

# A Distributed Dynamic Scheduling Algorithm for a Terabit Multicast Packet Switch

Feihong Chen, Necdet Uzun and Ali N. Akansu

New Jersey Center for Multimedia Research  
Department of Electrical and Computer Engineering  
New Jersey Institute of Technology  
University Heights, Newark, NJ 07102  
Email: {fxc0407, uzun}@oak.njit.edu, ali@megahertz.njit.edu

**Abstract.** In this paper, we present a novel switch design of a large scale multicast packet switch which is featured by a modular switch architecture and a distributed resource allocation algorithm. Switch inputs and outputs are grouped into small modules called Input Shared Blocks (ISBs) and Output Shared blocks (OSBs). Input link sharing and output link sharing are cooperated intelligently so that no speedup is necessary in central switch fabric (ATMCSF). Cell delivery is based on link reservation in every ISB. Dual round robin rings connect ISBs to provide a fast and fair link resource allocation among ISBs according to a Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm. QOBDLR is a distributed algorithm in which an ISB can dynamically increase/decrease its link reservation for a specific OSB according to its local available information. Arbitration complexity is  $O(1)$ . Switch performance is evaluated through simulations for an 256x256 switch. It is demonstrated that the proposed switch can achieve a comparable performance as the output queued switch under any traffic pattern.

## 1 Introduction

As the demanding bandwidth of Internet services continues to increase, switches and routers face challenging high capacity requirements. Many services, such as teleconferencing and entertainment video, are characterized by point-to-multipoint communication. As a promising candidate of the backbone core switching system for Broadband networks, ATM switches need to be scalable, cost-effective, and support multicasting. In this paper, we propose a novel switch design of a scalable multicast packet switch.

Switches and routers employing output queueing (OQ) proved to maximize throughput and optimize latency. These characteristics are very important for high throughput and supporting Quality of Service (QoS). However, output queued switches are limited by the bandwidth of commercially available memories, since they require to store incoming data from  $N$  inputs, i.e.,  $N$  times increase in the memory speed compared to an input buffered switch. Currently, it is practical to implement an output queued switch or router with an aggregated bandwidth of several 10Gb/s. However, it is impractical to build an output queued switch with a large number of ports and fast line rate.

On the other hand, input queued (IQ) switches become more attractive because switch fabric and input memory only need to run as fast as line rate. To

overcome head of line (HOL) blocking, virtual output queues are applied at every switch input together with some weighted input-output matching algorithms to achieve 100% maximized throughput [5][6][7][8]. But, most scheduling algorithms proposed for IQ switches use centralized schedulers, which collect traffic information from  $N$  switch inputs in every cell slot and need multiple iteration to determine the final input-output matching. Scheduling complexity of at least  $O(N^{2.5})$  becomes a main obstacle when a switch grows to a large size and has a very fast line rate. Situation can become even worse under multicast traffic.

In our previous work [9], we proposed a design of a large scale ATM switch using input and output link sharing. Switch inputs and outputs are grouped into small modules called Input Shared Block (ISB) and Output Shared Block(OSB). Under uniformly distributed input traffic, link sharing with round robin cell scheduling resolves output contention and eliminates the speedup requirement for central switch fabric. Switch leads to a comparable performance as the output queued switch under uniform multicast traffic. However, isolated ISBs prevent switch from achieving high performance under non-uniform multicast traffic.

