Maximizing Throughput for Optical Burst Switching Networks

Jikai Li, Chunming Qiao, Jinhui Xu, Dahai Xu
Department of Computer Science and Engineering
State University of New York at Buffalo
201 Bell Hall
Buffalo, NY 14260, USA
{jikaili,qiao,jinhui,dahaixu}@cse.buffalo.edu

Abstract— **A key problem in Optical Burst Switching (OBS) is to schedule as many bursts as possible on wavelength channels so that the throughput is maximized and the burst loss is minimized. Currently, most of the research on OBS (e.g., burst scheduling and assembly algorithms) has been concentrated on reducing burst loss in an "average-case" sense. Little effort has been devoted to understanding the worst-case performance. Since OBS itself is an open-loop control system, it may often exhibit a worst-case behavior when adversely synchronized, thus a poor worst-case performance can lead to an unacceptable system-wide performance. In this paper, we use competitive analysis to analyze the worst-case performance of a large set of scheduling algorithms, called best-effort online scheduling algorithms, for OBS networks, and establish a number of interesting upper and lower bounds on the performance of such algorithms. Our analysis shows that the performance of any best-effort online algorithm is closely related to a few factors, such as the range of offset time, burst length ratio, scheduling algorithm, and number of data channels. A surprising discovery is that the worst-case performance of any best-effort online scheduling algorithm is primarily determined by the maximum to minimum burst length ratio, followed by the range of offset time. Furthermore, if all bursts have the same burst length and offset time, all best-effort online scheduling algorithms generate the same optimal solution, regardless how different they may look like. Our analysis can also be extended to some non-best-effort online scheduling algorithms, such as the well-known Horizon algorithm, and establish similar bounds. Based on the analytic results, we give guidelines for several widely discussed OBS problems, including burst assembly, offset time setting and scheduling algorithm design, and propose a new channel reservation protocol called VFO to improve the worst-case performance. Our simulation shows that it is quite often for an online scheduling algorithm to exhibit its (near) worst-case performance. Thus improving the worst-case performance is essential. Our simulation also suggests that VFO reduces the average burst loss rate by as much as 35%.**

I. Introduction

To meet the increasing bandwidth demands and reduce costs, several optical network paradigms have been under intensive research. Of all these paradigms, optical circuit switching is relatively easy to implement but lacks flexibility to cope with the fluctuating traffic and the changing link status; Optical Packet Switching(OPS) is conceptually ideal, but the required optical technologies such as optical buffer and optical logic are too immature for it to happen anytime soon. An alternative approach called Optical Burst Switching (OBS) that combines the best of optical circuit switching and optical packet switching was proposed in [1] [2], and has received increasing amount of attention from both academia and industry worldwide [3]- [11].

In an OBS network, an ingress OBS node assembles client data units (e.g. IP packets) into bursts and sends out a corresponding control packet for each data burst. This control packet is delivered out-of-band and leads the data burst by an offset time o. The control packet carries, among other information, the offset time at the next hop, and the burst length l. At each intermediate node along the way from the ingress node to an egress node, the control packet reserves necessary resources (e.g., bandwidth on a desired output channel) for the following burst, which will be disassembled at the egress node.

A prevailing reservation protocol in OBS networks is called Just-Enough-Time (JET) [1]. In JET a control packet reserves an output wavelength channel for a period of time equal to the burst length l, starting at the expected burst arrival time r, which can be determined based on the offset time value and the amount of processing time the control packet has encountered at the node up to this point in time. If the reservation is successful, the control packet adjusts the offset time for the next hop, and is forwarded to the next hop; otherwise, the burst is blocked and will be discarded if there is no Fiber Delay Lines (FDLs).

Given the fact that OBS uses one-way reservation protocols such as JET, and that a burst can't be buffered at any intermediate node due to the lack of optical RAM (a FDL, if available at all, can only provide a limited delay and contention resolution capability), burst loss could be a serious problem for OBS networks, especially when the system is adversely synchronized and bursts repeatedly arrive in worst-case or near worst-case pattern. Thus, if the worst-case performance is too poor, an OBS network may not function well. An effective way to resolve this problem is to design scheduling systems with optimized worst-case performance.

To make this approach feasible, in this paper, we first consider the problem of determining the possible factors that could affect the worst-case performance of an online scheduling algorithm, and then use the obtained analytical results to design a new channel reservation protocol, called *Virtual Fixed Offset time (VFO)*, to achieve the desired performance. We particularly consider a large set of scheduling algorithms, called *best-effort online* scheduling algorithms, which includes most of the

well-known burst scheduling algorithms such as Latest Available Unused Channel with Void Filling (LAUC-VF), First Fit with Void Filling and Round Robin with Void Filling described in [3], Min-SV, Min-EV, Max-SV, Max-EV and Best-Fit described in [12] and the algorithm in [13]. By using competitive analysis, we establish a number of interesting upper and lower bounds on the worst-case performance of an arbitrary best-effort online scheduling algorithm. Our analysis reveals a surprising result showing that the worst-case performance is primarily determined by the burst length ratio and seconded by the range of burst offset time. Thus, if all the bursts have the same length and offset time, any best-effort online scheduling algorithm will give the optimal solution. Our simulation results on the worst-case performance of LAUC-VF agree with the phenomenon predicted by our theoretical analysis. The experimental comparison of our VFO protocol with JET suggests that VFO could reduce the burst loss rate of JET by as much as 35%.

The rest of this paper is organized as follows. Section II provides a more detailed description on how OBS networks work and what factors might cause data loss. Section III reviews some existing research on the scheduling problem in OBS networks, including various best-effort online scheduling algorithms and several theoretical results on a more general job interval selection problem. Section IV presents our analysis on the worst-case performance of best-effort online scheduling algorithms. A new channel reservation protocol VFO is proposed in Section V, and simulation results are presented in Section VI. Section VII concludes our work.

II. Problem Description

Given a data burst b_i, we assume that the control packet arrives o_i units of time (which is called the offset time) before the corresponding burst b_i arrives. In such a case, the reservation for the burst will not start at the current time (t), but at $r_i = o_i + t$ (i.e., when the burst actually arrives). If the burst's length is l_i, the reservation will be made until $f_i = r_i + l_i$. Because bursts may not arrive one after another continuously, each channel is likely to be fragmented with several reservation periods, separated by idle intervals (also called void). More specifically, each data channel initially corresponds to an open interval (considered to be a special case of a void interval) from time 0 to positive infinite. Let each void interval I_j be modelled as an ordered pair (s_j, e_j), where s_j and e_j are the starting end and ending time of the void interval I_j, respectively, with $e_j > s_j$. We say a void interval I_j is feasible to a data burst $b_i = (r_i, f_i)$, if and only if $s_j \leq r_i$, and $e_j \geq f_i$. Once the reservation is made using a feasible interval I_j, up to two new void intervals may be created, which are (s_j, r_i) and (f_i, e_j), respectively.

From the perspective of a scheduler in an OBS network, it is presented with a list of the bursts and processes them in the order of the arrival time of corresponding control packets. Let $B = (b_1, b_2 \ldots b_i \ldots b_n)$ be such an order. Generally speaking, all OBS scheduling algorithms are online, that is, when processing burst b_i, scheduler has the information of all previous bursts $b_1, b_2, \ldots, b_{i-1}$, but no knowledge of the following bursts $b_{i+1}, b_{i+2}, \ldots, b_n$. Scheduler uses such information to check whether there is a void feasible to b_i. If multiple voids can accommodate b_i, the scheduler assigns b_i to one of them according to certain criteria.

