A Novel Analytical Model for Switches With Shared Buffer

Zhenghao Zhang, *Student Member, IEEE*, and Yuanyuan Yang, *Senior Member, IEEE*

***Abstract*—Switches with a 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 of shared buffer switches has been studied 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. The model is based on the idea of induction: first find the behavior of two queues, then aggregate them into one block; then find the behavior of three queues while regarding two of the queues as one block, then aggregate the three queues into one block; then aggregate four queues, and so on. When all queues have been aggregated, the behavior of the entire switch will be found. This model has perfect accuracies under all tested conditions and has polynomial complexity.**

***Index Terms*—Aggregation, analytical model, Markov chain, shared buffer, switches.**

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, over the years, the performance analysis of shared buffer switches has intrigued many researchers and has become a classical problem in the literature. However, it has not been adequately solved. Existing models are either accurate but have exponential complexities or not very accurate.

A shared buffer switch is shown in Fig. 1(a). It has N input/output ports and a common buffer pool with B cell locations. It receives fixed length cells from the input ports and operates in a time-slotted manner. The arriving cells are multiplexed and stored in the buffer, where they are organized into N separate first-in first-out queues, one for each output port. The difficulty in analytically modeling a shared buffer switch is due to the strong dependence of queues in the buffer, which means that the numbers of cells stored in the N queues are a set of random variables strongly depending on each other. The dependence comes from both the dependence of the input and the dependence of the finite size buffer. The dependence of the input means, for example, that if there are x cells destined for output 1, then there

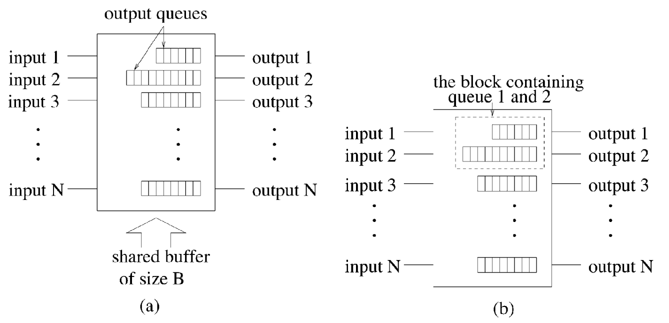


Fig. 1. (a). A shared buffer switch with N input ports and N output ports and a common buffer pool with B cell locations. (b). After the behaviors of queue 1 and queue 2 are found, consider them as one block.

cannot be over $N - x$ cells destined for other outputs at the same time slot. In other words, the knowledge of the number of cells destined to one output contains much information about the number of cells destined to other outputs. The dependence 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 dependence of the queues makes 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 element being the number of cells stored in each queue [5]. However, this makes the number of states grow exponentially with N. To make the analysis tractable, researchers tried different approximation models based on different assumptions about the queues or the cells. In [5] and [6], 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] and [4], it was assumed that all possible combinations for cells to be distributed in the buffer are equally likely. Since queues and cells may behave according to these assumptions only in some cases, the resulting analytical models perform satisfactorily only in some cases but not in all cases. In this paper, we will give the Aggregation model which, although is also an approximation model, does not make assumptions about dependencies of the queues nor about how cells are distributed and is very accurate in all tested cases. It is also efficient since its complexity is a polynomial function of the switch size.

Although the technical details and mathematical interpretations could be lengthy, the idea of this model 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 the case when the buffer is very large such that the cell loss probability will be very small and the buffer dependence can be neglected. However, note that the queues are still dependent on each other due to the input dependence. Now, if we only consider two queues in the

Manuscript received May 3, 2005; revised May 15, 2006; approved by IEEE/ACM Transactions on Networking Editor F. Neri. This work was supported in part by the U.S. National Science Foundation under Grants CCR-0073085 and CCR-0207999.

The authors are with the Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY 11794-2350 USA (e-mail: zhhzhang@ece.sunysb.edu; yang@ece.sunysb.edu).

Digital Object Identifier 10.1109/TNET.2007.896250

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 since the input traffic patterns to them are known (by the assumptions about input traffic) and other queues will grow so large such that some of the input cells to outputs 1 and 2 have to be dropped. After finding out the behaviors of these two queues, they can be "aggregated" together into a single block. This block will store the cells for output 1 and output 2, as illustrated in Fig. 1(b), and its behavior can be deduced from the behaviors of the two queues. Now consider three queues: queues for outputs 1, 2, and 3. Instead of three separate queues, they can be regarded as two components: one component is the block containing queue 1 and queue 2, and the other component is queue 3. Since the behaviors of both components are known, the behaviors of the three queues can be found quite precisely. Then again, the three queues can be aggregated into one block, which can be used to find the behavior of four queues. The process can be carried on until all N queues have been aggregated into one block, and by this time, the queueing properties of the entire switch will have been found. Note that although the idea is illustrated by assuming the buffer is large, we will show later that it can also be used for switches with small 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 block and the new queue. The whole process takes N steps; therefore, the complexity is a polynomial function of the switch size. Also note that although queues are dependent upon each other, they interact in such a way that after each step some unnecessary details can be omitted, thus reducing the complexity of the model. For example, when aggregating two queues, only the probability that queue 1 and queue 2 are of size $x+y$ is needed rather than the probability that queue 1 is of size x and queue 2 is of size 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 queue 1 and queue 2.

In this paper, we will mainly illustrate the Aggregation model under the widely used Bernoulli traffic and the ON–OFF Markovian traffic. It should be mentioned that although today's routers are mainly input-queued switches, the shared buffer switch remains an important option for switching and will likely become even more important when all-optical packet switching becomes mature. A preliminary version of this work can be found in [16].

The rest of the paper is organized as follows. Section II describes the operations of the switch. Section III describes the Aggregation model under uniform Bernoulli traffic. Section IV describes the Aggregation model under ON–OFF Markovian traffic. Section V shows how to apply the Aggregation model to optical WDM switches. Section VI shows how to apply the Aggregation model to a type of switch called the transmit-first switch. Section VII gives detailed comparisons between the Aggregation model and other existing models and also gives reasons for the very good accuracy of the Aggregation model. Finally, Section VIII concludes the paper.

II. Switch Operations

In this section, we describe the operations of a shared buffer switch. As shown in Fig. 1(a), the switch is assumed to have N input/output ports and a common buffer pool with B cell locations. It receives fixed length cells from the input ports and operates in a time-slotted manner. There are N first-in first-out queues in the buffer, one for each output port.

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. Such a switch is called the "receive-first switch" and is denoted as RFS. There is also the "transmit-first switch," or TFS, in which phase 1 and phase 2 are reversed. The Aggregation model can be applied to both the RFS and the TFS, and we will mainly illustrate it with the RFS.

Before executing phase 3, the switch 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 of which cells should be dropped 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, or be "pushed out." In general, algorithms that allow pushout have better performance than those do not [7]. The advantage of the random drop algorithm is that cells in longer queues are more likely to be dropped than cells in shorter queues, thus preventing some queues from completely occupying the buffer and starving others.

III. Aggregation Model Under Bernoulli Traffic

In this section, we will illustrate the Aggregation model using uniform Bernoulli traffic. The assumptions of the traffic are as follows:

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

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

A. Overview

There are N virtual queues in the buffer, one for each output. 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. (X, Y) can be regarded as a two-dimensional Markov chain, whose transition rate and steady-state distribution can be found by the assumptions of the input traffic. Then, the two queues can be combined into one block that stores the cells for output 1 and output 2. The state of the block can be represented by another pair of random variables (S, U), where S is the number of cells stored in the block and U is the number of nonempty queues in the block. To describe the behavior of the block, two conditional probabilities are computed $CT(S_m, U_m | S_0, U_0, n)$ and $CB(U_1 | S_m, U_m, S_1)$.

$CT(S_m, U_m|S_0, U_0, n)$ is the probability that the block goes from state (S_0, U_0) to intermediate state (S_m, U_m) after receiving n cells. $CB(U_1|S_m, U_m, S_1)$ is the probability that the block will have U_1 nonempty queues after dropping $S_m - S_1$ cells in intermediate state (S_m, U_m). Note that here we have made the first of the two assumptions in the Aggregation model, which is that *the transition of* (S, U) *over time has Markov property*. A detailed discussion with regard to this assumption is provided in Section VII-B.