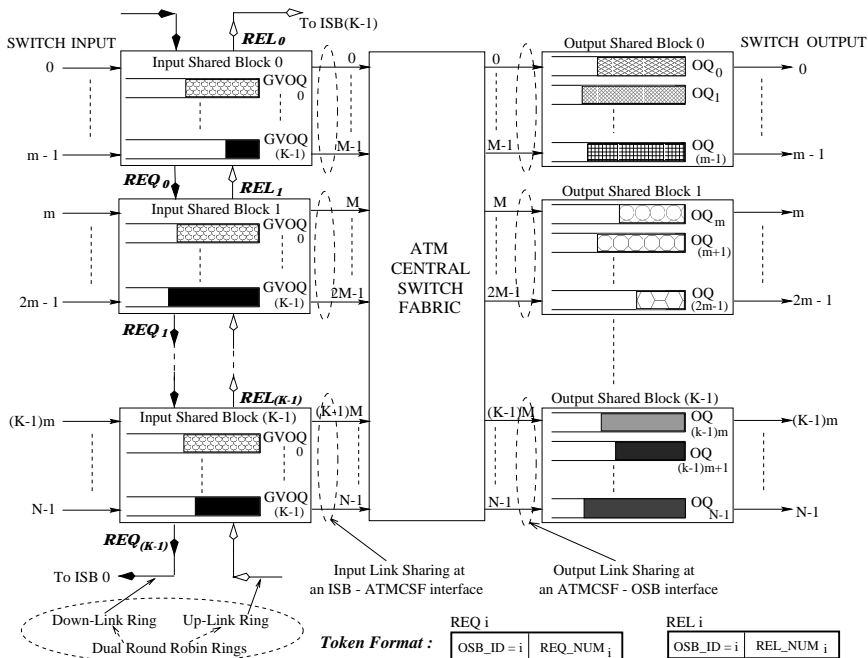
To support non-uniform traffic, in this paper, we propose an enhanced switch design in which ISBs are connected by dual round robin rings. Link request tokens (REQs) and link release tokens (RELs) circulate on the dual rings and pass ISBs one by one in a round robin manner. Every ISB should make link reservation in advance in order to obtain the desired links to the targeted OSBs. Cell delivery in every cell slot is based on link reservation in each ISB. We propose a Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm, in which every ISB can dynamically increase/decrease its link reservation for an OSB by “borrowing” and/or ”lending” links from each other through REQ and REL tokens. QOBDLR is a distributed algorithm in the sense that an ISB modifies its link reservation according to its own local available information. Arbitration complexity is only  $O(1)$ . Hence, QOBDLR can achieve a fast and fair link resource allocation among ISBs. Performance evaluation demonstrates that the proposed switch can achieve a comparable performance as the OQ switch under any traffic pattern. The proposed switch can be easily extended to a large scale up to Terabits capacity.

This paper is organized as follows. In Section 2, we introduce a novel switch architecture using link sharing and dual round robin rings. In Section 3, we describe cell delivery and present the Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm. In Section 4, switch performance is evaluated with simulations for 256x256 switch. Conclusion is drawn in Section 5.

## 2 Switch Architecture

Fig 1 depicts the modular switch architecture. The proposed switch consists of four major components : *Input Shared Block (ISB)*, *Output Shared Block (OSB)*, *ATM Central Switch Fabric (ATMCSF)*, and *Dual Round Robin Rings*. Switch inputs and outputs are respectively grouped into  $K$  ISBs and  $K$  OSBs, where  $K = \frac{N}{m}$ . ISBs are connected by dual rings on which  $K$  link request tokens (REQs) and  $K$  link release tokens (RELs) circulate in a round robin manner. At every ISB-ATMCSF interface, there are  $M$  *input links* shared by  $m$  related switch inputs. At every ATMCSF-OSB interface, there are  $M$  *output links* shared by  $m$  grouped switch outputs. In this paper, we only consider the case of  $m = M$ ,

and the study of  $M > m$  which implies a virtual speedup in ATMCSF is the subject of our ongoing work. Applying input link sharing and output link sharing together is the unique feature of the proposed switch. Link sharing eliminates speedup requirement in ATMCSF.



**Fig. 1.** The proposed modular switch architecture : an  $N \times N$  switch consists of  $K$  ISBs,  $K$  OSBs, ATMCSF, and dual round robin rings;  $K = \frac{N}{m}$  and  $m = M$  in this paper.

### 2.1 Input Shared Block

An ISB can be a shared memory receiving multicast cells<sup>1</sup> from  $m$  ( $= M$  in this paper) related switch inputs. A multicast cell is saved once in an ISB instead of keeping  $j$  identical cell copies (assume,  $j$  is the fanout of a multicast cell,  $1 \leq j \leq N$ ). We propose a *Grouped Virtual Output Queue (GVOQ)* scheme for shared memory management in an ISB.

As shown in Fig 1, every ISB only maintains  $K$  ( $= \frac{N}{m}$ ) grouped virtual output queues. Each grouped virtual output queue is a linked list of the multicast cells targeting a same OSB. If a multicast cell has more than one destination to an OSB, only a single connection carrying all desired destinations is attached to the related grouped virtual queue. Hence, a cell delivered from an ISB may carry multiple destinations, then will be stored into every related output queues when

<sup>1</sup> A multicast cell may have one or multiple destinations. Hence, multicast traffic includes unicast traffic.

the cell is received by an OSB. GVOQ follows FIFO principle to receive cells from switch inputs and deliver cells to ATMCSF.

Since an ISB-ATMCSF interface has a capacity of  $M$  links, an ISB can deliver at most  $M$  cells to ATMCSF in every cell slot. An ISB can send a cell through any of the  $M$  shared links. Input link sharing is able to avoid link starvation when some GVOQ is empty, because other GVOQs can utilize the idle link to deliver their cells. Input link sharing results in an improved performance.