The research interest of this work is to analyze the worst-case performance of current OBS scheduling algorithms and to find out what factors affect the performance. Based on the analysis, we provide guidelines on how to enhance performance, and design a new reservation algorithm to implement the guidelines.

To better understand the performance of an online scheduling algorithm, we classify burst losses in OBS networks into two categories. In the first category, burst losses occur when the total number of simultaneously arriving bursts exceeds the number of wavelengths. This kind of burst losses will be inevitable (assume no FDL is available). To capture such a scenario, we use *overlapping degree* to denote the number of bursts overlapped in one link. More specifically, given a link l and time t, the overlapping degree d_l^t of l at time t is the number of bursts in l which contain t in their intervals (r_i, f_i). For a time period $T = [t_i, t_j]$, we define the maximum overlapping degree of link l as follows,

$$D_l^T = max_{t \in T}\{d_l^t\}.$$

The value of *overlapping degree* is directly related to burst loss. The larger the *overlapping degree* is, the more likely an incoming burst will be dropped.

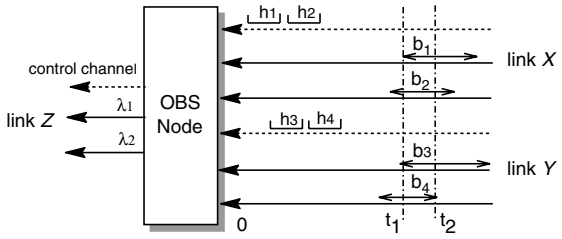
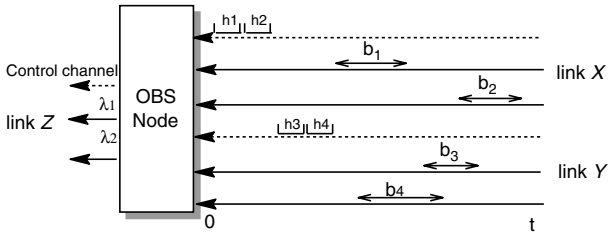


Fig. 1. an example of unavoidable burst losses

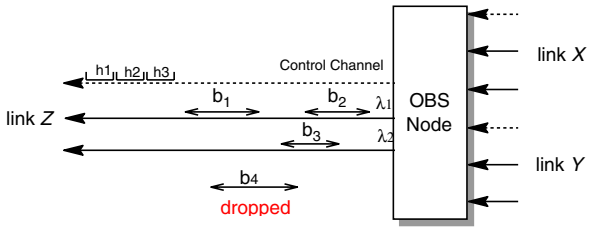
Fig.1 shows an example where LAUC-VF fails to schedule all bursts successfully. In this example, an OBS node has two incoming links and one outgoing link, and each link has two wavelengths for transmitting data and one for transmitting control packets. Assume four data bursts, b_1, b_2, b_3 and b_4, arrive from the incoming links and these four data bursts overlap with each other from time t_1 to time t_2 (i.e., $D_z^{(t_1,t_2)} = 4$). Two out of the four bursts will be dropped if no FDL is available at the OBS node. Such kind of burst loss happens because the overlapping degree is larger than the number of wavelength channels. The problem of how to reduce the overlapping degree in an OBS networks is out of the scope of this paper.

The second category of burst loss is caused by scheduling algorithms. Due to its online feature, a scheduling algorithm does not have the information on future incoming bursts, and thus could unwisely drop a burst which might be avoided if such information were available. Fig.2(a) shows an example, where the network settings are the same as the one in Fig. 1. Four bursts arrive in the order of b_1, b_4, b_3 and b_2, but their control packets arrive in the order of h_1, h_2, h_3 and h_4.

Assume that LAUC-VF is used to schedule the four bursts. The scheduler first reserves wavelength λ_1 for bursts b_1 and b_2 upon receiving their corresponding control packets h_1 and h_2. When control packet h_3 arrives, the scheduler has to reserve



(a) burst arrival pattern



(b) scheduling result

Fig. 2. an example showing the deficiency of an online algorithm

wavelength λ_2 for burst b_3. When control packet h_4 comes, all wavelengths have already been reserved. Thus, burst b_4 has to be dropped. Fig. 2(b) shows the scheduling result.

The reason for dropping b_4 is not due to lack of sufficient bandwidth. In fact, if b_1 and b_3 were assigned to wavelength λ_1, and b_2 and b_4 were assigned to wavelength λ_2, there would be no burst loss in this example. The real reason for this kind of burst losses is that when an online scheduling algorithm, such as LAUC-VF, schedules an incoming burst, it does not know about the following bursts, and thus might assign the burst to the wavelength appears to be the "best" for the time being, but such a scheduling maybe undesirable for future bursts.

III. Previous Work

So far, several scheduling algorithms have been proposed for OBS networks. Horizon [14] does not utilize any "closed" intervals, and thus is fast but not bandwidth efficient. On the other hand, LAUC-VF can schedule a burst in a closed interval (i.e., as long as it is possible) but has a much slower running time. To combine the advantages of these two algorithms, Xu *et. al.* [12] recently proposed a new algorithm called Min-SV which organizes voids by augmenting a load-balanced binary search tree. Similar idea was also extended to obtain a series of scheduling algorithms, including Min-EV, Max-SV, Max-EV, and Best Fit. Most of their algorithms have a loss rate as low as LAUC-VF, and run as almost fast as Horizon. Other scheduling algorithms include First Fit with Void Filling, Round Robin with Void Filling [3] and the algorithm presented in [13].

Although the above algorithms are very much different from each other, they (except Horizon) share the following two common features. First, they schedule bursts in an online fashion, that is, upon receiving a burst scheduling request, they either assign the corresponding burst to a wavelength or reject it before processing the next request. Second, they all use the best-effort strategy to schedule bursts. A burst will not be dropped if there exists a feasible void (which is not the case in Horizon). We call algorithms with these two features as *best-effort online* scheduling algorithms. We notice that although extensive simulation results are available on the performance of many best-effort online scheduling algorithms, there still lacks theoretical analysis on their worst-case performance. In this paper, we consider all best-effort online scheduling algorithms as a whole and analyze their worst-case performance. We assume that no preemption is allowed and bursts will not be broken up or segmented.

It is interesting to point out that although OBS is a new research area, the burst scheduling problem can actually be modelled as a special case of the Job Interval Selection Problem (JISP) [15] that has a rich research history. In the JIST problem, a set of jobs are provided as input to one or more machines. Each job includes a set of intervals on the time axis representing the possible time slots when the job can be executed. To schedule a job, one of the associated intervals is selected and assigned to the corresponding machine (no preemption or job breakup is allowed). For a set of jobs, the intervals assigned to each machine should not overlap. The objective is to schedule as many jobs as possible.

In the context of OBS networks, each burst b_i can be treated as a job and each wavelength as a machine. For OBS without FDLs, the scheduling problem is equivalent to the problem of finding maximum independent set in an interval graph [16]. Golumbic [16] showed this problem is in class P. Akin and Silverberg [17] gave an optimal offline algorithm, which runs in time $O(n^2 \log n)$, where n is the number of jobs. It was also shown that the problem is NP-Complete if each job has to be executed on a subset of machines. For the case that each job can only be executed on k machines, [17] gave an optimal *offline* algorithm running in $O(n^{k+1})$ time. Besides offline algorithms, there are also a few online algorithms. Long [18] used examples to show that no deterministic online algorithm can achieve good performance if there is no restrictions on jobs. Lipton [19] proposed a randomized online algorithm that is strongly 2-competitive for the case in which intervals must be one of two possible lengths. The authors of [20] [21] [22] gave several preemptive online scheduling algorithms. Despite all these efforts, the questions as to what factors affect, and to what extent they affect the worst-case performance of burst scheduling are still open.

