Analytical evaluation and implementation of a novel slotted optical switching scheme with two way reservations

J. D. Angelopoulos, Member, IEEE, K. Kanonakis, H. C. Leiligou, T. Orphanoudakis, Member, IEEE, T. Politi

National Technical University of Athens, Institute of Computer and Communication Systems
Heroon Polytechniou 9, 15773 Zografou Athens, Greece
Tel: +30-2107721511, e-mail: jangel, kokan, nelly, fanel, tpoliti @telecom.ntua.gr

ABSTRACT
Optical Burst Switching achieves multiplexing gain in the optical domain but cannot reach reasonable utilization before burst losses become unacceptable. In this paper, we analytically evaluate a novel method to avoid these losses by first sending over the control channel a short scout packet that simulates the events that the actual packet will experience. Once the scout message detects a drop at any of the intermediate nodes, the actual packet is not sent but the process repeated. The penalty is the delay for the implicit reservations and the occasional process repetition, limiting the applicability of this approach to a periphery of no more than just few thousand kilometers. A slotted approach is adopted to facilitate accounting of data and allow for pipelined operation which improves performance. We also evaluate the implementation complexity of the proposed scheme to prove its feasibility and the avoidance of congestion on the control channel.

Keywords: Optical Burst Switching, slotted operation, analytical performance evaluation

1. INTRODUCTION AND SYSTEM ARCHITECTURE
To achieve multiplexing gain directly in the optical domain, the Optical Burst Switching (OBS) paradigm [1] has been proposed. The principle of operation is simple: the source node sends a Burst Header ahead of each data burst in a control channel to prepare all nodes along the data path for the following burst, which therefore need not be buffered on the way to its destination. An important advantage is that the control processing is kept in the electrical domain while an equally important drawback is rather heavy burst loss [2] except at very low utilization levels due to its ambitious on-the-fly switching.

In our approach, initially presented in [3] first, slotted operation is proposed (as also done in [4]) to reduce the collision probabilities of OBS networks, by avoiding the quite large waste of partial collisions. Second, we propose a two-way reservation scheme which allows for sending payload on the certainty of no collision at any node while our efficient node scheduling algorithm provides precedence to bursts travelling to long destinations. However, the most important feature of the proposed solution is that slotting allows the implementation of a pipelined scheme between control message and data emission. The novelty in this work is that reservations operate per data slot. The concept can be described as “probe-and-go”, because before sending the data slot, the switching contentions along the path are probed by a “scout”. This is a control message sent in the control channel exactly as the OBS headers but also travelling back to the source node with the outcome of the scheduling by each node. To make this possible, an offset just longer than the round trip time is used. In the event of a negative outcome, the data slot will not be emitted. So, only the queues that received a positive scout will emit one slot to be switched according to the already prepared schedules in each node. Hence, the FIFO order is always preserved in payload slots.

Figure 1: The system architecture

The system architecture is similar to any typical OBS environment consisting of a number of periphery nodes where traffic is aggregated from Gigabit Ethernet or slower links and buffered in electronic buffers and aggregated and possibly segmented at the system periphery creating fixed size data packets (slots) [4]. Different queues are maintained per destination. Source routing is employed and knowledge about each route also

The work reported in this paper was supported in part by the EU project FP6-506760 “NOBEL”.

includes the round trip distance measured in slots obtained by ranging after every route update. Full wavelength conversion capabilities are assumed while no fiber delay lines are used to resolve output fiber/wavelength contention.

2. SCOUT FORMAT AND OPERATION

The scouts are situated inside control slots of the same size, also synchronized, to data slots. The format of the scout is shown in Figure 1. Each scout carries the necessary information for one data slot including the source and destination address (10 bit OBS node address covering a domain of 1024 nodes), the wavelength on which the data slot is to arrive (10 bits) and the offset Td (also 10 bits) indicating how many slots later it is to arrive (one slot above the known round-trip in slots). In addition, it carries 2 flags: one with the outcome of the reservation (Ack or Nack), and one indicating whether the scout is traveling in the forward direction (making reservations) or the reverse (i.e. returning with a positive or negative reservation), a 2bit number Na counting reservation attempts (i.e. how many times a scout received a negative reservation and had to be resent). With the 8bit Error Correction and the spare bits, the scout size reaches 64 bits. For a transmission rate of 2.5Gbps in the control wavelength and 100μs slots, each control slot has a size of 250,000bits and can host more than 3,900 scouts, making congestion in the control channel extremely unlikely.

As the scout enters each core node, the node switch learns that Td slots later a data slot will enter from the same port in the indicated wavelength claiming the also indicated output, implying a request to reserve a targeted future slot in the scheduling log of node 0, it is bound to return successfully, since there are no contentions with other data flows.

3. PERFORMANCE EVALUATION

We consider the 3-node topology of Figure 2(a). Only edge node 0 produces data slots towards edge node 2, while core node 1 executes the forwarding. The number of wavelengths in each link is $W$. It is clear that, once a scout manages to reserve a future slot in the scheduling log of node 0, it is bound to return successfully, since there are no contentions with other data flows.

If we assume that data slots are produced at node 0 according to a Poisson process with rate $\lambda$, it is well-known from Probability Theory that the random variable $A$ corresponding to the number of arrivals in a time period $s$ (here denoting the slot duration) follows the discrete Poisson distribution with parameter $\lambda$ (which actually shows the rate of arrivals per slot):

$$\Pr(A = k) = \frac{\lambda^k}{k!} \cdot e^{-\lambda}, \quad \lambda = \lambda \cdot s$$

We refer to the scouts that do not manage to reserve a slot and depart from node 0 as pending ones. These are kept in a dedicated queue waiting for their departure. The latter queue is of a limited capacity $B_s$ in order to keep the queuing delay bounded. All the above lead to a Markov chain for the number of pending scouts at the

Figure 2: (a) Network topology and (b) The Markov chain of pending scouts
beginning of each new time slot (after all the new scout arrivals in the previous slot and the subsequent scout departures at the end of it – hence the maximum state of the chain is \( B_s - W \) and not \( B_s \), as it is certain that after all the arrivals throughout the slot, \( W \) scouts will be serviced) which is shown in Figure 2(b).