## 2.2 Output Shared Block

As shown in Fig 1, an OSB is a shared memory containing  $m$  ( $= M$  in this paper) output queues. In every cell slot, each output queue delivers one cell out of the related switch output. Since an ATMCSF-OSB interface only supports  $M$  links, an OSB can accept at most  $M$  cells from the central switch fabric in every cell slot. ATMCSF can use any of the  $M$  shared links to transmit a cell to an OSB. Without output link sharing, if more than one cell goes to a same switch output, either cells are blocked, or it is necessary for the switch fabric to speedup. However, output link sharing can avoid both problems.

## 2.3 Central Switch Fabric

The central switch fabric should keep the same cell sequence for those cells which are delivered from an ISB to a same OSB. Apart from this, no other restrictions are placed on the central switch fabric. It can be any type of switch fabric (for example Abacus switch [4]), and no speedup is necessary because of input link sharing and output link sharing.

## 2.4 Dual Round Robin Rings

ISBs are connected by dual rings : a down-ward ring conveys link request tokens (REQs); and an up-ward ring carries link release tokens (RELs). At any time, there are  $K$  REQ tokens and  $K$  REL tokens circulating on the dual rings respectively and passing ISBs one by one in a round robin manner. Each OSB (e.g. the  $i^{th}$  OSB) is correlated with a REQ token (e.g.  $REQ_i$ ) and a REL token (e.g.  $REL_i$ ).

Both REQ token and REL token contain two fields (shown in Fig 1) : (1) "OSB\_ID" is the identification of an OSB; (2) "REQ\_NUM" indicates how many link requests are issued for the identified OSB. Or, "REL\_NUM" records the number of released links which are available to be reserved at the related ATMCSF-OSB interface.

# 3 Cell Scheduling

## 3.1 Cell Delivery

Cell delivery is based on link reservation in every ISB. Each ISB has a *link reservation vector* and a *queue occupancy vector*. We use  $LK\_RSV^i$  and  $Q^i$  to represent the two vectors in the  $i^{th}$  ISB ( $0 \leq i, j < K$ ) :

•  $LK\_RSV^i = [r_0^i, r_1^i, \dots, r_{(K-1)}^i]$ ; Link Reservation Vector in the  $i^{th}$  ISB. Where  $r_j^i$  indicates how many links at the ATMCSF-OSB  $j$  interface are reserved by the  $i^{th}$  ISB,  $0 \leq r_j^i \leq M$ .

•  $Q^i = [q_0^i, q_1^i, \dots, q_{(K-1)}^i]$ ; Queue Occupancy Vector in the  $i^{th}$  ISB. Where  $q_j^i$  shows queue length of the  $j^{th}$  GVOQ in the  $i^{th}$  ISB,  $q_j^i \geq 0$ .

In a cell slot, each ISB delivers cells to central switch fabric according to its link reservation. For example, if  $LK\_RSV^i$  is  $[2, 0, \dots, 4]$  in current cell slot, the  $i^{th}$  ISB will send 2 cells to OSB 0 and 4 cells to OSB  $(K-1)$ , but no cells are scheduled to other OSBs.

According to its queue occupancy vector, each ISB can dynamically increase/decrease its link reservation for a specific OSB by “borrowing” or “lending” links through REQ and/or REL tokens. To achieve a fast and fair link resource allocation among ISBs, we will propose a *Queue Occupancy Based Dynamic Link Reservation* (QOBDLR) algorithm in next section.

### 3.2 Link Reservation : QOBDLR Algorithm

#### Definition 1 : Link Reservation Rule.

Link reservation among  $K$  ISBs must satisfy two criteria :

- (1)  $\sum_{j=0}^{K-1} r_j^i \leq M$ , i.e. the total links reserved by the  $i^{th}$  ISB can not exceed  $M$  which is the maximum number of links at an ISB-ATMCSF interface;
- (2)  $\sum_{i=0}^{K-1} r_j^i \leq M$ , i.e. the total links reserved by all ISBs to the  $j^{th}$  OSB can not exceed  $M$  which is the maximum number of links at the ATMCSF-OSB interface.

