Performance Analysis of an Edge Optical Burst Switching Node with A Large Number of Wavelengths *

Lisong Xu, Harry G. Perros, George N. Rouskas
Department of Computer Science, North Carolina State University, Raleigh, NC 27695-7534, USA
Email: {lxu2,hp,rouskas}@csc.ncsu.edu

We consider an edge optical burst switching (OBS) node with or without converters, and with no buffering. The OBS node serves a number of users, each connected to the switch with a fiber link that supports multiple wavelengths. Each wavelength is associated with a 3-state Markovian burst arrival process which permits short and long bursts to be modeled. In a previous paper [18], we modeled the edge OBS node as a closed multi-class non-product-form queueing network, which we analyzed approximately by decomposition. In this paper, we develop computationally efficient approximate algorithms to analyze an edge node in the limiting case where the number of wavelengths is large. This limiting case is of practical importance since WDM technology will permit hundreds of wavelengths in a fiber.

1. Introduction

Optical burst switching (OBS) is a technology positioned between wavelength routing (i.e., circuit switching) and optical packet switching. All-optical circuits tend to be inefficient for traffic that has not been groomed or statistically multiplexed, and optical packet switching requires practical, cost-effective, and scalable implementations of optical buffering and optical header processing, which are several years away. OBS is a technical compromise that does not require optical buffering or packet-level parsing, and it is more efficient than circuit switching when the sustained traffic volume does not consume a full wavelength. The transmission of each burst is preceded by the transmission of a control packet, whose purpose is to inform each intermediate node of the upcoming data burst so that it can configure its switch fabric in order to switch the burst to the appropriate output port. An OBS source node does not wait for confirmation that an end-to-end connection has been set-up; instead it starts transmitting a data burst after a delay (referred to as *offset*), following the transmission of the control packet. We assume that OBS nodes have no buffers, therefore, in case of congestion or output port conflict, they may drop bursts.

OBS networks have received considerable attention recently. A number of wavelength reservation schemes have been proposed for OBS, including just-enough-time (JET) [11], Horizon [13], just-in-time (JIT) [15], and wavelength-routed OBS [7] which uses two-way reservations. The burst loss performance of OBS networks has been studied extensively [13,6,5,14,20], including recent work by the authors who developed novel and accurate analytical models for OBS switches with and without wavelength converters [18]. Various authors have also studied several issues related to OBS networks, including control architectures [16], wavelength scheduling algorithms [17], the effect of optical buffers [8], burst assembly [5] and traffic shaping at the edge of the network [14], QoS support [20,5].

The approximate algorithms we developed in [18] do not scale to nodes with hundreds of wavelengths per port. In this paper, we present computationally efficient approximate algorithms which scale to large numbers of wavelengths. In Section 2, we first describe the queueing model of an edge OBS node we developed in [18]. In Section 3, we develop algorithms for an edge node serving a large number of users, and in Section 4 we validate the accuracy of the approximations. We conclude the paper in Section 5.

2. A Queueing Network Model of an Edge OBS Node

In this paper, we model an edge OBS node employing the `Jumpstart` just-in-time (JIT) signaling protocol. The `Jumpstart` project [1] is a joint MCNC/NCSU research effort addressing the design,

*This work was supported by the Intelligence Technology Innovation Center under contract MDA904-00-C-2.

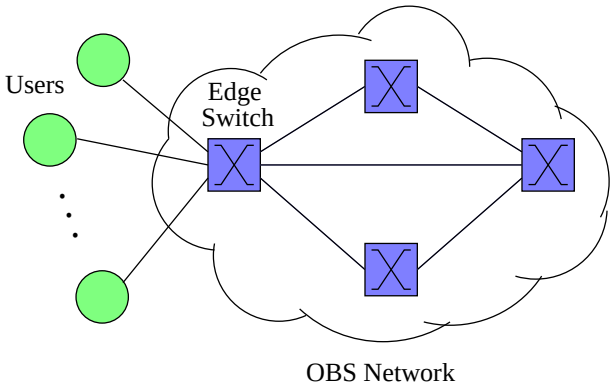

Figure 1. Users attached to an edge OBS node

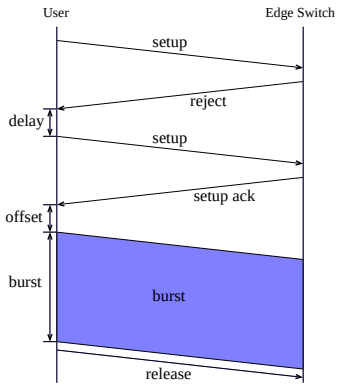

Figure 2. Signaling messages in `Jumpstart`

specification, performance evaluation, and hardware implementation of a signaling protocol for OBS networks. The signaling protocol is based on the work by Wei and McFarland [15] and it is described in [3]. We have implemented the protocol in FPGA, and deployed it in the ATDNet testbed in Washington, DC [2]. Below we describe the aspects of the `Jumpstart` signaling protocol that are relevant to the modeling an edge OBS node; for full details, the interested reader is referred to [3].

We consider an OBS network consisting of OBS nodes interconnected by bidirectional fiber links, as shown in Figure 1. Each fiber carries $W + 1$ wavelengths. One wavelength is used to transmit control packets, and the other W wavelengths are used to transmit data bursts. We assume that a user is capable of transmitting on any of these $W + 1$ wavelengths simultaneously.

