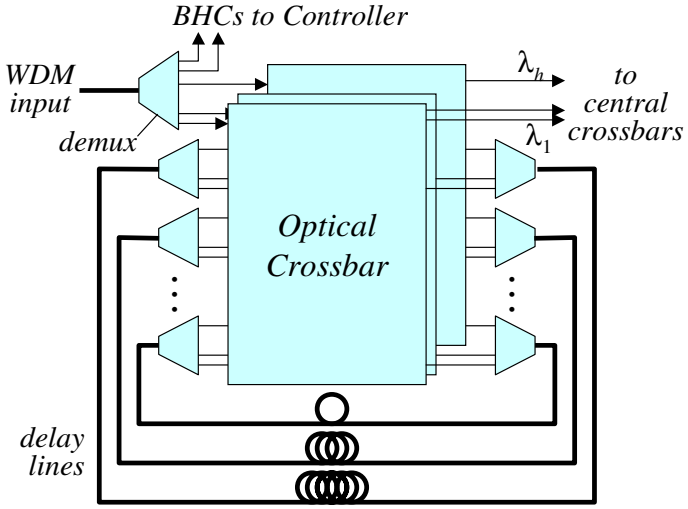


Fig. 5. Optical Timeslot Interchanger

replacing the delay lines of length 1, with delay lines of length $1, 2, \ldots, N$. This allows each timeslot to be switched through just a single delay line, reducing the number of switching operations to 2. Of course, it comes at the cost of increasing the total delay line length from N to approximately $N^2/2$.

A more practical nonblocking switch design uses delay lines of length 1, 2, 3,...,$(A-1)$, where A is an integer parameter, plus additional delay lines of length A, $2A$, $3A$,...,$(B-1)A$ time slots, where B is a second integer parameter. We call these two sets of delay lines the *short delay lines* and the *long delay lines*. Let us suppose a time slot has to be delayed by a value of T time slots. T can be expressed as a sum, $k_2A + k_1$, where $k_1 \in [0, A)$ and $k_2 \in [0, B)$. To delay the time slot by T, we pass the data through the long delay line of length k_2A and then pass it through the short delay line of length k_1. The maximum we can delay a signal using this configuration is $(B-1)A+(A-1)$ and since the maximum delay needed is $N-1$, this gives us the relation $AB \geq N$. The number of delay lines in this design is $A + B - 2$ and hence, choosing $A = B = \lceil\sqrt{N}\rceil$ gives us the minimum number of delay lines. It can be easily seen that these values of A and B generate all values between 0 and N because k_1 and k_2 are the digits of T when expressed in base $\lceil\sqrt{N}\rceil$ notation.

We next show that this design is nonblocking. Consider any two input timeslots i and j that are to be delayed by amounts d_i and d_j, where $i + d_i \neq j + d_j$. Suppose first that $\lceil d_i/A\rceil = \lceil d_j/A\rceil$. Then both time slots will pass through the same long delay line, but since they arrive at different times, they will emerge from the delay line at different times. Hence, they cannot conflict with each other when entering a short delay line. Since $i + d_i \neq j + d_j$, they must emerge from the short delay lines at different times, ensuring no conflict at the output. Now suppose that $\lceil d_i/A\rceil \neq \lceil d_j/A\rceil$. In this case, the time slots may emerge from their respective long delay lines at the same time, creating a potential conflict if they must be switched to the same short delay line. However, such a conflict can only occur if $i + d_i = j + d_j$, contradicting the condition on the overall delays. Hence, the design is nonblocking, assuming timeslots are always switched first through a long delay line, then through a short delay line.

The size of the crossbar required for this design is $(2\lceil\sqrt{N}\rceil - 1) \times (2\lceil\sqrt{N}\rceil - 1)$ (31×31 for $N = 256$) and the length of the fiber required is $N\lceil\sqrt{N}\rceil/2$ (2,048 when $N = 256$). Thus, we have reduced the size of the crossbar at the expense of increased fiber length, relative to the first design. This design also limits the number of switching operations that a timeslot is subjected to, to at most three. Also note that it is very easy to determine the switching operations needed to switch a timeslot.