#### Definition 2 : Link Reservation Slot, i.e. Rsv\_Slot.

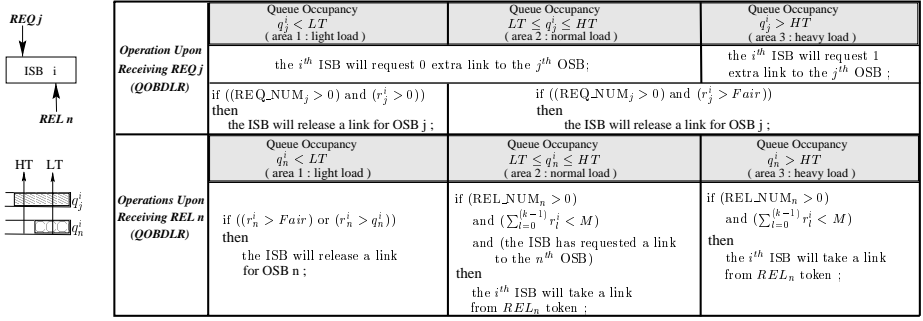
Rsv\_Slot is defined as a small time interval during which an ISB receives a pair of REQ and REL tokens. In a Rsv\_Slot, an ISB has the authority to modify its link reservation for the two OSBs which are identified by the received REQ and REL tokens. Rsv\_Slot is independent from Cell\_Slot, usually,  $Rsv\_Slot \ll Cell\_Slot$ . When a cell slot is due, every ISB delivers cells to ATMCSF according to its current link reservation vector.

QOBDLR algorithm is performed in every Rsv\_Slot. As a common model shown in Fig 2, the  $i^{th}$  ISB ( $0 \leq i < K$ ) is receiving a  $REQ_j$  and a  $REL_n$  tokens in current Rsv\_Slot, usually  $REQ_j$  and  $REL_n$  identify two different OSBs (i.e.  $j \neq n$ ). The  $i^{th}$  ISB only has the authority to modify its link reservation, i.e.  $r_j^i$  and  $r_n^i$ , for the  $j^{th}$  OSB and the  $n^{th}$  OSB in current Rsv\_Slot.

#### Operations Upon Receiving $REQ_j$ Token :

When receiving  $REQ_j$  token, the  $i^{th}$  ISB will evaluate its queue occupancy  $q_j^i$  against two thresholds : a high threshold (HT) and a low threshold (LT). Then, the  $i^{th}$  ISB decides whether to request an extra link and/or release a link to the  $j^{th}$  OSB.

If  $q_j^i > HT$ , the  $i^{th}$  ISB will request an additional link for the  $j^{th}$  OSB. Note that, if the  $i^{th}$  ISB had sent a link request before but has not obtained the desired link yet, the  $i^{th}$  ISB will not issue an extra new-born link request but just keep asking for one additional link if  $q_j^i > HT$ . And if the  $REQ_j$  token carries link requests (i.e.  $REQ\_NUM_j > 0$ ), the  $i^{th}$  ISB will schedule a single



	Queue Occupancy $q_n^i < LT$ (area 1 : light load)	Queue Occupancy $LT \leq q_n^i \leq HT$ (area 2 : normal load)	Queue Occupancy $q_n^i > HT$ (area 3 : heavy load)
<b>Operation Upon Receiving <math>REQ_j</math> (QOBDLR)</b>	the $i^{th}$ ISB will request 0 extra link to the $j^{th}$ OSB;	the $i^{th}$ ISB will request 1 extra link to the $j^{th}$ OSB ;	the $i^{th}$ ISB will request 1 extra link to the $j^{th}$ OSB ;
	if $((REQ\_NUM_j > 0) \text{ and } (r_j^i > 0))$ then the ISB will release a link for OSB j ;	if $((REQ\_NUM_j > 0) \text{ and } (r_j^i > Fair))$ then the ISB will release a link for OSB j ;	
<b>Operations Upon Receiving <math>REL_n</math> (QOBDLR)</b>	Queue Occupancy $q_n^i < LT$ (area 1 : light load)	Queue Occupancy $LT \leq q_n^i \leq HT$ (area 2 : normal load)	Queue Occupancy $q_n^i > HT$ (area 3 : heavy load)
	if $((r_n^i > Fair) \text{ or } (r_n^i > q_n^i))$ then the ISB will release a link for OSB n ;	if $(REL\_NUM_n > 0)$ and $(\sum_{i=0}^{(k-1)} r_i^i < M)$ and (the ISB has requested a link to the $n^{th}$ OSB) then the $i^{th}$ ISB will take a link from $REL_n$ token ;	if $(REL\_NUM_n > 0)$ and $(\sum_{i=0}^{(k-1)} r_i^i < M)$ then the $i^{th}$ ISB will take a link from $REL_n$ token ;

**Fig. 2.** QOBDLR algorithm performed in a Rsv\_Slot. For example, the  $i^{th}$  ISB is receiving  $REQ_j$  and  $REL_n$  token in current Rsv\_Slot.

link release for OSB j if the ISB reserves more than  $Fair$ <sup>2</sup> links to OSB j (i.e.  $r_j^i > Fair$ ).

If  $LT \leq q_j^i \leq HT$ , the  $i^{th}$  ISB may release a link if  $REQ_j$  token carries link requests and if its link reservation for the OSB j is more than  $Fair$  links; Otherwise, the ISB will keep the same reservation as before.