IV. Worst-Case Analysis

In this section, we present our results on the worst-case analysis of best-effort online scheduling algorithms for OBS networks without FDLs. We start our discussion with notations and assumptions.

Let B be a burst set and b_i and b_j are two arbitrary bursts in B. We define a binary relation between b_i and b_j as follows.

$$b_i \preceq b_j \quad \text{if} \quad f_i \leq r_j,$$

where f_i is the finishing time of b_i and r_j is the starting (or ready) time of b_j. The relation is an *interval order* [23]. If for pair of bursts $b_i, b_j \in B$, we have either $b_i \preceq b_j$ or $b_j \preceq b_i$,

we say B is a linearly ordered set, or a *chain* [23]. It is easy to see that if B is a chain, all bursts in B can be assigned to one initially empty channel.

Given a burst set B, w data channels, and a scheduling algorithm A, a feasible scheduling of B includes w subsets $B_1^A, B_2^A, \cdots, B_w^A$ of B, where each subset B_i^A is a chain, and for any pair of subsets B_i^A and B_j^A $(i \neq j)$, $B_i^A \cap B_j^A = \emptyset$. We call $S_{B,w}^A = \{B_1^A, B_2^A, \cdots, B_w^A\}$ an *assignment* of B by algorithm A. $S_{B,w}^A$ is *extendable* (by burst b) if there exists a unscheduled burst $b \in B \backslash \bigcup_{i=1}^{w} B_i^A$ and a chain $B_i^A \in S_{B,w}^A$ such that $B_i^A \cup \{b\}$ forms a new chain. For a given assignment $S_{B,w}^A$, if no $b \in B \backslash \bigcup_{i=1}^{w} B_i^A$ can extend it, then $S_{B,w}^A$ is *maximal*.

Clearly, for any best-effort online algorithm A and any burst set B, the assignment $S_{B,w}^A$ must be maximal; otherwise, there would exist at least one unscheduled burst $b \in B \backslash \bigcup_{i=1}^{w} B_i^A$ such that b extends $S_{B,w}^A$, contradicting the definition of best-effort online algorithm.

For a given sequence of bursts B and an online scheduling algorithm A, let $A(B)$ and $OPT(B)$ be the number of bursts successfully scheduled by A and an *optimal* offline algorithm OPT, respectively. Notice that the optimal solution can be achieved by running an offline algorithm [17], and represents a performance upper bound for all online algorithms. To measure the quality of the scheduling by online algorithm A, we consider the ratio $R_A(B) \equiv OPT(B)/A(B)$. The competitive ratio R_A of the online algorithm A is defined as follows.

$$R_A \equiv inf\{r \geq 1 : R_A(B) \leq r \text{ for all sets } B\}.$$

This definition of competitive ratio is also called the absolute worst-case performance ratio of A [24].

In the above definition, we assume each burst has the same importance and this case will be referred to as the "unweighted case". One way to represent the importance of bursts is to associate with each burst a weight, such as the length of the burst.

In the above weighted case, we define the weight of an assignment $S_{B,w}^A$ of a burst set B as follows.

$$\begin{aligned} \text{weight}(S_{B,w}^A) &= \sum_{b \in \bigcup_{i=1}^{w} B_i^A} length(b) \\ &= \sum_{b \in \bigcup_{i=1}^{w} B_i^A} \|b\| \end{aligned}$$

Let $R'_A(B) \equiv \text{weight}(S_{B,w}^{OPT})/\text{weight}(S_{B,w}^A)$. The (weighted) competitive ratio R'_A of algorithm A is defined as follows.

$$R'_A \equiv inf\{r \geq 1 : R'_A(B) \leq r \text{ for all sets } B\}$$

To analyze the worst-case performance of a best-effort online algorithm A, we consider two important parameters, the burst length ratio and the range of offset time. Let $[s, t]$ be the range of burst lengths for an OBS node and $[u, v]$ be the range of burst offset time, where $t \geq s > 0$, and $v \geq u > 0$. We denote the burst length ratio Δ as the ratio of the longest burst length over the shortest burst length, i.e., $\Delta = \frac{t}{s}$, and β as the difference of v and u, i.e., $\beta = v - u$. Note that when $\beta = 0$, data bursts arrive in the same order as their control packets.

For a burst set B_i and a burst $b \notin B_i$, if b overlaps one or more bursts of B_i, we say b *overlaps* B_i. The subset of bursts in B_i overlapped by b is denoted by

$$B_i \oplus b = \{b' \in B_i, \text{ b overlaps b'}\}.$$

For a pair of burst sets B_i and B_j, the set of bursts in B_i overlapped by bursts in B_j is denoted by

$$B_i \odot B_j = \bigcup_{b \in B_j} B_i \oplus b,$$

and the set of bursts in B_i that are not overlapped by any burst in B_j is

$$\overline{B_i \odot B_j} = B_i \backslash (B_i \odot B_j).$$

Since a burst in B_i can be overlapped by several bursts in B_j, we have the following inequalities

$$|B_i \odot B_j| = \bigcup_{b \in B_j} |B_i \oplus b| \leq \sum_{b \in B_j} |B_i \oplus b|,$$

and

$$|\overline{B_i \odot B_j}| = |B_i| - |B_i \odot B_j| \geq |B_i| - \sum_{b \in B_j} |B_i \oplus b|.$$

Below we present our competitive analysis for best-effort online scheduling algorithms.

A. Unweighted Case

To simplify our presentation, in the following discussion we assume $\beta = \infty$ by default. The assumption $\beta = \infty$ implies that bursts can be presented to the scheduling algorithm in an arbitrary order. Our analysis can be easily applied to the cases where β has a different value.

The following is a simple but useful observation.

Observation 1: Given a chain B_i, a burst b can overlap at most $\lceil \Delta \rceil + 1$ bursts in B_i, i.e., $|B_i \oplus b| \leq \lceil \Delta \rceil + 1$.

Lemma 1: If $w = 1$, $R_A = \lceil \Delta \rceil + 1$ for any best-effort online scheduling algorithm A.

Proof: We first show that $R_A \geq \lceil \Delta \rceil + 1$ (Note that to prove the lower bound of R_A, it is sufficient to give an instance). Consider the example in Fig. 3. In this instance, we have $\lceil \Delta \rceil + 2$ bursts. One of them is long, and the rest are all short. The long burst overlaps all the short bursts and the set of short bursts forms a chain. Obviously, the optimal assignment is to select all short bursts. For a best-effort online algorithm A, if the longest burst is presented to A first, A will schedule it to the wavelength and discard all short bursts. Therefore, $R_A \geq \lceil \Delta \rceil + 1$ for all A.

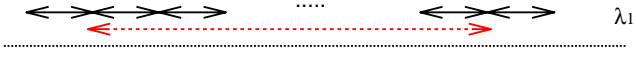


Fig. 3. R_A equals $\lceil \Delta \rceil + 1$

Next, we show $R_A \leq \lceil \Delta \rceil + 1$ by contradiction. For any burst set B, let $\{B_1^{OPT}\}$ be the optimal offline assignment, and $\{B_1^A\}$ be the assignment by algorithm A. Without loss of generality, we assume $OPT(B) = n$. Suppose $R_A > \lceil \Delta \rceil + 1$.

