A Novel Analytical Model for Electronic and Optical Switches with Shared Buffer

Zhenghao Zhang and Yuanyuan Yang
Department of Electrical & Computer Engineering, State University of New York, Stony Brook, NY 11794, USA

Abstract— **Switches with shared buffer have lower packet loss probabilities than other types of switches when the sizes of the buffers are the same. In the past, the performance analysis for electronic shared buffer switches has been carried out extensively. However, due to the strong dependencies of the output queues in the buffer, it is very difficult to find a good analytical model. Existing models are either accurate but have exponential complexities or not very accurate. In this paper, we propose a novel analytical model called the Aggregation Model for switches with shared buffer. This model can be used for analyzing both electronic and optical switches, and has perfect accuracies under all tested conditions and has polynomial time complexity. It is based on the idea of induction: first find the behavior of 2 queues, then aggregate them into one block; then find the behavior of 3 queues while regarding 2 of the queues as one block, then aggregate the 3 queues into one block; then aggregate 4 queues and so on. When a sufficient number of queues have been aggregated, the behavior of the entire switch is found. We believe that the new model represents the best analytical model for shared buffer switches so far.**

I. Introduction

Switches with shared buffer have lower packet loss probabilities than other types of switches when the sizes of the buffers are the same. Due to its practical interest, the performance analysis of shared buffer switches has become a classical problem in the literature and over the years has intrigued many researchers. However, it has not been adequately solved. Existing models are either accurate but have exponential complexities or not very accurate.

The difficulty of analytically modeling a shared buffer switch is due to the strong dependencies of the queues in the buffer. For example, consider the switch shown in Fig.1(a). It has N input ports and N output ports, and a common buffer pool with B cell locations. The arriving cells are multiplexed and stored in the buffer, where they are organized into N separate queues, one for each output port. The strong dependency means that the numbers of cells stored in the N queues are a set of random variables strongly depending on each other. The dependency comes from both the dependency of the input and the dependency of the finite size buffer. The dependency of the input means, for example, that if it is known that there are x cells destined for output 1, then there cannot be over $N-x$ cells destined for other outputs. In other words,

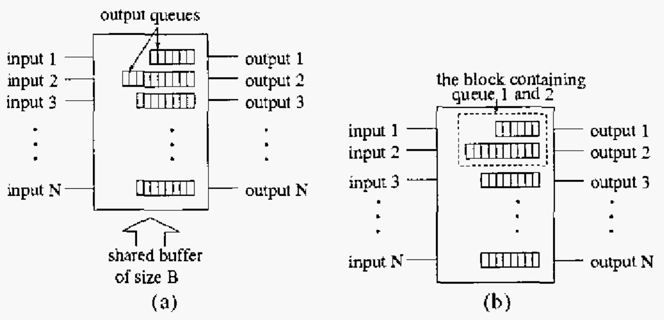


Fig. 1. (a). A shared buffer switch with N input ports and N output ports and a common buffer pool with B cell locations. (b). After the behaviors of queue 1 and queue 2 are found, regard them as one block.

the knowledge of the number of cells destined to one output contains much information about the number of cells destined to other outputs. The dependency of the buffer is similar: knowing that there are y cells in output queue 1 rules out the possibility that there are over $B-y$ cells in other queues.

The strong dependencies of the queues make exact analytical modeling impossible except by the vector method which uses a $1 \times N$ vector to represent the state of the switch, with each component being the number of cells stored in each queue [6]. However, this makes the number of states grow exponentially with N. To make the analysis tractable, other researchers tried to circumvent the strong dependency by making assumptions on the queues. In [6], [5] it was assumed that the queues are independent of each other. In [1], it was assumed that the cells stored in the buffer have independent random destinations. In [3], [4], it was assumed that all possible combinations of how cells were stored in the buffer are equally likely. All these assumptions fail to acknowledge the fact that the queues, as well as the cells stored in the queues, are indeed dependent upon each other. As a result, the accuracies of these models are not very good under certain conditions. In this paper we will give the Aggregation Method to solve the problem in a completely different way. In this model we do not make arbitrary assumptions on the queues in the buffer. Moreover, the model has polynomial running time as a function of the switch size and gives very accurate results under all tested cases.

Although the technical details and mathematical interpretations could be lengthy, the idea of our method is quite simple and can be described as follows. The basic idea is to find the behavior of the queues in an inductive way. First consider when the buffer is very large. In this case the cell loss probability will be very small and the buffer dependency can be neglected. Note that the queues are still dependent on each other due to the input dependency. Now if we only consider 2 queues in the buffer, say, queues for output 1 and output 2, a very good prediction about the behaviors of these two queues can be obtained without much difficulty: the input patterns to these two queues are known (by the assumptions about input traffic), and no other queue will interfere with them, i.e., no other queue will grow to a size so large such that some of the input cells to these two outputs have to be dropped. After finding out the behavior these two queues, we "aggregate" them together, or to consider them as a single block. This block will store the cells for output 1 and output 2, as illustrated in Fig.1(b), and at this moment, its behavior is known quite well. Now consider 3 queues: the queues for outputs 1, 2, and 3. Instead of regarding them as 3 separate queues, we regard them as two components: one component is the block containing queues 1 and 2, and the other component is queue 3. The behavior of each of the two components is known, and again we can find out the behaviors of the three queues quite precisely. Then again we aggregate the three queues into one block, and then consider 4 queues. The process can be carried on until all N queues have been aggregated into one block, and by this time, we have found the queueing properties of the entire switch. We will show later that this idea can also be used for switches with smaller buffers.

The advantage of this method is that, regardless of the value of N, in each step, there are always only two components that need to be considered. The whole process takes N steps, therefore the complexity of this method is a polynomial of N. Also note that though the queues are dependent upon each other, they interact in such a way that after a step some unnecessary details can be "omitted" and only those that will be needed in the future need to be stored. For example, when considering two queues, rather than storing the probability that queue 1 is of size x and queue 2 is of size y, we only store the probability that queue 1 and queue 2 are of size $x + y$, because other queues typically do not care how these $x + y$ cells are distributed as long as they know that there are $x + y$ cells in these two queues.

It should be pointed out that the idea of aggregation is not limited only to electronic switches, and can be used in a wide variety of applications. In this paper we will also apply it to optical Wavelength Division Multiplexing (WDM) switches. WDM technique is now widely regarded as the candidate for future high speed communication networks due to its nearly unlimited bandwidth [12]. In a WDM packet switch, input and output links are optical fibers. On a fiber there are k wavelengths, each carrying independent data [12]. Buffers are implemented with Fiber Delay Lines (FDL), which are capable of delaying packets for a certain amount of time. The incoming packets are fixed length cells stored in the optical domain.

In the past the performance of WDM packet switches with

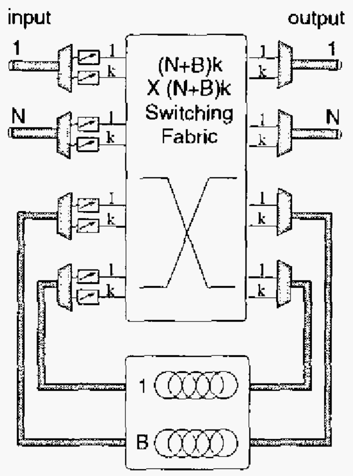


Fig. 2. An optical WDM switch with shared buffer.

dedicated buffer for each output link has been studied by [14], [13]. However, since FDLs are expensive and bulky, to reduce the cost and size of the switch, it is more desirable to let them shared by all outputs [15]. Fig.2 shows such a WDM switch with shared buffer. It has N input/output fibers and B shared FDLs, each capable of delaying a packet for one time slot. Before entering the switching fabric, a packet can be converted from one wavelength to another by wavelength converters. Aside from the technical details, this switch can be considered as a switch with N input/output ports and Bk buffer locations, with each port capable of receiving/sending up to k packets at a time slot. To the best of our knowledge, the performance of this type of switch has not been previously studied analytically. We will show that the Aggregation Model can also be used for it and can give very accurate results.

In the rest of the paper, we will first illustrate the Aggregation Model under uniform Bernoulli traffic for electronic switches. We will then show that it can also be used under other types of traffic. Finally we will apply it to optical WDM switches and also to other switch models.