Following the `Jumpstart` JIT signaling protocol [3], a user first sends a `setup` message to its edge OBS node. We assume that an OBS node consists of a non-blocking space-division switch fabric, with no optical buffers. If the edge node can switch the burst, it returns a `setup ack` message to the user. The `setup ack` message contains the offset field that informs the user how long it should wait before transmitting its burst. It is possible, however, that a `setup` message be refused. In this case, the edge node returns a `reject` message. The user goes through a random delay, and it then re-transmits the `setup` message. In our model, we assume that the user continues to re-transmit the `setup` message until it receives a `setup ack` message. (Alternatively, we can assume that the user drops the burst if it receives a `reject` message. This case can be easily taken into account in our model. For further details, see [19].)

In `Jumpstart` [3], the OBS node allocates resources within its switch fabric for a burst at the moment that it decides to accept the `setup` message. The source may indicate the length of its burst transmission in the `setup` message. This information is used by the OBS node to determine when to free the resources allocated to the burst. Alternatively, the source may simply send a `release` message to the node to indicate the end of its transmission. In this case, the OBS node frees the resources allocated to the burst upon receipt of the `release` message. Our model can take into account either method, due to the inherent abstractions in the underlying queueing network.

The sequence of messages exchanged between a user and its edge node is shown in Figure 2. A user can be in one of three states: **(1)** *idle*, i.e., no bursts to transmit; **(2)** *busy* transmitting a burst; or **(3)** *blocked*, i.e., undergoing a delay before it re-transmits a `setup` message. If a user can simultaneously transmit bursts on different wavelengths, then it can be in a different state for each burst wavelength.

2.1. The Burst Arrival Process

Each burst wavelength from a user to an OBS edge switch is associated with a burst arrival process. We use the three-state Markov process shown in Figure 3 to model arrivals on a given burst wavelength. The arrival process may be in one of three states: `short burst`, `long burst`, or `idle`. If it is in the `short burst` (respectively, `long burst`) state, then the user is in the process of transmitting a short (respectively, long) burst on this wavelength. If it is in the `idle` state, then the user is not transmitting any burst on this wavelength. The duration of a burst, whether short or long, is assumed to be exponentially distributed. In this model, we assume that the source becomes idle after the transmission of each burst. That is, the source does not transmit bursts back-to-back. This assumption can be easily removed by

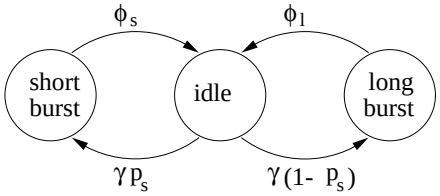

Figure 3. The burst arrival process

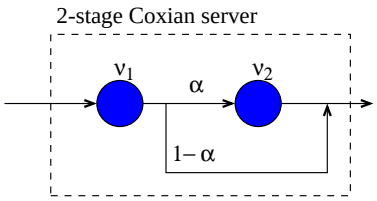

Figure 4. 2-stage Coxian server

modifying the three-state Markov process in Figure 3 to include transitions between the `short burst` and `long burst` states. Also, more complicated burst arrival processes can be modeled by introducing additional states and appropriate transitions between them. For instance, instead of using only two burst lengths (short and long), we may introduce $k > 2$ different burst lengths, each associated with a different state of the Markov process. Non-exponentially distributed burst lengths can also be accounted for by describing the length of a burst by a Coxian distribution. The analysis of the queueing network model that represents an edge OBS node can be extended to these more general burst arrival processes.

The burst arrival process of Figure 3 can be characterized completely by the four parameters: **(1)** the mean duration $1/\gamma$ of the `idle` state, **(2)** the mean duration of the `short burst` state $1/\phi_s$, **(3)** the mean duration of the `long burst` state $1/\phi_l$, and **(4)** the probability p_s that a burst is a small burst. In this paper, we will use the following four parameters to characterize the arrival process: **(1)** the *load* l of the burst arrival process, defined as the percentage of time that a wavelength is used for transmitting bursts, **(2)** the mean burst size s of both short bursts and long bursts, **(3)** the *ratio* r of the mean long burst duration to the mean short burst duration, i.e., $r = \phi_s/\phi_l$, and **(4)** the probability p_s that a burst is a small burst. Given one set of parameters, one can obtain the other. For details, see [19].

Since a customer may request either a short or a long burst with probabilities p_s and $1-p_s$, respectively, the burst time distribution is a two-stage hyper-exponential distribution. It is well-known that this distribution can be represented by a two-stage Coxian server (see Figure 4). We will let ν_1 and ν_2 denote the service rate of the first and second stage of the Coxian server, respectively, and a denote the probability that, upon completion of the first service stage, the customer will proceed to the second stage. The values of ν_1, ν_2, and a are uniquely determined by the values of $1/\phi_s, 1/\phi_l$, and p_s [18].

2.2. A Queueing Network Model of an Edge OBS Node

Let P and $N \leq P$, denote the number of input/output ports of an edge node and the number of the users connected to the edge node, respectively. The traffic on each incoming wavelength from a user to the edge node is generated by the burst arrival process described in Section 2.1. Since each user can simultaneously transmit bursts on all its W burst wavelengths, the user is associated with W different burst arrival processes. Therefore, an edge node with N users has a total of NW burst arrival processes.

