

# Performance of The Implementation of A PipeLine Buffering System In Optical Burst Switching Networks

Hailong Li, Hanmeng Neo, Thng Li-Jin Ian  
Department of Electrical and Computer Engineering  
National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260

**Abstract**—*Optical Burst Switching (OBS)* is a new paradigm proposed to efficiently support the ever-growing broadband traffic over WDM networks. In OBS networks, an important issue is how to improve the efficiency of the scheduling algorithm. In this paper, a new control packet buffering scheme “*PipeLine System*” is introduced to buffer a certain number of control packets in order to collect more information before scheduling is carried out. A new scheduling scheme “*Virtual Scheduling Technique*” (*VST*) is subsequently proposed to be implemented together with the *PipeLine System* to improve the overall performance in terms of the burst drop probability. The *VST* technique reduces unused voids significantly, while maintaining *First-In-First-Out (FIFO)* processing discipline on the control packets.

**Keywords**—*OBS, PipeLine, QoS, Scheduling*

## I. INTRODUCTION (HEADING 1)

The explosive growth in the Internet technology has made it imperative to have a network which supports very high bandwidth. *Wavelength Division Multiplexing (WDM)* based on optical networks is becoming a viable solution to meet this ever-increasing bandwidth demand. *Optical Burst Switching (OBS)* [2] is a new paradigm proposed to efficiently support an all-optical Internet for the ever-growing broadband traffic. *OBS* combine the benefit of optical packet switching and optical circuit switching, so that it can maintain the efficiency of optical packet switching while reducing the need to depend overly on optical buffers. In the architecture of *OBS* paradigm, IP packets are assembled into burst data at the network ingress node and the bursts are routed over a bufferless optical transport core using dynamic wavelength assignment. The ingress node sends out a control packet which is followed by a burst after some offset time,  $OT \geq \Delta h$ , where  $\Delta$  is the processing time of the control packet at each intermediate node and  $h$  is the number of hops along its route. This scheme ensures that the burst data cannot overtake the corresponding control packet. In *OBS* network, the resources are allocated using one way reservation through the use of the control packet since the sender of a burst does not wait for an acknowledgement of its reservation request. This guarantees that the forwarded burst will undergo at most a pre-defined delay at the ingress node plus latency due to propagation time.

Another main concern in *OBS* is to provide some *QoS* support for multimedia and real time applications. There are several techniques to provide *QoS* in the form of priority classes in *OBS* and the most commonly used technique is to assign an extra *QoS* offset time [3] to the higher priority

classes. This technique does not require additional protocol processing to differentiate priority and is implemented in the *WDM* physical layer.

The offset times among all the bursts in the network are different since each burst do not traverse the same route or encounter the same number of hops. In addition, additional *QoS* scheme will also add extra offset time and node scheduling may involve the use of an *FDL* which will also affect the offset time. In terms of drop probability, we demonstrate that it is better to schedule the bursts based on burst arrival time rather than on the control packet arrival time for an *OBS* with widely different offset time differences.

This paper is organized as follows. In Section II, we propose the “*PipeLine System*” which allows the scheduling of bursts according to burst arrival time while maintaining *FIFO* processing discipline on the control packets. In Section III, the numerical simulations are presented to demonstrate the usefulness of the technique. We conclude the paper in Section IV.

## II. IMPLEMENTATION OF PIPELINE SYSTEM

### A. Channel Scheduling Algorithm

In *OBS* literature, many channel scheduling algorithms have been proposed as described in [1,4]. Amongst these algorithms, the most popular ones are *First Fit (FF)* algorithm, *Latest Available Unscheduled Channel (LAUC)* and *Latest Available Unused Channel with Void Filling (LAUC-VF)* algorithm. In this paper, the *LAUC-VF* algorithm is used as the default channel scheduling algorithm to schedule the bursts of the control packets being buffered by the *PipeLine System*. The basic idea of the *LAUC-VF* algorithm is to minimize voids by selecting the latest available unused data channel for each arriving data burst. Given the arrival time  $t$  of a data burst with duration  $L$  to the optical switching matrix, the scheduler first determines outgoing data channels that are available for the time period of  $(t, t+L)$ . If there is at least one such data channel, the scheduler then selects the latest available data channel, i.e. the channel having the smallest gap between  $t$  and the end of the last data burst just before  $t$ . In comparing the performance of the three scheduling algorithms mentioned above, it was found that *LAUC-VF* algorithm has the best performance and leads to maximum throughput [1].