Then, $A(B) \leq \lceil \frac{n}{\lceil\Delta\rceil+1} \rceil - 1$. Since each burst in B_1^A can overlap at most $\lceil\Delta\rceil + 1$ bursts in B_1^{OPT}, the number of bursts in $\{B_1^{OPT}\}$ that do not overlap $\{B_1^A\}$ is

$$\begin{aligned}
|\overline{B_1^{OPT} \odot B_1^A}| &= |B_1^{OPT}| - |B_1^{OPT} \odot B_1^A| \\
&\geq |B_1^{OPT}| - \sum_{b \in B_1^A} |B_1^{OPT} \oplus b| \\
&\geq n - (\lceil \frac{n}{\lceil\Delta\rceil + 1} \rceil - 1) \times (\lceil\Delta\rceil + 1) \\
&\geq 1
\end{aligned}$$

This implies that $\overline{B_1^{OPT} \odot B_1^A}$ contains at least one burst which can extend $\{B_1^A\}$, contradicting the fact that $\{B_1^A\}$ is maximal. ■

***Lemma** 2:* For any best-effort online scheduling algorithm A and any integer $w \geq 1$, $R_A \geq \lceil\Delta\rceil + 1$.

Proof: To prove this lower bound, we consider the example in Fig. 4. In this example, we have two types of bursts, (a) $w(\lceil\Delta\rceil + 1)$ short bursts, and (b) w long bursts. The short bursts form w chains with each of $\lceil\Delta\rceil+1$ bursts, and each long burst overlaps all the short bursts. The offline optimal solution is to select all short bursts, i.e., $OPT(B) = w(\lceil\Delta\rceil + 1)$. If the control packets of long bursts arrive earlier than those of short bursts, the online algorithm will select the w long bursts as the solution, thus making $R_A(B) = \lceil\Delta\rceil + 1$.

Fig. 4. R_A larger than or equal to $\lceil\Delta\rceil + 1$

■

Remark: Although the above lemma indicates that it is possible to have the competitive ratio larger than $\lceil\Delta\rceil + 1$, according to our studies, it is rather difficult to come up with a burst set with $R_A(B) > \lceil\Delta\rceil + 1$.

***Lemma** 3:* For any best-effort online scheduling algorithm A and any integer $w \geq 1$, $R_A \leq \lceil\Delta\rceil + 2$.

Proof: For any burst set B, let $S_{B,w}^A = \{B_1^A, \cdots, B_w^A\}$ and $S_{B,w}^{OPT} = \{B_1^{OPT}, \cdots, B_w^{OPT}\}$ be the assignments obtained by A and an offline optimal algorithm, respectively. Assume $OPT(B) = n$. Suppose $R_A > \lceil\Delta\rceil + 2$. Then, $A(B) \leq \lceil \frac{n}{\lceil\Delta\rceil+2} \rceil - 1$.

For any $1 \leq i \leq w$, the burst set scheduled to channel i by the optimal algorithm but not by A is $\overline{B_i^{OPT} \odot B_i^A}$. Note that each burst b in $\overline{B_i^{OPT} \odot B_i^A}$ must belong to some B_j^A, $j \neq i$ and $1 \leq j \leq w$, otherwise $S_{B,w}^A$ would not be maximal (since $B_i^A \cup \{b\}$ forms a new chain). Thus, we have

$$\begin{aligned}
&\left| \bigcup_{i=1}^{i=w} \overline{B_i^{OPT} \odot B_i^A} \right| \\
&= \sum_{i=1}^{i=w} \left| \overline{B_i^{OPT} \odot B_i^A} \right| \\
&= n - \sum_{i=1}^{i=w} |B_i^{OPT} \odot B_i^A| \\
&\geq n - (\lceil\Delta\rceil + 1) \times (\lceil \frac{n}{\lceil\Delta\rceil + 2} \rceil - 1) \\
&\geq \lceil \frac{n}{\lceil\Delta\rceil + 2} \rceil
\end{aligned}$$

Since $\left| \bigcup_{i=1}^{i=w} \overline{B_i^{OPT} \odot B_i^A} \right| > A(B)$, there exists at least one burst $b \in \bigcup_{i=1}^{i=w} \overline{B_i^{OPT} \odot B_i^A}$ but not in any B_j^A, $1 \leq j \leq w$. This means b can extend assignment $S_{B,w}^A$. Since A is a best-effort online algorithm, $S_{B,w}^A$ is maximal and cannot be extended. Thus, a contradiction. ■

Combining the above lemmas, we have the following theorem.

***Theorem** 1:* For any best-effort online scheduling algorithm A, $\lceil\Delta\rceil + 1 \leq R_A \leq \lceil\Delta\rceil + 2$.

The above results hold for the case $\beta = \infty$. For different β values, we can use similar argument to obtain the following theorem. Details of the proof are omitted from this paper.

***Theorem** 2:* For any best-effort online scheduling algorithm A and any β value, we have $\lceil\Delta\rceil \leq R_A \leq \lceil\Delta\rceil + 2$.

As an extension of our analysis, we also applied our technique to Horizon algorithm, which is not a best-effort online algorithm, and obtained the following theorem. Details of the proof are left for the full paper.

***Theorem** 3:* For Horizon algorithm, $\lceil\Delta + \frac{\beta}{s}\rceil \leq R_A \leq \lceil\Delta + \frac{\beta}{s}\rceil + 2$, where s is the shortest burst length.

B. Weighted Case

For any pair of bursts b_i and b_j, if $r_i < r_j < f_i$, we say burst b_i *leads* burst b_j; if $r_i < f_j < f_i$, we say burst b_i *tails* burst b_j; if $r_i \leq r_j < f_j \leq f_i$, we say burst b_i *contains* burst b_j.

***Lemma** 4:* Given a chain B and a burst b, if $\Delta > 1$, then $\text{weight}(B \oplus b) \leq (2\Delta + 1) \times \|b\|$.

Proof: It is easy to see that there is at most one burst b_l in chain B leads b and at most one burst b_t in chain B tails b. There might be one or more bursts in B contained by b. Let $b_1, b_2, \cdots, b_c$ be the set of contained bursts. Clearly, their total length is less than or equal to $\|b\|$. Thus, we have the following inequality.

$$\begin{aligned}
\text{weight}(B \oplus b) &= \|b_l\| + \sum_{i=1}^{i=c} \|b_i\| + \|b_t\| \\
&\leq \Delta\|b\| + \|b\| + \Delta\|b\| \\
&= (2\Delta + 1) \times \|b\|
\end{aligned}$$

■

***Lemma** 5:* For any best-effort online scheduling algorithm A and any $\Delta > 1$, $R'_A \geq 2\Delta + 1 - \epsilon$, where ϵ is a positive constant arbitrarily close to 0.

Proof: We prove this lower bound by giving an instance. Consider Fig. 5. In this example, the burst set B contains two

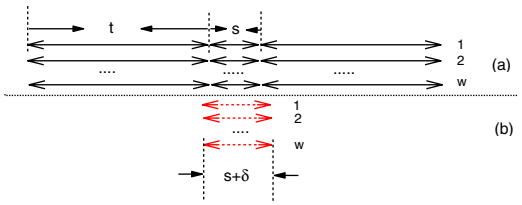


Fig. 5. R'_A larger than or equal to $2\Delta + 1 - \epsilon$

types of bursts, (a) $3w$ bursts of length either s or t (i.e., the shortest or longest burst length), (b) w bursts of length $s + \delta$, where $0 < \delta \leq t - s$. The type (a) bursts forms w chains with each of three bursts. Each type (b) burst overlaps every type (a) burst.