Let us first consider an edge OBS node with no converters. In this case, a burst on an incoming wavelength can only be switched to the *same* wavelength on each output port, and user bursts arriving to the edge switch on different wavelengths do not interfere with each other. Consequently, the edge node can be decomposed into W sub-systems, one per burst wavelength, and this decomposition is exact. Each sub-system $w, w = 1, \cdots, W$, is a $P \times P$ switch with N users, but each input and output port has a single wavelength, which corresponds to wavelength w of the original edge switch. Therefore, each sub-system has N burst arrival processes.

The queueing network model of a sub-system is shown in Figure 5; it consists of $P+1$ nodes numbered $0, 1, \cdots, P$. Node 0 is an infinite server node, and it represents the burst arrival processes which are in the `idle` state. Node $i, i = 1, \cdots, P$, represents the (single) wavelength of output port i. Each node i consists of a single *transmission server* and an *infinite server*. The customer (if any) occupying the transmission server represents the arrival process whose burst is being transmitted by output port i. The customers (if any) in the infinite server represent those arrival processes which are undergoing a delay before their users re-transmit the corresponding `setup` messages. The total number of customers in this closed queueing network model of a sub-system is equal to N (the total number of burst arrival processes in the sub-system).

Let us now follow the path of a customer through the queueing network model in Figure 5. Let us

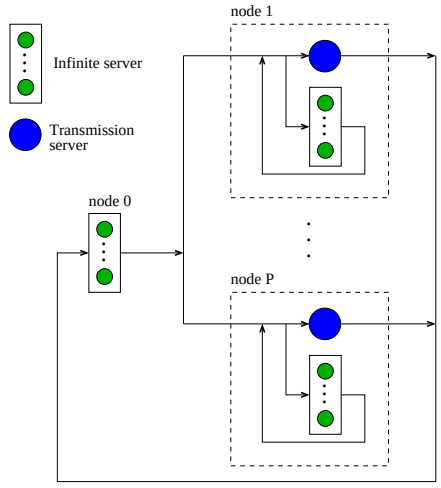

Figure 5. Queueing network model of a subsystem of an edge switch without converters

Parameter	User 1	User 2	User 3	User 4
load l	0.2	0.4	0.6	0.8
mean s	2	4	6	8
ratio r	20	40	60	80
p_s	0.2	0.4	0.6	0.8
q_5	1	1	1	1
q_1, q_2, q_3, q_4	0	0	0	0

Figure 6. Parameters of the burst arrival processes for verifying Property 2

assume that the customer starts in the `idle` state, i.e., it is in node 0. The time it spends in the `idle` state is exponentially distributed with mean $1/\gamma$. Upon completion of its service at node 0, it moves to node i with probability q_i; this corresponds to the transmission of a `setup` message for a burst with output port i. If the single transmission server at node i is free, the customer enters service immediately. The service time is exponentially distributed with a mean of $1/\phi_s$ or $1/\phi_l$ with probabilities p_s or $1-p_s$, corresponding to the transmission of short or long burst, respectively. If the transmission server is busy (i.e., output port contention occurs), the customer enters the infinite server at node i, where it undergoes an exponential delay with mean $1/\omega$; this delay models the delay until the retransmission of the `setup` message. Upon completion of the exponential delay, the customer again tries to seize the transmission server. If the transmission server is busy, the customer joins the infinite server again, and it undergoes another delay, and so on, until it succeeds to get hold of the transmission server. The customers in the infinite server are often referred to in the literature as *orbiting* customers.

When all N customers have the same burst arrival process, the closed queueing network model consists of a single class of N customers and $P+1$ nodes. If each customer has a different burst arrival process and/or branching probabilities, then the queueing network becomes a multi-class queueing network with $P+1$ nodes and N classes, where each class contains exactly one customer. In [18], we described approximate algorithms for the analysis of the single-class and multi-class queueing network.

In the case of converters, a `setup` message for output port i of the switch is accepted as long as at least one wavelength on this output port is free. Otherwise, the `setup` message is rejected, and the user undergoes a delay before retransmitting the message. Clearly, the above decomposition of an edge switch into sub-systems per wavelength is no longer possible, since user bursts arriving on different wavelengths may interfere with each other. The edge switch *as a whole* is modeled by a closed queueing network which is very similar to the one shown in Figure 5. The new queueing network consists of $P+1$ nodes and a total of NW customers (since there are now NW arrival processes). Node 0 in the new queueing network is identical to node 0 in the network of Figure 5. Similarly, each node $i, i = 1, \cdots, P$, in the new queueing network corresponds to each of the output ports of the edge switch. The difference is that each node $i, i = 1, \cdots, P$, consists of an infinite server and W (rather than one) transmission servers, each corresponding to one of the W wavelengths of output port i. For the edge OBS node with converters, approximate algorithms for both the single-class and multi-class cases are reported in [18].