When considering three queues, queue 1 to queue 3, the first two queues can be regarded as a block and the state of the three queues can be represented by (S, U, Z), where S is the number of cells stored in the block, U is the number of nonempty 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 and the steady-state distribution of this three-dimensional Markov chain can be found. Similar to the previous step, the three queues can now be combined into a single block and only two random variables (S, U) can be used 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 nonempty queues from queue 1 to queue 3. The two conditional probabilities used to describe the behavior of the new block (containing three queues) $CT(S_m, U_m|S_0, U_0, n)$ and $CB(U_1|S_m, U_m, S_1)$ can be found, where $CT(S_m, U_m|S_0, U_0, n)$ is the probability that the block goes from state (S_0, U_0) to intermediate state (S_m, U_m) after receiving n cells, and $CB(U_1|S_m, U_m, S_1)$ is the probability that the block will have U_1 nonempty queues after dropping $S_m - S_1$ cells. When considering four queues, again the first three queues will be considered as one block, and the state of the four queues can be represented by triple (S, U, Z). The process can be carried on until all N queues have been combined into one single block. Then, useful information such as cell loss probability and average delay can be found with the behavior of the N-queue block.

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 the queue for output 1 and the 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). The probability that there are a cells arrived for output 1 and b cells arrived for output 2 follows multinomial distribution and can be found as

$$p_{a,b}(a,b) = \frac{N!}{a!b!(N-a-b)!}\left(\frac{\rho}{N}\right)^{a+b}\left(1-\frac{2\rho}{N}\right)^{(N-a-b)}. \tag{1}$$

After receiving a cells and sending out 1 cell if not empty, queue 1 will have $X_m = [X_0 + a - 1]^+$ cells, where $[x]^+ = x$ if $x \geq 0$ and $[x]^+ = 0$ if $x = 0$. Similarly, queue 2 will have $Y_m = [Y_0 + b - 1]^+$ cells.

If there is no buffer overflow in the switch, the next state of the Markov chain is (X_m, Y_m). Otherwise, some of the cells may be dropped by the buffer management algorithm, and the two queues may store fewer cells than X_m and Y_m. Note that 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 no information about other queues. Therefore, some approximation has to be used. We assume that *if* $X_m + Y_m \leq B$, *no cells will be dropped from queue 1 and queue 2; otherwise* $X_m + Y_m - B$ *cells will be dropped from queue 1 and queue 2*. This assumption is the second of the two assumptions used in the Aggregation model and can be called the "zero external interference assumption," since it is equivalent to assuming that only queue 1 and queue 2 are in the buffer, or, the "external queues," i.e., queue 3 to queue N, will never grow to a size large enough to interfere with queue 1 and queue 2. A detailed discussion with regard to this assumption is provided in Section VII-B. The probability that the switch dropped v_x cells from queue 1 and v_y cells from queue 2, where $v_x + v_y = X_m + Y_m - B$, $0 \leq v_x \leq X_m$, and $0 \leq v_y \leq Y_m$ is

$$P\_D(v_x, v_y; X_m, Y_m) = \frac{\binom{X_m}{v_x}\binom{Y_m}{v_y}}{\binom{X_m + Y_m}{X_m + Y_m - B}}. \tag{2}$$

Combining these discussions, the transition rate of the Markov chain from state (X_0, Y_0) to state (X_1, Y_1) can be determined as follows. We say integer a satisfies the queuing relation defined by X_m and X_0 if $X_m = [X_0 + a - 1]^+$ and denote the set of such integers as $Q(X_m, X_0)$. $Q(X_m, X_0)$ is introduced for presentational convenience only. In fact, for most values of X_m and X_0, $Q(X_m, X_0)$ is simply $\{X_m - X_0\}$; only when $X_m = X_0 = 0$ will $Q(X_m, X_0)$ have two elements which are 0 and 1. Clearly, if $X_1 + Y_1 < B$, the transition rate is

$$\sum_{(a,b)} p_{a,b}(a,b) \tag{3}$$

where $a \in Q(X_1, X_0)$ and $b \in Q(Y_1, Y_0)$, since if $X_1 + Y_1 < B$, according to our assumption, no cells would have been dropped from queue 1 and queue 2 and the probability that the Markov chain will go from (X_0, Y_0) to (X_1, Y_1) is the probability that the two queues have received a proper number of cells. If $X_1 + Y_1 = B$, some cells may have been dropped from the queues, i.e., the queues may have visited some intermediate state (X_m, Y_m) before reaching (X_1, Y_1), where $X_m \geq X_1$, $Y_m \geq Y_1$. The probability that they will go to intermediate state (X_m, Y_m) is $\sum_{(a,b)} p_{a,b}(a,b)$, where $a \in Q(X_m, X_0)$ and $b \in Q(Y_m, Y_0)$. At intermediate state (X_m, Y_m), to go to state (X_1, Y_1), there must be $X_m - X_1$ cells dropped from queue 1 and $Y_m - Y_1$ cells dropped from queue 2, which occurs with probability $P\_D(X_m - X_1, Y_m - Y_1; X_m, Y_m)$. Therefore, the transition probability when $X_1 + Y_1 = B$ is

$$\sum_{(X_m, Y_m)} \sum_{(a,b)} P\_D(X_m - X_1, Y_m - Y_1; X_m, Y_m) p_{a,b}(a,b) \tag{4}$$

where $a \in Q(X_m, X_0)$, $b \in Q(Y_m, Y_0)$, $X_m \geq X_1$, $Y_m \geq Y_1$.

After obtaining the transition rates, the steady-state distribution of the Markov chain, $\pi(X, Y)$, can be easily found. As described earlier, we will now combine these two queues into one block. Another pair of random variables (S, U) will be used as the state of this block, where S is the number of cells stored in this block and U is the number of nonempty queues in this block, where $0 \leq S \leq B$, $0 \leq U \leq 2$. For a given (S, U), (X, Y) that satisfies $X + Y = S$ and $u(X) + u(Y) = U$ is called a "substate" of (S, U), where $u()$ is the step function

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

There can be more than one substate of (S, U), and all such substates 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 $\sum_{(X,Y)} \pi(X, Y)$, where (X, Y) denotes the substate of (S, U).

For future computations, two conditional probabilities will be needed to describe the behavior of the block. The first one is $CT(S_m, U_m \mid S_0, U_0, n)$, the probability that the block will go from state (S_0, U_0) to intermediate state (S_m, U_m) after receiving n cells. (Note that the intermediate state (S_m, U_m) is the state of the block before the buffer management algorithm is run and is not a state of the Markov chain.) Let (X_0^i, Y_0^i) denote the ith substate of (S_0, U_0) and (X_m^j, Y_m^j) denote the jth substate of (S_m, U_m). Note that given the block is in state (S_0, U_0), the probability that it is in substate (X_0^i, Y_0^i) is $\frac{\pi(X_0^i, Y_0^i)}{\pi(S_0, U_0)}$. Let $T(X_m^j, Y_m^j \mid X_0^i, Y_0^i, n)$ denote the probability that after receiving n cells, the two queues will go from (X_0^i, Y_0^i) to (X_m^j, Y_m^j). Clearly, we have

$$CT(S_m, U_m \mid S_0, U_0, n) = \sum_i \sum_j \frac{\pi\left(X_0^i, Y_0^i\right)}{\pi\left(S_0, U_0\right)} \times T\left(X_m^j, Y_m^j \mid X_0^i, Y_0^i, n\right). \quad (5)$$

Also note that

$$T\left(X_m^j, Y_m^j \mid X_0^i, Y_0^i, n\right) = \sum_{(a,b)} \binom{n}{a} 0.5^a 0.5^b \quad (6)$$

where $a + b = n$ and $a \in Q(X_m^j, X_0^i)$ and $b \in Q(Y_m^j, Y_0^i)$, and the expression inside the summation is the probability that out of the n arriving cells, a go to output 1 and b go to output 2.