### B. PipeLine System

By adopting an approach similar to the normal queuing system, a new buffering system called the PipeLine System is proposed to buffer some control packets and at the same time ensure that the flow of the control packets is of a FIFO nature. New scheduling algorithms can also be implemented together with this buffering system, given the additional information obtained by buffering the control packets. In the implementation of the PipeLine System, every new control packet that arrives at each intermediate node will be appended at the entrance of the pipeline. Subsequently, the control packet that is residing at the exit of the pipeline will be forced to leave the pipeline to move on to the next node. The control packets are released in a periodic and orderly manner so that they are not too bursty in the downstream nodes so as to cause a buffer overflow.

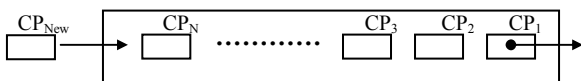


Fig. 1. Architecture of the PipeLine System

Fig. 1 illustrates the use of the PipeLine System to buffer  $N$  control packets at a particular node. Each time a new control packet  $CP_{New}$  arrives at the node, some scheduling algorithm and processing will be carried out on the control packet at the exit of the pipeline (i.e.  $CP_1$ ). Upon completion of these processes, the  $CP_1$  will be pushed out of the pipeline to proceed to the next node in the network. Subsequently, the control packets remaining in the pipeline (i.e.  $CP_2, CP_3 \dots CP_N$ ) will be shifted up to replace the vacancy previously occupied by  $CP_1$ . Following this,  $CP_{New}$  will be appended at the end of the pipeline so as to ensure a FIFO flow of the control packets.

Since the delay incurred in buffering each control packet is dependent on the inter-arrival time of control packets, it is essential to restrict the buffering time of the control packets by implementing a maximum buffer time of  $T_{max}$  for each control packet. Normally,  $T_{max} > N / \lambda$ , where  $N$  is the size of buffer, and  $\lambda$  is the average traffic arrival rate. Once the residence time of a control packet exceeds  $T_{max}$ , it will be forced to exit the pipeline even though there is no new arrival of a control packet. Thus, by implementing the maximum buffer time, we can set the required offset time  $OT = h \times T_{max}$  (i.e. worst case situation whereby each control packet is being buffered for  $T_{max}$  at each node). With the implementation of the PipeLine System at each node, we are now able to exercise more control on scheduling, since by buffering the control packets, more information on burst arrival pattern can be gathered for a  $T_{max}$  period.

C. Scheduling By Burst Arrival Time (SBBAT)

In the pipeline system, all the buffered control packets are accessed to find out their corresponding burst arrival time. The purpose of accessing the burst arrival time of each buffered control packet is to reorder them so that the scheduler will schedule the bursts starting from the control packet with the

earliest burst arrival time by using the LAUC-VF algorithm. In this manner, the reservation requests of control packets will be processed in the order of burst arrival time rather than in the order of the arrival of control packets. The need for such an implementation arises from the fact that there exists a variable offset time between each control packet and its data burst, this means that the arrival of the first control packet does not necessarily imply that its corresponding burst will be the first to arrive. Thus it is essential to reorder the control packets according to their burst arrival time so that the voids between the bursts can be effectively minimized.