3. An Edge Node with a Large Number of Wavelengths

The algorithms reported in [18] have a good accuracy, but they are computationally intensive and they cannot be used to analyze an OBS edge node with a large number of wavelengths, say 64 or more. In this paper, we describe approximate algorithms for the analysis of the queueing network shown in Figure 5

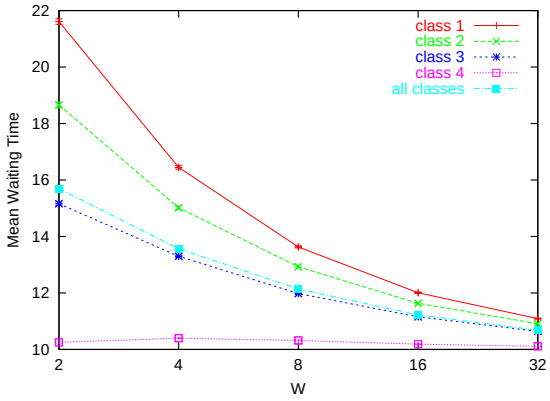

Figure 7. Mean waiting time for different classes

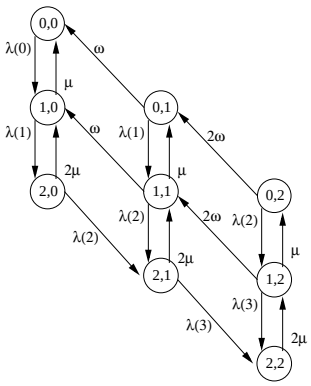

Figure 8. Node state transition diagram

when the number W of wavelengths per fiber is large. These algorithms are based on two properties of the above queueing network when W is large, described below. The algorithms for the single-class and multi-class queueing network when W is large are described in Sections 3.2 and 3.3, respectively.

3.1. Two Properties when W is Large

The two properties described in this section have been verified by simulation. They have not been rigorously proved.

> *Property 1: As the number W of wavelengths increases, the performance of the queueing network under study becomes dependent on the mean burst size, and it is relatively insensitive to the burst size distribution.*

A similar property was reported in [4]. They found using simulation that, if arbitrarily distributed service times are replaced by exponentially distributed service times in a closed queueing network, the deviation in the performance measures is tolerable. This property is referred to as the *robustness* of the closed queueing network. nodes the mean deviation is zero. The authors reported that this approximation is more accurate if **(i)** the squared coefficient of variation c^2 of the service time is not very large, **(ii)** there is a bottleneck in the queueing network, and **(iii)** the number of customers increases. Note that in our case, as W increases, both the number of customers in the queueing network and the number of servers in each node increase.

We have found out by simulation that Property 1 holds for $W > 32$. Therefore, in this case we can analyze the queueing network shown in Figure 5 by assuming that the size of a burst has an exponential, rather than Coxian, distribution. This can be implemented by setting $r = 1$ in which case both short and long bursts are exponentially distributed with the same mean s. The algorithm for the single-class queueing network described in Section 3.2 is based on this approximation.

The second property is for the multi-class case of the queueing network shown in Figure 5. As we explained above in Section 2.2, the multi-class queueing network corresponds to the case of an edge OBS node where each customer has a different burst arrival process or branching probabilities.

> *Property 2: As the number W of wavelengths increases, the mean waiting time of a class of customers in node i, $i = 1, \cdots, P$, tends to the mean waiting time of all classes in the same node.*

We illustrate this behavior for the queueing network that represents an edge switch with $N = 4$ users, $P = 5$ input/output ports, and mean orbiting rate $\omega = 0.1$. For simplicity, we assume that there are four classes and that the wavelengths of the same user belong to the same class. Figure 6 lists the parameters of the burst arrival process for each user (class). Figure 7 shows the mean waiting time of each class 1-4 and of all classes at output port 5, as the number W of wavelengths increases from 2 to 32. We observe that when $W = 2$, the difference between the mean waiting time of the four classes is large. As W increases, however, this difference decreases, and when W is equal to 32 the mean waiting times of

the four classes are very close to the mean waiting time of all classes. This property implies that when we analyze an OBS edge switch with a large number of wavelengths, we can use a single-class queueing network to approximate the multi-class queueing network. The algorithm for a multi-class queueing network described in Section 3.3 is based on this property and also on Property 1.

This property can be intuitively explained as follows. Let us consider a product-form closed queueing network with two classes, and let (N_1, N_2) be the population vector, where N_1 and N_2 are the number of class 1 and class 2 customers, respectively, in the queueing network. The closed queueing network consists of M/M/1 FIFO nodes and M/G/∞ nodes. At M/M/1 nodes, both classes have the same service time distribution, but at M/G/∞ nodes, classes may have different service time distributions.

Consider M/M/1 node i, and let μ_i denote its service rate. Let $W_{ic}(N_1, N_2)$ denote the mean waiting time of a class-c customer at node i in a network with a population vector (N_1, N_2), and let $L_i(N_1, N_2)$ denote the corresponding mean number of customers at node i. By the *arrival theorem*, the distribution of the number of customers seen by a class-c customer at the time of arrival to node i is the same as that of the number of customers at node i with one less class-c customer in the network. Therefore, the mean waiting time of a class-1 customer and a class-2 customer can be obtained as follows:

$$W_{i1}(N_1, N_2) = L_i(N_1 - 1, N_2)/\mu_i, \quad W_{i2}(N_1, N_2) = L_i(N_1, N_2 - 1)/\mu_i \tag{1}$$

Consider now the case where $N_1 = 1$ and $N_2 = 1$. Then, $W_{i1}(1,1) = L_i(0,1)/\mu_i$, $W_{i2}(1,1) = L_i(1,0)/\mu_i$. $L_i(0,1)$ (respectively, $L_i(1,0)$) is the mean number of customers at node i in a network with only a class-2 (respectively, class-1) customer. Since this closed queueing network has a product-form solution, both $L_i(0,1)$ and $L_i(1,0)$ can be calculated easily. They are dependent on the service time of each class at every node in the network. Recall that both class 1 and class 2 have the same service time distribution at M/M/1 nodes, but they have different service time distribution at M/G/∞ nodes. Therefore, $L_i(0,1) \neq L_i(1,0)$, and consequently $W_{i1}(1,1) \neq W_{i2}(1,1)$.