References [9] and [10] describe a timeslot interchanger design that is *rearrangeably nonblocking* meaning that it can be configured to permute a set of N timeslots in an arbitrary way, assuming that the required permutation is given in advance. This approach can be implemented using two set of delay lines of length $1, 2, \ldots, N/4$ and a single delay line of length $N/2$, where N is assumed to be a power of 2. This gives $2(\log_2 N)-1$ delay lines and a total delay line length of $(3N/2) - 2$. On the other hand, it requires $2(\log_2 N) - 1$ switching operations (15 for $N = 256$) and it requires that the full permutation be known in advance, or that the entire switch configuration be changed as new timeslots are received, making it difficult to apply in the TSOBS context.

Table I summarizes the four nonblocking TSI designs discussed above and shows their complexity characteristics. Only the third design is a real candidate for practical use, and even it is somewhat expensive, both in terms of total fiber length and crossbar complexity.

C. Blocking OTSIs

Blocking OTSIs are an alternative to nonblocking OTSIs, offering lower complexity, at the cost of some small non-zero blocking probability. In the TSOBS context, the impact of a blocking TSI will be to reduce the statistical multiplexing performance slightly.

Perhaps the most natural choice of delays for a blocking TSI is the set $\{1, 2, 4, \ldots, N/2\}$. This allows any time-slot to be switched to any of the output timeslots, provides small total delay (255 for $N = 256$) and small crossbar size (8×8 for N=256). We show below that an OTSI with these delays can be operated so as to achieve a small average number of switching operations (≤ 3 under most conditions), and that the impact of blocking on the statistical multiplexing performance is very small.

For a blocking TSI, we need to define a search procedure to find a sequence of delay lines through which we can switch an arriving timeslot, to deliver it to a free output timeslot. This needs to be done without creating any conflicts at any of the delay lines. The key to implementing the search procedure is a *schedule* that allows us to keep track of which delay lines are available for use at each point in time. The schedule is represented by an array of bits, $sched[i, t]$ with $k + 1$ rows and mN columns, where k is the number of delay lines, N is the number of timeslots per frame and m is an upper bound on the number of timeslots that can be used for a single burst. For $i \in [1, k]$, we

Delay line lengths	Crossbar Size		Fiber Length (in timeslots)		Switching Operations	
		$N = 256$		$N = 256$		$N = 256$
$N \times 1$	$N + 1$	257	N	256	N	256
$1, 2, \ldots, N-1$	N	256	$N^2/2$	32896	2	2
$1, \ldots, A, 2A, \ldots, (B-1)A$	$2\lceil\sqrt{N}\rceil - 1$	31	$N\lceil\sqrt{N}\rceil/2$	2048	3	3
$2 \times (1, 2, 4, \ldots, N/4), N/2$	$2\log_2 N$	16	$(3N/2) - 2$	382	$2(\log_2 N) - 1$	15
Blocking TSI: $1, 2, \ldots, N/2$	$\log_2 N$	8	$N - 1$	255	variable	2 to 3

TABLE I

TABLE SHOWING THE COMPLEXITY OF THE TSI DESIGNS

let $sched[i, t] = 1$ if the ith delay line is "busy" at time t. We say a delay line is busy at a given time, if there is some timeslot scheduled to exit the delay line at that time. For simplicity, we define $t = 0$ to be the current time. We let $sched[0, t] = 1$ if the output link of the OTSI is busy at time t.

Fig. 6 shows an example of the schedule array for $k = 3$, $N = 8$ and $m = 1$. The delay values for each of the delay lines are shown next to their rows. The schedule array implicitly defines a directed graph $G = (V, E)$ that can be used to find a sequence of delay lines leading to a free output timeslot, that can be used for an arriving burst. The vertex set V, is $\{u_0, \ldots, u_{N-1}\}$. Each vertex corresponds to a potential delay that a timeslot may be subjected to. The set of edges E consists of all pairs (u_i, u_j) for which there is some delay line h with delay $j - i$, and $sched[h, j] = 0$. Fig. 7 gives an example of the graph defined by the schedule in Fig. 6.