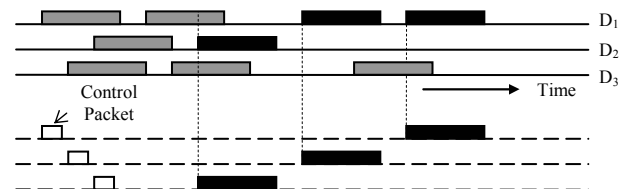


Fig. 2(a). Scheduling by order of Control Packet arrival



Fig. 2(b). Scheduling by order of burst arrival

As an example, Fig. 2(a) illustrates the scenario whereby the bursts are scheduled the moment their control packets arrive at the node. Comparing this with Fig. 2(b), we see that the voids are greatly reduced when the bursts are scheduled according to their burst arrival time. Thus, by scheduling bursts according to their burst arrival time, we are now able to improve the void-filling capability of conventional scheduling algorithms.

D. Virtual Scheduling Technique (VST)

A new scheduling process called Virtual Scheduling Technique (VST) is proposed to work in the PipeLine System. The primary purpose of VST is to solve the conflict between SBBAT (scheduling by burst arrival time) and the FIFO processing of control packets. This conflict arises because the control packet arrival order is not necessarily the same as the burst arrival order in OBS with offset time differences. It means that, in the PipeLine System, the burst with earliest burst arrival time, which is to be scheduled by the scheduler, does not necessarily correspond to the first control packet that arrives at the node. The purpose of VST is to provide as much flexibility as possible in order to maximize the void filling capability of LAUC-VF. Since  $CP_1$  is at the exit of the pipeline, VST must fully schedule (i.e. the time duration and the output channel for the burst are fixed) the burst corresponding to  $CP_1$ . For those bursts that have an arrival time earlier than  $CP_1$ 's burst but whose control packets are

still held up in the pipeline, i.e. behind that of CP<sub>1</sub>, they must be virtually scheduled (i.e. the burst is being assigned to a *tentative* output data channel by using the scheduling algorithm, i.e. LAUC-VF) by the order of burst arrival to conform to the SBBAT principle. This tentative channel assignment reserves bandwidth for the burst but also allows for flexibility in fitting other bursts as well when the need arises. This is because a virtually scheduled burst can be assigned another available data channel or an available FDL in order to fit another burst. Such flexibility cannot be found in a fully scheduled burst. All other bursts, as long as their control packets are behind that of CP<sub>1</sub>, and their burst arrival time are also later than that of CP<sub>1</sub>'s burst will remain as unscheduled. Once the control packet is at the exit of the pipeline, the burst is fully scheduled according to:

- If it has already been virtually scheduled before, then its tentative channel assignment will become permanent.
- Otherwise, if it has never been scheduled before, then LAUC-VF is invoked to fully schedule the burst.

VST allows a window of opportunity for another round of re-scheduling for those bursts whose control packets have not left the node yet in order to fit more bursts.

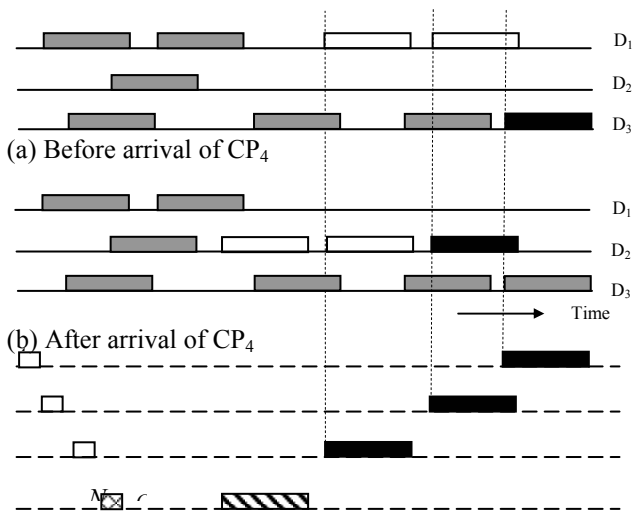


Fig. 3. Void reduction in using "VST"