Consider now the case where $N_1 = 1000$ and $N_2 = 1000$. We have: $W_{i1}(1000, 1000) = L_i(999, 1000)/\mu_i$, $W_{i2}(1000, 1000) = L_i(1000, 999)/\mu_i$. $L_i(999, 1000)$ (resp., $L_i(1000, 999)$) is the mean number of customers at node i in a network with 999 class-1 customers and 1000 class-2 customers (resp., 1000 class-1 customers and 999 class-2 customers). We can see that there are 1998 common customers between the population vectors (999,1000) and (1000,999), 999 each of class 1 and 2. Since 99.95% of customers are the same in both population vectors, the percentage difference between $L_i(999, 1000)$ and $L_i(1000, 999)$ is smaller than the percentage difference between $L_i(0,1)$ and $L_i(1,0)$. The same is true for the pair of vectors $W_{i1}(1000, 1000)$ and $W_{i2}(1000, 1000)$ compared to vectors $W_{i1}(1,1)$ and $W_{i2}(1,1)$. In other words, the mean waiting time of different classes tends to the overall mean as the number of customers increases.

3.2. Analysis of the Single-Class Queueing Network when W is Large

We analyze the queueing network model described in Section 2 when the number W of wavelengths is large and the node has converters. (The case of no converters results in a simpler queueing network than the one analyzed here.) As we discussed in Section 3.1, due to Property 1, we assume that $r = 1$, that is, both short and long bursts are exponentially distributed with the same mean. We analyze this queueing network using Marie's algorithm [9,10]. The idea in Marie's method is to replace each non-BCMP node by a *flow equivalent node* with a load-dependent exponential service rate, obtained by calculating the conditional throughput of the non-BCMP node in isolation under a load-dependent arrival rate. In the following subsection, we calculate the conditional throughput of each node i, $i = 1, \cdots, P$. Node 0 is an infinite server, that is, a BCMP node, so we do not need to construct a flow equivalent node for it.

3.2.1. The Conditional Throughput

Let us consider node i, $i = 1, \cdots, P$, of the queueing network shown in Figure 5. Let $\lambda_i(n_i)$ be the arrival rate into this node when there are a total of n_i customers in the node. Due to Property 1, we assume that the service time of the transmission server is exponentially distributed with mean $1/\mu_i = s$. The state of node i can be described by $(n_i^{(t)}, n_i^{(o)})$, where $n_i^{(t)} = 0, 1, \cdots, W$, indicates the number of busy transmission servers, and $n_i^{(o)} = 0, 1, \cdots, (N-1)W$, gives the number of orbiting customers occupying the infinite server. In order to simplify the notation, and since we are only concerned with the analysis of node i in isolation, we drop the index i throughout the rest of this subsection.

Let $\upsilon(n)$ denote the conditional throughput of the node with n customers. We have that:

$$\upsilon(n) \;=\; \frac{p(n-1)\lambda(n-1)}{p(n)} \tag{2}$$

where $p(n)$ is the steady-state probability that there are a total of n customers in the node, $p(n) = \sum_{n^{(t)}+n^{(o)}=n} p(n^{(t)}, n^{(o)})$. We obtain the steady-state probability $p(n^{(t)}, n^{(o)})$ of state $(n^{(t)}, n^{(o)})$ by solving the node numerically. Figure 8 shows the state transition diagram of a node with $N = 2$ and $W = 2$. We note that the transition rate matrix is a block tri-diagonal matrix, and each diagonal block is also a tri-diagonal matrix. Therefore, we use the block Gauss-Seidel method [12] to solve it. Since each diagonal block is a tri-diagonal matrix, it is easy to get its LU decomposition, and then each block equation can be solved by a forward and backward substitution.

3.2.2. The Iterative Algorithm

We analyze the queueing network following Marie's algorithm:

- **Step 1:** Initialize the service rate $\mu_i(n_i)$ of flow equivalent server $i, i = 1, 2, \cdots, P$, to n_i/s, and set the service rate $\mu_0(n_0)$ of flow equivalent server 0 to γn_0.
- **Step 2:** For each node i, $i = 1, 2, \cdots, P$, do the following steps:
  - **Step 2.1:** Calculate the arrival rate $\lambda_i(n_i)$ of node i by short-circuiting node i in the substitute product-form queueing network, where each node j has an exponential service time $\mu_j(n_j)$.
  - **Step 2.2:** Calculate the probability $p_i(n_i^{(t)}, n_i^{(o)})$ of node i using the block Gauss-Seidel method.
  - **Step 2.3:** Calculate the conditional throughput $\upsilon_i(n_i)$ of node i using expression (2).
- **Step 3:** Check the following two convergence conditions. If both convergence criteria are satisfied, then stop. Otherwise, set $\mu_i(n_i)$ to $\upsilon_i(n_i)$ for all $i = 1, 2, \cdots, P$, and go back to Step 2.
  - The first convergence condition ensures that the sum of the mean number of customers at each nodes is equal to the total number of customers in the queueing network:

$$\left| \frac{NW - \sum_{i=0}^{P} \sum_{j=0}^{NW} j p_i(j)}{NW} \right| < \epsilon \tag{3}$$