To find the best sequence of delay lines, we essentially perform a breadth-first search on this graph, starting from node u_0. Such a search constructs a *shortest path tree* in the graph, as illustrated in Fig. 8. The unshaded nodes have delay values that correspond to free timeslots on the output link. The path in the tree to such an output defines a sequence of delay lines that can be used to reach that output. The delay line corresponding to an edge (u_i, u_j) on such a path, is the delay line with delay value $j - i$. The number of switching operations is minimized by selecting a path of minimum length from u_0 to an unshaded vertex. When there are two or more unshaded vertices on minimum length paths from u_0, we select the vertex u_i with the smallest value of i, in order to minimize the delay that a timeslot is subjected to. In Fig. 8, u_3 is selected at the conclusion of the search, and the timeslot is then switched through the delay lines of length 1 and 2.

The search can be done using the schedule array. The procedure does construct the shortest path tree, but does not explicitly construct the graph. A code fragment implementing the required search procedure is shown below. In this procedure, q is a list of nodes from which the breadth-first search needs to be extended next, $p(u_i)$ is the parent of u_i in the shortest path tree constructed by the search procedure and $n(u_i)$ is the number of edges on the path from u_0 to u_i in the tree. The variables $\delta_1 < \delta_2 < \cdots < \delta_k$ are the k different delay values and s is the number of timeslots that the burst being scheduled requires.

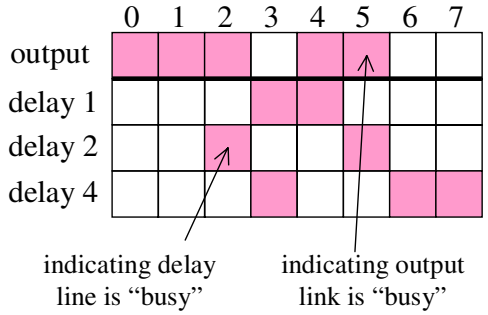


Fig. 6. Example of the schedule array for $k = 3$, $N = 8$ and $m = 1$

```
q := [u_0];
p(u_i) := Ø for all i;
n(u_i) := ∞ for all i;
b := −1;
while q ≠ [ ] do
    u_i := q[1]; q := q[2..]; // remove u_i from q
    if sched[0, d(u_i)] = 0 then
        if b = −1 or
            n(u_i) < n(u_b) or
            n(u_i) = n(u_b) and i < b then
            b := i;
        end;
    end;
    for h = 1 to k loop
        Let u_j be the vertex with j = i + δ_h;
        if sched[h, j] = 0
            and n(u_i) + 1 < n(u_j) then
            p(u_j) := u_i; n(u_j) := n(u_i) + 1;
            if u_j ∉ q then
                q := q & [u_j]; // add u_j to q
            end;
        end;
    end;
end;
```

When a search terminates successfully, b is the delay value associated with the selected output timeslot. If the path from u_0 to u_b (defined by the parent pointers) goes through nodes $u_0 = u_{i_1}, u_{i_2}, \ldots, u_{i_r} = u_b$ the timeslot is switched through the delay lines with delay values $i_2 - i_1, i_3 - i_2, \ldots, i_r - i_{r-1}$. The schedule must be updated to indicate the busy status of the

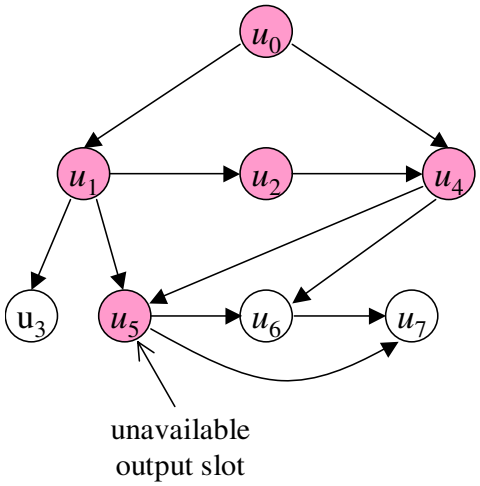


Fig. 7. Directed graph corresponding to example schedule.