II. The Aggregation Model for Electronic Switches under Bernoulli Traffic

In this section we will illustrate the Aggregation Model using uniform Bernoulli traffic for electronic switches. The assumptions of the traffic are:

- The arrival at the input is Bernoulli with parameter ρ ($0 \leq \rho \leq 1$), i.e., at a time slot, the probability that there is a cell arriving at an input port is ρ and independent of other time slots.
- The destination of a cell is uniformly distributed over all N outputs.
- Inputs are independent of each other.

The switch is modeled as running in a three-phase manner: In phase 1, it accepts the arrived cells. In phase 2, it transmits the cells at the head of queues. In phase 3, it runs a buffer management algorithm and drops some cells if necessary. We call it the "receive first switch" and denote it by RFS. Later on we will also consider "transmit first switch" or "TFS", in which phase 1 and phase 2 are reversed.

When the switch has just finished phase 2, we say it is in the "intermediate state". In the intermediate state, if the number of cells that have to be buffered exceeds the buffer size, some of the cells have to be dropped. The decisions are made by a buffer management algorithm. We adopt a "random drop with pushout" algorithm which, if the buffer size is exceeded by V, will randomly drop V cells out of the total $B+V$ cells. Note that by this algorithm, not only the newly arrived cells but also cells already in the buffer could be dropped. This is what meant by "pushout". In general, algorithms that allow pushout have better performance than those do not [7].

Before diving into the formulas and equations, we first give an overview of the method.

A. An Overview

There are N virtual queues in the buffer, one for each output. We first consider two queues, queues for output 1 and for output 2. The state of these two queues can be represented by a pair of random variables, (X,Y), with X being the number of cells stored in queue 1 and Y being the number of cells stored in queue 2. We model it as a two-dimensional Markov chain. The transition rate and the steady state distribution of this Markov chain can be found. Then we combine the two queues into one block which stores the cells for output 1 and output 2. We will then use another pair of random variables, (S,U), to represent the state of the block, where S is the number of cells stored in the block and U is the number of non-empty queues in the block. To describe the behavior of the block, two conditional probabilities are needed, $CT(S_1,U_1|S_0,U_0,n)$ and $CB(U_2|S_1,U_1,S_2)$. $CT(S_1,U_1|S_0,U_0,n)$ is the probability that the block goes from state (S_0,U_0) to intermediate state (S_1,U_1) when there are n cells arrived for it. $CB(U_2|S_1,U_1,S_2)$ is the probability that if the intermediate state is (S_1,U_1), after dropping S_1-S_2 cells, the block will have U_2 non-empty queues.

Now consider three queues, queue 1 to queue 3. Regarding the first two queues as a block, we can use (S,U,Z) to represent the state of the three queues, where S is the the number of cells stored in the block, U is the number of non-empty queues in the block and Z is the number of cells stored in queue 3. With the transition probability for (S,U) obtained in the previous step, the transition rate of this 3-dimensional Markov chain can be found, with which we find the steady state distribution of this Markov chain. Now similar to the previous step, we combine the three queues into a single block and use only two random variables, (S,U) to represent the state of this block, where S is the number of cells stored in queue 1 to queue 3 and U is the number of non-empty queues from queue 1 to queue 3. We now update the two conditional probabilities used to describe the behavior of the new block containing three queues, $CT(S_1,U_1|S_0,U_0,n)$ and $CB(U_2|S_1,U_1,S_2)$, where $CT(S_1,U_1|S_0,U_0,n)$ is the probability that the block containing 3 queues goes from state (S_0,U_0) to intermediate state (S_1,U_1) when there are n cells arrived for it, and $CB(U_2|S_1,U_1,S_2)$ is the probability that after dropping S_1-S_2 cells the block containing 3 queues will have U_2 non-empty queues. When considering 4 queues, again the first 3 queues will be considered as one block and we can still use triple (S,U,Z) to represent the state of the four queues. After obtaining the steady state distribution of the Markov chain we can combine the four queues into one block. The process can be carried on until all N queues have been combined into one single block. Then we can find useful information such as cell loss probability and average delay of the switch.

Note that throughout the process only 3 random variables are used, as compared to N random variables of the vector method. Of course, this model is not an exact model as the vector method since we use only 2 random variables to model up to N queues and we assume Markov property of these random variables. However, as can be seen later, our model is indeed very accurate under all network configurations and arrival rates.

B. Detailed Description

In the following we give detailed description of our model. First consider only two queues.

1) Getting Started – Two queues: The state of queue for output 1 and queue for output 2 is represented by random variable pair (X,Y). We will first give the transition rate of this two dimensional Markov chain.

Suppose the Markov chain is currently in state (X_0,Y_0) where $X_0+Y_0 \le B$ and there are a cells arrived for output 1 and b cells arrived for output 2. The Markov chain will go to intermediate state (X_1,Y_1) where

$$X_1 = \begin{cases} 0, & X_0=0 \text{ and } a<2 \\ X_0+a-1 & \text{otherwise} \end{cases} \tag{1}$$

and

$$Y_1 = \begin{cases} 0, & Y_0=0 \text{ and } b<2 \\ Y_0+b-1 & \text{otherwise} \end{cases} \tag{2}$$

For convenience we write the conditional probability that given there are a cells arrived for output 1 and b cells arrived for output 2, the Markov chain will transit from (X_0,Y_0) to (X_1,Y_1) as $T_{a,b}(X_1,Y_1;X_0,Y_0|a,b)$:

$$T_{a,b}(X_1,Y_1;X_0,Y_0|a,b) = \begin{cases} 1, & \text{if Eq. (1) and (2) satisfied} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

The transition rate from (X_0,Y_0) to (X_1,Y_1) can be found by summing over all possible pairs of (a,b):

$$T(X_1,Y_1;X_0,Y_0) = \sum_{(a,b)} T_{a,b}(X_1,Y_1;X_0,Y_0|a,b)p_{a,b}(a,b) \tag{4}$$

where $p_{a,b}(a,b)$ is the probability that there are a cells arrived for output 1 and b cells arrived for output 2.

$p_{a,b}(a,b)$ can be found as follows. Let m be the total number of cells arrived for output 1 and output 2. Then

$$p_m(m) = \begin{pmatrix} N \\ m \end{pmatrix} (2\rho/N)^m (1 - 2\rho/N)^{N-m} \qquad (5)$$

where $m \in [0, N]$. Given there are m cells for output 1 and output 2, the probability that a out of these m cells are for output 1 is

$$p_{a|m}(a|m) = \begin{pmatrix} m \\ a \end{pmatrix} 0.5^a 0.5^{m-a} \qquad (6)$$

where $a \in [0, m]$ and

$$p_{a,b}(a,b) = p_{a|m}(a|a+b) p_m(a+b) \qquad (7)$$

So far we have given the transition rate from an initial state (X_0, Y_0) to some intermediate state (X_1, Y_1). (X_1, Y_1) is an intermediate state since it is the state before running the buffer management algorithm. If there is no buffer overflow in the switch, (X_1, Y_1) will be the state of the two queues in the next time slot. Otherwise some of the cells may be dropped by the buffer management algorithm and the two queues may store fewer cells than X_1 and Y_1. In this paper we refer to the dropping of cells as the "moving back" of the Markov chain.

It will be difficult, if not impossible, to know exactly how many cells will be dropped from queue 1 and queue 2, because at this moment we have absolutely no information about queues other than queue 1 and queue 2. Thus some approximation has to be used. We will assume that if $X_1 + Y_1 \le B$, no cells will be dropped from queue 1 and queue 2. Otherwise suppose $X_1 + Y_1 = B + V$, V cells will be dropped from queue 1 and queue 2.

Since the dropping is random, when $X_1 + Y_1 = B + V$, the probability that the switch dropped v_x cells from queue 1 and v_y cells from queue 2 is

$$\frac{\begin{pmatrix} X_1 \\ v_x \end{pmatrix} \begin{pmatrix} Y_1 \\ v_y \end{pmatrix}}{\begin{pmatrix} X_1 + Y_1 \\ V \end{pmatrix}} \qquad (8)$$

where $v_x + v_y = V$ and $0 \le v_x \le X_1$, $0 \le v_y \le Y_1$. After the dropping the Markov chain will move back to (X_2, Y_2) where $X_2 = X_1 - v_x$ and $Y_2 = Y_1 - v_y$.