The other one is $CB(U_1 \mid S_m, U_m, S_1)$, which is the probability that the block will have U_1 nonempty queues after dropping $S_m - S_1$ cells at intermediate state (S_m, U_m). Note that the probability that the two queues are in intermediate state (X_m, Y_m) is

$$\sum_{(X_0,Y_0)} \sum_{(a,b)} \pi(X_0, Y_0) p_{a,b}(a, b) \quad (7)$$

where $a \in Q(X_m, X_0)$ and $b \in Q(Y_m, Y_0)$. The probability that the block is in intermediate state (S_m, U_m) is $\pi(S_m, U_m) = \sum_{(X_m,Y_m)} \pi(X_m, Y_m)$, where (X_m, Y_m) denotes the substate of (S_m, U_m). We have

$$CB(U_1 \mid S_m, U_m, S_1) = \sum_j \sum_k \frac{\pi\left(X_m^j, Y_m^j\right)}{\pi(S_m, U_m)} \times D\left(X_1^k, Y_1^k \mid X_m^j, Y_m^j, S_m - S_1\right) \quad (8)$$

where (X_m^j, Y_m^j) denotes the jth substate of (S_m, U_m), (X_1^k, Y_1^k) denotes the kth substate of (S_1, U_1), and $D(X_1^k, Y_1^k \mid X_m^j, Y_m^j, S_m - S_1)$ is the probability that after dropping $S_m - S_1$ cells, the two queues will go from (X_m^j, Y_m^j) to (X_1^k, Y_1^k). Clearly

$$D\left(X_1^k, Y_1^k \mid X_m^j, Y_m^j, S_m - S_1\right) = P\_D\left(X_m^j - X_1^k, Y_m^j - Y_1^k; X_m^j, X_m^j\right). \quad (9)$$

2) Iteration—More Queues: Suppose I queues have been aggregated into one block. After finishing computing for two queues $I = 2$. We can now study $I + 1$ queues by regarding queue 1 to queue I as a block and use (S, U, Z) to represent the state of the queues, where S is the number of cells stored in the block, U is the number of nonempty queues in the block, and Z is the number of cells stored in queue $I + 1$.

Suppose initially the Markov chain is in state (S_0, U_0, Z_0). The probability that there are n cells arrived for output 1 to output I and c cells arrived for output $I + 1$ also follows multinomial distribution and can be found as

$$p_{n,c}(n, c) = \frac{N!}{n!c!(N - n - c)!} \left(\frac{I\rho}{N}\right)^n \left(\frac{\rho}{N}\right)^c \times \left(1 - \frac{(I + 1)\rho}{N}\right)^{(N-n-c)}. \quad (10)$$

After receiving c cells and sending out 1 cell if not empty, queue $I + 1$ will contain $Z_m = [Z_0 + c - 1]^+$ cells. After receiving n cells and sending out cells in nonempty queues, the probability that the first I queues will contain S_m cells and will have U_m nonempty queues is $CT(S_m, U_m | S_0, U_0, n)$, which is the conditional probability obtained in the previous iteration.

Similarly to the two-queue case, we make the "zero external interference assumption," which means that if $S_1 + Z_1 \leq B$, no cells will be dropped from queue 1 to queue $I + 1$; otherwise $S_1 + Z_1 - B$ cells will be dropped from these queues. The transition rate of the Markov chain from state (S_0, U_0, Z_0) to state (S_1, U_1, Z_1) can be determined as follows. If $S_1 + Z_1 < B$, no cells would have been dropped from these queues, and the transition rate is simply

$$\sum_{(n,c)} p_{n,c}(n, c) CT(S_1, U_1 | S_0, U_0, n) \quad (11)$$

where $c \in Q(Z_1, Z_0)$. If $S_1 + Z_1 = B$, some cells may have been dropped from the queues, i.e., the queues may have visited some intermediate state (S_m, U_m, Z_m) before reaching (S_1, U_1, Z_1), where $S_m \geq S_1$, $U_m \geq U_1$ and $Z_m \geq Z_1$. The probability that they will go to intermediate state (S_m, U_m, Z_m) is $\sum_{(n,c)} p_{n,c}(n, c) CT(S_m, U_m | S_0, U_0, n)$, where $c \in Q(Z_m, Z_0)$. At intermediate state (S_m, U_m, Z_m), to go to state (S_1, U_1, Z_1), first there must be $S_m - S_1$ cells dropped from the block and $Z_m - Z_1$ cells dropped from queue $I + 1$, which occurs with probability $P\_D(S_m - S_1, Z_m - Z_1; S_m, Z_m)$; second, after dropping $S_m - S_1$ cells, the block must contain U_1 nonempty queues, which occurs with probability

$CB(U_1 \mid S_m, U_m, S_1)$ and is obtained from the previous iteration. Therefore, the transition probability when $S_1 + Z_1 = B$ is

$$\begin{aligned}\sum_{(S_m,U_m,Z_m)} \sum_{(n,c)} & P\_D(S_m - S_1, Z_m - Z_1; S_m, Z_m) \\ & \times CB(U_1 \mid S_m, U_m, S_1) p_{n,c}(n,c) \\ & \times CT(S_m, U_m \mid S_0, U_0, n) \end{aligned} \tag{12}$$

where $S_m \geq S_1, U_m \geq U_1, Z_m \geq Z_1, c \in Q(Z_m, Z_0)$.

After obtaining the transition rates, 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 nonempty queues from queue 1 to $I + 1$. (We use a prime to denote variables and expressions related to the block with $I + 1$ queues.) The probability that the block is in state (S', U') can be found by $\pi'(S', U') = \sum \pi(S, U, Z)$, 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 substate of (S', U').

Similar to the two-queue case, two conditional probabilities are needed for the next iteration. The first one is $CT'(S'_m, U'_m \mid S'_0, U'_0, n')$, which is the probability that when n' cells arrived for output 1 to output $I + 1$, the block will go from (S'_0, U'_0) to (S'_m, U'_m). Let (S^i_0, U^i_0, Z^i_0) denote the ith substate of (S'_0, U'_0) and (S^j_m, U^j_m, Z^j_m) denote the jth substate of (S'_m, U'_m). Similar to the two-queue case, we have

$$\begin{aligned} CT'(S'_m, U'_m \mid S'_0, U'_0, n') = \sum_i \sum_j & \frac{\pi\left(S^i_0, U^i_0, Z^i_0\right)}{\pi(S'_0, U'_0)} \\ & \times T\left(S^j_m, U^j_m, Z^j_m | S^i_0, U^i_0, Z^i_0, n\right) \end{aligned} \tag{13}$$

where $T(S^j_m, U^j_m, Z^j_m \mid S^i_0, U^i_0, Z^i_0, n')$ is the probability that after receiving n' cells, the $I + 1$ queues go from (S^i_0, U^i_0, Z^i_0) to (S^j_m, U^j_m, Z^j_m), and

$$\begin{aligned} T & \left(S^j_m, U^j_m, Z^j_m \mid S^i_0, U^i_0, Z^i_0, n'\right) \\ & = \sum_{(n,c)} CT\left(S^j_m, U^j_m \mid S^i_0, U^i_0, n\right) \\ & \quad \times \binom{n'}{n} \left(\frac{I}{I+1}\right)^n \left(\frac{1}{I+1}\right)^c \end{aligned} \tag{14}$$

where $n + c = n'$ and $c \in Q(Z^j_m, Z^i_0)$.

The other one is $CB'(U'_1 \mid S'_m, U'_m, S'_1)$, which is the probability that the block will have U'_1 nonempty queues after dropping $S'_m - S'_1$ cells at intermediate state (S'_m, U'_m). Note that the probability that the $I + 1$ queues are in a certain intermediate state, i.e., $\pi(S^j_m, U^j_m, Z^j_m)$ and $\pi(S'_m, U'_m)$, can be found similarly to the two-queue case. Also, as in the two-queue case, we have

$$\begin{aligned} CB'(U'_1 \mid S'_m, U'_m, S'_1) = \sum_j \sum_k & \frac{\pi(S^j_m, U^j_m, Z^j_m)}{\pi(S'_m, U'_m)} \\ & \times D\left(S^k_1, U^k_1, Z^k_1 \mid S^j_m, U^j_m, Z^j_m, S'_m - S'_1\right) \end{aligned} \tag{15}$$

where (S^j_m, U^j_m, Z^j_m) denotes the jth substate of (S'_m, U'_m), (S^k_1, U^k_1, Z^k_1) denotes the kth substate of (S'_1, U'_1), and $D(S^k_1, U^k_1, Z^k_1 \mid S^j_m, U^j_m, Z^j_m, S'_m - S'_1)$ is the probability that after dropping $S'_m - S'_1$ cells, the $I + 1$ queues will go from (S^j_m, U^j_m, Z^j_m) to (S^k_1, U^k_1, Z^k_1). We have

$$\begin{aligned} D & \left(S^k_1, U^k_1, Z^k_1 \mid S^j_m, U^j_m, Z^j_m, S'_m - S'_1\right) \\ & = P\_D\left(S^j_m - S^k_1, Z^j_m - Z^k_1; S^j_m, Z^j_m\right) \\ & \quad \times CB\left(U^k_1 \mid S^j_m, U^j_m, S^k_1\right). \end{aligned} \tag{16}$$

Finally, let $I = I + 1$

$$\begin{aligned} CT(S_m, U_m \mid S_0, U_0, n) &= CT'(S'_m, U'_m \mid S'_0, U'_0, n') \\ CB(U_1 \mid S_m, U_m, S_1) &= CB'(U'_1 \mid S'_m, U'_m, S'_1) \end{aligned}$$

and continue to the next iteration.