If  $q_j^i < LT$ , and if  $REQ_j$  token carries link requests, the  $i^{th}$  ISB will release one of its occupied link to OSB j.

In general, a new-born link request for the  $j^{th}$  OSB will be added into  $REQ_j$  token, hence,  $REQ\_NUM_j$  will be increased by 1. On the other hand, if the  $i^{th}$  ISB releases a link to satisfy a link request in  $REQ_j$  token,  $REQ\_NUM_j$  will be reduced by 1. Usually, the link scheduled to be released can not be passed to other ISBs in current Rsv\_Slot. The  $i^{th}$  ISB will record this pending link release, and will add this released link into  $REL_j$  token when the  $i^{th}$  ISB receives  $REL_j$  token in some Rsv\_Slot(s) later.

### Operations Upon Receiving $REL_n$ Token :

When receiving  $REL_n$  token, the  $i^{th}$  ISB will evaluate its queue occupancy  $q_n^i$  with HT and LT to decide an intended modification on  $r_n^i$ .

If  $q_n^i > HT$ , the  $i^{th}$  ISB will grab an additional link for the  $n^{th}$  OSB as long as following conditions are hold : (1)  $REL_n$  token carries available links (i.e.  $REL\_NUM_n > 0$ ); (2) the total number of links reserved by the  $i^{th}$  ISB is less than M (i.e.  $\sum_{l=0}^{(k-1)} r_l^i < M$ ).

If  $LT \leq q_n^i \leq HT$ , the  $i^{th}$  ISB will take an extra link for the  $n^{th}$  OSB if following requirements are satisfied : (1)  $REL_n$  token carries available links (i.e.  $REL\_NUM_n > 0$ ); (2) the ISB has requested a link for the OSB ; (3) the total number of links reserved by the ISB is smaller than M (i.e.  $\sum_{l=0}^{(k-1)} r_l^i < M$ ).

If  $q_n^i < LT$ , the  $i^{th}$  ISB will schedule a link release if its current reservation for the  $n^{th}$  OSB is either greater than Fair (i.e.  $r_n^i > Fair$ ) or greater than its current queue occupancy (i.e.  $r_n^i > q_n^i$ ).

For the case of  $q_n^i > HT$ , it may happen that the  $i^{th}$  ISB takes a link from  $REL_n$  token but it had not sent a link request before. Under such circumstance,

<sup>2</sup>  $Fair = \lfloor \frac{M}{K} \rfloor$ , i.e. the M links at an ATMCSF-OSB interface are fairly allocated to K ISBs

we term the  $i^{th}$  ISB as a 'thieving ISB', which "steals" an available link that may have been released for a link request issued by another ISB, namely the 'victim ISB'. The 'victim ISB' who has sent a link request and is waiting for an available link may never receive its desired link if there is a 'thieving ISB' on the way snatching an available link from  $REL_n$  token. To resolve this problem, the 'thieving ISB' should send a link request for the  $n^{th}$  OSB to motivate a link release not for itself but for the 'victim ISB'. Usually, this compensated link request for the  $n^{th}$  OSB can not be inserted into  $REQ_j$  token in current Rsv\_Slot. The ISB has to record this pending link request and wait for receiving  $REQ_n$  token to send this link request to other ISBs.

For the case of  $q_n^i < LT$ , the  $i^{th}$  ISB's releasing a link for the  $n^{th}$  OSB is due to its own low traffic load and it does not need to be triggered by link requests in  $REQ_n$  token. It means that the  $i^{th}$  ISB releases a link for the  $n^{th}$  OSB without any knowledge of how many link requests are indicated in  $REQ_n$  token. In order to achieve an efficient link utilization, it is required that  $REL\_NUM_n \leq REQ\_NUM_n$ , i.e. all available links will be used up by link requests and will be needed by some ISBs. Bearing this in mind, when the  $i^{th}$  ISB releases a link for the  $n^{th}$  OSB due to  $q_n^i < LT$ , there are actually  $(REQ\_NUM_n - 1)^3$  or  $(REQ\_NUM_n - 2)^4$  link requests are expecting available links for the  $n^{th}$  OSB. Hence, the  $i^{th}$  ISB should decrease  $REQ\_NUM_n$  by either 1 or 2. However, the  $i^{th}$  ISB does not hold  $REQ_n$  token in current Rsv\_Slot. The  $i^{th}$  ISB has to record this pending reduction of link requests and wait for receiving  $REQ_n$  token to modify  $REQ\_NUM_n$ .

Due to operations for the received  $REL_n$  token, when the  $i^{th}$  ISB receives  $REQ_n$  token in some Rsv\_Slot(s) later, the ISB first has to update  $REQ\_NUM_n$  with the pending increment/decrement of link requests for the  $n^{th}$  OSB.