When computing the transition rate of the Markov chain, we first find $T(X_1, Y_1; X_0, Y_0)$, the transition rate from state (X_0, Y_0) to state (X_1, Y_1) by Equation (4). If $X_1 + Y_1 \le B$, the transition rate from state (X_0, Y_0) to state (X_1, Y_1) is incremented by $T(X_1, Y_1; X_0, Y_0)$. Otherwise let $Back(X_2, Y_2; X_1, Y_1)$ be the probability that (X_1, Y_1) will move back to (X_2, Y_2). Then the transition rate from state (X_0, Y_0) to state (X_2, Y_2) is incremented by $T(X_1, Y_1; X_0, Y_0) \times Back(X_2, Y_2; X_1, Y_1)$.

After obtaining the transition rate, $\pi(X, Y)$, the steady state distribution of the Markov chain can be easily found, either by directly inverting the transition matrix or using a fixed point method. As described earlier, we will now combine these two queues into one block. We use another pair of random variables, (S, U), as the state of this block, where S is the number of cells stored in this block and U is the number of non-empty queues in this block where $0 \le S \le B$, $0 \le U \le 2$. For a given (S, U), (X, Y) which satisfies $X + Y = S$ and $u(X) + u(Y) = U$ is called a "sub-state" of (S, U), where $u()$ is the step function:

$$u(x) = \begin{cases} 0, & x \le 0 \\ 1 & \text{otherwise} \end{cases}$$

There can be more than one sub-states of (S, U) and all such sub-states will merge into one state. Note that this is where the simplification over the vector method begins. The probability that the block is in state (S, U), or $\pi(S, U)$, is obtained by summing $\pi(X, Y)$ for all its sub-states.

We will need two conditional probabilities to describe the behavior of the block for the computation in the next step. First we will need to find the transition rate from (S_0, U_0) to intermediate state (S_1, U_1), given that there are n cells arrived for output 1 and output 2. Denoted by $CT(S_1, U_1 | S_0, U_0, n)$, it can be found as follows. Let (X_0^0, Y_0^0) be a sub-state of (S_0, U_0). Given that the block is in state (S_0, U_0), the probability that it is in sub-state (X_0^0, Y_0^0) is

$$\theta = \pi(X_0^0, Y_0^0) / \pi(S_0, U_0)$$

For a sub-state of (S_1, U_1) denoted by (X_1^j, Y_1^j), let $T_n(X_1^j, Y_1^j; X_0^0, Y_0^0 | n)$ be the probability that given there are n arrived cells for output 1 and output 2, the block transits from (X_0^0, Y_0^0) to (X_1^j, Y_1^j). It can be found by

$$T_n(;|n) = \sum_{a+b=n} T_{a,b}(;|a,b) p_{a|m}(a|a+b) \qquad (9)$$

where $T_{a,b}(;|a,b)$ is given in Equation (3) and $p_{a|m}(a|a+b)$ is given in Equation (6). Let η be the summation of $T_n(X_1^j, Y_1^j; X_0^0, Y_0^0 | n)$ over all sub-states of (S_1, U_1). Increment $CT(S_1, U_1 | S_0, U_0, n)$ by $\theta\eta$. Then repeat this procedure for the rest of sub-states of (S_0, U_0).

The other probability we will need to know is when the intermediate state is (S_1, U_1), given the block will move back to a state containing S_2 cells, what is the probability that it will have U_2 non-empty queues? Denote it by $CB(U_2 | S_1, U_1, S_2)$, first, let θ be the probability that the block is in sub-state (X_1^0, Y_1^0) of (S_1, U_1). Knowing that there are $v = S_1 - S_2$ cells dropped, we can use Equation (8) to find the probability that v_x cells are dropped from queue 1 and v_y cells are dropped from queue 2, where $v_x + v_y = v$ and $0 \le v_x \le X_1^0$, $0 \le v_y \le Y_1^0$. Denote this probability as γ. Increment $CB(U_2 | S_1, U_1, S_2)$ by $\theta\gamma$ where $U_2 = u(X_1^0 - v_x) + u(Y_1^0 - v_y)$ and $u()$ is the step function. Then repeat this procedure for the rest of sub-states of (S_1, U_1).

2) The Iteration – More queues: Now suppose we have aggregated I queues into one block. When just completed the computing for two queues, $I = 2$. We now study $I + 1$ queues by regarding queue 1 to queue I as a block. The state of the $I + 1$ queues is represented by (S, U, Z), where S is the the number of cells stored in the block. U is the number of non-empty queues in the block and Z is the number of cells stored in queue $I+1$.

Similar to the 2-queue case, we will first need to find out the transition rate of this Markov chain. Suppose it is in state (S_0, U_0, Z_0). Given there are n cells arrived for output 1 to output I and c cells for output $I+1$, queue $I+1$ will go from Z_0 to Z_1 where

$$Z_1 = \begin{cases} 0, & Z_0 + c \leq 1 \\ Z_0 + c - 1 & \text{otherwise} \end{cases} \tag{10}$$

and the probability that the $I+1$ queues will transit to state (S_1, U_1, Z_1) is

$$T_{n,c}(S_1, U_1, Z_1; S_0, U_0, Z_0 | n, c) = CT(S_1, U_1 | S_0, U_0, n) R(Z_1, Z_0, c) \tag{11}$$

where

$$R(Z_1, Z_0, c) = \begin{cases} 1, & Z_1, Z_0, \text{ and } c \text{ satisfy Equation (10)} \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

and $CT(S_1, U_1 | S_0, U_0, n)$ is the conditional probability obtained from the previous step. The transition rate of the intermediate switch state can be found by

$$T(S_1, U_1, Z_1; S_0, U_0, Z_0) = \sum_{(n,c)} T_{n,c}(S_1, U_1, Z_1; S_0, U_0, Z_0 | n, c) p_{n,c}(n, c) \tag{13}$$

$p_{n,c}(n,c)$ can be found in a similar way to $p_{a,b}(a,b)$. Let m be the total number of cells arrived for output 1 to output $I+1$. Then

$$p_m(m) = \binom{N}{m} \left(\frac{(I+1)\rho}{N}\right)^m \left(1 - \frac{(I+1)\rho}{N}\right)^{N-m} \tag{14}$$

where $m \in [0, N]$. Given there are m cells for output 1 to output $I+1$, the probability that n out of these m cells are for output 1 to output I is

$$p_{n|m}(n|m) = \binom{m}{n} \left(\frac{I}{I+1}\right)^n \left(\frac{1}{I+1}\right)^{m-n} \tag{15}$$

where $n \in [0, m]$ and

$$p_{n,c}(n,c) = p_{n|m}(n|n+c) p_m(n+c) \tag{16}$$

For moving back, we make the same assumptions as in the 2-queue case, that is, if $S_1 + Z_1 \leq B$, no cells will be dropped from these queues, otherwise suppose $S_1 + Z_1 = B + V$, and V cells will be dropped from these queues.

Note that the switch may not actually work according to this assumption. When $S_1 + Z_1 \leq B$, cells from these queues may still be dropped since other queues may be storing too many cells, and when $S_1 + Z_1 = B + V$, fewer than V cells could be dropped from these queues since the switch may decide to drop some cells from other queues. However, the probabilities of the above events are relatively small, and most importantly, as I approaches $N-1$, these probabilities will decrease to zero and this assumption will become true.

Computing the moving back probability is somewhat more complicated than the 2-queue case because one more random variable, U, the number of non-empty queues in the block is involved. Suppose $S_1 + Z_1 = B + V$. First use Equation (8) to obtain the probability that out of these V cells, v_s are from the block and v_z are from queue $I+1$. Then the $(J+1)_{th}$ queue will move to $Z_2 = Z_1 - v_z$. The block will store $S_2 = S_1 - v_s$ cells, and the probability that it will have U_2 non-empty queues is $CB(U_2 | S_1, U_1, S_2)$ which was obtained in the previous step. With the moving back probability and the transition probability to the intermediate states, the transition probability of this Markov chain can be found in a similar way to the 2-queue case.