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 nonempty queues. $CT(S_m, U_m \mid S_0, U_0, n)$ can also be obtained, which is the probability that when there are S_0 cells and U_0 nonempty queues in the buffer, after receiving n cells and sending out cells in nonempty queues, the switch will store S_m cells and has U_m nonempty queues. 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

$$\begin{aligned} \alpha = \sum_{S_0=0}^{B} \sum_{U_0=0}^{N} \sum_{n=0}^{N} \sum_{S_m=0}^{B} \sum_{U_m=0}^{N} & \pi(S_0, U_0) \\ & \cdot CT(S_m, U_m \mid S_0, U_0, n)[S_m - B]^+ p_n(n) \end{aligned} \tag{17}$$

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

$$p_n(n) = \binom{N}{n} \rho^n (1 - \rho)^{N-n} \tag{18}$$

for $n \in [0, N]$. To see why (17) holds, note that the probability that the switch is in state (S_0, U_0) is $\pi(S_0, U_0)$. After receiving n new cells and sending out cells in nonempty queues, the probability that it will go to intermediate state (S_m, U_m) is $CT(S_m, U_m \mid S_0, U_0, n)$. If $S_m \leq B$, no cell will be dropped; otherwise, exactly $S_m - B$ will be dropped.

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 sent out. Since the buffer management algorithm we adopt allows pushout, cells that are in buffer could be dropped without being sent out. Therefore, we modify the definition of delay time as the number of time slots a cell stays in the buffer before being sent out or being dropped.

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

$$E(W) = E(S)/(N\rho) \tag{19}$$

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

$$E(S) = \sum_{S=0}^{B} \sum_{U=0}^{N} S\pi(S,U). \tag{20}$$

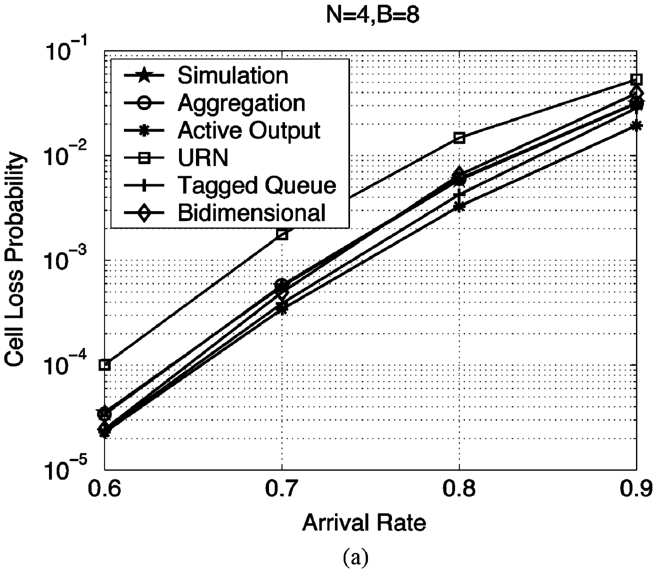
D. Validation of the Model

We have conducted extensive simulations to validate the Aggregation model. In our simulations, if the cell loss probability is in the order of 10^{-4} or higher, the simulation program is run for 10 000 000 time slots. If the cell loss probability is below 10^{-4} (only in three cases in this paper), the simulation program is run until it has encountered 1000 lost cells. The number of times slots run in the simulations is determined by the cell loss probability because cell loss is a rare event, while cell delay can be collected for each cell. For comparison purpose, we have also implemented four other models, called the Active Input model [1], the URN model [3], the Tagged Queue model [5], and the Bidimensional model [4]. Figs. 2 and 3 show the analytical results along with the simulation results. We can see that for both cell loss probability and average delay, and for both the small switch and the large switch, the Aggregation model is very accurate since its curves almost overlap with the simulation curves. On the other hand, the performances of other models may be good when the switch is small (Fig. 2, $N = 4$ and $B = 8$) but will deteriorate when the switch becomes large (Fig. 3, $N = 16$ and $B = 32$). We did not provide the results for the Tagged Queue model in Fig. 3 because it is actually of exponential complexity and takes too much time for larger switches. A detailed comparison of the Aggregation model and other models will be given in Section VII.

IV. The Aggregation Model Under On–Off Markovian Traffic

So far, we have considered the Aggregation model under Bernoulli traffic. The idea of aggregation can also be applied to other traffic types, and in this section, we will illustrate how to apply it to *on–off Markovian traffic*, which is also widely used in the literature. Under this type of traffic, an input port alternates between two states, the "idle" state and the "busy" state. In a time slot, if the input port is in "idle" state, there will be no arriving cell at this input port; otherwise, that is, if the input port is in "busy" state, there will be one arriving cell at this input port. Cells arriving in consecutive busy time slots are called a stream, and all cells belonging to the same stream go to the same destination. Since the state transition is Markovian, an input port can be modeled as a two-state Markov chain. When in the busy state, the probability that it will go to the idle state in the next time slot 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 it will go to the busy state in the next time slot is q, and the probability that it will stay in the idle state is $1 - q$. It can be derived that the average stream length is $1/p$, the average idle period length is $1/q$, and the traffic load is $\rho = q/(p + q)$.

When applying the aggregation method to on–off Markovian traffic, the general scheme is exactly the same as that for Bernoulli traffic. However, it becomes more complex because



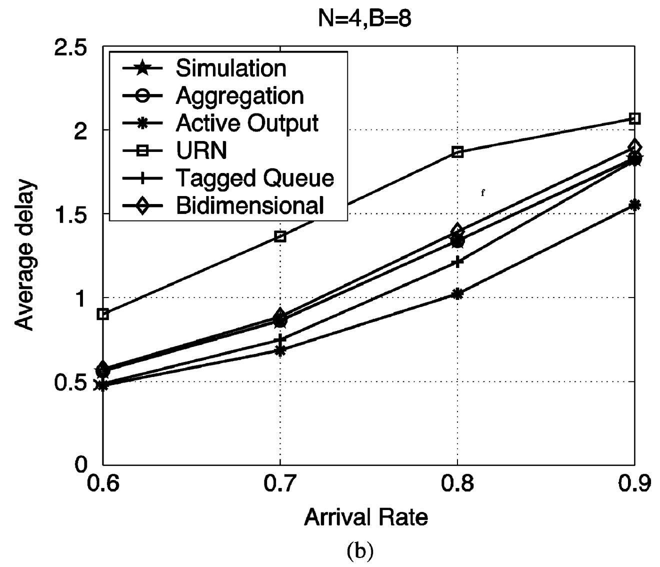


Fig. 2. Cell loss probability and average delay under Bernoulli traffic when $N = 4, B = 8$.

the inputs at different time slots are no longer independent. Therefore, some random variables have to be added to describe the state of the input.

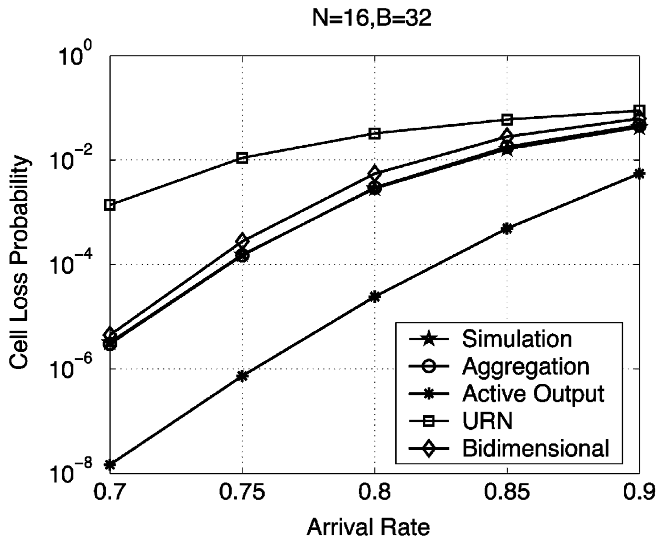
A. Input State Transitions

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 for $I \in [1, N]$. (α, β, χ) is a Markov chain. The transition rate from state $(\alpha_0, \beta_0, \chi_0)$ to state $(\alpha_1, \beta_1, \chi_1)$ can be found as follows.