Fig. 3(a) shows the arrival of the control packets in the order of CP<sub>1</sub>, CP<sub>2</sub>, CP<sub>3</sub> and CP<sub>4</sub>. The scheduler will perform virtual scheduling on B2 and B3 before fully scheduling burst B1 based on the benefits that can arise by scheduling according to burst arrival time. We note that the bursts B2 and B3 are scheduled "virtually" as their control packets still remains inside the pipeline after CP<sub>1</sub> is forwarded to the next node. This allows the scheduler the choice of re-scheduling B2 and B3 if necessary. When a new control packet CP<sub>4</sub> arrives, the scheduler will carry out VST on B3 and B4 before fully scheduling B2. With reference to Fig. 3(b), the scheduler performs a virtual schedule on B4 first (since it has the earliest burst arrival time), then it virtually reschedules B3 to use channel D<sub>2</sub> (to minimize void) before fully re-scheduling B2 to use channel D<sub>2</sub> as well. Notice that due to the previous

tentative channel assignment of B2 and B3, a new channel assignment with lesser voids can be achieved. The advantage of VST is to allow the scheduler to reserve time duration for burst B2 and B3 on data channel D<sub>1</sub> and this reserved time duration can be used by a new data burst by rescheduling B2 or B3 whereby, if so doing, voids of the data channels can be further minimized.

To summarize the implementation of the PipeLine system with the VST algorithm, a description for the algorithm is presented below. There are two parameters being introduced in the description, namely  $n$  and  $k$  where  $n$  is the current buffering depth of the PipeLine system and  $k$  is the queue ID number of each control packet to ensure a FIFO flow.

**Begin {PipeLine System with the VST algorithm}**  
**{ $n=0$ , Queue ID Number  $k=0$ }**

Step 1: Check if residence time of any CP exceeds  $T_{max}$   
 If residence time  $< T_{max}$ , goto Step 2  
 If residence time  $\geq T_{max}$ , goto Step 4

Step 2: Wait for the arrival of a new Control Packet, CP<sub>New</sub>.  
 $n=n+1$

Step 3:  $k=k+1$ ;  
 Assign  $k$  to CP<sub>New</sub> as ID number to mark it, so that it will be the last to leave the PipeLine .  
 If ( $n = N$ )  
     goto Step 4  
 Else goto Step 1

Step 4: Carry out "VST" on the bursts that have earlier burst arrival time than the burst of the Control Packet with the smallest ID number (i.e. CP<sub>1</sub>).

Step 5: Fully schedule the burst of CP<sub>1</sub>.  
 If (schedule is success) Forward CP<sub>1</sub> to next node.  
 Else  
     Drop CP<sub>1</sub>  
 Endif  
 $n=n-1$ ;  
 goto Step 1.

**END**

III. SIMULATION RESULTS AND DISCUSSION

In this section, the performance of the PipeLine system is presented. We consider drop probability and the capability of the pipeline system for providing priority QoS using extra offset time. The optical router used in the simulation environment is set to operate with a transmission rate of 10Gbps, 16 data channels and 1 control channel. Reference [6] has shown that after assembling IP traffic into burst at the OBS ingress node, the burst arrival time can still be assumed to be exponentially distributed for short time scale and the burst size can be assumed to have a Gaussian distribution [1,5] if the source IP traffic is Self-similar. Hence, in our

simulation, we assume an exponential distribution for burst inter-arrival time and a Normal distribution for burst size.