If \( \beta \) denotes the step (either positive or negative) in the amount of pending scouts between two consecutive slots, then we have (note that we assume \( P(A = n) \) to be zero for negative \( n \)):

\[
p_{i,\beta} = \begin{cases} 
\sum_{m=0}^{W-i} P(A = m), & \beta = -i \\
\sum_{m=B_s-W}^{\infty} P(A = m), & \beta = -i + B_s - W 
\end{cases}
\]

We denote the \((B_s - W + 1) \times (B_s - W + 1)\) transition probability matrix of the above chain as \( P \) and calculate its elements using equation (1). Let \( \pi \) be the steady-state probability vector for the number of pending scouts. Assuming that this vector exists, it has to follow the equation: \( \pi = \pi \cdot P \). Thus, if by \( \pi[n] \) we denote the state probability vector at epoch \( n \), we recursively calculate the equation \( \pi[n+1] = \pi[n] \cdot P \) (beginning from an arbitrary state probability vector \( \pi[0] \)), until all elements in the vector converge with a certain tolerance \( \varepsilon \), and regard the final vector as the steady-state probability vector \( \pi \). Then, the average number \( N_s \) of pending scouts in equilibrium equals \( E\{\pi_i\} \).

Next, we need to translate that metric into the mean queuing delay of the corresponding (i.e. the ones that produced the scouts under discussion) data slots. Let’s consider a specific scout and the average time between its creation and its successful return from the destination node: As a data slot can arrive at any time between the beginning and the end of a time slot, an average delay of a half slot is introduced. Then, using Little’s Theorem, we get the average time spent by the scout in the queue as \( \gamma = \frac{1}{\alpha} \), where \( \gamma \) stands for the throughput of scouts, i.e. rate of scouts entering the queue, and is related to rate \( \alpha \) via the scouts queue overflow probability \( p_o \):

\[
\gamma = (1 - p_o) \cdot \alpha , \text{ where } p_o = \sum_{i=0}^{B_s-W} \pi_i \cdot \sum_{n=1}^{\infty} n \cdot P(A = B_s - i + n)
\]

Finally, there is the deterministic round trip delay \( r_t \) (including the processing delay at all nodes) for the successful scout to reach its destination and return. Thus we reach the mean queuing delay of a data slot:

\[
D_{tot} = \frac{N_s}{\gamma} + r_t + \frac{s}{2}
\]

The results of the presented analysis were compared to the ones produced by a simulation model, in a scenario with \( W = 2 \), \( s = 100 \mu s \), \( B_s = 50 \) and \( r_t = 65 \) slots. Figure 3, which depicts the mean queuing delay of data slots versus the produced load (as a percentage of a 10Gbps link capacity) validates our analysis.

Figure 3: Queuing delay of data slots for \( W = 2 \), \( s = 100 \mu s \) and \( B_s = 50 \).
4. IMPLEMENTATION ARCHITECTURE

Due to strict timing constraints the node controller which is responsible for the processing of the scout messages is implemented in hardware and more specifically in ASIC technology. The controller interfaces the O/E converters of the control wavelengths of the input and output fibers to receive as well as the optical switch fabric as shown in Figure 4(a) for the case of four input fibers. The node controller receives the content of the control channels, processes it and transmits back the updated scout messages.

The control wavelengths content is temporarily stored in a 64-bit wide (equal width to a scout message) FIFO memory. Then, the received information is filtered: Backward traveling scouts are just forwarded to the appropriate control channel transmitter unit, while the forward traveling scouts (trying to reserve slots) are passed to the “FW scouts processing unit” which calculates the switch matrix. Each column of the switch matrix corresponds to a slot time and each row corresponds to an output fiber and wavelength. The value of the element \( SM[i, j] \) represents the input fiber and wavelength that will be connected to output fiber and wavelength \( i, j \) time slots later. Since the switch matrix should be able to accommodate the largest possible \( T_d \), the memory requirements for \( SM \) are: \( W \cdot F \cdot T_d \cdot \log_2(W \cdot F) \) where \( F \) is the number of connected fibers. For example, for 4 fibers and 64 wavelengths, \( SM \) is 2048\cdot \max T_d.

Figure 4: The implementation architecture: a) overall, b) FW scouts processing unit

The heart of the controller is the “FW scouts processing unit” which executes the “probe-and-go” algorithm to define the resource allocations and resolve conflicts. The internal organization of this unit is shown in Figure 4(b). The received forward traveling scouts are stored in separate memories per targeted slot and output so that scouts targeting the same output for the same slot can be found in the same memory block. Having sorted the scouts per output and addressed slot, a block of logic undertakes the responsibility of scouts processing following the proposed algorithm. To accelerate the whole procedure and keep it below the slot duration, a different block per output and slot may be employed (as shown in Figure 4(b)) or alternatively if the relevant processing time is affordable, a lower number of instantiations of this block can be implemented. We have implemented this architecture for 4 input fibers and 64 wavelengths and the design can fit and achieve the desired rate even when implemented in FPGA technology.

5. CONCLUSIONS

It has been shown through analytical evaluation that slotted operation and two way reservations allow for pipelining the reservation messages with the data slots transmission, which results in improved performance. It moreover facilitates the implementation in hardware.

REFERENCES