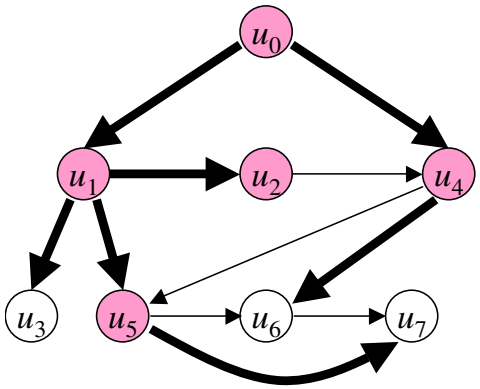


Fig. 8. Search constructing the shortest path tree (shown in bold)

selected delay lines. This is done by setting $sched[j_q][i_q] = 1$, for $1 < q \leq r$, where j_q is the index of the delay line with delay value $i_q - i_{q-1}$.

The search procedure can be terminated early under certain conditions. In particular, if the node u_i, removed from the front of q, has $n(u_i) > n(u_b)$, then there is no point in continuing the search, since no shorter paths to "exit nodes" will be found. We can also terminate the search early if $n(u_i)$ exceeds some prespecified limit on the path length. In this case, the search fails without finding a solution, forcing us to drop the burst being scheduled.

Table I gives the complexity of using a blocking TSI and shows the size of the crossbar required, the length of the delay lines, and the number of switching operations. We can see that the blocking design gives us a decisive cost advantage over the nonblocking ones.

D. Design Issues for the Synchronizers

As a result of the varying length of fibers that terminate at input ports, chromatic dispersion within the fiber, temperature variation leading to phase drift of the optical signals and other fiber transmission non-linearities, the frames arrive at the input ports at random times and completely misaligned with each other. In order to enable the required synchronous switching operations, they must first be realigned. This is the function of the Syncronizers in Fig.4.

The synchronizers can be implemented using a space-division optical switch and a finely calibrated set of delay lines. That is, the synchronizer can use the same basic structure as an OTSI. A key observation is that the synchronizers need not align the incoming data streams on frame boundaries. It is sufficient to align them on timeslot boundaries, since the electronic controller can be designed to compensate for alignment differences that are just integer multiples of the timeslot interval. This is important, because it significantly reduces the number of wavelengths needed.

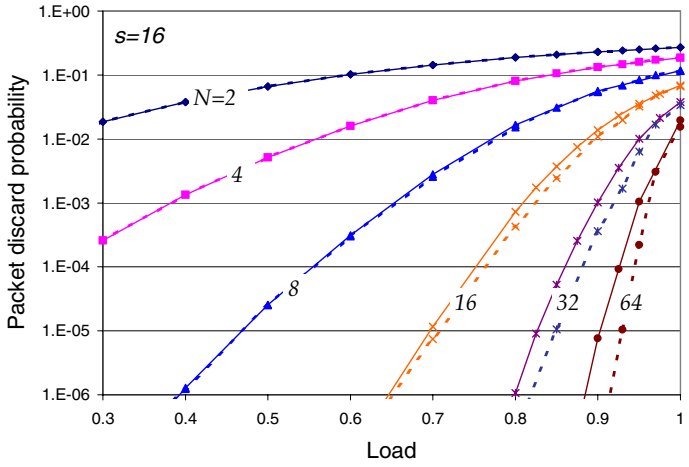
The key design parameter for the synchronizer is the ratio of the timeslot interval to the required alignment accuracy. Defining this ratio to be S, the synchronizer must be able to produce delay values equal to the first S integer multiples of the required alignment accuracy. So for example, if the timeslot duration is 1 μs and the required alignment accuracy is 20 ns, we need 50 distinct delay values. The same choices that apply to the OTSI apply here. Binary delays yield a minimal set of delay lines (6 in the example), but may require $1 + \log_2 S$ switching operations. Using a set of *short* and *long* delay lines allows the number of switching operations needed for synchronization to be cut to 3, but requires more delay lines (14 in the example). Reference [11] addresses the issues with synchronization in more detail.