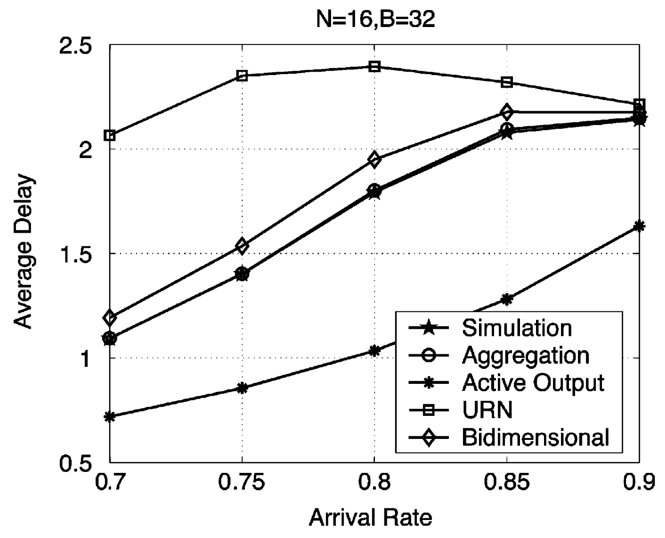
Let $p_{Rt}(r_t \,|\, \chi_0)$ be the probability that there are totally r_t input ports jumped from the idle state to the busy state, given that there were χ_0 busy input ports in the previous time slot.

$$p_{Rt}(r_t \,|\, \chi_0) = \binom{N - \chi_0}{r_t} q^{r_t} (1 - q)^{N - \chi_0 - r_t}. \tag{21}$$

Given r_t, the probability that there are r_1 new busy input ports sending cells to output 1 to I and r_2 new busy input ports



(a)



(b)

Fig. 3. Cell loss probability and average delay under Bernoulli traffic when $N = 16, B = 32$.

sending cells to output $I + 1$ is

$$p_{R1R2}(r_1, r_2 \,|\, r_t) = \frac{r_t!}{r_1! r_2! (r_s - r_1 - r_2)!} \times \left(\frac{I}{N}\right)^{r_1} \left(\frac{1}{N}\right)^{r_2} \left(1 - \frac{(I+1)}{N}\right)^{(r_t - r_1 - r_2)}. \quad (22)$$

Therefore, the probability that there are totally r_t new busy input ports and among them there are r_1 input ports sending cells to output 1 to I and r_2 input ports sending cells to output $I + 1$ is

$$p_R(r_1, r_2, r_t \,|\, \chi_0) = p_{R1R2}(r_1, r_2 \,|\, r_t) p_{Rt}(r_t \,|\, \chi_0). \quad (23)$$

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 the busy state to the idle state

$$p_{L1}(l_1 \,|\, \alpha_0) = \binom{\alpha_0}{l_1} p^{l_1} (1-p)^{\alpha_0 - l_1}. \quad (24)$$

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 the busy state to the idle state

$$p_{L2}(l_2 \,|\, \beta_0) = \binom{\beta_0}{l_2} p^{l_2} (1-p)^{\beta_0 - l_2}. \quad (25)$$

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

$$p_{Lr}(l_r \,|\, \delta_0) = \binom{\delta_0}{l_r} p^{l_r} (1-p)^{\delta_0 - l_r}. \quad (26)$$

Therefore, the probability that there are totally l_t input ports jumped from the busy state to the idle state and l_1 of them used to send cells to output 1 to output I and l_2 of them used to send cells to output $I + 1$ is

$$\begin{aligned} p_L(l_1, l_2, l_t \,|\, \alpha_0, \beta_0, \chi_0) &= p_{L1}(l_1 \,|\, \alpha_0) p_{L2}(l_2 \,|\, \beta_0) \\ &\quad \times p_{Lr}(l_t - l_1 - l_2 \,|\, \chi_0 - \alpha_0 - \beta_0). \end{aligned} \quad (27)$$

After obtaining (23) and (27), the transition probability from $(\alpha_0, \beta_0, \chi_0)$ to state $(\alpha_1, \beta_1, \chi_1)$ is

$$\sum p_L(l_1, l_2, l_t \,|\, \alpha_0, \beta_0, \chi_0) p_R(r_1, r_2, r_t \,|\, \chi_0) \quad (28)$$

where (r_1, r_2, r_t) and (l_1, l_2, l_t) satisfy $\alpha_1 - \alpha_0 = r_1 - l_1$, $\beta_1 - \beta_0 = r_2 - l_2$, and $\chi_1 - \chi_0 = r_t - l_t$.

B. Aggregation Method for ON–OFF Markovian 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 very similar to the Bernoulli traffic case, we only briefly outline those are unique to ON–OFF Markovian traffic.

When considering two queues, the state of the two queues is represented by five random variables, $(\alpha, \beta, \chi, X, Y)$, where α is the number of busy inputs addressing to output 1, β is the number of busy inputs addressing to 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 rate from state $(\alpha_0, \beta_0, \chi_0, X_0, Y_0)$ to state $(\alpha_1, \beta_1, \chi_1, X_1, Y_1)$ can be obtained in *two phases*, that is, first finding the transition probability of the inputs, and then finding the transition probability of the queues, and the transition rate will be the product of the two. The transition probability of the inputs can be found first since the input transition is independent of the state of the switch. Then, the probability that the inputs will go from state $(\alpha_0, \beta_0, \chi_0)$ to $(\alpha_1, \beta_1, \chi_1)$ is given in (28). After that, the probability that the two queues will go from (X_0, Y_0) to (X_1, Y_1) given that queue 1 received α_0 cells and queue 2 received β_0 is needed. Clearly, when $X_1 + Y_1 < B$, this probability is 1 only if $X_1 = [X_0 + \alpha_0 - 1]^+$ and $Y_1 = [Y_0 + \beta_0 - 1]^+$; otherwise it is zero. When $X_1 + Y_1 = B$, let $X_m = [X_0 + \alpha_0 - 1]^+$ and

$Y_m = [Y_0 + \beta_0 - 1]^+$. If $X_m \geq X_1$ and $Y_m \geq Y_1$, this probability is $P\_D(X_m - X_1, Y_m - Y_1; X_m, Y_m)$; otherwise, it is zero.

After obtaining the transition matrix, the steady-state distribution can be found. Like in Bernoulli traffic, the two queues will now be combined into one block, that is, $(\alpha, \beta, \chi, X, Y)$ will be replaced by (α', χ, S, U), where α' is the total number of busy inputs addressing to output 1 and output 2, S is the number of cells stored in the block, and U is the number of nonempty queues in the block. The two conditional probabilities for describing the behavior of the block can be found in a very similar way as that for Bernoulli traffic. The difference is that the first conditional probability will add χ to the conditions and will be $CT(S_m, U_m \mid S_0, U_0, \alpha, \chi)$. Also, the second conditional probability will add α and χ to the conditions and will be $CB(U_1 \mid S_m, U_m, S_1, \alpha, \chi)$.

When considering $I + 1$ queues, where $I \geq 2$, the state of the queues is represented by six random variables $(\alpha, \beta, \chi, S, U, Z)$, where α is the number of busy inputs addressing to output 1 to output I,, β is the number of busy inputs addressing to 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 nonempty queues from queue 1 to queue I, and Z is the number of cells stored in queue $I + 1$. The transition rate from state $(\alpha_0, \beta_0, \chi_0, S_0, U_0, Z_0)$ to state $(\alpha_1, \beta_1, \chi_1, S_1, U_1, Z_1)$ is also obtained in two phases and the transition probability of the inputs is given in (28). The probability that the $I + 1$ queues will go from (S_0, U_0, Z_0) to (S_1, U_1, Z_1) given that queue 1 to queue I received α_0 cells and queue $I + 1$ received β_0 can be found as follows. When $S_1 + Z_1 < B$, the transition rate is $CT(S_1, U_1 \mid S_0, U_0, \alpha_0, \chi_0)$ if $\beta_0 \in Q(Z_1, Z_0)$; otherwise, it is zero. When $S_1 + Z_1 = B$, the transition rate is

$$\sum_{(S_m, U_m)} P\_D(S_m - S_1, Z_m - Z_1; S_m, Z_m) \times CB(U_1 \mid S_m, U_m, S_1, \alpha_0, \chi_0) \times CT(S_m, U_m \mid S_0, U_0, \alpha_0, \chi_0)$$

where $Z_m = [Z_0 + \beta_0 - 1]^+$, $Z_m \geq Z_1$, $S_m \geq S_1$, $U_m \geq U_1$. Then, 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 $CT(S_m, U_m \mid S_0, U_0, n, T)$ and $CB(U_1 \mid S_m, U_m, S_1, n, T)$ will be updated.

