Article number 5395466, Pages 609-614 ; DOI- http:/dx.doi.org/10.1109/ICETET.2009.46
Second International Conference on Emerging Trends in Engineering and Technology, ICETET-09
Nagpur; 16 December 2009 through 18 December 2009; Category number E3884; Code 79542

Best Fit Void Filling Algorithm in Optical Burst Switching Networks

M. Nandi, A. K. Turuk, D. K. Puthal and S. Dutta
Department of Computer Science and Engineering,
National Institute of Technology, Rourkela, India
E-mail: {mrinal.nandi, akturuk, hal.css and saibal.13}@gmail.com

Abstract:
Optical Burst Switching is a promising technology in Optical Network. Scheduling of data burst in data channels in an optimal way is one of a key problem in Optical Burst Switched networks. The main concerns in this paper is to schedule the incoming bursts in proper data channel such that more burst can be scheduled so burst loss will be less. There are different algorithms exists to schedule data burst on data channels. Latest available unscheduled channel with void filling and minimum end void are the best among other existing non-segmentation based void filling algorithms. Though it gives less burst loss, but not utilizing the existing voids efficiently. In this paper we propose a new approach, which will give less burst loss and also utilize voids in efficient way. Also analyze the performance of this proposed scheduling algorithm and compare it with the existing void filling algorithms with respect to burst loss by simulation. It is shown that the proposed algorithm gives some better performances compared to the existing algorithms.
Keywords: Optical Burst Switching, Scheduling Algorithm, Void Filling Algorithms, LAUC-VF, Min-EV.

I. Introduction

Optical burst switching (OBS) is emerging as the switching technology for next generation optical networks. Advantages of optical packet switching and circuit switching are combined in OBS and overcoming their limitations. Data (or payload) is separated from control packet. A control packet is sent before the payload to reserve the resources on the path to the destination of payload. When a control packet arrives at an intermediate node a wavelength scheduling algorithm is used by the scheduler to schedule the data burst on an outgoing wavelength channel. The required information to schedule a data burst are arrival time and duration of data burst, which are obtained from the control packet. On the other hand, scheduler keeps availability of time slots on every wave length channel and schedule a data burst in a channel depending upon the scheduling algorithm it uses. Different scheduling algorithms have been proposed in literature to schedule payload/ data burst. They differ in burst loss and complexity. Depending upon the channel selection strategy, they can be classified as *Horizon* and *Void filling* algorithm. *Horizon* algorithm considers the channels which have no scheduled data burst at or after current time t and the channels are called *Horizon* channels. *Void filling* algorithms consider the channels which have unused duration in between two scheduled data bursts. These are called *Void* channels. The example of *Horizon* algorithms are FFUC, LAUC and *Void filling* algorithms are FFUC-VF, LAUC-VF and Min-EV. *Horizon* algorithms are easy to implement and burst loss ratio is high. Where as burst loss ratio is lower in *Void filling* algorithms but complex switching are required to implement. Among the *void filling* algorithms, burst loss ratio is lower in LAUC-VF and Min-EV. LAUC-VF schedule a data burst in a *void* channel such that the time difference between arrival data bursts starting time and previous scheduled data bursts end time is minimum. Where as Min-EV schedule a data burst in a *void* channel, such that the time difference between a scheduled data bursts start time and arrival data bursts end time is minimum. Both, LAUC-VF and Min-EV consider only one side of a void. There may be a possibility, in which a smaller data burst will be scheduled in a larger void where as a bigger data burst will be dropped. This will lead to higher burst blocking and lower channel utilization.

In this chapter we propose a new channel scheduling algorithm, which attempts to make efficient utilization of existing void within a channel. Thus, giving rise to higher channel utilization and lower blocking probability. Rest of the paper is organized as follows. Section 2 explains the existing scheduling algorithms. Limitations of the existing void filling algorithms are explained in Section 3. Working of the propose best fit void filling algorithm is explained in Section 4. We compare our proposed scheme with LAUC-VF and Min-EV schemes. Comparison and simulation results are presented in Section 5. Finally, some conclusions are drawn in Section 6.