IV. Performance of a TSOBS Router

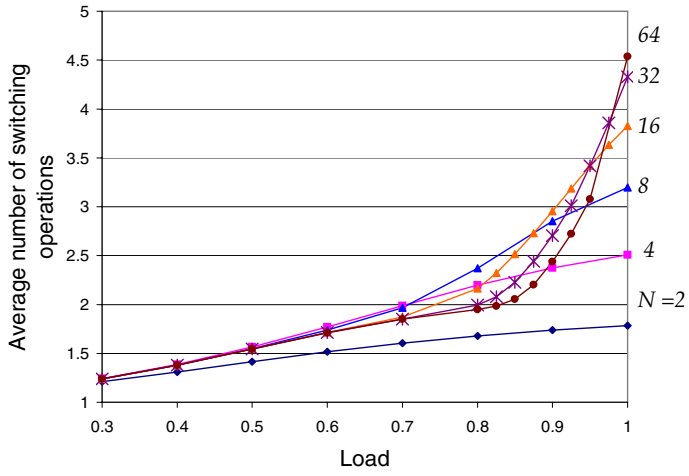
In this section, we study the performance of a TSOBS router using the blocking OTSI design discussed in the previous section. When a BHC is received by the controller announcing the imminent arrival of a burst at one of the input links, the controller does an address lookup to determine the appropriate outgoing link. It must then determine the set of timeslots that are available on the wavelength being used by the burst, both on the outgoing link and at the output of the OTSI for the input link where the burst is to arrive. It then performs the search discussed in the last section to find a set of delay lines through which it can switch the burst, in order to shift it into an available outgoing timeslot.

To evaluate the performance of the OTBS router using a blocking OTSI, we performed simulations using different OTSI configurations. Our primary performance metric is the burst discard probability, which is the fraction of bursts that need to be discarded, due to blocking at either the outgoing link or due to the OTSI. We also measured the number of switching operations that were used to switch the bursts through the OTSIs. The simulations were done for uniform random traffic with binomially distributed arrivals and deterministic burst lengths of one timeslot. The number of input and output links was 16.

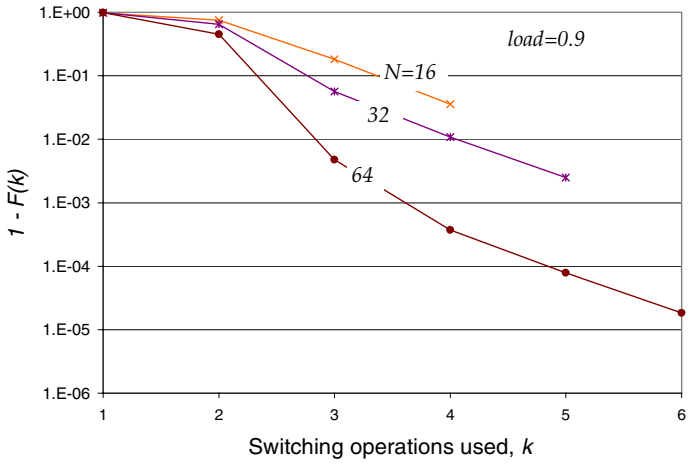
Fig. 9(a) shows the packet discard probabilities for a range of different frame sizes. The OTSIs used delay values of $1, 2, 4, \ldots$ with the largest delay value being equal to half of the frame size. Also shown are the discard probabilities with nonblocking OTSIs. We can see that we do not lose much in the way of performance by using the blocking OTSIs instead of the nonblocking



(a) Packet discard probability vs. Load



(b) Average switching operations vs. Load



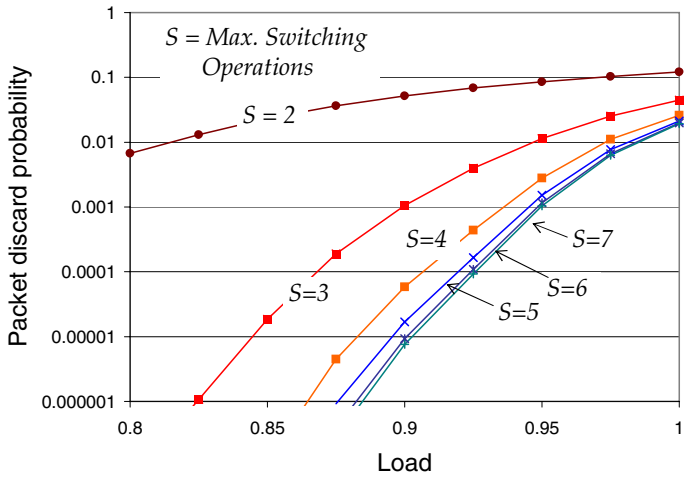
(c) $1 - F(k)$ vs. k, k = number of switching operations