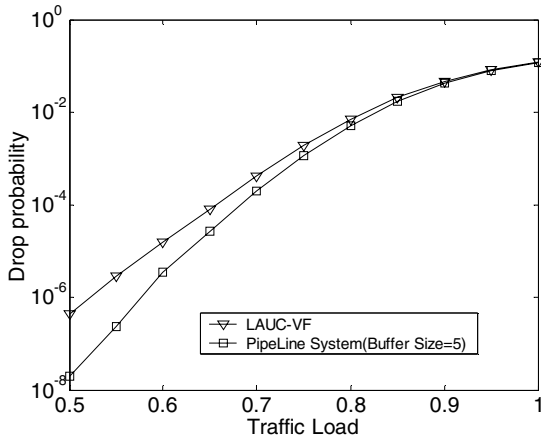


Fig. 4. Performance of a Classless PipeLine System {Buffer Size = 5 Control packets}

Fig. 4 shows the performance of the conventional LAUC-VF scheduling system and the PipeLine System with a buffer size of 5 control packets in a classless scenario. A single FDL with delay parameter corresponding to the average burst length duration was employed in the optical node. The burst lengths conform to a Gaussian distribution. From the plot, the drop probability of the PipeLine System with a buffer size of 5 control packets is better than the LAUC-VF scheduling algorithm by around one order at a low traffic intensities of 0.5 – 0.6.

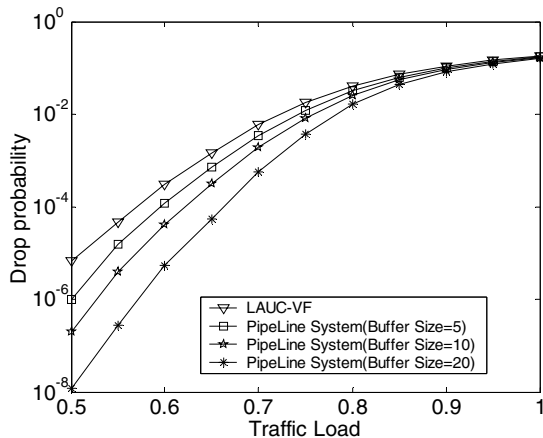


Fig. 5. Performance on drop probability for LAUC-VF and PipeLine System with buffer size {5,10,20}.

Fig. 5 shows the drop probability of a 6 priority classes traffic scenario under increasing traffic intensities ranging from 0.5 – 1.0 for LAUC-VF scheduling and VST scheduling for a PipeLine System with buffer sizes of {5, 10, 20}. The optical node employed 3 FDLs which correspond to burst length duration of one times, two times and three times that of the average burst length duration. It is noted that the drop probability decreases significantly as the buffer size of the PipeLine system increases. This is because more information can be obtained by buffering more control packets and this allows the VST scheduler to make better decisions when

scheduling the data bursts. Also we note that at higher traffic loads, the improvement in the drop probability diminishes.

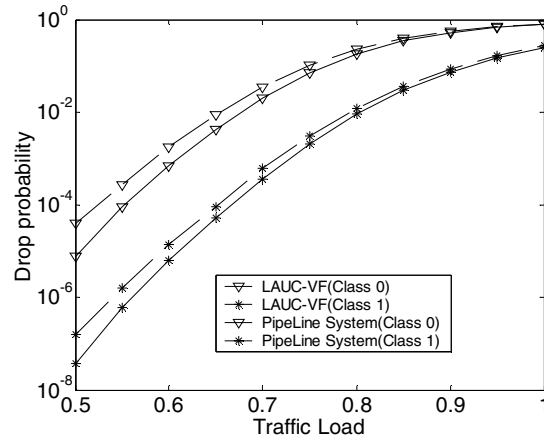


Fig. 6. Effect of PipeLine System on service differentiation.

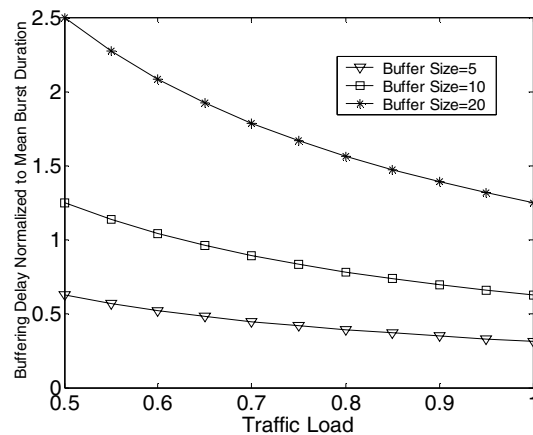


Fig. 7. Average Control Packet Buffering Delay for PipeLine System with buffer size {5,10,20}.