II. Scheduling Algorithms

When a control packet arrives at a core node, a wavelength channel scheduling algorithm is used to determine a wavelength channel on an outgoing link for the corresponding data burst. The information required by the scheduler such as the expected arrival time of the data burst and its duration are obtained from the control packet. The scheduler keeps track of the availability of time slots on every wavelength channel. It selects one among several idle channels. The selection of wavelength channel needs to be done in an efficient way so as to reduce the burst loss. At the same time, the scheduler must be simple and should not use any complex algorithm, because the routing nodes operate in a very high-speed environment handling a large amount of burst traffic. A complex scheduling algorithm may lead to the early data burst arrival situation wherein the data burst arrives before its control packet is processed and eventually the data burst is dropped [1].

In this section we discuss various scheduling algorithms proposed in literature [2, 3]. These algorithms differ in their complexity and performance in terms of burst loss. A wavelength channel is said to be unscheduled at time *t* when no data burst is using the channel at or after time *t*. Algorithms which consider unscheduled channels are called Horizon algorithm. A channel is said to be unused for the duration of voids between two successive data bursts and after the last data burst assigned to the channel. Algorithms which consider voids within channels are called void filling algorithm. According to scheduling strategy used scheduling algorithms can be classified as follows:

- Horizon or Without void filing [2].
- With void filling [3].

Representative of Horizon algorithms are: First Fit Unscheduled Channel (FFUC) [4, 2, 3, 5], Latest Available Unused Channel (LAUC) [6, 5] and that of void filling algorithms are: First Fit Unscheduled Channel with Void Filling (FFUC-VF) [3], Latest Available Unused Channel with Void Filling (LAUCVF) [4, 7, 8] and Minimum End Void (Min-EV) [8].

Working of algorithms is illustrated with the help of Figure 1. In Figure 1, control packet arrives at a node at time t_{CB}. Duration of payload is t_{burst} and the offset time for the data burst is t_{offset}. The offset time is calculated as:

$$t_{offset} = H \times \Delta \quad (1)$$

where H is number of hops from source to destination and Δ is the time required for processing and switching the control packet. The time at which the first bit of payload arrives at the node is $t_{CB} + t_{offset}$ and the last bit arrive at $t_{CB} + t_{offset} + t_{burst}$. We define unscheduled channel and void channel as following:

Unscheduled channel: A wavelength channel is said to be unscheduled at time t when no data burst is using the channel at or after t.

Void channel: If a channel is unused for duration between two successive data bursts.

III. Limitations Of The Existing Scheduling Algorithms

Horizon scheduling algorithms consider the unscheduled channels to schedule a data burst. It does not consider the availability of void within a channel, which could otherwise be used in channel scheduling. For example consider the Figure 2. In this figure there two data bursts *a* and *b* are scheduled on channel *1* and data burst *c* on channel *2*. For horizon scheduling algorithms, channel *1* is available at time instant *t* and channel *2* is at *t'*. Suppose a data burst *x* arrives. Horizon scheduling algorithms will schedule the data burst *x* on channel *2* as shown in Figure 3. They do not consider the voids within a channel. In channel *1* there exist a void between data bursts *a* and *b* within which the data burst *x* could have been scheduled. Thus, horizon scheduling algorithms are not efficient in terms of channel utilization and gives rise to higher burst loss.

On the other hand, void filling algorithms consider both unscheduled and void channel to schedule data bursts. For the scenario as shown in Figure 2, void filling algorithms will schedule data burst *x* on channel *1*. Thus, increases the channel utilization. Any data burst arriving between *t'* and *t* could be schedule on channel *2*, which otherwise could have been dropped in horizon algorithms. Thus, horizon scheduling algorithms are not efficient in terms of burst loss and channel utilization in comparison to void filling algorithms.

Though void filling algorithms are efficient than horizon scheduling algorithms, but they are not the optimal scheduling algorithms. The limitations of the void filling algorithms such as LAUC-VF and Min-EV algorithms lies in the fact that they consider only one side of a void. LAUC-VF, consider the void created between incoming data bursts start time and previous scheduled data bursts end time. Whereas Min-EV, consider the void created between scheduled data bursts start time and incoming data bursts end time. Due to this smaller size data bursts may be scheduled in a larger void whereas bigger size data bursts may get blocked. In the following subsection a brief description of the limitations in terms of blocking and channel utilization of LAUC-VF and Min-EV void filling algorithms is presented.

A. Blocking in LAUC-VF and Min-EV

In OBS data bursts are of variable lengths. If a smaller data burst arrive earlier than a larger size data burst then void filling algorithm may schedule the smaller data burst on a larger void and the larger size data burst may be dropped due to unavailability of data channel. This can happens in void filling algorithms due to their consideration of one side of a void.