C. Rewinding

The Aggregation model can be improved as follows. Recall that the assumption we have made that could lead to model inaccuracy is the zero external interference assumption; thus, the model can be improved if the external queues, i.e., queue $I + 1$ to queue N, can somehow be taken into account. Note that 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 $CT_I(S_m, U_m \mid S_0, U_0, \alpha, \chi)$ and $CB_I(U_1 \mid S_m, U_m, S_1, \alpha, \chi)$, where subscript I is used to indicate that they are for a block containing I queues. For simplicity, in the following, they will be written as $CT_I()$ and $CB_I()$, respectively. After going through $N - 1$ steps, $CT_I()$ and $CB_I()$ for all $2 \leq I \leq N$ will have been found. We can then run the model again by regarding the switch as two blocks, one with I queues and the other with $N - I$ queues. Since at this time the behavior of these two blocks is known, the state distribution of the two blocks can be found. Note that the zero external interference assumption is not needed at this time since all queues in the switch are considered. With the steady-state distribution, the probability that when the first block is in a certain state, the second block stores a certain number of cells can be found, and this probability can, in turn, be used to obtain better $CT_I()$ and $CB_I()$ without the zero external interference assumption. This recalculation may start with $I = 2$, then for $I = 3$ to $I = N - 1$, and is called "rewinding."

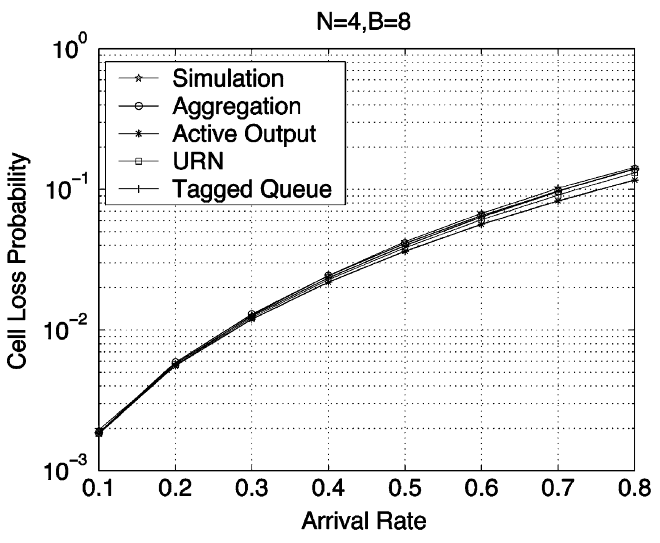
The rewinding modification is similar to a fixed-point method: first make a guess about the solution, then use this guess to find a better solution in hope that it will indeed become better. We found that the rewinding will improve the accuracy of the Aggregation model, and typically only one round of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

D. Validations of the Model

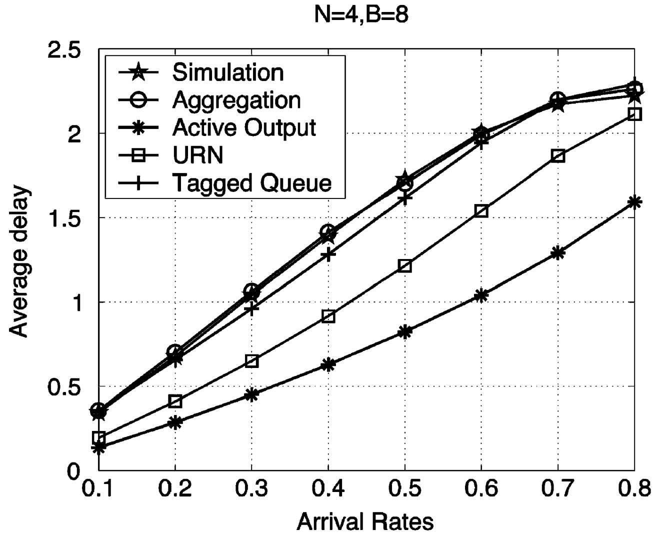
Fig. 4 shows the cell loss probability and the average delay time of a switch, where $N = 4$ and $B = 8$ as a function of the traffic load when the average stream length is 5. It can be seen that the results of the Aggregation model are very close to the simulation. The cell loss probabilities of other models are close to the simulation, but the average delays are not. Note that the Bidimensional model is not shown here, because [4] did not consider ON–OFF Markovian traffic. A detailed comparison will be given in Section VII.

V. Aggregation Model for Optical Switches

The idea of aggregation is not limited only to electronic switches, and in this section, we will apply it to wavelength-division-multiplexing (WDM) switches used in optical packet switching networks. The WDM technique is now widely regarded as the candidate for future high-speed communications 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 (FDLs), which are capable of delaying packets for a certain amount of time. The incoming packets are fixed-length cells in the optical domain. In the past, the performance of WDM packet switches with dedicated buffer for each output link has been studied in [13] and [14]. However, since FDLs are expensive and bulky, to reduce the cost and the size of the switch, it is more desirable to let them be shared by all outputs [15]. Fig. 5 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. Whenever possible, an arriving packet will be sent, by the switching fabric, to its destined output fiber. However, if



(a)



(b)

Fig. 4. Cell loss probability of and average delay under ON–OFF Markovian traffic when $N = 4, B = 8$. The average stream length is 5.

it cannot be sent to the output fiber, it will be sent to one of the delay lines. After being delayed for one time slot, it will come out of the delay line and compete with other delayed packets as well as the newly arrived packets for access to the output fiber again. If it fails, it will be sent to a delay line again to wait for the next time slot. 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 in a time slot. To the best of our knowledge, the performance of this type of switch has not been studied analytically.

In mathematical terms, the only difference between the WDM switch and the electronic switch is that at one time slot, a queue can send out up to k cells instead of only one. Therefore, the only modification needed is about the equation governing the queue behavior: after receiving a cells, a queue that used to have X_0 cells will have $X_m = [X_0 + a - k]^+$ cells in the intermediate state. Variable U in the pair of random variables (S, U)

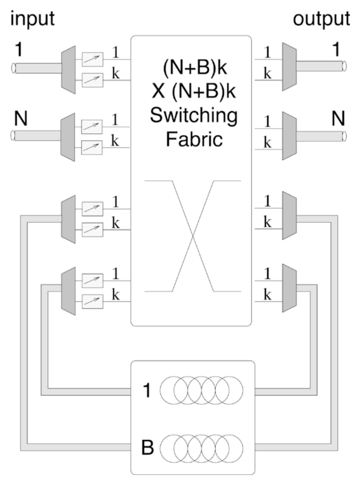


Fig. 5. An optical WDM switch with shared buffer.

representing the state of an aggregated block of queues 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 eight input/output fibers, four FDLs, and four wavelengths per fiber under Bernoulli traffic. It can be seen that the Aggregation model matches perfectly with the simulations.

VI. Transmit-First Switch

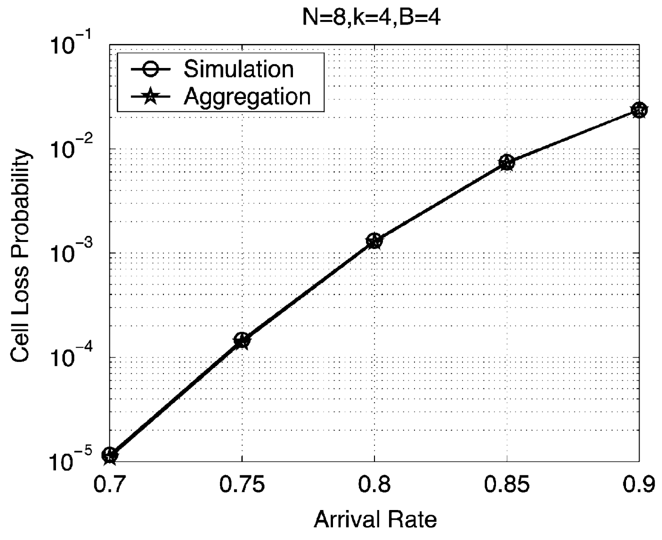
So far, we have considered receive-first switches, which first accept all arrived cells, then transmit cells at the head of queues. There are also transmit-first switches, which first transmit cells at the head of queues, then accept arrived cells. The extension of the Aggregation model from the receive-first switch to the transmit-first switch is immediate. Similar to Section V, only the equation about queuing behavior needs to be modified. In a transmit-first switch, after receiving a cells, a queue that used to have X_0 cells will have $X_m = [X_0 - 1]^+ + a$ cells in the intermediate state. Fig. 7 shows the analytical and simulation results for an electronic transmit-first switch under Bernoulli traffic when $N = 16$, $B = 32$. Again, we can see that the Aggregation model is very accurate.