After obtaining the transition probabilities, the steady state distribution of the Markov chain can be found. We will then aggregate the $I+1$ queues into one block. The state of block will be represented by (S', U'), where S' is the number of cells stored in queue 1 to $I+1$ and U' is the number of non-empty queues from queue 1 to $I+1$. The probability that the block is in state (S', U') can be found by

$$\pi'(S', U') = \sum \pi(S, U, Z) \tag{17}$$

where (S, U, Z) satisfies $S + Z = S'$ and $U + u(Z) = U'$. Following our convention each such triple (S, U, Z) is called a sub-state of (S', U').

Finally, similar to the two-queue case, we have to prepare two conditional probabilities for the next step. The first is $CT'(S'_1, U'_1 | S'_0, U'_0, n')$, which is the probability that when n' cells arrived for output 1 to output $I+1$, the block will transit from (S'_0, U'_0) to (S'_1, U'_1). Again, let (S_0^0, U_0^0, Z_0^0) be a sub-state of (S'_0, U'_0). Knowing that the block is in state (S'_0, U'_0), the probability that is in sub-state (S_0^0, U_0^0, Z_0^0) is

$$\theta = \pi(S_0^0, U_0^0, Z_0^0) / \pi'(S'_0, U'_0)$$

Let (S_1^j, U_1^j, Z_1^j) be a sub-state of (S'_1, U'_1). $T_n(S_1^j, U_1^j, Z_1^j; S_0^0, U_0^0, Z_0^0 | n')$, which is the transition probability from state (S_0^0, U_0^0, Z_0^0) to state (S_1^j, U_1^j, Z_1^j) given that there are totally n' cells arrived for output 1 to output $I+1$ can be found by

$$T_n(;|n') = \sum_{n+c=n'} T_{n,c}(;|n,c) p_{n|m}(n|n') \tag{18}$$

where $T_{n,c}(;|n,c)$ is given in Equation (11) and $p_{n|m}(n|n')$ is given in Equation (15). Find such transition probabilities for all sub-states of (S'_1, U'_1) and let η be the summation of them. Increment $CT'(S'_1, U'_1 | S'_0, U'_0, n')$ by $\theta\eta$. Then repeat this for the rest of sub-states of (S'_0, U'_0).

The second conditional probability is that when the intermediate state is (S'_1, U'_1), knowing the block will move back to a state containing S'_2 cells, what is the probability that it will have U'_2 non-empty queues? Denote this probability by $CB'(U'_2 | S'_1, U'_1, S'_2)$. First let θ be the probability that the block is in sub-state (S_1^0, U_1^0, Z_1^0) of (S'_1, U'_1). Knowing that there are $V = S'_1 - S'_2$ cells dropped, we can use Equation (8) to find the probability that v_s cells are dropped from the block and v_z cells are dropped from queue $I+1$, where $v_s + v_z = V$ and $0 \leq v_s \leq S_1^0$, $0 \leq v_z \leq Z_1^0$. Denote this probability by γ. Now for the block, the intermediate state is (S_1^0, U_1^0), and it will move back to a state storing $S_2^0 = S_1^0 - v_s$ cells. The probability that after moving back, it will have U_2^0 non-empty queues is $\sigma = CB(U_2^0|S_1^0, U_1^0, S_2^0)$, which is obtained from the previous step. Increment $CB'(U_2'|S_1', U_1', S_2')$ by $\theta\gamma\sigma$ where $U_2' = U_2^0 + u(Z_1^0 - v_z)$ and $u()$ is the step function. Then repeat this procedure for the rest of sub-states of (S_1', U_1').

C. Using the Model

After iterating the process until $I = N-1$, the stationary distribution $\pi(S, U)$ can be found, where S is the number of cells stored in the buffer and U is the number of non-empty queues. We can also obtain $CT(S_1, U_1|S_0, U_0, n)$, which is the probability that when there are S_0 cells and U_0 non-empty queues in the buffer, upon receiving n cells, the switch will store S_1 cells and has U_1 non-empty queues before moving back. With this information we are now in the position to derive the cell loss probability and the average delay time of the switch.

1) Cell Loss Probability: The cell loss probability, denoted by α, can be found by

$$\alpha = \sum_{S_0=0}^{B} \sum_{U_0=0}^{N} \sum_{n=0}^{N} \sum_{S_1=0}^{B} \sum_{U_1=0}^{N} \pi(S_0, U_0) \cdot CT(S_1, U_1|S_0, U_0, n)(S_1 - B)p_n(n) \quad (19)$$

where $S_1 > B$. $p_n(n)$ is the probability that there are totally n cells arrived at this switch

$$p_n(n) = \binom{N}{n} \rho^n (1-\rho)^{N-n}, \quad (20)$$

where $n \in [0, N]$. The probability that the switch is in state (S_0, U_0) is $\pi(S_0, U_0)$. Upon receiving n new cells, the probability that it will transit to (S_1, U_1) is $CT(S_1, U_1|S_0, U_0, n)$. If $S_1 \leq B$, no cell will be dropped. Otherwise exactly $S_1 - B$ will be dropped. α is then found by summing over all possible n, U and S.

2) Average Delay: The delay time of a cell is usually defined as the number of time slots it stays in the buffer before being transmitted. Since the buffer management algorithm we adopt allows pushout, cells that have to be put into the buffer could be dropped without being actually transmitted. Therefore, we modify the definition of delay time as the number of time slots a cell stays in the buffer before being transmitted or being dropped.

It can be shown that in such a system Little's Formula still holds. That is,

$$E(W) = E(S)/(N\rho) \quad (21)$$

where $E(W)$ is the average delay time and $E(S)$ is the average number of cells stored in the buffer. $E(S)$ can be found as

$$E(S) = \sum_{S=0}^{B} \sum_{U=0}^{N} S\pi(S, U) \quad (22)$$

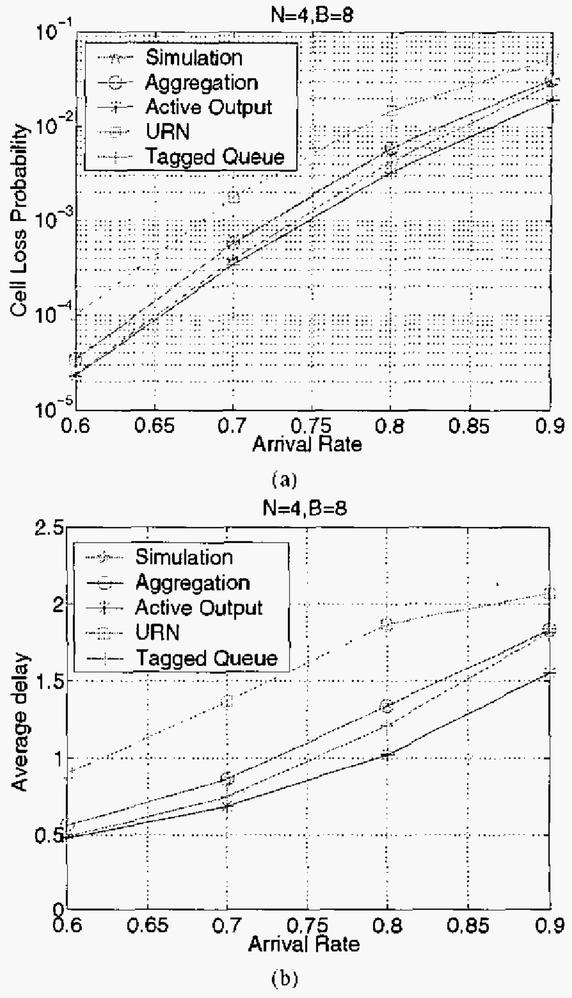


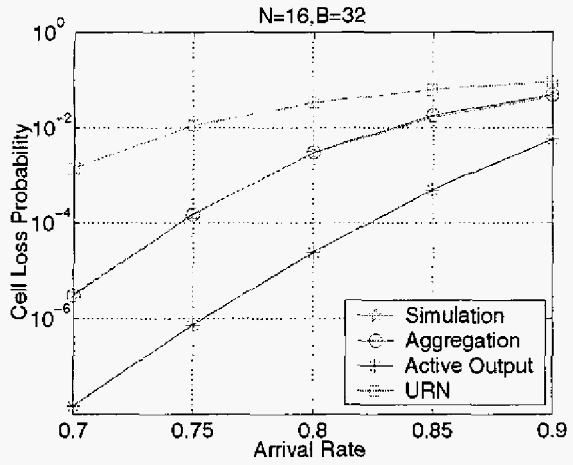
Fig. 3. Cell loss probability and average delay under Bernoulli traffic when $N = 4$, $B = 8$.