For example consider the Figure 5. In this figure data burst *b0* and *b1* are schedule on channel *1*, *b2* and *b3* on channel *2* and *b4* and *b5* on channel *3*. On channel *1* the end time of data burst *b0* is t_1 and start time of data burst *b1* is t_2. Data burst *b2* has end time of t_3 and data burst *b3* has start time of t_4 on channel *2*. Similarly, for data burst *b4*, t_5 is the end time and for data burst *b5*, t_6 is the start time.

Suppose three data bursts *B0*, *B1* and *B2* arrive at a node. Arrivals of control packet for data bursts are shown in control channel. Control packet *CB0* for data burst *B0* has arrived first then *CB1* for data burst *B1*, and finally *CB2* for data burst *B2* arrived in that order. Start time and end time of data burst *B0* is t_{b0} and t_{e0}, for data burst *B1* is t_{b1} and t_{e1} and for data burst *B2* is t_{b2} and t_{e2}.

Scheduling of the data burst onto a channel depend on the type of scheduling algorithm node is using. That is, whether node is using LAUC-VF or Min-EV algorithm. We present below two different cases. One for scheduling with LAUC-VF and the other for Min-EV algorithms. Since the data burst *B0*, *B1* and *B2* arrive in that order, the scheduler will schedule data burst *B0* first, then *B1* and followed by *B2* in that order.

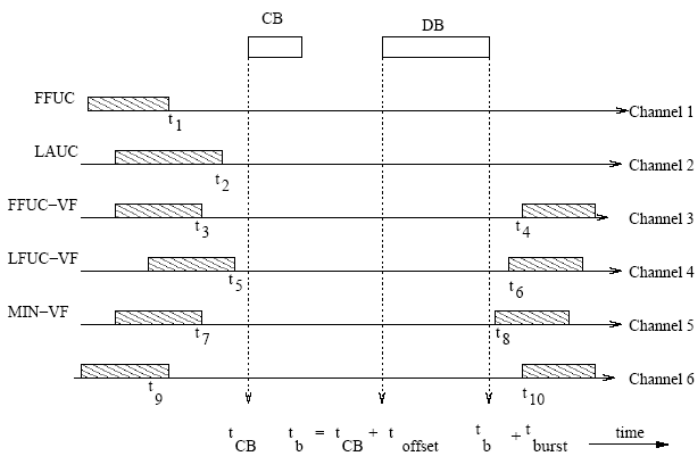

Fig: 1 Illustration of Burst Scheduling Algorithms

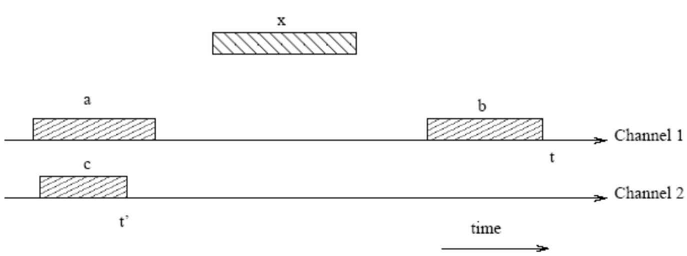

Fig: 2 A scheduling scenario

Case 1: Scheduling using LAUC-VF
LAUC-VF algorithm tries to schedule a data burst on a void, such that difference between the start time of a new data burst and the end time of a previous scheduled data burst whose end time is prior to the new data burst start time will be minimum.