Obviously, the optimal assignment selects all type (a) bursts and drops all type (b) bursts. However, for a best-effort online algorithm A, if type (b) bursts are scheduled first, then all type (b) bursts will be selected as the assignment which leaves no room for selecting any type (a) burst. Thus we have

$$\begin{aligned} R'_A(B) &= \frac{w \times (2t + s)}{w \times (s + \delta)} \\ &= \frac{\frac{2t}{s} + \frac{s}{s}}{\frac{s}{s} + \frac{\delta}{s}} \\ &= \frac{2\Delta + 1}{1 + \frac{\delta}{s}} \end{aligned}$$

When δ approaches 0, the ratio approaches $(2\Delta + 1)^-$, i.e.,

$$\lim_{\delta \to 0^+} R'_A = (2\Delta + 1)^-$$

■

***Lemma** 6:* For any best-effort online scheduling algorithm A, $R'_A \leq 2\Delta + 2$.

Proof: For any burst set B, let $S^{OPT}_{B,w} = \{B^{OPT}_1, \cdots, B^{OPT}_w\}$ and $S^A_{B,w} = \{B^A_1, \cdots, B^A_w\}$ be the assignments by an optimal online algorithm OPT and algorithm A, respectively. Assume $weight(S^{OPT}_{B,w}) = n$. Suppose $R'_A > 2\Delta + 2$. Then $weight(S^A_{B,w}) < \frac{n}{2\Delta+2}$.

For each channel i, the set of bursts assigned to channel i by OPT but not by A is $\overline{B^{OPT}_i \odot B^A_i}$. If a burst b in $\overline{B^{OPT}_i \odot B^A_i}$ were not in any B^A_j, $1 \leq j \leq w$, then $S^A_{B,w}$ would be extended by b and thus could not be maximal, a contradiction. Thus each burst in $\overline{B^{OPT}_i \odot B^A_i}$ must belong to some $B^A_j, j \neq i$. Summing over all channels, we have

$$\begin{aligned} &\text{weight}(\bigcup_{i=1}^{i=w} \overline{B^{OPT}_i \odot B^A_i}) \\ &= \sum_{i=1}^{i=w} \text{weight}(\overline{B^{OPT}_i \odot B^A_i}) \\ &= n - \sum_{i=1}^{i=w} \text{weight}(B^{OPT}_i \odot B^A_i) \\ &> n - (2\Delta + 1) \times (\frac{n}{2\Delta + 2}) \\ &= \frac{n}{2\Delta + 2} \end{aligned}$$

This means weight$(\bigcup_{i=1}^{i=w} \overline{B^{OPT}_i \odot B^A_i}) > weight(S^A_{B,w})$. Thus there must be at least one burst, say b, in $\bigcup_{i=1}^{i=w} \overline{B^{OPT}_i \odot B^A_i}$ which extends $S^A_{B,w}$. This contradicts the fact that $S^A_{B,w}$ is maximal. Hence, $R'_A \leq 2\Delta + 2$ ■

Based on the above discussion, we have the following theorem.

***Theorem** 4:* For any best-effort online scheduling algorithm A and $\Delta > 1$, $2\Delta + 1 - \epsilon \leq R'_A \leq 2\Delta + 2$, where ϵ is a positive constant arbitrarily close to 0.

Our techniques can be easily applied to the case where $\Delta = 1$, and obtain the following theorem. Details are left for the full paper.

***Theorem** 5:* For any best-effort online scheduling algorithm A and $\Delta = 1$, $2\Delta \leq R'_A \leq 2\Delta + 1$.

For an arbitrary β value, we have the following theorems.

***Theorem** 6:* For any best-effort online scheduling algorithm A and any β value, if $\Delta > 1$, then $\Delta + 1 - \epsilon \leq R'_A \leq 2\Delta + 2$, where ϵ is a positive constant arbitrarily close to 0.

***Theorem** 7:* For any best-effort online scheduling algorithm A and any β value, if $\Delta = 1$, $\Delta \leq R'_A \leq 2\Delta + 1$.

For Horizon algorithm, we have the following theorem.

***Theorem** 8:* For Horizon algorithm, $\Delta + \frac{\beta}{s} + 1 \leq R'_A \leq 2\Delta + \frac{\beta}{s} + 2$, where s is the shortest burst length.

C. *Worst-Case Analysis for* $\Delta = 1$ *and* $\beta = 0$

The analytic results in the unweighted and weighted cases indicate that burst length ratio Δ is the most influential factor for the worst-case performance of any best-effort online algorithm. The larger Δ is, the worse performance an online algorithm may have. Thus, to achieve optimal worst-case performance using best-effort online scheduling algorithm, all bursts should have the same length, that is, $\Delta = 1$.

Another factor which affects the worst-case performance is β. The larger the value of β, the more likely bursts will arrive out of order. To better understand the influence of β, we consider the case $\beta = 0$ and $\Delta = 1$. In this case, all bursts arrive in the same order as their control packets.

For this special case, we first consider a known algorithm GOL ((Greedy On-line Algorithm) proposed by Faigle and and Nawijn [20]. GOL is an online preemptive algorithm, and schedules the bursts in the order of their arrival time, i.e., in

the order of r_i. In general, it is not a best-effort online algorithm since it may preempt bursts. When $\beta = 0$ and $\Delta = 1$, GOL becomes a best-effort online algorithm.

GOL contains the following main steps. Assume data bursts arrive in the order of $b_1, b_2, \cdots, b_m$.

1) Assign b_1 to an arbitrary channel.
2) For burst b_i, try to Assign it to any free channel. If no channel can accommodate b_i, find a scheduled burst b with the largest finishing time f.
   a) If $f_i \geq f$, then discard b_i.
   b) If $f_i < f$, then replace b by b_i.

For unweighted case, the following theorem have been proved in [20].

***Theorem** 9:* (Faigle and Nawijn) Algorithm GOL minimizes the number of losses.

For the case $\beta = 0$ and $\Delta = 1$, we have the following lemma.

***Lemma** 7:* When $\beta = 0$ and $\Delta = 1$, if any best-effort online algorithms A drops a burst b, then all other best-effort online algorithms drop b.

Proof: We prove this lemma by contradiction. Suppose this lemma does not hold for burst set $B = \{b_1, b_2, \cdots, b_m\}$ with $r_1 \leq r_2 \leq \cdots \leq r_m$. Then there must exist some bursts which are dropped by A but not by some other best-effort online algorithm, say A'. Let b_i be such a burst with the smallest index.

Let $B_{i-1} = \{b_1, b_2, \cdots, b_{i-1}\}$ be the set of bursts that have already been scheduled by A and A', and $S^A_{B_{i-1},w}$ and $S^{A'}_{B_{i-1},w}$ be their assignments, respectively. If A drops b_i, then each B^A_j, $1 \leq j \leq w$, has exactly one burst overlapping b_i. Therefore, the number of bursts in $S^A_{B_{i-1},w}$ overlapped by b_i is w. Since b_i is the first burst dropped by A but not by A', $S^A_{B_{i-1},w}$ and $S^{A'}_{B_{i-1},w}$ have the same set of bursts (although they may be scheduled to different channels by A and A'). Thus after b_i is scheduled by A', the overlapping degree becomes $w+1$, which is impossible to form a feasible scheduling with only w channels. ■

The following theorem holds for both unweighted and weighted cases. (Note that when $\Delta = 1$, the two cases are equivalent.)

***Theorem** 10:* If $\beta = 0$ and $\Delta = 1$, all best-effort online algorithms have the same optimal scheduling.

V. New Reservation Protocol

From previous discussion, we know that the worst performance is strongly related to Δ and β. Theorem 10 indicates that to achieve the best worst-case performance, we should have $\Delta = 1$ and $\beta = 0$.