Fig. 9. Charts for different number of time slots per frame, N

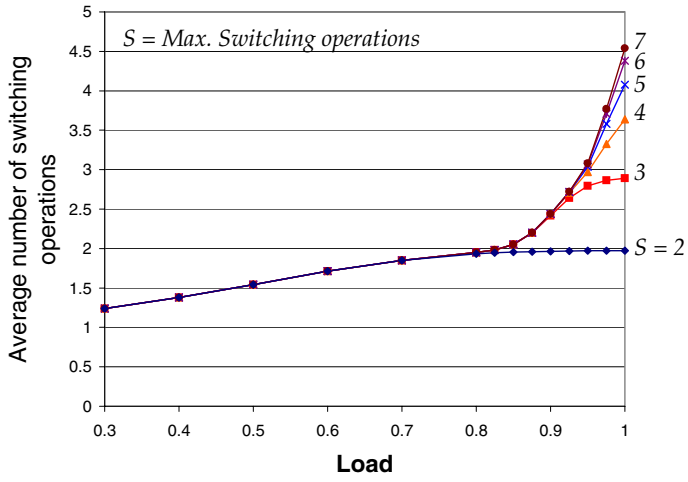
ones, considering that the cost of using nonblocking OTSIs is significantly higher. Figure 9(b) shows the average number of switching operations that each burst is subjected to. For loads up to about 70% the average number remains below 2, meaning that the average burst passes through just one delay line and for loads up to about 90% the average number remains below 3, meaning that the average burst passes through two delay lines only. Fig 9(c) shows the tail of the distribution of the number of switching operations. Specifically, we define $F(k)$ to be the fraction of bursts that require at most k switching operations. So, $1-F(k)$ is the fraction of bursts that require more than k switching operations. Note that the chart uses a logarithmic scale for the values of $1-F(k)$. Fig 9(c) shows $1-F(k)$ when the offered load on the output links is 90%. For $N = 64$, less than 45% of the bursts require more than two switching operations, so almost 55% use at most two, meaning they only use a single delay line and less than 0.5% of the bursts require more than three switching operations, so almost 99.5% use at most three, meaning they only use two delay lines.

The set of results in Fig. 10 shows the effect of placing an upper bound on the number of switching operations that are allowed for each burst. These results are for a system with a frame size of 64. Fig. 10(a) shows that if we restrict the number of switching operations too much, we cause a large increase in the burst discard probability, but with a limit of 3, the burst discard probability is almost the same as when there is no limit. The utilization at a blocking probability of 10^{-6} is approximately 0.83 when the number of switching operations is restricted to 3 and is 0.88 when it is 7, giving about 6% reduction. This is consistent with what we observed in Fig 9(c). Fig. 10(b) shows the average number of switching operations when the number of switching operations is limited. For loads up to 85% the limit has a negligible effect on the number of switching operations, but for loads greater than 90% it produces a significant reduction. Fig. 10(c) shows the fraction of bursts using k or more switching operations, when the number is limited.