D. Validation of the Model

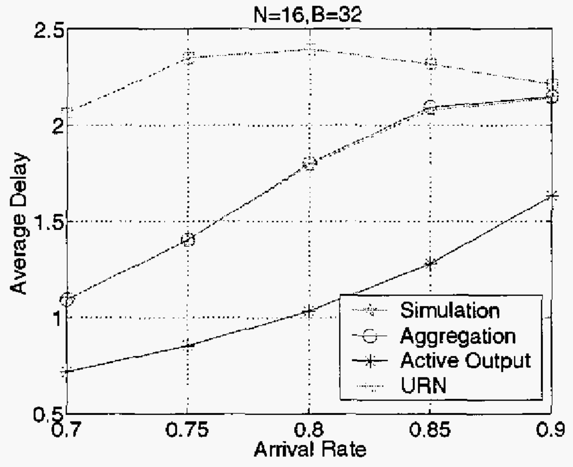
We conducted extensive simulations to validate the Aggregation model and the results are shown in Fig. 3 to Fig. 4. For comparison purpose, we also implemented 3 other models, called the Active Input Model [1], the URN Model [3], and the Tagged Queue Model [6], and showed their results in the figures. We can see that for both cell loss probability and average delay and for both small switch and large switch, the Aggregation Model is very accurate: its curves almost overlap with the simulation curves. Other models, in general, do not perform very well, especially when N and B grow large. We did not provide the results for the Tagged Queue Model when $N = 16$ and $B = 32$, because in this case it takes too much time since this model is actually of exponential complexity. We will give a detailed comparison of the Aggregation model and other models in Section VI.

III. The Aggregation Model under Bursty Traffic

So far we have considered the Aggregation Model under Bernoulli traffic. The idea of aggregation is actually not related



(a)



(b)

Fig. 4. Cell loss probability and average delay under Bernoulli traffic when $N = 16$, $B = 32$.

to traffic types, and in this section we will show how to apply it to Bursty Traffic for electronic switches. Under bursty traffic, an input port is assumed to be alternating between two states, the "idle" state and the "busy" state. When in "idle" state no cell arrives at this input port. When in "busy" state, cells continuously arrive at this input port and go to the same destination. This stream of cells is called a burst.

When applying the aggregation method to bursty traffic, the general scheme is exactly the same as for Bernoulli traffic. However, it becomes more complicated because of the input correlations and we have to add some additional random variables to describe the state of the input. Therefore, first we give the input state transition probabilities.

A. Input State Transitions

It is usually assumed that the durations of the busy and idle periods are random variables following geometric distribution with parameter p and q, respectively. An input port can be modeled as a two state Markov chain. When in the busy state, the probability that in the next time slot it will go to the idle state is p and the probability that it will stay in the busy state is $1-p$. Similarly, when in the idle state, the probability that in the next time slot it will go to the busy state is q and the probability that it will stay in the idle state is $1-q$. The average burst length L is $1/p$, and the average idle period length is $1/q$. Therefore the traffic load is $\rho = q/(p+q)$.

We use triple (α, β, χ) to represent the state of the input, where α is the number of cells arrived for output 1 to output I, β is the number of cells arrived for output $I+1$ and χ is the total number of input cells arrived at this switch. Here $I \in [1, N-1]$. (α, β, χ) is a Markov chain. The transition probability from state $(\alpha_0, \beta_0, \chi_0)$ to state $(\alpha_1, \beta_1, \chi_1)$ can be found as follows.

Let $p_{Rt}(r_t|\chi_0)$ be the probability that there are totally r_t input ports jumped from idle state to busy state, given that there were χ_0 busy input ports in the previous time slot.

$$p_{Rt}(r_t|\chi_0) = \binom{N-\chi_0}{r_t} q^{r_t}(1-q)^{N-\chi_0-r_t}, \qquad (23)$$

Given r_t, let $p_{Rs}(r_s|r_t)$ be the probability that out of these r_t newly turned busy input ports, r_s are sending cells to output 1 to output $I+1$

$$p_{Rs}(r_s|r_t) = \binom{r_t}{r_s}\left(\frac{I+1}{N}\right)^{r_s}\left(1-\frac{I+1}{N}\right)^{r_t-r_s}, \qquad (24)$$

since the destinations are randomly chosen. Again, given that there are r_s new busy inputs to output 1 to output $I+1$, the probability that there are r_1 inputs for output 1 to output I is

$$p_{R1}(r_1|r_s) = \binom{r_s}{r_1}\left(\frac{I}{I+1}\right)^{r_1}\left(1-\frac{I}{I+1}\right)^{r_s-r_1}, \qquad (25)$$

With Equations (23) to (25), the probability that there are totally r_t new busy input ports and r_1 of them go to output 1 to I and r_2 of them go to output $I+1$ is

$$p_R(r_1, r_2, r_t|\chi_0) = p_{Rt}(r_t|\chi_0)p_{Rs}(r_1+r_2|r_t)p_{R1}(r_1|r_1+r_2) \qquad (26)$$

Now let $p_{L1}(l_1|\alpha_0)$ be the probability that given there were α_0 input ports sending cells to output 1 to output I, l_1 of them jumped from busy state to idle state,

$$p_{L1}(l_1|\alpha_0) = \binom{\alpha_0}{l_1} p^{l_1}(1-p)^{\alpha_0-l_1}, \qquad (27)$$

Similarly, let $p_{L2}(l_2|\beta_0)$ be the probability that given there were β_0 input ports sending cells to output $I+1$, l_2 of them jumped from busy state to idle state,

$$p_{L2}(l_2|\beta_0) = \binom{\beta_0}{l_2} p^{l_2}(1-p)^{\beta_0-l_2}, \qquad (28)$$

There were $\delta_0 = \chi_0 - \alpha_0 - \beta_0$ input ports sending cells to output port $I+2$ to output N. Let $p_{Lr}(l_r|\delta_0)$ be the probability that l_r of these input ports jumped from busy state to idle state.

$$p_{Lr}(l_r|\delta_0) = \binom{\delta_0}{l_r} p^{l_r}(1-p)^{\delta_0-l_r}, \qquad (29)$$

With Equations (27) to (29), the probability that there are totally l_t input ports jumped from busy state to idle state and l_1 of them used to address output 1 to output I and l_2 of them used to address output $I+1$ is

$$p_L(l_1, l_2, l_t|\alpha_0, \beta_0, \chi_0) = p_{L1}(l_1|\alpha_0)p_{L2}(l_2|\beta_0)p_{Lr}(l_t - l_1 - l_2|\delta_0) \quad (30)$$

After obtaining Equations (26) and (30), $TI(\alpha_1, \beta_1, \chi_1; \alpha_0, \beta_0, \chi_0)$, which is the transition probability from $(\alpha_0, \beta_0, \chi_0)$ to state $(\alpha_1, \beta_1, \chi_1)$, can be found by

$$TI(\alpha_1, \beta_1, \chi_1; \alpha_0, \beta_0, \chi_0) = \sum p_L(l_1, l_2, l_t|\alpha_0, \beta_0, \chi_0)p_R(r_1, r_2, r_t|\chi_0) \quad (31)$$

over all possible (r_1, r_2, r_t) and (l_1, l_2, l_t) satisfying $\alpha_1 - \alpha_0 = r_1 - l_1$, $\beta_1 - \beta_0 = r_2 - l_2$ and $\chi_1 - \chi_0 = r_t - l_t$.