### 3.3 Remarks

During system initialization, all link resources are assigned to ISBs, i.e.  $\sum_{i=0}^{(k-1)} r_l^i = M$  and  $\sum_{i=0}^{(k-1)} r_l^i = M$  for  $\forall i, j$ . But, how to initialize link reservation vectors is not important because an ISB will dynamically "borrow" and/or "lend" links from/to other ISBs according to its traffic load.

Cell delivery and link reservation are independent operations. When a Cell\_Slot is due, every ISB sends cells to ATMCSF based on its current link reservation vector. But, an ISB can modify its link reservation in every Rsv\_Slot, usually  $Rsv\_Slot < Cell\_Slot$ .

In QOBDLR algorithm, the high threshold HT and the low threshold LT are predefined system parameters and are consistent after their initialization. How to make the optimal choice on the values of HT and LT is beyond this paper. In this paper, we select HT and LT as 4 and 2 for the example of an 256x256 switch. We show that the switch with QOBDLR algorithm judged by the HT and LT is able to achieve a **fair and fast** link resource allocation among ISBs.

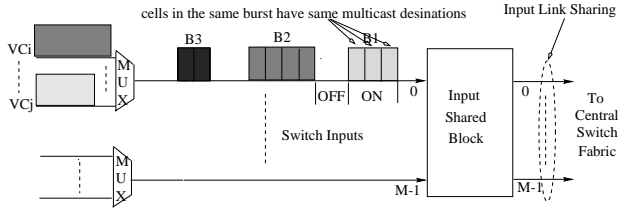
<sup>3</sup> If the  $i^{th}$  ISB has not sent a link request for the  $n^{th}$  OSB, then  $(REQ\_NUM_n - 1)$  link requests are demanding available links after the  $i^{th}$  ISB releases a link.

<sup>4</sup> If the  $i^{th}$  ISB has issued a link request for the  $n^{th}$  OSB, then  $(REQ\_NUM_n - 2)$  link requests are demanding available links after the  $i^{th}$  ISB releases a link.

## 4 Switch Performance

### 4.1 Traffic Model

Switch performance is evaluated under both uniform and non-uniform traffic. As shown in Fig 3, cells coming from different VCs are multiplexed in bursts which are interleaved to contribute as arrival traffic at every switch input. We employ the ON(active)/OFF(idle) model to describe the burst-idle process. The back-to-back cells in a ON duration belong to a same VC so that they have same destinations. No cells arrive in idle period.



**Fig. 3.** Traffic Model : Multicast Burst Traffic

To build a uniform traffic, we set a small value of MBS which is the maximum burst size (i.e. the number of cells in a ON duration). Cells' destinations are uniformly distributed among  $N$  switch outputs.

On the contrary, non-uniform traffic is featured by “hot spot” phenomenon : cells in an ISB prefer to go to some switch outputs, but not to other outputs. Three scenarios are likely to build a non-uniform burst traffic : (1) If maximum burst size MBS is very large, then cells in an ON period will keep targeting the same destinations for a relatively long time. Cell destinations are not uniformly distributed among  $N$  switch outputs in this time duration. (2) If bursts are correlated with each other, i.e. cells in a successive ON periods have the same destinations. Even though MBS may be small, cells accumulated in several bursts will make the traffic non-uniformly distributed among  $N$  output ports. (3) In an extreme case, an ISB has cells only destined to a specific OSB, but has no cells target for other ISBs. This is so called “1 ISB  $\rightarrow$  1 OSB hot spot” traffic.

In this paper, VBR source is used to generate the ON-OFF traffic with following traffic parameters : MBS, i.e. the maximum burst size; PCR, i.e. the peak cell rate (the number of cells/sec) which satisfies that  $PCR \leq LCR$ ; LCR, i.e. the line cell rate which approximates  $\frac{15520000}{53 \times 8} = 366,793$  (cells/sec) for an OC-3 link; ACR, i.e. the average cell rate. We define  $F_{out}$  as the fanout of a cell.  $F_{out}$  has a uniform distribution from 0 to  $C_{max}$ . The average fanout load  $F = (C_{max} + 1)/2$ , where  $C_{max}$  is the maximum copies allowed for a multicast cell. The effective input load is defined as  $\rho = ACR * F / LCR$ ,  $\rho \leq 1$ .