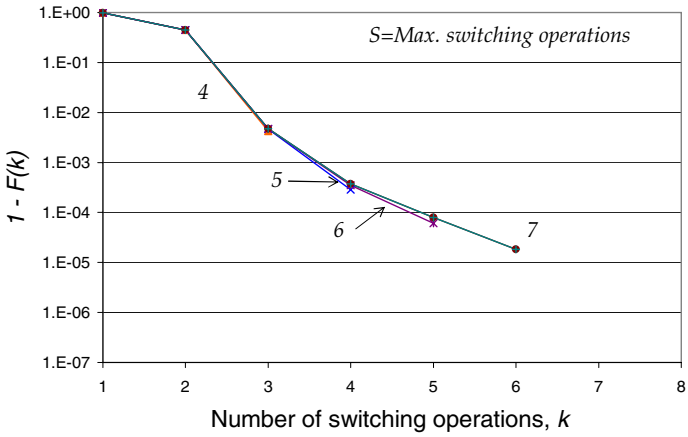
The final set of simulations were performed to quantify the effect of reducing the number of delay lines available. For this set of simulations, the frame size was fixed at 64. The number of delay lines was varied from 1 to 6, with the longer delay lines being omitted, when the number is restricted. So, for example, when 3 delay lines are used, the delay values are 1, 2 and 4. Fig. 11(a) shows the packet discard probabilities. We see that four delay lines gives a comparable blocking performance to six. This is significant, since with four delay lines, the total delay line length is reduced by a factor 4. Fig 11(b) shows the effect on the number of switching operations when the number of delay lines is limited. We can see that decreasing the number of delay lines from six to four increases the number of switching operations at high loads. At loads of 90% the average number of switching operations with four delay lines is about 3.3 and is 2.5 for six delay lines, whereas it is 1.87 and 1.85 at loads of 70% for four and six delay lines respectively. Thus, given the benefits we do not lose much by way of the number of switching operations by reducing the number of delay lines to four. Fig 11(c) shows the tail of the distribution of the number of switching op-



(a) Packet discard probability vs. Load



(b) Average switching operations vs. Load



(c) $1 - F(k)$ vs. k, k = number of switching operations

Fig. 10. Charts for different values of maximum number of switching operations allowed, S

erations. Here, the differences are more evident, but the absolute magnitudes remain small.

V. Conclusions

This paper has introduced a promising variant of optical burst switching, in which switching is done in the time domain, rather than the wavelength domain. This eliminates the need for wavelength converters, the largest single cost component of systems, which switch in the wavelength domain. This may allow optical burst switching to become cost-competitive with electronic packet switching, potentially a very significant development, since no previous optical packet switching architecture has shown any real promise of becoming cost-competitive with electronic alternatives.

Our performance results show that a system with as few as 64 timeslots can provide excellent statistical multiplexing performance, even with blocking OTSIs with just four delay lines. With a timeslot duration of 1 μs, each OTSI would require a total delay line length equal to the distance that light travels in fiber in 15 μs (under 4 km). For routers terminating wide area optical links spanning hundreds or thousands of kilometers, this is a very modest overhead. The average number of switching operations that bursts are subjected to is also quite modest, less than four switching operations per hop, for loads up to 90%. This makes it likely that most bursts could be switched from end-to-end with no intermediate conversion to electronic form.

Substantial additional work is needed to determine if time-sliced optical burst switching will live up to the promise that this study seems to show. Variable length bursts can be expected to produce inferior performance to what we have observed for single timeslot bursts. We have also not yet made a thorough study of the cost of the electronic control subsystem. This could significantly affect the apparent advantages of this approach. The ability of network interfaces to aggregate individual user data packets into larger bursts may also prove to be a critical factor in the practicality of these systems.

References

[1] J. S. Turner, "Terabit Burst Switching," *Journal of High Speed Networks*, 1999.

[2] D. Blumenthal, P. Prucnal, and J. Sauer, "Photonic packet switches: Architectures and experimental implementation," *Proc. IEEE*, vol. 82, pp. 1650-1667, Nov. 1994.

[3] P. Gambini, *et al.*, "Transparent optical packet switching: Network architecture and demonstrators in the KEOPS project," *IEEE Journal on Selected Areas in Communication*, vol. 16, pp. 1245-1259, Sep. 1998.

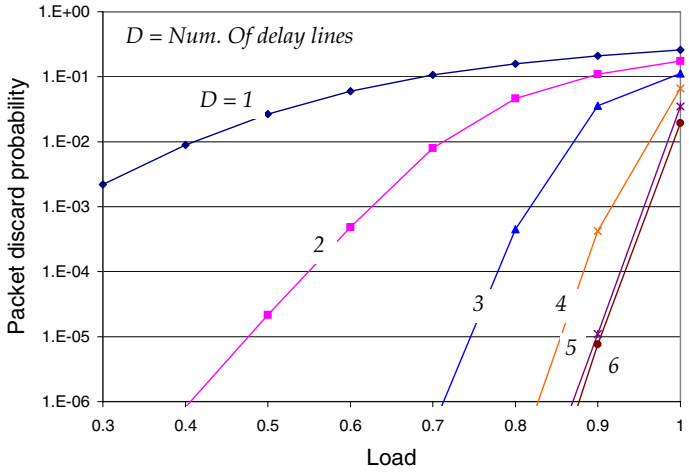
[4] F. Masetti, "System functionalities and architectures in photonic packet switching," *Photonic Networks*, Giancarlo Prati (editor), Springer Verlag 1997.