  - The second convergence condition ensures that the conditional throughput of each node is consistent with the topology of the queueing network:

$$\left| \frac{s_i - \frac{1}{P+1} \sum_{j=0}^{P} s_j}{\frac{1}{P+1} \sum_{j=0}^{P} s_j} \right| < \epsilon \quad i = 0, 1, \cdots, P, \qquad s_i = \begin{cases} \frac{1}{p_i} \sum_{j=0}^{NW} p_i(j)\mu_i(j), & i = 1, \cdots, P \\ \sum_{j=0}^{NW} p_i(j)\mu_i(j), & i = 0 \end{cases} \tag{4}$$

3.2.3. The Mean Waiting Time

We now show how to calculate the mean waiting time of a customer at node i. This is used in the solution of the multi-class network in Section 3.3.

The mean waiting time $T_i^{(w)}$ of a customer at node $i, i = 1, 2, \cdots, P$, is the average time from the instance when the customer enters the node to the instance when the customer gets service at the transmission server. The mean response time $T_i^{(r)}$ of a customer at node $i, i = 1, 2, \cdots, P$, is the average time from the instance when the customer enters the node to the instance when the customer leaves the node. By applying Little's law, we obtain $T_i^{(w)} = T_i^{(r)} - 1/\mu_i = N_i^{(r)}/\lambda_i - 1/\mu_i$, where $N_i^{(r)} = \sum_{j=0}^{(N-1)W} \sum_{k=0}^{W} (k+j) p_i(k, j)$ is the mean number of customers in node i, and $\lambda_i = \sum_{j=0}^{(N-1)W} \sum_{k=0}^{W} \lambda_i(k+j) p_i(k, j)$ is the mean arrival rate into node i. Note that, the utilization of the transmission server at node i, $\lambda_i/(W\mu_i)$, can also be obtained as:

$$\frac{\lambda_i}{W\mu_i} \;=\; \frac{1}{W} \sum_{j=0}^{(N-1)W} \sum_{k=0}^{W} k p_i(k, j), \quad \text{from where we have:} \quad \frac{1}{\mu_i} \;=\; \frac{1}{\lambda_i} \sum_{j=0}^{(N-1)W} \sum_{k=0}^{W} k p_i(k, j) \tag{5}$$

By substituting (5) to the expression for $T_i^{(w)}$, we can obtain

$$T_i^{(w)} = \frac{1}{\lambda_i} \sum_{j=0}^{(N-1)W} \sum_{k=0}^{W} j p_i(k,j) \tag{6}$$

3.3. Analysis of the Multi-Class Queueing Network

In this section, we extend the above analysis of the queueing network model of the edge OBS switch to the case where each customer has a different burst arrival process. This is taken into account by associating each customer with a different class. The resulting queueing network is a closed non product-form queueing network with multiple classes, each of which has only a single customer. The number of classes is NW, which by current technological standards can be very large. For example, the number of classes for 8 users and 128 wavelengths is 1024. As before, we assume that the OBS node has converters.

According to Property 2, as the number of wavelengths increases, the mean waiting time of a class at a node becomes closer to the mean waiting time of all classes at the node. This property implies that when we analyze a queueing network with a large number of classes, we can use a single-class queueing network to approximate the multi-class queueing network. Therefore, in Section 3.3.1, we describe an aggregation technique for constructing a single-class queueing network which is equivalent to the multi-class queueing network, and in Section 3.3.2, we present an iterative algorithm for analyzing the multi-class network.

3.3.1. Class Aggregation

For the equivalent single-class queueing network, we have to specify the branching probability $p_i^{(agg)}$ that a customer leaving node 0 will enter node i, and the mean service rate $\mu_i^{(agg)}$ at node i, $i = 1, 2, \cdots, P$. Let $p_i^{(c)}$ denote the branching probability that a class-c customer, $c = 1, \cdots, NW$, leaving node 0 will enter node i in the original multi-class queueing network, $s^{(c)}$ denote the mean burst size of class c, and $1/\gamma^{(c)}$ denote the mean duration of the idle state of class c. Suppose that we know the mean waiting time, $T_i^{(w)}$, of all classes of customers at node i. Then, we can approximately calculate the throughput $H^{(c)}$ of class c in the network as

$$H^{(c)} = 1/(\frac{1}{\gamma^{(c)}} + \sum_{i=1}^{P} (T_i^{(w)} + s^{(c)}) p_i^{(c)}) \tag{7}$$

In the equivalent single-class queueing network, the branching probabilities $p_i^{(agg)}$, $i = 1, \cdots, P$, can be obtained as $p_i^{(agg)} = \frac{\sum_c H^{(c)} p_i^{(c)}}{\sum_c H^{(c)}}$. Then, we obtain the mean service rate $\mu_i^{(agg)}$ using the expressions: $1/\mu_0^{(agg)} = \sum_c (H^{(c)}/\gamma^{(c)}) / \sum_c H^{(c)}$ and $1/\mu_i^{(agg)} = \sum_c H^{(c)} p_i^{(c)} s^{(c)} / \sum_c H^{(c)} p_i^{(c)}$, $i = 1, \cdots, P$.