B. The Aggregation Method for Bursty Traffic

After obtaining the transition probability of the inputs, we can start deriving the transition probability of the entire switch. Since much of the technical details is similar to, if not the same as, the Bernoulli traffic case, we only outline the parts which are different.

When considering two queues, the state of the switch are represented by 5 random variables, $(\alpha, \beta, \chi, X, Y)$, where α is the number of busy inputs addressing output 1, β is the number of busy inputs addressing output 2, χ is the total number of busy inputs, X is the number of cells stored in queue 1 and Y is the number of cells stored in queue 2. The transition probability from state $(\alpha_0, \beta_0, \chi_0, X_0, Y_0)$ to intermediate state $(\alpha_1, \beta_1, \chi_1, X_1, Y_1)$ is

$$TI(\alpha_1, \beta_1, \chi_1; \alpha_0, \beta_0, \chi_0)T_{a,b}(X_1, Y_1; X_0, Y_0|\alpha_0, \beta_0)$$

where $TI(\alpha_1, \beta_1, \chi_1; \alpha_0, \beta_0, \chi_0)$ is given in Equation (31) and $T_{a,b}(X_1, Y_1; X_0, Y_0|\alpha_0, \beta_0)$ is given in Equation (3). If $X_1 + Y_1 > B$, same as in the Bernoulli traffic case, the Markov chain will move back to some state $(\alpha_1, \beta_1, \chi_1, X_2, Y_2)$ where $X_2 + Y_2 = B$. The probability of moving back from (X_1, Y_1) to (X_2, Y_2) is exactly the same as in the Bernoulli traffic.

After obtaining the transition matrix, the steady state distribution can be obtained. Like in Bernoulli traffic, the two queues will now be combined into one block. (X, Y) will be replaced by (S, U), where S is the number of cells stored in the block and U is the number of non-empty queues in the block. (α, β) will be replaced by $n = \alpha + \beta$ which is the total number of cells addressing outputs 1 and 2. The conditional probability that given that there are n cells for this block, it will go from (S_0, U_0) to (S_1, U_1) can be found in a similar way as in Bernoulli traffic, except that here be more precise, we also record the total number of cells arrived at the switch. Therefore the conditional probability will be $CT(S_1, U_1|S_0, U_0, n, T)$ where T is the total number of cells arrived at the switch and the rest of variables are defined in the same way as in Bernoulli traffic. The conditional probability concerning moving back is also added with two more conditions, the total number of cells arrived at the switch, T, and the total number of cells addressing the block, n, and is written as $CB(U_2|S_1, U_1, S_2, n, T)$.

When considering $I+1$ queues where $I \geq 2$, the state of the queues is represented with 6 random variables, $(\alpha, \beta, \chi, S, U, Z)$, where α is the number of busy inputs addressing output 1 to output I, β is the number of busy inputs addressing output $I+1$, χ is the total number of busy inputs, S is the number of cells stored in queue 1 to queue I, U is the number of non-empty queues from queue 1 to queue I and Z is the number of cells stored in queue $I+1$. The transition probability from state $(\alpha_0, \beta_0, \chi_0, S_0, U_0, Z_0)$ to intermediate state $(\alpha_1, \beta_1, \chi_1, S_1, U_1, Z_1)$ is

$$TI(\alpha_1, \beta_1, \chi_1; \alpha_0, \beta_0, \chi_0)T_{n,c}(S_1, U_1, Z_1; S_0, U_0, Z_0|\alpha_0, \beta_0)$$

where $T_{n,c}(S_1, U_1, Z_1; S_0, U_0, Z_0|\alpha_0, \beta_0)$ is given in Equation (11). Moving back is exactly the same as the Bernoulli traffic case. Then we combine the $I+1$ queues into one block: (S, U, Z) will be replaced by (S', U') where $S' = S + Z$ and $U' = U + u(Z)$, and (α, β) will be replaced by $n = \alpha + \beta$. After that the two conditional probabilities $CT(S_1, U_1|S_0, U_0, n, T)$ and $CB(U_2|S_1, U_1, S_2, n, T)$ will be updated.

C. The Rewinding Modification

The cell loss probability of bursty traffic is much larger than Bernoulli traffic. Therefore, when the buffer size is not large enough, the zero buffer assumption, or the assumption that there is no buffer dependency usually fails. In other words, some of the queues will grow to a very large size and the switch may have to drop cells in other queues. However, the aggregation method still works fairly well under these conditions. The reason is that, first, we only need the conditional transition probabilities, and to obtain these probabilities we will first go through a normalization procedure which tends to make them accurate. Another reason is that when using the zero buffer dependency assumption, at the I_{th} iteration, we are actually assuming there are only $I+1$ queues in the buffer. This assumption will become closer and closer to the truth as the iteration goes on, since when the number of considered queues increases, the number of unknown queues will decrease, and in the last step when $I = N-1$, there is indeed no interference from other queues since there is no other queue at all.

With the observations above, we can improve the accuracy of the aggregation method as follows. The behavior of a block containing I queues is found after the $(I-1)_{th}$ iteration and is fully described by the two conditional probabilities, $CT_I(S_1, U_1|S_0, U_0, n, T)$ and $CB_I(U_2|S_1, U_1, S_2, n, T)$. Here we added subscript I to indicate that these are the transition probabilities of a block containing I queues. For simplicity, in the following we write them as $CT_I()$ and $CB_I()$, respectively. After going through $N-1$ steps, we have obtained the $CT_I()$ and $CB_I()$ for all $2 \leq I \leq N$. Now we can consider the switch as two blocks, one with I queues and the other with $N-I$ queues. Since the behavior of these two blocks is known, the steady state distribution of the two blocks can be found. Note that the zero buffer dependency assumption is not needed at this time since all the queues in the switch are considered. With this we can find the probability that the second block stores j cells when the first block is in a certain state. With this probability we can recalculate the $CT_I()$ and $CB_I()$ without the zero buffer dependency assumption, since now we have the knowledge about the number of cells stored in other $N-I$ queues. This recalculation may start with $I=2$, then for $I=3$ to $I=N-1$. We call this "rewinding."

The rewinding modification is similar to a fixed point method: first make a guess about the solution, then use this guess to find a better solution in hope that it will indeed become better. We found that the rewinding will improve the accuracy of the aggregation method, and typically only one round of rewinding is needed. Therefore, in our program for bursty traffic this modification is included. However, for Bernoulli traffic it seems unnecessary because the model is already very accurate.

D. *Validations of the Model*

In Fig. 5 we show the cell loss probability and the average delay time of a switch with $N=4$ and $B=8$ as a function of the traffic load when the average burst length is 5 time slots. We can see that the results of the Aggregation Model are very close to the simulation. The results of other models are also shown. The cell loss probabilities of these models are close to the simulation, but the average delays are not. A detailed comparison will be given in SectionVI.

IV. The Aggregation Model for Optical Switches

The Aggregation Model can be readily applied to optical WDM switches. As in Section I, we model WDM switch with N input/output fibers and B shared FDLs as a switch with N input/output ports and Bk buffer locations, with each port capable of receiving/sending up to k cells at a time slot. By modifying the three equations governing the queuing behaviors, Equations (1), (2) and (10), the Aggregation Model can be used for this type of switch. For example, Equation (1) should be changed to

$$X_1 = \begin{cases} 0, & X_0 + a \le k \\ X_0 + a - k & \text{otherwise} \end{cases} \tag{32}$$

Equations (2) and (10) should also be modified in a similar way. Also, in the pair of random variables (S, U) which represent the state of an aggregated block of queues, U now should be interpreted as the number of cells ready to be transmitted in the block. Fig.6 shows the cell loss probability and the average delay of a WDM switch with 8 input/out fibers, 4 FDLs and 4 wavelengths per fiber under Bernoulli traffic. We can see that the Aggregation Model matches perfectly with the simulations.