In general, it is relatively easy to make all bursts have the same length. But it is not trivial to have all bursts with the same offset time. In a network using the JET protocol, the offset time of a burst changes all the way from the ingress node to the egress node to take into account the processing delay encountered by its corresponding control packet . The JET protocol itself cannot support a fixed offset time value for a given burst, not to mention for all bursts.

In order to simplify signaling protocol design and system implementation, Xu *et al.* [25] proposed a protocol called Only Destination Delay (ODD), which associates with each burst a constant offset time and uses FDLs at each intermediate node to delay the burst by a time period equal to the exact processing time. Two major difficulties make this scheme hard to implement. First, the exact processing time of an incoming burst is in general not a constant, which depends on the status of the scheduling algorithm and the traffic load. Therefore, it is difficult to obtain the precise delay time. Second, for a link with w data channels, if w bursts arrive simultaneously, w control channels are needed to make sure that each burst has the same offset time. This means that in order to support a constant offset time, the number of control channels should be equal to the number of data channels, which is an impractical requirement.

Instead of assigning a constant offset time to each burst, we propose a new channel reservation protocol, called *Virtual Fixed Offset time (VFO)*, which still uses variable offset times but mimics the behavior of having bursts with a constant offset time. We first introduce the main idea of our protocol and then give a detailed description.

The basic idea of VFO is from the following observation. When bursts have a constant offset time, they will be processed in the order of their arrival time. Therefore, unlike other existing OBS reservation algorithms, VFO does not process bursts in the order of the arrival time of control packets. Instead, VFO schedules bursts in the order of burst arrival time. Upon receiving a control packet, VFO does not schedule the corresponding burst immediately. First, it sorts the arriving control packets according to their burst arrival times. Then, it schedules the burst with the earliest arrival time. Obviously, if VFO can schedule bursts *strictly* according to their arrival times, the scheduling will be the same as in the case where all bursts have a constant offset time.

An immediate question about the above idea is how to schedule bursts strictly according to their arrival times, or in other words, how to ensure that no burst will arrive before the burst currently being scheduled. The answer is that if a burst has the smallest offset time u, clearly it need not be delayed and should be processed immediately; if the burst has the largest offset time v, it has to be delayed by a time period $\beta = v - u$ to guarantee that no other bursts will arrive before it. Thus, to make this scheme feasible, we need to know β or equivalently the u and v values.

However, it is practically difficult to determine the exact u value. This difficulty is due to the fact that in practice, the number of control channels is much smaller than the number of data channels. Under such an assumption, when multiple bursts (or more accurately multiple control packets) arrive at the same time, some control packets have to be electronically buffered before being processed. Each time when a control packet is buffered, the offset time of the corresponding burst is decreased. This means that the u value of a burst will keep decreasing. In an extreme case, u may become 0 and $\beta = v$. This leaves the scheduler no time to schedule the bursts arriving at the same time as their control packets.

Thus in order to schedule bursts according to their arrival times, sorting bursts by their arrival times and delaying their scheduling are not sufficient. This is mainly due to the difficulty in determining the exact delay of a control packet at a node. To overcome this obstacle, we use FDL to delay bursts at each node and intentionally increase their offset time. Note that

FDLs are used here for the purpose of increasing burst offset time rather than resolving burst contention. To avoid having a very long delay time, it is necessary to keep the β value as small as possible.

We notice that the idea of sorting bursts by their arrival time and delaying their scheduling has previously been used by Chen and Turner in their Horizon algorithm with reordering [26]. In their algorithm, each burst b_i is simply scheduled at time $r_i - \zeta$ for some fixed constant ζ to emulate the case where every burst has a constant offset time. After scheduling, each burst has the same offset time ζ, thus facing similar difficulties as ODD. Furthermore, since bursts have different destinations, they need different amount of offset times to complete the scheduling at all intermediate nodes on the way to their destinations. Thus, it is unlikely to find a fixed ζ satisfying all bursts.

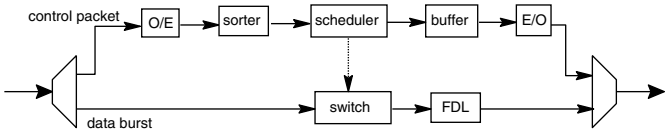


Fig. 6. core node architecture for VFO

Below we give a more detailed description to our approach.

Fig. 6 illustrates the architecture of VFO at each OBS node. For each control packet h_i, it takes five steps to finish the scheduling. First, it goes through an O/E conversion, and then is forwarded to a sorter where h_i is sorted (along with other accumulated control packets) by the arrival time of its corresponding burst. At certain time point, the scheduler removes a control packet from the sorter and assigns a feasible wavelength to its corresponding burst, if any, and sets up the switch accordingly. If the scheduling fails, the control packet as well as the corresponding burst is destroyed; otherwise, it is sent to an electronic buffer even if the control channel is immediately available. Once certain timing constraint (to be discussed) is satisfied, the control packet is re-activated from the electronic buffer and sent to the E/O conversion. Each successfully scheduled data burst will be delayed by an FDL by a fixed amount of time T.

The remaining difficulty of our scheme is to determine the timing for each step. Since the offset time of a burst changes during the whole process, to find the exact timing, we need to keep tracking the offset time of each burst. For this purpose, we stick a time stamp to each control packet h_i once it is converted from optical domain to electronic domain (assume the offset time of h_i at this moment is o_i). The current offset time of h_i can then be computed by the following equation.

$$\text{offset time} = \text{time stamp} - \text{current clock time} + o_i$$

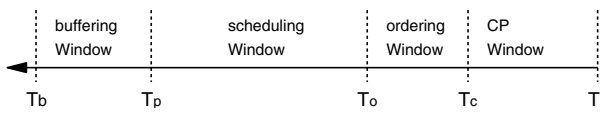


Fig. 7. the phases a control packet will go through in VFO

Fig. 7 shows how the offset time of a control packet h_i changes in VFO as it goes through different phases. We define five particular values, T_b, T_p, T_o, T_c and T to divide the life cycle of h_i (in VFO) into four different phases with each closely related to a step. The offset time of h_i may reduce from T to T_b, depending on the status of h_i. When h_i is in the optical domain, the offset time will always be in the range (T_c, T), where T_c and T are two constants. Setting a fixed offset time range for all bursts provides us two kinds of benefits. First, it provides more flexibility to deal with the contention caused by simultaneously arriving control packets competing for the control channel. In such a scenario, the scheduler can still send out the control packets by assigning them with different offset time. Second, with fixed T_c and T, it enables us to determine the exact delay for each control packet so that no burst will be scheduled out of order. We call the range (T_c, T) a CP Window (or control packet window).

When h_i is in the sorter, VFO requires that its offset time be within the range (T_o, T_c), where $T_c - T_o$ is larger than the maximum amount of time needed for sorting a single control packet (i.e., inserting h_i to a sorted list of control packets). Thus the sorting operation should be done before the offset time of h_i decreases to T_o. Once the sorting is finished, the control packet will not be immediately moved to the scheduler. Instead, it will stay in the sorter until its offset time becomes T_o. Since the offset time of each control packet in the optical domain is within the range (T_c, T) and sorting is done before the offset time decreases to T_o, the control packets will be scheduled by the arrival time of their corresponding data bursts.