3.3.2. The Iterative Algorithm

As we showed in the previous subsection, if we know the throughput of each class in a multi-class queueing network, we can aggregate it into a single-class queueing network. However, in order to calculate the throughput of each class in the multi-class queueing network, we need the mean waiting time of all class customers, which can be obtained by solving the single-class queueing network. Therefore, we use the following iterative algorithm to solve the multi-class queueing network.

- **Step 1:** Initialize the throughput $H^{(c)}$ of each class c to 1.
- **Step 2:** Construct the equivalent single-class queueing network by aggregating all classes into one class as we explained above, and solve it using the algorithm described in Section 3.2.
- **Step 3:** Calculate the mean waiting time of all classes using expression (6).
- **Step 4:** Set $H_{old}^{(c)}$ to $H^{(c)}$, $c = 1, 2, \cdots, NW$.
- **Step 5:** Calculate $H^{(c)}$ using expression (7), $c = 1, 2, \cdots, NW$.
- **Step 6:** Check whether the throughput $H^{(c)}$ satisfy the following convergence criterion: $\sqrt{\sum_c (H^{(c)} - H_{old}^{(c)})^2} / \sqrt{\sum_c \left(H^{(c)}\right)^2} < \epsilon$. If yes, then stop. Otherwise, repeat from Step 2.

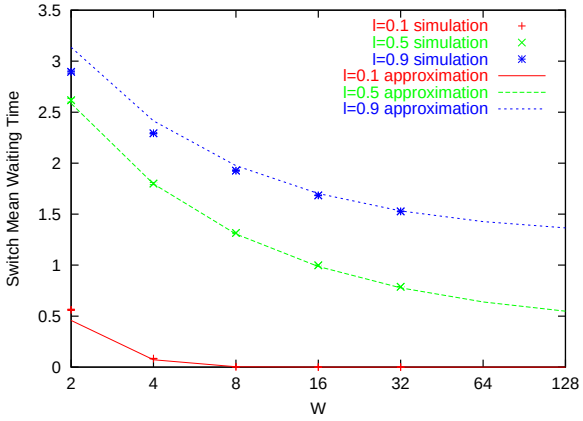

Figure 9. Effect of traffic load

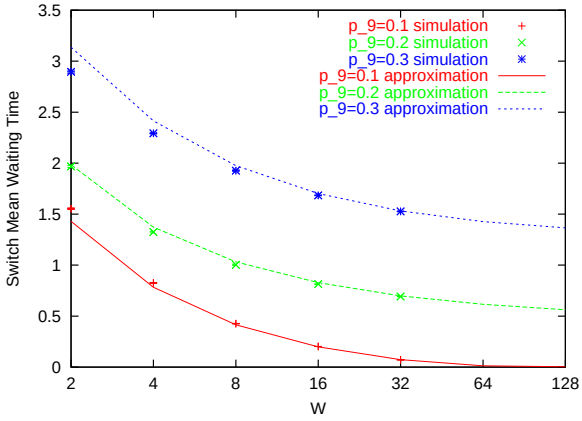

Figure 10. Effect of traffic pattern

4. Numerical Results

In this section, we present results to illustrate how the different system parameters affect the performance of the edge OBS node. In order to investigate the accuracy of our iterative algorithms, we also compare the approximate results to results obtained from a simulation program of an edge OBS switch. Simulation results are plotted along with 95% confidence intervals estimated by the method of batch means. The number of batches was 30, with each batch run lasting until each wavelength has transmitted at least 100,000 bursts.

A comprehensive set of results for various performance measures and for a wide range of values of the system parameters can be found in [19]. In this section we present results for an edge node with $N = 8$ users and $P = 10$ input/output ports. The queueing model of this edge switch has $P + 1 = 11$ nodes. Nodes 1 to 10 represent the output ports of the edge switch, and node 0 represents the burst arrival processes in the idle state. For the approximate algorithms, we vary the number of wavelengths from 2 to 128 in powers of two. However, limited by the running time of the simulation program, we only ran simulations with W values up to 32. As expected, the approximate algorithm does not give good results for values of $W < 32$ in the graphs presented below. We plot the results starting at $W = 2$, in order to show how its accuracy increases as W increases. Overall, our experiments we present indicate that the error due to our approximations decreases as W increases, and when $W = 32$, the approximate results are almost identical to the simulation results. We also note that when $W = 32$, the simulation takes between 60 and 80 hours to complete, whereas the approximate algorithm takes only a few minutes.

We first describe three experiments for an edge node in which all wavelengths are associated with the same burst arrival process. In the first experiment, the parameters of the process are: load $l = 0.1$, mean burst size $s = 1$, burst size ratio $r = 100$, and short burst probability $p_s = 0.9$. The destination probabilities are $q_9 = q_{10} = 0.3$ and $q_i = 0.05$ for $i = 1, \cdots, 8$. The mean orbiting time $1/\omega$ is set to be 10 times of the mean burst size (i.e., $1/\omega = 10$, or $\omega = 0.1$). The second and third experiments have the same parameters as the first one, except that the load l is set to 0.5 and 0.9, respectively.