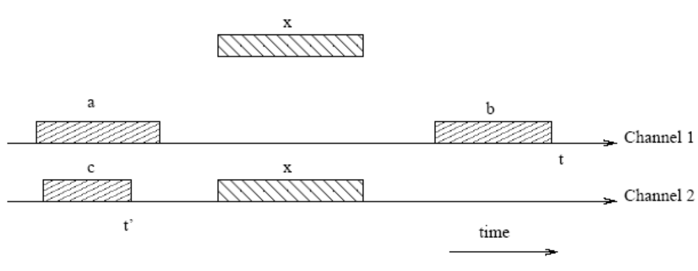

Fig: 3 Scheduling by horizon algorithms

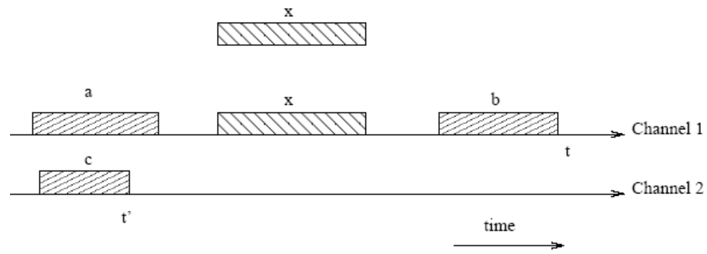

Fig: 4 Scheduling by void filling algorithms

Data burst *b0*, *b2* and *b4* have their end time prior to data burst *B0*'s start time. Differences between the start time of *B0* and end time of *b0*, *b2* and *b4* are $(t_{b0} - t_1)$, $(t_{b0} - t_3)$ and $(t_{b0} - t_5)$ respectively. Of these LAUC-VF, schedule the data burst on a channel, that has the minimum difference. Difference between the start time of data burst *B0* and end time of data burst *b0* is minimum. That is $(t_{b0} - t_1)$ is the minimum value of the three values $(t_{b0} - t_1)$, $(t_{b0} - t_3)$ and $(t_{b0} - t_5)$. So LAUC-VF schedule the data burst *B0* on channel *1*. When the request CB1 for data burst B1 arrives, there is no available channel to schedule the data burst B1, hence B1 is dropped. Data burst B2 can be schedule in channel 2.

Case 2: Scheduling using MIN-EV
In Min-EV scheduling algorithm, an incoming data burst is schedule on a channel, such that the start time of a already scheduled data burst and end time of an incoming data burst is minimum. Here we consider only those schedule data bursts whose start time is after the end time of the incoming data burst. In Figure 5, data bursts *b1*, *b3* and *b5* have start time after the end time of data burst *B0*.
Difference between the end time of data burst *B0*, and the start time of data burst *b1*, *b3* and *b5* are $(t_2 - t_{e0})$, $(t_4 - t_{e0})$ and $(t_6 - t_{e0})$ respectively. Of these $(t_4 - t_{e0})$ is the minimum. So the data burst *B0* is schedule on channel *2*. Similarly data burst *B1* is schedule on channel *1*. However, data burst *B2* can not be schedule as there is no wavelength channel is available.

B. Channel Utilization in LAUC-VF and MIN-EV
In Figure 5 the duration of void in channel *1*, *2* and *3* are $(t_2 - t_1)$, $(t_4 - t_3)$ and $(t_6 - t_5)$ respectively. Higher the fraction of void utilized higher will be channel utilization. Fraction of void utilized is the ratio of the data burst duration scheduled on the void to the void duration.
In Figure 5 LAUC-VF schedule data burst *B0* in the void of channel *1*. The fraction of void utilized is $(t_{e0} - t_{b0})/(t_2 - t_1)$. Of these the fraction $(t_{e0} - t_{b0})/(t_2 - t_1)$ is smaller. Scheduling data burst *B0*, in channel *1*, *2* and *3*, the fraction of void utilized will be $(t_{e0} - t_{b0})/(t_2 - t_1)$, $(t_{e0} - t_{b0})/(t_4 - t_3)$ and $(t_{e0} - t_{b0})/(t_6 - t_5)$ respectively. This is because $(t_2 - t_1) > (t_4 - t_3) > (t_6 - t_5)$. Thus scheduling data burst *B0* in channel *1*, gives rise to inefficient channel utilization. Moreover, this creates a void $(t_2 - t_{e0})$ of considerable duration.

Min-EV algorithm schedule data burst *B0* in channel *2*. Fraction of void utilized is higher than that of scheduling on channel *1* and lower than scheduling on channel *3*. Scheduling *B0* in channel *2*, void of channel *3* remains utilized.

Thus, it is observed that the channel utilization is lower in both LAUC-VF and Min-EV. This is because both algorithms consider only one side of a void i.e., either the start or end side of a void. Next we propose a new channel scheduling algorithm which considers both end of a void in scheduling and it utilizes void efficiently.