In the scheduling step, if burst b_i is successfully scheduled on some wavelength, the control packet h_i will be buffered in electronic RAM until the control channel is free and the offset time of h_i is within the range $(T_b, 0)$, where T_b is a negative value equal to $T_c - T$. Note that, although the offset time of h_i is negative, it does not mean that we should discard burst b_i, since b_i is currently delayed by an FDL for time T. Once h_i is converted to optical domain, the offset time will again be in the range (T_c, T).

So far, we have showed how VFO works without giving the values to the five parameters, T_b, T_p, T_o, T_c and T. In general, these values depend on the implementation of future OBS networks, and thus it is unrealistic for us to give the exact values at this moment. Instead, we provide some guidance below on how to determine these values.

T_p is the offset time when the scheduling step ends. It should be positive to allow the data bursts be switched to the assigned channels. Since VFO dose not use T_p explicitly, the choice for T_p is flexible. T_o is the longest time needed by the scheduler to schedule one burst, and can be obtained from the hardware specification of the scheduler. T_c is the sum of T_o and the longest time needed for sorting a control packet. Thus it can also be obtained from the hardware specification of the sorter. T depends on T_c and the requirement on the length of (T_c, T). The length of (T_c, T) should be long enough to accommodate all the control packets whose corresponding bursts arrive simultaneously. In an OBS network with w data channels, w bursts may arrive at the same time. Thus the length should be at least $\frac{w \times e}{d}$, where e is the control packet duration and d is the number of control channels. As for T_b, since $T_b = T_c - T$, once T_c and T are determined, T_b can be easily computed.

VI. Experimental Results

In this section, we present our simulation results on our competitive analysis and VFO protocol. The simulation includes two parts. The first part simulates the competitive performance of LAUC-VF, where LAUC-VF is selected as the representative of best-effort online scheduling algorithms and AS algorithm [17] is selected as the representative of optimal offline scheduling algorithms. The second part of our simulation compares the performance of VFO with that of JET.

A. Competitive Performance of LAUC-VF

In this simulation, we assume that the observed OBS node has two incoming links injecting randomly generated bursts and one outgoing link. Both burst length and burst interarrival time follow Pareto distribution, and the offset time of bursts follows a uniform distribution in range $[0.1ms, 1ms]$. The average burst duration is $1ms$. There are 10 data channels on each link. For each setting (e.g., a fixed Δ value and offered link load), we generate $10,000$ sets of bursts, each consisting of 150 data bursts. For each burst set B, we count the number of successfully scheduled bursts by LAUC-VF and AS, and compute its $R_A(B)$ value. Of the $10,000$ sets of bursts, we pick the set with the largest $R_A(B)$ value. Then, we select the largest $R_A(B)$ instead of the mean to observe the worst-case performance.

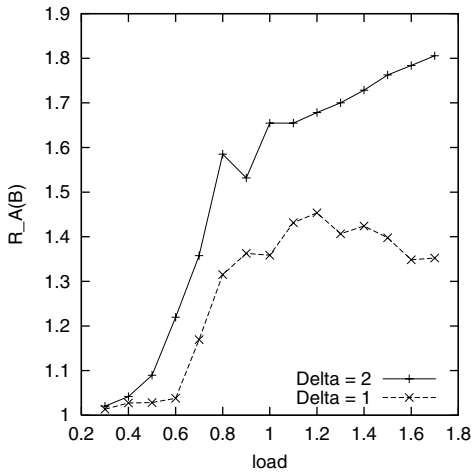


Fig. 8. competitive performance of LAUC-VF

Fig.8 shows how $R_A(B)$ changes under different Δ values and offered link load. The result indicates that $R_A(B)$ tends to increase with the offered link load. We attribute this to the fact that when the link load is low, the bursts are widely distributed and the overlapping degree is small. Thus most of the bursts will be successfully scheduled by LAUC-VF, and consequently makes $R_A(B)$ small. When the link load increases, the bursts are more likely to overlap. Further, with a heavier link load, LAUC-VF has a higher probability to schedule bursts to the wrong channels, thus resulting in a larger $R_A(B)$ value.

Fig.8 also indicates that the $R_A(B)$ for $\Delta = 2$ is larger than that for $\Delta = 1$. This is consistent with our theoretical bounds on R_A. Part of the reason is because when Δ is larger, a wrongly selected burst may block more bursts.

This simulation also suggests that a best-effort online scheduling algorithm like LAUC-VF can indeed encounter some bad (if not the worst) instances and produces very poor scheduling results. For example, Fig.8 shows that the scheduling results can be 80% worse than the optimal solution. Thus it is important for OBS networks to provide a mechanism (such as setting Δ to be 1 and using better channel reservation protocol like VFO) to eliminate bad or worst behavior.

B. VFO vs. JET

To compare the performance of VFO and JET, we design two simulations. The first is based on the same single-node topology used in previous simulation, and the second is based on the NSFNET network. In both simulations, we assume that the burst inter-arrival time follow Pareto distribution. We also assume that there are 10 data channels on each link. For a given offered link load, bursts are generated in the same order for different protocols. All simulations use LAUC-VF to schedule bursts. Our simulations focus on examining the burst loss rate under different channel reservation algorithms.

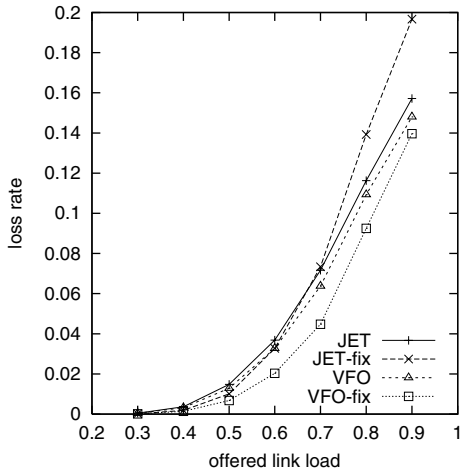


Fig. 9. average loss rate of JET and VFO

Fig.9 shows the loss rate of JET and VFO under the simpler topology. In order to determine how burst length affects the performance, we generate two types of bursts: (1) bursts with a fixed length $1ms$ and (2) bursts with variable lengths (which follows Pareto distribution), with the same average burst length $1ms$. We use VFO-fix and JET-fix to denote the performance of VFO and JET for type (1) bursts, and VFO and JET for type (2) bursts. The figure shows that the loss rates of both VFO and JET are rapidly increasing functions of the offered link load. In both cases, VFO consistently outperforms JET. Particularly, VFO reduces the loss rate of JET by as much as 35% for type (1) bursts and 10% for type (2) bursts. As predicted by our theoretical analysis, VFO-fix has the best performance among all cases. This suggests that to achieve the best performance, we should have all bursts with the same length (i.e., $\Delta = 1$) and schedule bursts in the order of their arrival times.

To compare the performance of JET and VFO in a more complicated network setting, we simulate JET, JET-fix, VFO and

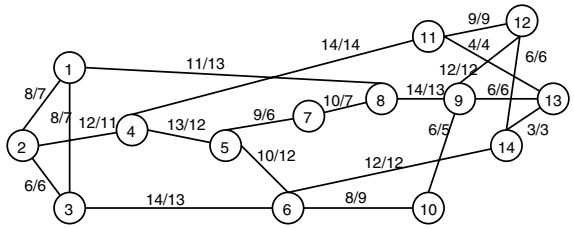


Fig. 10. number of paths through each link in the NSFNET

VFO-fix in the 14-node NSFNET shown in Fig.10. In this network, each link has 10 data channels. We assume each node is both an ingress node and a core node. There is one OBS path from each node to every other node. The two OBS paths going opposite directions between a pair of nodes may be routed asymmetrically when there are two or more shortest paths between that pair. The OBS paths used in our simulation are computed by using a shortest path algorithm. The number of OBS paths passing through each link (in opposite directions) are shown in Fig.10. Each node generates the same average amount of traffic which is a fraction of the link capacity. In our simulation, this average amount varies from 0.3 to 0.9. The traffic generated by each node is randomly distributed among all other nodes.