Figure 9 shows the switch mean waiting time obtained from these three experiments. The switch mean waiting time is the average waiting time of all users before they transmit a burst to the switch. As expected, the higher the load, the longer the mean waiting time. We also observe that, as the number W of wavelengths increases, the mean waiting time decreases as well. Recall that in a switch with wavelength converters, a user has to wait and retransmit the `setup` message if and only if all wavelengths at the destination output port are busy. Intuitively, the larger the number of wavelengths, the smaller the probability that all wavelengths at an output port are busy. Therefore, as W increases, the switch mean waiting time decreases.

We also ran three experiments to illustrate the effect of the traffic pattern. In the first experiment, we assume that all wavelengths are associated with the same burst arrival process with the following parameters: load $l = 0.9$, mean burst size $s = 1$, burst size ratio $r = 100$, short burst probability $p_s = 0.9$,

destination probabilities $q_i = 0.1, i = 1, \cdots, 10$, and mean orbiting rate $\omega = 0.1$. The second experiment has the same parameters as the first one, except that the destination probabilities are $q_9 = q_{10} = 0.2$ and $q_i = 0.075$ for $i = 1, \cdots, 8$. In the third experiment, we set the destination probabilities to $q_9 = q_{10} = 0.3$ and $q_i = 0.05$. In other words, the first traffic pattern is uniform, while the last two are hot-spot traffic patterns with ports 9 and 10 receiving more traffic than the other 8 ports. Figure 10 shows the switch mean waiting time for the three traffic patterns. We observe that the larger the q_9 and q_{10}, the longer the mean waiting time, i.e., a switch with a hot-spot traffic experiences a longer mean waiting time than a switch with a uniform traffic.

5. Concluding Remarks

We have presented a new queueing network model of an edge OBS node, and we have developed efficient approximate algorithms to analyze both the single-class and multi-class instances in the case of very large numbers of wavelengths. We have presented results which demonstrate that our algorithms are accurate as the number of wavelengths approaches and/or exceeds 32. Also, our approximate algorithms are orders of magnitude faster than simulation. Therefore, our models can be used for extensive "what-if" analysis that would not be possible otherwise.

REFERENCES

1. http://jumpstart.anr.mcnc.org.
2. I. Baldine *et al.* Just-in-time optical burst switching implementation in the ATDnet all-optical networking testbed. In *Globecom 2003*. (Submitted).
3. I. Baldine, G. N. Rouskas, H. G. Perros, and D. Stevenson. `JumpStart`: A just-in-time signaling architecture for WDM burst-switched networks. *IEEE Communications*, 40(2):82–89, February 2002.
4. G. Bolch *et al. Queueing Networks and Markov Chains.* John Wiley & Sons, Canada, 1998.
5. K. Dolzer and C. Gauger. On burst assembly in optical burst switching networks - a performance evaluation of Just-Enough-Time. In *Proceedings of ITC-17*, pages 149–161, September 2001.
6. K. Dolzer, C. Gauger, J. Späth, and S. Bodamer. Evaluation of reservation mechanisms for optical burst switching. International Journal of Electronics and Communications, 55(1), January 2001.
7. M. Duser and P. Bayvel. Analysis of a dynamically wavelength-routed, optical burst switched network architecture. *IEEE/OSA Journal of Lightwave Technology*, 20(4):574–585, April 2002.
8. C. Gauger. Dimensioning of FDL buffers for optical burst switching nodes. In *Proceedings of the 5th IFIP Optical Network Design and Modeling Conference (ONDM 2002)*, Torino, February 2002.
9. R. Marie. An approximate analytical method for general queueing networks. *IEEE Transactions on Software Engineering*, 5(5):530–538, September 1979.
10. R. Marie. Calculation equilibrium probabilities for $\lambda(n)/C_k/1/N$ queues. *ACM Sigmetrics Performance Evaluation Review*, 9(2):117–125, 1980.
11. C. Qiao and M. Yoo. Optical burst switching (OBS)-A new paradigm for an optical Internet. *Journal of High Speed Networks*, 8(1):69–84, January 1999.
12. W. Stewart. *Numerical Solutions of Markov Chains.* Princeton University Press, New Jersey, 1994.
13. J. S. Turner. Terabit burst switching. *Journal of High Speed Networks*, 8(1):3–16, January 1999.
14. S. Verma, H. Chaskar, and R. Ravikanth. Optical burst switching: a viable solution for terabit IP backbone. *IEEE Network*, pages 48–53, November/December 2000.
15. J. Y. Wei and R. I. McFarland. Just-in-time signaling for WDM optical burst switching networks. *Journal of Lightwave Technology*, 18(12):2019–2037, December 2000.
16. Y. Xiong, M. Vandenhoute, and H.C. Cankaya. Control architecture in optical burst-switched WDM networks. *IEEE Journal on Selected Areas in Communications*, 18(10):1838–1851, October 2000.
17. J. Xu, C. Qiao, J. Li, and G. Xu. Efficient channel scheduling algorithms in optical burst switched networks. In *Proceedings of IEEE INFOCOM*, 2003.
18. L. Xu, H. G. Perros, and G. N. Rouskas. A queueing network model of an edge optical burst switching node. In *Proceedings of IEEE INFOCOM 2003*, April 2003.
19. Lisong Xu. *Performance Analysis of Optical Burst Switched Networks.* PhD thesis, NCSU, 2002.
20. M. Yoo, C. Qiao, and S. Dixit. QoS performance of optical burst switching in IP-over-WDM networks. *Journal on Selected Areas in Communications*, 18(10):2062–2071, October 2000.