### 4.2 Performance Evaluation

Using OPNET/MIL3 simulation platform [11], we simulated an 256x256 ( $N = 256$ ) switch which consists of 8 ISBs and 8 OSBs ( $K = 8$ ). Each ISB/OSB is of



size  $32 \times 32$  ( $m = M = 32$ ). Meantime, we compare the proposed switch with a switch design in our previous work [9]<sup>5</sup>. As a comparison, we also simulated an  $256 \times 256$  output queued switch under the same traffic scenarios. There are two reasons for us to select OQ switch as a comparison reference : (1) OQ switches proved to maximize throughput and optimize latency under any traffic pattern; (2) in the literature so far, few of existing switch designs is dedicated for a distributed large scale switch and is investigated under any traffic condition. Therefore, we believe that it is fair and effective to compare our switch design with an OQ switch under same traffic patterns.

We investigate switch performance under both uniform and non-uniform traffic. Following performance statistics are estimated :

- **Throughput** : switch throughput which is statistically measured on N switch outputs ;
- $D_{E-to-E}$  : the average end-to-end cell delay ( # of cell slots) which is defined as the latency for a cell going through the switch;
- $D_{ISB}$  : the average cell delay in ISBs ( # of cell slots); ATMCSF is assumed to deliver cells from input shared links to output shared links in a cell slot, hence, end-to-end cell delay  $D_{E-to-E}$  is resulted from two parts : the *cell delay in ISBs (i.e.  $D_{ISB}$ )*, and the *cell delay in OSBs*.
- $S_{OSB}$  : the average occupancy of an OSB (# of cells); an OSB has  $M$  output queues,  $S_{OSB}$  indicates the total number of cells waiting in an OSB.

Table 1 shows switch performance under **uniform traffic** with different input load  $\rho$ . Both unicast traffic and multicast traffic are applied. In unicast uniform traffic, every cell arriving at a switch input only carries one destination. But, in multicast uniform traffic, a coming cell may have multiple destinations. Cells' destinations are uniformly distributed among N switch outputs.

In the simulation, the proposed enhanced switch performs link reservation with the slowest rate, i.e.  $Rsv\_Slot = Cell\_Slot$ . Hence, in a cell slot, a token can only pass through one ISB. Any faster link reservation rate (i.e.  $Rsv\_Slot < Cell\_Slot$ ) will result in better performance than what we simulated here. The values of HT and LT are selected as 4 and 2 respectively.

Under uniform traffic, both the Switch in [9] and the proposed enhanced switch can achieve a comparable performance as the OQ switch. On **throughput** performance, the OQ switch always obtains the maximized throughput  $\rho$ , while our switch designs closely approach to the OQ switch with less than 0.6% throughput degradation. In general, the **end-to-end cell delay**  $D_{E-to-E}$  increases with input load  $\rho$ . Compared with the lower bound of  $D_{E-to-E}$  achieved in the OQ switch, the  $D_{E-to-E}$  in our switch designs cause 2~20 more cell slots. Longer cell delay is due to lower throughput. Since employing dynamic link reservation, the proposed switch outperforms the Switch in [9] in the performance of throughput, cell delay and delay jitter. In addition, Table 1 shows that the end-to-end cell delay in our designs is mainly due to the latency in the OSBs

<sup>5</sup> The switch in [9] has a similar switch architecture as the proposed enhanced switch, but ISBs are isolated from each other without any connection. In a cell slot, an ISB is only responsible for delivering cells to a certain OSB according to a round robin one-to-one mapping from K ISBs to K OSBs. The switch had been demonstrated to be able to obtain a good performance under uniform traffic, but it has a weakness to support non-uniform traffic.

rather than the cell delay in ISBs. It is a good feature of the proposed switches because cells are forwarded to OSBs in a faster manner and most of the cells are queued in OSBs. Hence, the proposed switches explicit a capability to mimic the OQ switch, and OSBs may incorporate per VC queueing with appropriate cell schedulers to provide QoS guarantees. This is a subject of our ongoing work.