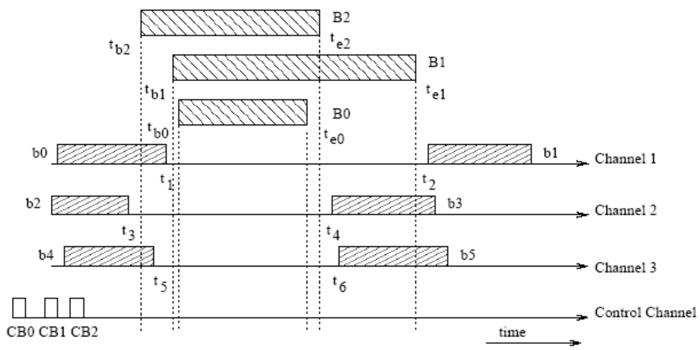

Fig: 5 Failures of LAUC-VF and Min-EV Algorithms

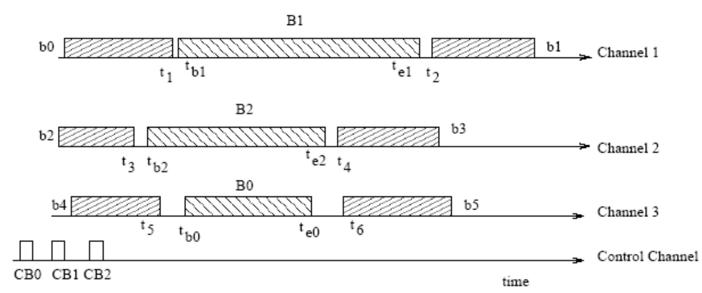

Fig: 6 Scheduling by BFVF algorithm

IV. BEST FIT VOID FILLING ALGORITHM

In this section we propose a new scheduling algorithm called Best Fit Void Filling (BFVF), which attempts to maximize the channel utilization and minimize the burst loss. Our propose algorithm first selects all possible void channels, on which the data burst can be scheduled. Then selects one of the possible void channel such that the void utilization factor is maximum. We calculate the void utilization factor as:

$$utilization = (a \times 100)/x \quad (2)$$

Where a is the data burst length and x is the void length.

In Figure 5, data burst *B0* can be schedule any one of the channel *1*, *2* and *3*. Void utilization factor for *B0* on channel *1*, *2* and *3* are $(t_{e0} - t_{b0})/(t_2 - t_1)$, $(t_{e0} - t_{b0})/(t_4 - t_3)$ and $(t_{e0} - t_{b0})/(t_6 - t_5)$ respectively. Void utilization factor for channel *3* is maximum, since $(t_6 - t_5) < (t_4 - t_3) < (t_6 - t_5)$. So BFVF algorithms selects channel *3* to schedule the data burst *B0*. Similarly data burst *B1* is schedule on channel *1* and *B2* on channel *2*. In our propose algorithm all three data burst *B0*, *B1* and *B2* can be scheduled on channel *3*, *1* and *2* respectively as shown in Figure 6. Thus the channel utilization is higher and burst loss ratio is lower in our propose scheme than in LAUC-VF and Min-EV.

We workout an example to show the void utilization in LAUC-VF, Min-EV and our proposed BFVF algorithm. We assume the following numerical values:

$t2 - t1 = 12\mu s$

$t4 - t3 = 10\mu s$

$t6 - t5 = 8\mu s$

and length of data burst B0 is = $5\mu s$

Void utilization in

LAUC-VF, *utilization* = (5 ∗ 100) /12 = 41.67%

MIN-EV, *utilization* = (5 ∗ 100) /10 = 50%

BFVF, *utilization* = (5 ∗ 100) /8 = 62.5%

This shows that void utilization is higher in our proposed BFVF algorithm.

Formally, we describe BFVF algorithm below. The following notations are used in our algorithm:

$length_b$: Length of the incoming data burst,

$length_v(i)$: Void length in channel *i*,

$start_b$: Start time of a data burst,

$start_v(i)$: Start time of void in channel *i* and

data channel : Data channel selected by the algorithm to schedule the data burst.

Best Fit Void Filling Algorithm

Input: $start_b$, $length_b$

Output: *data channel*

Step 1: Select all possible schedulable void channels. A void channel i is said to be schedulable if $start_b > start_v(i)$ and $length_b < length_v(i)$. If no schedulable void channel exists then goto Step 4.

Step 2: Calculate the channel utilization factor for all schedulable void channel found in Step 1.

Step 3: Find a channel *j* such that it has the maximum channel utilization factor as found in Step 2. Output channel *j* as the required data channel.

Stop.

Step 4: Schedule the data burst according to LAUC algorithm. Stop.