V. Transmit First Switch

So far, we have considered receive first switches, which first accept all arrived cells, then transmit cells at the head of queues. There are also transmit first switches, which first transmit cells at the head of queues, then accept arrived cells. The extension of the Aggregation Model from the receive

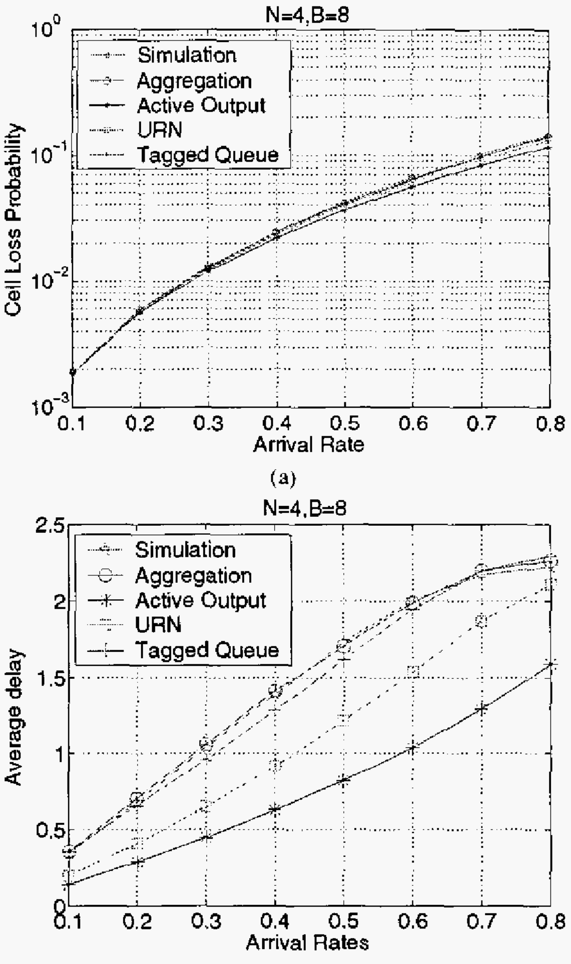


Fig. 5. Cell loss probability of and average delay under bursty traffic when $N=4$, $B=8$. The average burst length is 5 time slots.

first switch to the transmit first switch is immediate. Similar to Section IV, only Equations (1), (2) and (10), need to be modified. For example, Equation (1) should be changed to

$$X_1 = \begin{cases} a, & X_0 = 0 \\ X_0 + a - 1 & \text{otherwise} \end{cases} \tag{33}$$

which states the fact that no cell will be transmitted if the queue is empty, otherwise one cell will be transmitted. Equations (2) and (10) should also be modified in a similar way. Fig. 7 shows the analytical and simulation results for a electronic transmit first switch under Bernoulli traffic when $N=16$, $B=32$. Again we can see that the Aggregation Model is very accurate.

VI. Comparisons with other Existing Models

In this section, we compare our model with three other existing analytical models for shared buffer switches we are aware of. Note that all these models are for electronic switches, therefore, the comparison will be for electronic switches only.

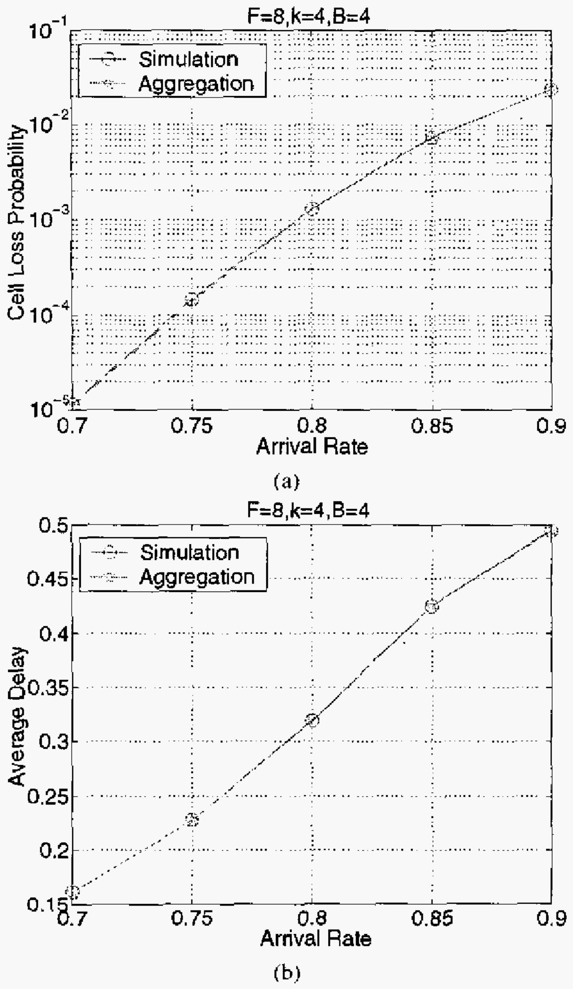


Fig. 6. Cell loss probability and average delay of a WDM switch under Bernoulli traffic when $F = 8$, $B = 4$, $k = 4$.

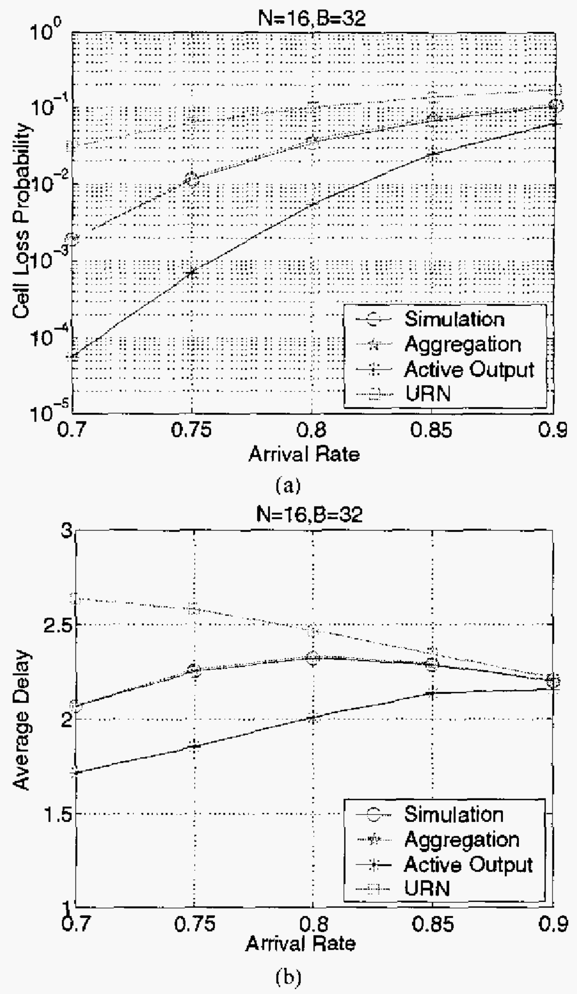


Fig. 7. Cell loss probability and average delay of a electronic transmit first switch under Bernoulli traffic when $N = 16$, $B = 32$.

To the best of our knowledge, previously there has not been any analytical model for optical WDM switches with shared buffer, and ours is the first such model. Also note that our model for WDM switches gives very accurate results, as can be seen in Fig.6,

As in [3], we call the three models the Active Output Model, URN Model, and the Tagged Queue Model. The Active Output Model was first introduced by Turner [1] and may be the earliest model for the shared buffer switches. In this model, it is assumed that the cells stored in the buffer have independent random destinations. The URN Model was introduced by Fong in [3] which assumed that all possible combinations of how cells were stored in the buffer are equally likely. The same assumption was also used in [4]. The Tagged Queue Model was introduced by Pattavina and Gianatti in [6], [5], which assumed that the queues in the buffer are independent of each other. We have implemented these models and shown their results along with our model and the simulations in the figures. We did not implement the vector method as it is of exponential complexity and, since known to be exact, it should yield the same results as the simulations.

From the figures, first we can see that our Aggregation Model is a very accurate model, since in every figure its curve matches perfectly with the simulation curve. Other three models perform well in some occasions, for example, in Fig.5(a) for cell loss probability under bursty traffic in a small switch; but perform not so well in other occasions, for example, in Fig.4 for both cell loss probability and average delay under Bernoulli traffic in a larger switch. Therefore, we can say that these three models are capable of giving satisfactory results under some type of traffic for some performance measures, but not for all types of traffic and all performance measures.