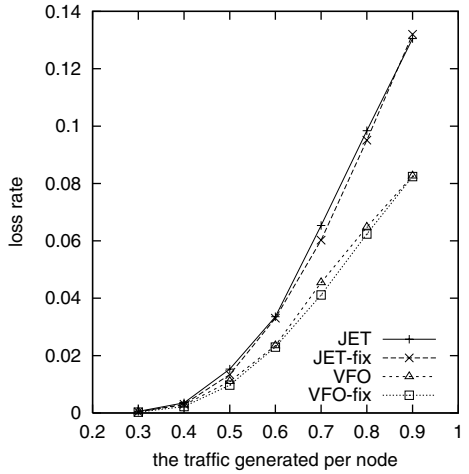


Fig. 11. average loss rate of VFO and JET in NSFNET network

Fig.11 shows the simulation results for VFO and JET. The relative performances of VFO and JET observed here are almost the same as these in Fig.9. The only difference is that the performance difference between VFO and VFO-fix or JET and JET-fix is significant reduced. Part of the reason may lie in the fact that each node in this network is an ingress node as well as a core node. The locally generated bursts are injected into the network whenever there is a feasible void in the outgoing links, thus reducing the benefits of having fixed-length bursts over having variable-length bursts.

VII. Conclusion

In this paper, we have given competitive analysis of a large set of online scheduling algorithms, called best-effort online scheduling algorithms, which include most of the existing scheduling algorithms such as LAUC-VF, and established a number of non-trivial lower and upper bounds on the competitive performance of all best-effort online scheduling algorithms. For OBS networks without FDLs, we have shown that the worst-case performance of an arbitrary best-effort online scheduling algorithm is closely related to a few factors, such as the range of offset time, the burst length ratio, scheduling strategy, and the number of data channels. Among all these factors, the burst length ratio is the dominating factor and followed by the range of burst offset time. Our analytic results suggest that to achieve the best worst-case performance, all bursts should have the same length and bursts should be scheduled strictly according to their arrival time. Based on our analysis, we proposed a new channel reservation protocol, called *Virtual Fixed Offset time (VFO)*, to improve the worst-case performance of online scheduling algorithms. Our simulations have shown that VFO reduces the loss rate of JET by as much as 35%.

As an extension, we have also applied our analysis to non-best-effort online scheduling algorithms, such as Horizon, and obtained similar analytic results.

References

[1] M. Yoo, M. Jeong and C. Qiao, "A high speed protocol for bursty traffic in optical networks," *SPIE's All-Optical Communication Systems:Architecture, Control and Protocol Issues*, vol. 3230, pp. 79–90, Nov, 1997.

[2] C. Qiao and M. Yoo, "Optical burst switching(OBS)-a new paradigm for an optical Internet," *Journal High Speed Networks*, vol. 8, pp. 69–84, 1999.

[3] Y. Xiong, M. Vandenhoute, and H.C. Cankaya, "Control architecture in optical burst-switched wdm networks," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1838–1851, 2000.

[4] J.Turner, "Terabit burst switching," *Journal High Speed Networks*, vol. 8, pp. 3–16, 1999.

[5] L. Xu, H. Perros, and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Communications Magazine*, vol. 39,no.1, pp. 136–142, Jan. 2001.

[6] A. Detti and M. Listanti, "Impact of segments aggregation on tcp reno flows in optical burst switching networks," in *IEEE Infocom 2002*, pp. 1803–1812.

[7] C. Hsu, T. Liu, and N. Huang, "Performance analysis of deflection routing in optical burst-switching networks," in *IEEE Infocom 2002*, pp. 66–73.

[8] C. Chang, D. Lee, C. Tu , "Using Switched Delay Lines for Exact Emulation of FIFO Multiplexers with Variable Length Bursts," in *IEEE Infocom 2003*, San Francisco, April, 2003

[9] Z. Rosberg, H. Vu, M. Zukerman and J. White, "Blocking Probabilities of Optical Burst Switching Networks Based on Reduced Load Fixed Point Approximations," in *IEEE Infocom 2003*,San Francisco, April, 2003

[10] L. Xu, H. Perros and G. Rouskas, "A Queueing Network Model of an Edge Optical Burst Switching Node," in *IEEE Infocom 2003*,San Francisco, April, 2003

[11] J. Ramamirtham and J. Turner "Time Sliced Optical Burst Switching," in *IEEE Infocom 2003*,San Francisco, April, 2003

[12] J. Xu, C. Qiao, J. Li and G. Xu, "Efficient Channel Scheduling Algorithms in Optical Burst Switched Networks" *IEEE Infocom 2003*, San Francisco, April, 2003

[13] Masanori Iizuka, Makoto Sakuta, Yoshiyuki Nishino and Iwao Sasase, "Labeled optical burst switching for ip-over-wdm integration," in *Proceedings of IEEE Globecom 2002*, Taipei

[14] J. Turner, "Terabit burst switching progress report (9/98-12/98)," in *Washington University at St.Louis Technical Report*, 1998.

[15] J. Chuzhoy, R. Ostrovsky and Y. Rabani, "Approximation Algorithms for the Job Interval Selection Problem and Related Scheduling Problems," in *IEEE Symposium on Foundations of Computer Science(FOCS)*, pp.348-356, 2001

[16] M. C. Golumbic, "Algorithmic Graph Theory and Perfect Graphs." Academic Press, 1980

[17] E. M. Arkin and E. B. Silverberg, "Scheduling jobs with fixed start and end times," in *Discrete Applied Mathematics*, Vol.18, pp. 1-8, 1987

[18] D. Long and M. Thankur, "Scheduling real-time disk transfers for continuous media applications," in *Twelfth IEEE Symposium on Mass Storage Systems*, pp. 227-232, April, 1993
[19] R. J. Lipton and A. Tomkins, "Online Interval Scheduling," in *Proc. 5th Ann. ACM-SIAM Symp. On Discrete Algorithms(SODA)*, pp.302-311, 1994
[20] U. Faigle and W.M. Nawijn. "Note on Scheduling intervals on-line," in *Discrete Applied Mathematics*, 58:13-17,1995
[21] G. Koren and D. Shasha. "An Optimal on-line scheduling algorithm for overloaded real-time systems," in *SIAM Journal on Computing*, 24:318-339,1995
[22] S. Baruah, G. Koren, B. Mishra, A. Raghunathan, L. Rosier and D. Shasha "On-line Scheduling in the Presence of Overload," in *IEEE Symposium on Foundations of Computer Science(FOCS)*, pp.100-110, 1991
[23] Peter C. Fishburn, "Interval Orders and Interval Graphs," John Wiley & Sons, 1985, ISBN 0-471-81284-6
[24] Dorit S. Hochbaum, et al. "Approximation Algorithms for NP-hard Problems," PWS Publishing Company, 1995, ISBN 0-534-94968-1
[25] L. Xu, H. G. Perros and G. N. Rouskas "A Simulation Study of Access Protocols for Optical Burst-Switched Ring Networks," *Computer Networks*, 42:143-160, 2003
[26] Y. Chen, J. Turner "WDM Burst Switching for Petabit Capacity Routers," *Proceeding of Milcom*, 1999