VII. Comparisons With Other Existing Models and a Deeper Understanding of the Aggregation Model

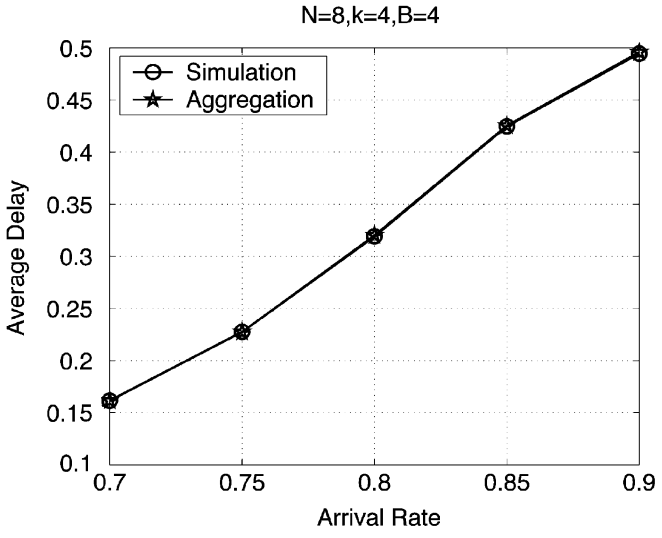
In this section, we will first compare the Aggregation model with four other existing analytical models for shared buffer switches we are aware of. Then we will give explanations for the very good performance of the Aggregation model.

A. Comparisons With Other Existing Models

We are aware of four approximation models for shared buffer switches, which are called the Active Output model, the URN model, the Tagged Queue model, and the Bidimensional model. The Active Output model was first introduced by Turner in [1] and may be the earliest model for the shared buffer switches. In this model, it is assumed that cells stored in the buffer have independent random destinations. The URN model was introduced by Fong in [3] and assumes that all possible combinations for cells to be distributed in the buffer are equally likely. The

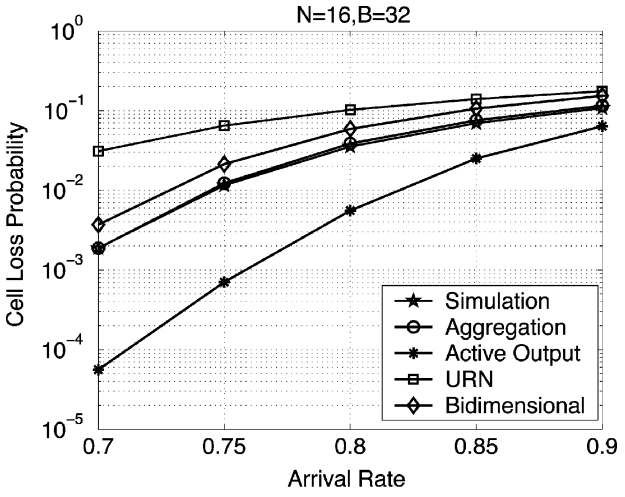


(a)

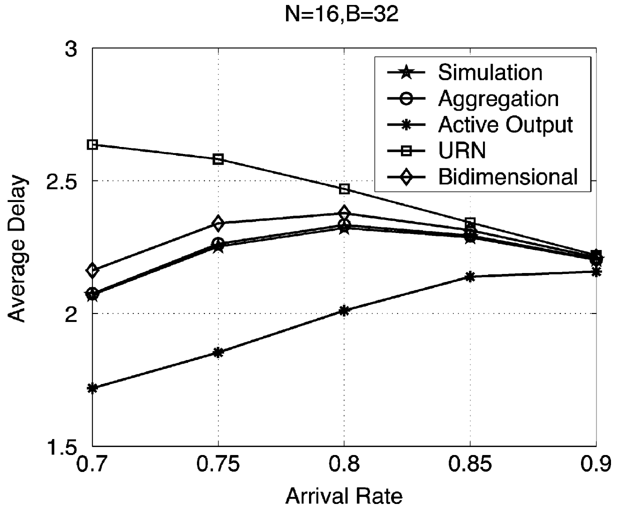


(b)

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



(a)



(b)

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

Tagged Queue model was introduced by Pattavina and Gianatti in [5] and [6] and assumes that queues in the buffer are independent of each other. The Bidimensional model was introduced by Bianchi and Turner in [4] and assumes that all possible combinations for cells to be distributed in the buffer are equally likely (it should be mentioned that the Bidimensional model is earlier than the URN model), and like the Aggregation model, also uses the number of nonempty queues as one of its two state variables of the Markov chain. It should be pointed out that in our implementation of the Active Output model and URN model, as suggested in [3], the idea of the "tagged queue" in the Tagged Queue model is incorporated to improve the model accuracy, that is, a queue called the "tagged queue" is singled out, and the rest $N-1$ queues are modeled as a block and the behavior of the block is obtained by the methods of different models. The results of these models have been shown in the figures. We did not implement the vector method as it is of exponential complexity and, since it is known to be exact, it should yield the same results as the simulations.

From the figures, first it can be seen that our Aggregation model is very accurate, since in every figure, its curve matches perfectly with the simulation curve. Other models may perform well in some occasions [for example, in Fig. 4(a)] for cell loss probability under the ON–OFF Markovian traffic in a small switch but also may perform not so well in other occasions (for example, in Fig. 3) both for cell loss probability and average delay under Bernoulli traffic in a larger switch. Therefore, we can say that these four 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.

An interesting phenomena that might have been noticed in Fig. 4 is that the performances of the Active Output model, the URN model, and the Tagged Queue model for finding the cell loss probability are better under ON–OFF Markovian traffic than under Bernoulli traffic. However, it is not because the transition rates of these models are better under ON–OFF Markovian traffic than under Bernoulli traffic. In fact, we have found that the transition rates of these models do not agree very well with the simulations both under Bernoulli traffic and under ON–OFF Markovian traffic. Similar facts with regard to the ON–OFF Markovian traffic can also be found in [3]. The reason for this improvement is that under the ON–OFF Markovian traffic, two more random variables describing the state of the input were added. However, note that under ON–OFF Markovian traffic, the average delays given by these models are still not very good, which is because their buffer transition probabilities are indeed not very accurate. Note that in these models, the cell loss probability is derived only from the tagged queue and is thus less dependent on the buffer transition probabilities than the average delay does.

Next, we consider the complexities of the models. The complexities of the Aggregation model, the Active Output model, the URN model, and the Bidimensional model are all polynomial functions of the switch size. The complexity of the Tagged Queue model is an exponential function of the switch size, which is explained in more details in the Appendix. The complexities of the polynomial models are mainly determined by the size of the state space. For Bernoulli traffic, the sizes of the state spaces of the Active Output model, the URN model, and the Bidimensional model are all $(B+1)(B+2)/2$, while the size of the state space of the Aggregation model is $(B+1)(B+2)(N-1)/2$. For ON–OFF Markovian traffic, the sizes of the state spaces of the Active Output model and the URN model are all $(N+1)(N+2)(B+1)(B+2)/4$, while the size of the state space of the Aggregation model is $[(B+1)(B+2)(N-1)/4]\sum_{i=0}^{N}(i+1)(i+2)$. Therefore, among the polynomial models the Aggregation model has the highest complexity. It should be mentioned that if implemented without the tagged queue, the Active Output model and the URN model may use only one variable for Bernoulli traffic and two variables for ON–OFF Markovian traffic, and the sizes of the state spaces will be reduced to $(B+1)$ and $(N+1)(B+1)$, but they will be less accurate.

B. Deeper Understanding of the Aggregation Model

We have seen that the Aggregation model is very accurate. A natural question is, Being that it is an approximation model, why does it have such a good performance? This question can be answered by examining the assumptions made by the Aggregation model since the performance of an approximation model depends on how good its assumptions approximate the reality.