Fig. 6 illustrates the performance of the PipeLine System on service differentiation of the network under same traffic scenario of Fig. 5 with buffer size of five. Because the drop probability of the higher 4 priority traffic is zero in simulation results when load is less than 0.9 and the number of arrived bursts is  $10^6$ , we only show the performance of lowest 2 priority. It can be observed from the graph that the Pipeline system can still provide service differentiation. The plot show that the dropping probabilities of both class 0 and class 1 bursts in the PipeLine System are lower than the dropping probabilities under conventional LAUC-VF scheduling. The pipeline system is designed to provide QoS by setting the basic offset time of the bursts to be equivalent to  $h \times T_{max}$  and their corresponding extra QoS offset time, with the assumption of a worst case scenario that all the control packets will be buffered for a period of  $T_{max}$  at each hop. PipeLine System ensure the QoS differentiation, because PipeLine System still maintains the feature of FIFO, which means the higher priority traffic can reserve bandwidth before the lower priority traffic like conventional system.

Fig. 7 shows the average buffering delay for each control packet under increasing traffic load ranging from 0.5 – 1.0 for the PipeLine System with buffer sizes of {5, 10, 20}. Even though the drop probability decreases significantly as the buffer size of the PipeLine system increases, there is an upper limit to the buffer size that can be employed as the delay experienced by each control packet is proportional to the buffer size of the PipeLine system. From the graph, we see that the average buffer delay of each control packet decreases with increasing traffic intensities for each buffer size.

#### IV. CONCLUSION

In this paper, a new buffering system called PipeLine System has been proposed to buffer a designated number of control packets at intermediate OBS nodes. A new scheduling technique called VST is contributed to allow the scheduler to schedule bursts according to their burst arrival time while providing flexibility in re-scheduling the bursts of control packets remaining in the PipeLine. From simulation results, it is shown that the PipeLine System has better drop probabilities compared to conventional LAUC-VF scheduling algorithm, and QoS is guaranteed as well. Although the performance of the PipeLine System can be increased by

increasing the buffer size, it is noted that there is a certain limit to the number of control packets that can be buffered, so that the control packets will not be overly delayed and the residence time of each control packet is restricted by setting a maximum buffer time,  $T_{max}$ , for each control packet.

#### REFERENCES

- [1] Y. Xiong, M. Vandenhoude, and H. C. Cankaya, "Control Architecture in Optical Burst-Switched WDM Networks". IEEE Journal On Selected Areas In Communications, Vol. 18, No. 10, October 2000.
- [2] C. Qiao and M. Yoo, "Optical burst switching (OBS)- a new paradigm for an optical Internet", J. High Speed Networks, vol.8, no.1, pp.69-84, 1999.
- [3] Myungsik Yoo, Chunming Qiao, Sudhir Dixit, "QoS Performance of Optical Burst Switching in IP-Over-WDM networks," IEEE Journal on Selected Areas in Communications, Vol 18, No. 10, Oct 2000.
- [4] L. Tancevski, A. Ge, G. Castanon, and L. Tamil, "A new scheduling algorithm for asynchronous, variable length IP traffic incorporating void filling", in Proc. OFC'99.
- [5] Xiang Yu, Yang Chen and Chunming Qiao, "A Study of Traffic statistics of Assembled Burst Traffic in Optical Burst Switched Network", Proceedings of SPIE Optical Networking and Communications Opticomm, Vol. 4874, 2002.
- [6] M. Izal and J. Aracil. "On the Influence of Self-similarity on Optical Burst Switching Traffic". Proceeding of GLOBECOM, 2002.