Step 1 of the algorithm is to find a schedulable *void* channel. If no such void channel is available then the data burst is scheduled as in LAUC-VF algorithm.

V. SIMULATION AND RESULTS

We compare the performance of our proposed BFVF algorithms with that of LAUC-VF and Min-EV algorithm through simulation. For simulation, we have considered, obs-ns [9] simulator that runs on the top of ns2 [10] simulator. Performance matrix considered for comparison are: (i) burst loss ratio *vs.* load (in number of data burst sent) (ii) link utilization *vs.* load (in number of data burst sent). Topology considered for simulation is shown in Figure 7 Following parameters are assumed for simulation: *eight* wavelength channels per link (*seven* data channels and *one* control channel), capacity of each channel is 5 Gbit/s, self-similar traffic are considered to generate burst [10], propagation delay between any two adjacent nodes is 1 ms, processing time of control packet is 1.5 μs.

We plot the burst loss ratio *vs.* load in Figure 8. Burst loss ratio is calculated as number of burst loss divided by number of burst sent. It is observed from the Figure 8 that the burst loss ratio is increases with increases in load, in all the three schemes. However the increase in our proposed BFVF scheme is lower than that of LAUC-VF and Min-EV algorithm. This is due to the efficient utilization of void channels in our BFVF scheme, which is not the case in LAUC-VF and Min-EV algorithm. In LAUC-VF and Min-EV algorithm a smaller data burst may be scheduled to a larger void, which is not the case in of BFVF algorithm.

Figure 9 shows the link utilization *vs.* load. Link utilization is calculated as the ratio of the traffic load on the link to the links capacity. It can also be calculated as the ratio of the link busy time to the total time. It is observed from the Figure 9 that the link utilization increases with load, in all the three schemes. However the increase in our proposed BFVF scheme is higher than that of LAUC-VF and Min-EV algorithm. This is due to efficiently scheduling data bursts in appropriate void channels. In case of LAUC-VF and Min-EV a smaller data burst may be scheduled to a larger void, whereas a bigger data burst may be dropped. For that link utilization in LAUC-VF and Min-EV is lower than proposed BFVF algorithm.

Time complexity of our proposed algorithm is same as in LAUC-VF and Min-EV and is equal to $O(w\log n)$, where n is the number of data bursts currently scheduled on every data channel and w is the number of data channels in a link.

VI. CONCLUSION

In this chapter we discuss performance of horizon and void filling scheduling algorithm. It is found that the void filling scheduling algorithm performs better than the horizon scheduling algorithms. However, there are limitations to the existing void filling scheduling algorithms. This limitation is mainly due to that; the existing schemes consider either the start time of the new data burst and end time of the previously scheduled data burst or start time of previously scheduled data burst and the end time of the new data burst. They do not take into account the data burst length and void length.

We proposed an algorithm called BFVF, which takes the arrival data burst length and void length into account in scheduling. Proposed scheme calculates the void utilization factor, and schedule the new data burst into a void channel having maximum void utilization factor. For non-availability of void channel, scheduling takes place as in LAUC scheduling algorithm.

The proposed scheme is compared with LAUC-VF and Min-EV. It is found that the proposed scheme perform better in term of burst loss ratio and channel utilization.