Next we consider the complexities of the models. As explained earlier, the complexity of the aggregation model is a polynomial function of the switch size. Under Bernoulli traffic, the Markov chain of the Aggregation Model is three dimensional with $(B + 1)(B + 2)(N - 1)/2$ states. Under bursty traffic, three more dimensions have to be added to describe the state of the input and the Markov chain is six dimensional with $O(N^3(B+1)(B+2)(N-1)/2)$ states. The complexity of the Active Output Model and the URN Model are also polynomial functions of the switch size. The Tagged Queue model, though also conceptually simple, has a time complexity growing exponentially with the switch size. To see this, we need to explain how the Tagged Queue model was implemented.

In this model, a queue called the "Tagged Queue" is singled out, and the rest $N-1$ queues are modeled as a block. The behavior of the tagged queue is given in Equation (1). The behavior of the block containing $N-1$ queues is described by $P_l(v|l)$ which is the conditional probability that when there are l cells in the block, v cells are ready to be transmitted. $P_l(v|l)$ is approximated by using the independence assumption of the queues. Let q_i be the number of cells stored in queue i. States satisfying $\sum_{i=2}^{N} q_i = l$ are said to be in set I_l. States in I_l satisfying $\sum_{i=2}^{N} u(q_i) = v$ are said to be in set $I_{l,v}$ where $u()$ is the step function. The probability that the block is in state $[q_2, q_3, \ldots, q_N]$ is approximated by using the independence assumption: $\prod_{i=2}^{N} P_Q(q_i)$, where $P_Q(x)$ is the probability that the tagged queue is storing x cells at the beginning of a time slot which can be found approximately with the zero buffer dependency assumption. The probability that the block is storing l cells and has v cells ready to transmit is approximated by summing probability over all the states in $I_{l,v}$. The probability that the block is storing l cells is approximated by summing probability over all the states in I_l. Then $P_l(v|l)$ is approximated as the ratio of the former over the latter. We can see that to find $P_l(v|l)$, one may have to go through all the possible states of the $N-1$ queues. With each queue having maximum length B, it will require $O(B^N)$ time. This number could be reduced by carefully avoiding redundant states, but will still be roughly in the order of $O(\frac{e^{\sqrt{B}}}{B})$ which is still exponential [10], since the number of different states it has to visit is the number of different ways to put B indistinguishable balls into $N-1$ indistinguishable boxes.

The advantage of singling out a tagged queue is that at the very least, the behavior of this tagged queue can be known quite well. This idea was first introduced in [6], [5], and is how the Tagged Queue Model got its name. The tagged queue idea was not used in other models originally, and it is incorporated here to improve their performance. We can see that these models are like our Aggregation Model in the last iteration, except for the differences in obtaining the behavior of the block. We obtain this information in a step by step manner, starting from a block containing only two queues, while other models make assumptions about the entire block.

As a conclusion, the Aggregation Model has an accuracy similar to the exact vector method and has polynomial time complexity, while the three other models are not accurate and may have exponential time complexity. In this sense, we can say the newly proposed aggregation model is the best analytical model for shared buffer switches.

Before ending this section, we would like to explain an interesting phenomena. It might have been noticed that the performances of the three models compared are better under Bursty traffic than under Bernoulli traffic. This is a surprise to us since the Bernoulli traffic is simpler than the Bursty traffic. In order to rule out the possibilities of coding errors, we did the following test. For Bernoulli traffic, we plugged the simulation $P_l(v|l)$ into the programs for these models and by doing so, both the cell loss probability curves and the average delay curves moved much closer to the simulation curve. Therefore, the reason that these three models are not performing well under Bernoulli traffic is because of their $P_l(v|l)$. In fact, we have compared the $P_l(v|l)$ obtained by these models with the simulation $P_l(v|l)$ and found that they do not agree well on most l and v. Since under bursty traffic the cell loss probabilities of these models are much closer to the simulations than under Bernoulli traffic, one might expect that the $P_l(v|l)$ of these models should become more accurate under bursty traffic. However, this is not the case. When comparing the $P_l(v|l)$, we found that they also do not agree well with simulations on most l and v, same as under Bernoulli traffic. The same fact can also be seen in [3]. The reason for this improvement, we feel, might be that two more random variables describing the state of the input were added, and that these models derive the cell loss probability based on the statistics of the tagged queue which is known relatively well. However, note that under bursty traffic the average delays given by these models are still not precise. The reason is that, to find average delay all queues have to be considered, and the final result will not be accurate if $P_l(v|l)$ is not.

VII. Conclusions

In this paper we have presented an analytical model called the Aggregation Model for switches with shared buffer. We considered both electronic and optical switches. This model finds the behavior of the queues in the shared buffer in a step by step manner. The time complexity of this model is a polynomial function of the switch size. We verified this model through extensive simulations and the results showed that it is very accurate, similar to the exact vector method which has exponential time complexity. In this sense, the new model is the best analytical model for shared buffer switches by far. In our future work we will find ways to further reduce the complexity of this model and consider other buffer management algorithms.

Acknowledgments

This work was supported in part by the U.S. National Science Foundation under grant numbers CCR-0073085, CCR-0207999 and ECS-0427345.

References

[1] J.S. Turner, "Queueing analysis of buffered switching networks," *IEEE Trans. Communications*, vol. 41, no. 2, pp. 412-420, 1993.

[2] J.A. Schormans and J.M. Pitts, "Overflow probability in shared cell switched buffers," *IEEE Communications Letters*, vol. 4, no. 5, pp. 167-169, 2000.

[3] S. Fong and S. Singh, "Performance analysis of shared buffer ATM switch with different cell departure models," *Proc. IEEE International Conference on Communications, 1998, ICC 98*, vol. 3, pp. 1824-1828, 1998.

[4] G. Bianchi and J.S. Turner, "Improved queueing analysis of shared buffer switching networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 482-490, 1993.

[5] A. Pattavina and S. Gianatti, "Performance analysis of ATM banyan networks with shared queueing. II. Correlated/unbalanced offered traffic," *IEEE/ACM Trans. Networking*, vol. 2, no. 4, pp. 411-424, 1994.

[6] S. Gianatti and A. Pattavina, "Performance analysis of ATM banyan networks with shared queueing. I. Random offered traffic," *IEEE/ACM Trans. Networking*, vol. 2, no. 4, pp. 398-410, 1994.

[7] S. Sharma and Y. Viniotis, "Optimal buffer managemen-t policies for shared-buffer ATM switches," *IEEE/ACM Trans. Networking*, vol. 7, no. 4, pp. 575-587, 1999.

[8] S.C. Liew, "Performance of various input-buffered and output-buffered ATM switch design principles under bursty traffic: simulation study," *IEEE Trans. Communications*, vol. 42, no. 2/3/4, pp. 1371-1379, 1994.

[9] C.S. Lin, B.D. Liu and Y.C. Tang, "Design of a shared buffer management scheme for ATM switches," *Proc. 15th Annual IEEE International ASIC/SOC Conference, 2002*, pp: 261-264, Sept. 2002.

[10] S. Ahlgren and K. Ono, "Addition and Counting: The Arithmetic of Partitions," *Notices of the American Mathematical Society*, vol. 48, no. 9, pp. 978-984, Oct. 2001.

[11] Y.N. Singh, A. Kushwaha and S.K. Bose, "Exact and approximate analytical modeling of an FLBM-based all-optical packet switch," *Journal of Lightwave Technology*, vol. 21, no. 3, pp. 719-726, March 2003.

[12] L. Xu, H. G. Perros and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Communications Magazine*, pp. 136-142, 2001.

[13] S.L. Danielsen, et. al, "Analysis of a WDM packet switch with improved performance under bursty traffic conditions due to tunable wavelength converters," *Journal of Lightwave Technology*, vol. 16, no. 5, pp. 729-735, 1998.

[14] S.L. Danielsen, et. al, "WDM packet switch architectures and analysis of the influence of tunable wavelength converters on the performance," *Journal of Lightwave Technology*, vol. 15, no. 2, pp. 219-227, 1998.

[15] D. K. Hunter and I. Andronovic, "Approaches to optical Internet packet switching," *IEEE Communications Magazine*, vol. 38, no. 9, pp. 116-122, 2000.