We make only two assumptions in the Aggregation model. The first is that we assume (S,U) has a Markov property where (S,U) is used to represent the state of the aggregated queues. This assumption is not true in some cases since examples can be found that the value of (S,U) is dependent on the value of (S,U) two time slots ago. However, it can be shown that it is still quite a good approximation. To give a quantitative measure of the model accuracy, we define the *total square error (TSE)* of a model as $\sum_{i=0}^{B}[p(i)-p^*(i)]^2$, where $p(i)$ and $p^*(i)$ denote the probability that there are totally i cells in the buffer found by the model and the simulations, respectively. The distribution of buffer occupancy is used because the cell loss probability and cell delay are all derived from it. We have tried the "semianalytical model" by plugging the one-step transition matrix of (S,U) found by simulations into the program for solving the Markov chain. Table I shows the TSE of the semianalytical model along with other models under Bernoulli traffic where "semi," "Bi," "Tri," and "Aggr" denote the semianalytical model, the Bidimensional model, the Tridimensional model (to be explained soon), and the Aggregation model, respectively. We can see that the buffer distribution of the semianalytical model is very close to the simulation, since, for example, when $N=8$, $B=16$, $\rho=0.8$, the TSE of the semianalytical model is in the order of 10^{-11} (5.1338×10^{-11}). Note that for TSE, values in the order of 10^{-6} can already be considered very small since when shown in figures, the two curves will overlap with each other and become indistinguishable. Thus, although strictly speaking, (S,U) is not a Markov chain; it can be approximated by the Markov chain with extremely small error.

TABLE I
TOTAL SQUARE ERROR (TSE) OF DIFFERENT MODELS UNDER BERNOULLI TRAFFIC

	Semi	Bi	Tri	Aggr
$N=4, B=8$, $\rho=0.6$	3.778 $\times10^{-13}$	6.746 $\times10^{-5}$	1.585 $\times10^{-5}$	1.034 $\times10^{-6}$
$N=4, B=8$, $\rho=0.8$	8.619 $\times10^{-12}$	8.639 $\times10^{-4}$	2.817 $\times10^{-4}$	3.594 $\times10^{-7}$
$N=8, B=16$, $\rho=0.6$	2.272 $\times10^{-12}$	7.310 $\times10^{-4}$	5.090 $\times10^{-4}$	7.936 $\times10^{-7}$
$N=8, B=16$, $\rho=0.8$	1.362 $\times10^{-11}$	3.600 $\times10^{-3}$	2.500 $\times10^{-3}$	8.159 $\times10^{-6}$

The other assumption is the "zero external interference assumption" used when deriving the transition rate of the Markov chain. Recall that we have assumed that when considering $I+1$ queues, if the number of cells that have to be buffered in these queues, i.e., S_m+Z_m, exceeds the buffer capacity B, S_m+Z_m-B cells will be dropped from them; otherwise, no cell will be dropped from them. Note that the switch may not actually work according to this assumption, because when $S_m+Z_m\le B$, cells from these queues may still be dropped since other queues may be storing too many cells, and when $S_m+Z_m>B$, more than S_m+Z_m-B cells may be dropped from these queues since it may happen that other queues are not empty and the switch decides to drop more cells from queue 1 to queue $I+1$. However, the probabilities of the above events are relatively small, and most important, as the iteration goes on, I will become larger and larger, and this assumption will become closer and closer to the truth since the number of unknown queues will become smaller and smaller. In the last iteration, $I=N-1$, and there is indeed no interference from external queues since there is no external queues at all.

Another question is whether the Aggregation model achieves good performance because it uses more variables as the state of the Markov chain. To check this, we have taken the idea of the Bidimensional model and implemented the "Tridimensional" model for Bernoulli traffic, which has the exactly the same state space as the Aggregation model and can be regarded as the Bidimensional model enhanced with the tagged queue idea. In the Tridimensional model, same as the Aggregation model in the last iteration, the state of the switch is represented by (S, U, Z), where the meanings of the random variables are exactly the same as in the Aggregation model, and the transition of the block with $N-1$ queues, i.e., (S, U), is derived according to the Bidimensional model. The Bidimensional model is chosen because it has better accuracy compared with other models. We have found that the TSE of the Tridimensional model is quite large. For example, as can be seen in Table I, when $N = 8$, $B = 16$, $\rho = 0.8$, under Bernoulli traffic, the TSE of the Tridimensional model is in the order of 10^{-3} (2.5×10^{-3}). Note that values in the order of 10^{-3} are quite large for TSE since if shown in figures, the two distribution curves will be separated by significant distances. The TSE of the Aggregation model, on the other hand, is in the order of 10^{-6} (8.1585×10^{-6}). We have also found that the TSE of the Bidimensional to be also in the order of 10^{-3} (3.6×10^{-3}) and is slightly larger than that of the Tridimensional model. Therefore, we can conclude that 1) there is very little improvement only by adding one variable to the state space and 2) the Aggregation model outperforms the Tridimensional model, and similarly, other models, because its transition rate is more accurate.

VIII. Conclusion

In this paper, we have presented the Aggregation model for switches with shared buffer. This model finds the behavior of queues in the shared buffer in a step by step manner. The complexity of this model is a polynomial function of the switch size. We have conducted extensive simulations and the results showed that the new model is very accurate. Our future work includes further reducing the complexity of this model and considering other buffer management algorithms.

Appendix

In this Appendix, we analyze the complexity of the Tagged Queue model. In the Tagged Queue model, a queue called the "Tagged Queue" is singled out, and the rest $N-1$ queues are modeled as a block. The tagged queue is a single queue, and its behavior can be easily found. The behavior of the block containing $N-1$ queues is described by $P_l(v \mid l)$, which is the conditional probability that when there are l cells in the block, v cells are ready to be transmitted. $P_l(v \mid l)$ is approximated by using the independent 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 external interference 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, $P_l(v \mid l)$ is approximated as the ratio of the former over the latter. Note that to find $P_l(v \mid l)$, 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 $O\left(\frac{e^{\sqrt{B}}}{B}\right)$, 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.

Acknowledgment

The authors would like to thank anonymous reviewers of this paper for very insightful suggestions and comments for improving the paper.

References

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

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

[3] S. Fong and S. Singh, "Modeling cell departure for shared buffer ATM switch," in *Proc. IEEE Int. Conf. Communications (ICC 98)*, 1998, vol. 3, pp. 1824–1828.

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

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

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

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

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

[9] C. S. Lin, B. D. Liu, and Y. C. Tang, "Design of a shared buffer management scheme for ATM switches," in *Proc. 15th Annu. IEEE Int. ASIC/SOC Conf.*, Sep. 2002, pp. 261–264.

[10] S. Ahlgren and K. Ono, "Addition and counting: The arithmetic of partitions," *Notices Amer. Mathematical Soc.*, 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," *J. Lightw. Technol.*, vol. 21, no. 3, pp. 719–726, Mar. 2003.

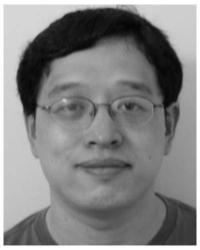
[12] L. Xu, H. G. Perros, and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Commun. Mag.*, vol. 39, no. 1, pp. 136–142, Jan. 2001.

[13] S. L. Danielsen, "Analysis of a WDM packet switch with improved performance under bursty traffic conditions due to tunable wavelength converters," *J. Lightw. Technol.*, vol. 16, no. 5, pp. 729–735, May 1998.

[14] S. L. Danielsen, "WDM packet switch architectures and analysis of the influence of tunable wavelength converters on the performance," *J. Lightw. Technol.*, vol. 15, no. 2, pp. 219–227, Feb. 1998.

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

[16] Z. Zhang and Y. Yang, "A novel analytical model for electronic and optical switches with shared buffer," in *Proc. IEEE INFOCOM 2005*, Miami, FL, Mar. 2005, pp. 420–431.



Zhenghao Zhang (S'03) received the B.Eng. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 1996 and 1999, respectively, and the Ph.D. degree in electrical engineering from the State University of New York at Stony Brook in 2006.

From 1999 to 2001, he worked in industry as a software engineer for embedded systems. He is currently a Postdoctoral Researcher in the Computer Sciences Department of the Carnegie Mellon University, Pittsburgh, PA. His research interests include scheduling in optical and wireless networks, performance modeling of optical and wireless networks, and network security.

Yuanyuan Yang (SM'98) received the B.Eng. and M.S. degrees in computer science and engineering from Tsinghua University, Beijing, China, and M.S.E. and Ph.D. degrees in computer science from The Johns Hopkins University, Baltimore, MD.

She is a Professor of computer engineering and computer science at the State University of New York at Stony Brook. Her research interests include wireless networks, optical networks, high-speed networks, and parallel and distributed computing systems. Her research has been supported by the National Science Foundation (NSF) and U.S. Army Research Office (ARO). She has published over 160 papers in major journals and refereed conference proceedings and holds six U.S. patents in these areas.

Dr. Yang is currently an Editor for the IEEE TRANSACTIONS ON COMPUTERS and the *Journal of Parallel and Distributed Computing*, and has served as an Editor for the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. She has also served on program/organizing committees of numerous international conferences in her areas of research.