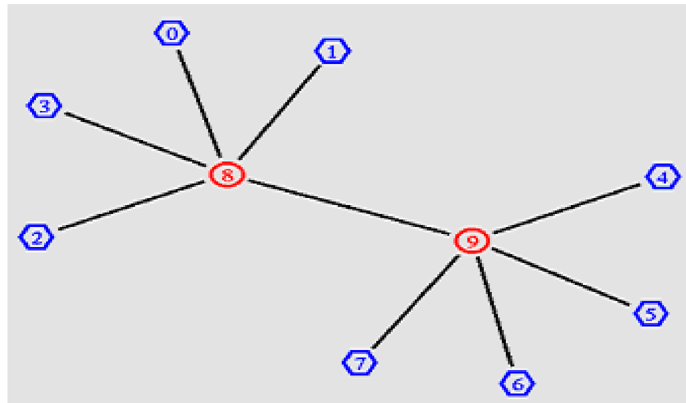

Fig: 7 Topology considered for simulation

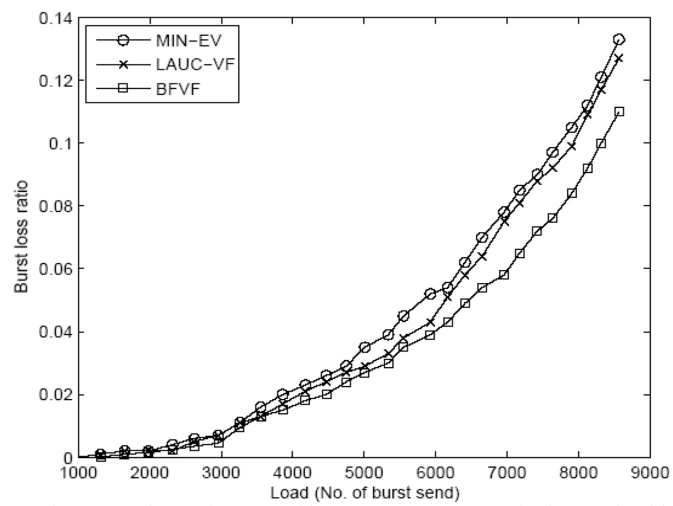

Fig: 8 Burst loss ratio vs. Load in BFVF, LAUC-VF and Min-EV algorithm

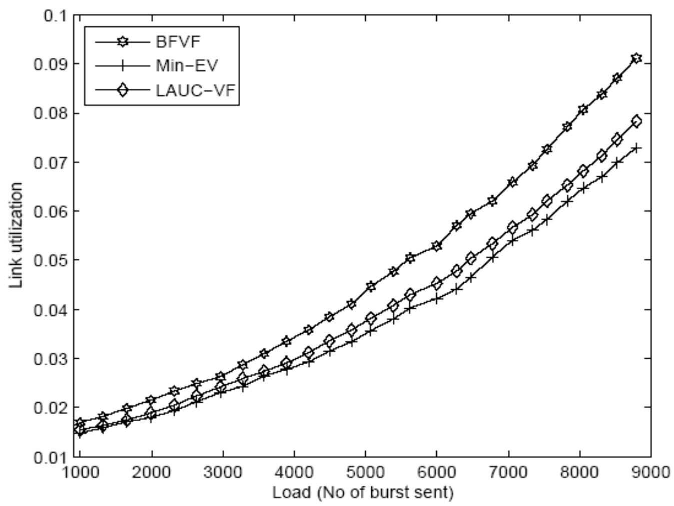

Fig: 9 Link utilization vs. Load in BFVF, LAUC-VF and Min-EV algorithm

REFERENCES

[1] K. Koduru, "New Contention Resolution Techniques for Optical Burst Switching," Master's thesis, Louisiana State University, May 2005.

[2] K. Dozer, C. Gauger, J. .Spath, and S. Bodamer, "Evaluation of Reservation Mechanisms for Optical Burst Switching," AEU International Journal of Electonics and Communications, vol. 55, no. 1, January 2001.

[3] M. Ljolje, R. Inkret, and B. Mikac, "A Comparative Analysis of Data Scheduling Algorithms in Optical Burst Switching Networks," in Proceeding of Optical Network Design and Modeling, 2005, 2005, pp. 493 – 500.

[4] Y. Xiong, M. Vandenhoute, and H. C. Cankaya, "Control Architecture in Optical Burst Switched WDM Networks," IEEE JSAC, vol. 18, no. 10, pp. 1838 – 1851, October 2000.

[5] M. Yoo, C. Qiao, and S. Dixit, "QoS Performance of Optical Burst Switching in IP-Over-WDM Networks," IEEE Journal on Selected Areas in Communications, vol. 18, no. 10, October 2000.

[6] W. M. Golab and R. Boutaba, "Resource Allocation in User-Controlled Circuit-Switched Optical Networks," LNCS Spinger-Verlag, vol. 16, no. 12, pp. 2081 – 2094, December 1998.

[7] M. Yang, S. Q. Zheng, and D. Verchere, "A QoS Supporting Scheduling Algorithm for Optical Burst Switching DWDM Networks," in Proceeding of GLOBECOM 01, 2001, pp. 86 – 91.

[8] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient Channel Scheduling Algorithms in Optical Burst Switching Networks," in Proceeding of IEEE INFOCOM, 2003, vol. 3, 2003, pp. 2268 – 2278.

[9] obs-ns Simulator. http://www.wine.icu.ac.kr/ obsns/.

[10] ns2 Simulator. http://www.isi.edu/nsnam/ns.