[5] J. Yates, J. Lacey, and D. Everitt, "Blocking in multiwavelength TDM networks," *Fourth International Conference on Telecommunication Systems, Modeling and Analysis*, pp. 535-541, March 1996.

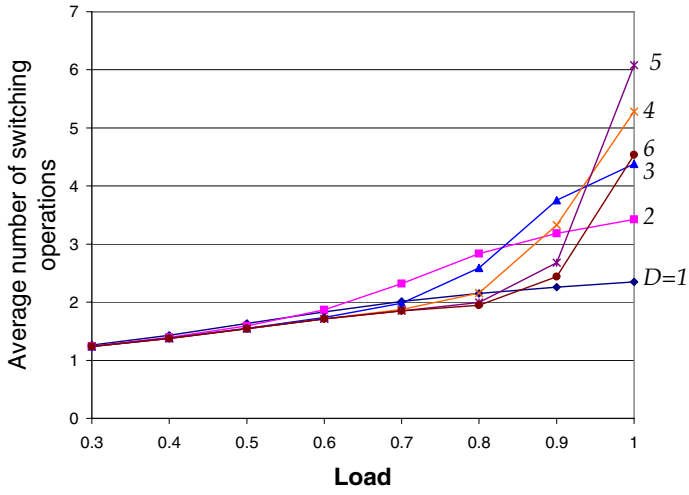
[6] R. Srinivasan, A. K. Somani, "A generalized framework for analyzing time-space switched optical networks," *Proceedings of INFOCOM 2001*, pp. 179-188, April 2001.

[7] B. Wen, K. M. Sivalingam, "Routing, Wavelength and time-slot assignment in time division multiplexed wavelength-routed optical WDM networks," *Proceedings of INFOCOM 2002*, June 2002.

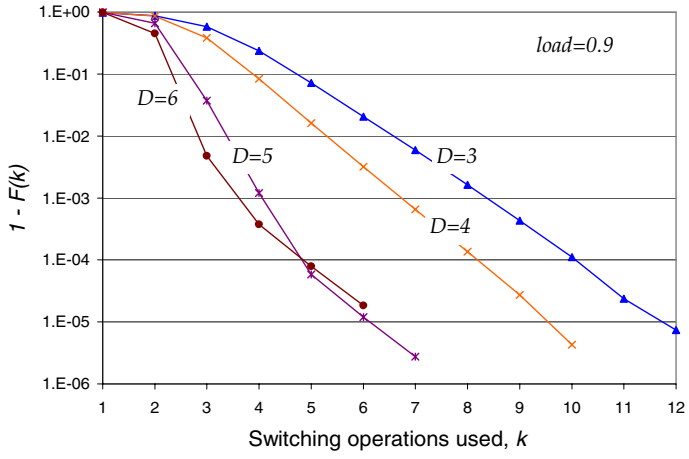
[8] R. Ramaswami and K. N. Sivarajan, *Optical Networks*, Morgan Kaufman, 2002.



(a) Packet discard probability vs. Load



(b) Average switching operations vs. Load



(c) $1 - F(k)$ vs. k, k = number of switching operations

Fig. 11. Charts for different number of delay lines, D

[9] D. K. Hunter and D. G. Smith, "New architectures for optical TDM switching," *IEEE/OSA Journal of Lightwave Technology*, vol.11, no.3, pp. 495-511, March 1993.

[10] H. F. Jordan, D. Lee, K. Y. Lee, and S. V. Ramanan, "Serial array time slot interchangers and optical implementations," *IEEE Transaction on Computers*, vol. 43, no. 11, pp. 1309-1318, November 1994.

[11] B. Bostica, M. Burzio, P. Gambini, and L. Zucchelli, "Synchronization issues in optical packet switched networks," *Photonic Networks*, G. Prati, ed., Springer-Verlag, 1997.