		Uniform, Unicast, Burst Traffic (F = 1, Fout = 1, MBS = 16)			Uniform, Multicast, Burst Traffic (F = 4, Cmax = 16, MBS = 16)		
		OQ Switch	Switch in [9]	The proposed switch Rsv_Slot = Cell Slot ; HT = 4 , LT = 2	OQ Switch	Switch in [9]	The proposed switch Rsv_Slot = Cell Slot ; HT = 4 , LT = 2
$\rho = 0.99$	Throughput	0.99	0.98440	0.98495	0.99	0.98903	0.98930
	$D_{E-to-E}$ (Min, Max)	47.25 (1.3, 197)	65.52 (6.95, 242)	62.3 (1.7, 222.8)	25 (1.0, 156)	35 (5.6, 203)	34 (2.0, 189.2)
	$D_{ISB}$ (Min, Max)	N/A	22.8 (1.0, 91)	20.2 (1.0, 63)	N/A	4.9 (1.0, 18.1)	4.2 (1.0, 25.7)
	$S_{OSB}$ (Min, Max)	N/A	1362 (1152, 1545)	1357 (1120, 1565)	N/A	768 (591, 969)	758 (572, 964)
$\rho = 0.90$	Throughput	0.90	0.89866	0.89892	0.90	0.89913	0.89932
	$D_{E-to-E}$ (Min, Max)	11.38 (1.1, 104)	18.90 (6.25, 116)	17.93 (2.0, 119)	9.5 (1.0, 97)	15.3 (3.5, 133)	14.2 (1.5, 117)
	$D_{ISB}$ (Min, Max)	N/A	5.8 (1.0, 30)	5.0 (0.1, 20)	N/A	5.9 (1.0, 17)	4.7 (1.0, 26)
	$S_{OSB}$ (Min, Max)	N/A	331.5 (249, 423)	316.5 (213, 409)	N/A	274 (201, 355)	263.5 (172, 365)
$\rho = 0.70$	Throughput	0.70	0.69918	0.69924	0.70	0.69949	0.69988
	$D_{E-to-E}$ (Min, Max)	4.01 (1.0, 46.5)	8.4 (6.6, 70)	5.2 (2.0, 67)	3.6 (1.0, 38)	8.1 (2.0, 76)	4.6 (2.0, 63)
	$D_{ISB}$ (Min, Max)	N/A	4.5 (1.0, 14)	2.2 (1.0, 10)	N/A	4.5 (1.0, 12.8)	1.3 (1.0, 12.1)
	$S_{OSB}$ (Min, Max)	N/A	100 (71, 133)	87 (59, 116)	N/A	86.5 (60, 117)	75 (52, 102)
$\rho = 0.50$	Throughput	0.50	0.49953	0.49987	0.50	0.49963	0.49994
	$D_{E-to-E}$ (Min, Max)	2.25 (1.0, 26.7)	3.1 (2.0, 48.2)	2.9 (2.0, 38.2)	2.0 (1.0, 23.5)	3.6 (1.0, 47)	2.3 (1.0, 33)
	$D_{ISB}$ (Min, Max)	N/A	2.1 (1.7, 13)	1.2 (1.7, 12)	N/A	2.1 (1.0, 10.95)	1.05 (1.0, 11.2)
	$S_{OSB}$ (Min, Max)	N/A	45 (30, 64)	35 (23, 52)	N/A	38.3 (22.3, 56.7)	29.6 (18.8, 46.8)

**Table 1.** Performance Comparison under Uniform Traffic with Different Input Load  $\rho$ . Both unicast and multicast traffic are applied.

Now, we investigate switch performance under non-uniform traffic in Table 2. The traffic applied is "1 ISB  $\rightarrow$  1 OSB HotSpot Traffic": the input traffic to the  $i^{th}$  ISB is dedicated to the  $i^{th}$  OSB, i.e. the cells in the  $i^{th}$  ISB only target the  $i^{th}$  OSB but no cell goes to other OSB. Both unicast traffic pattern and multicast traffic pattern are introduced.

Table 2 shows that the proposed enhanced switch can achieve a comparable performance as the OQ switch, but the Switch in [9] suffers a lot and can not survive under this non-uniform traffic. The reason for that is, the Switch in [9] only allows an ISB to deliver cells to its matched OSB according to a round robin one-to-one mapping. If the ISB does not have cells to go to the OSB, other

		<b>1 ISB - 1 OSB HotSpot, Unicast, Burst Traffic</b> (F = Fout = 1, Cmax = 16, MBS = 40, HotSpotBstLen = 20*5)			<b>1 ISB - 1 OSB HotSpot, Multicast, Burst Traffic</b> (F = 4, Cmax = 16, MBS = 40, HotSpotBstLen = 20*5)		
		<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4, LT = 2	<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4, LT = 2
$\rho = 0.99$	Throughput	0.99	0.13234	0.98593	0.99	0.53693	0.98986
	$D_{E-to-E}$ (Min, Max)	152 (1.0, 315)	1431 (39, 2087)	217 (2.0, 397)	113 (1.0, 338)	337 (13.5, 624)	157 (2.0, 482)
	$D_{ISB}$ (Min, Max)	N/A	1370 (1.0, 1964)	88 (1.0, 178)	N/A	313 (1.0, 596)	1.1 (1.0, 11.0)
	$S_{OSB}$ (Min, Max)	N/A	84 (5.0, 113)	4123 (3771, 4459)	N/A	188 (61.0, 169)	3842 (2576, 5028)
$\rho = 0.90$	Throughput	0.90	0.13098	0.89799	0.90	0.53381	0.89990
	$D_{E-to-E}$ (Min, Max)	72 (1.0, 178)	1070 (31, 1710)	125 (2.0, 227)	73 (1.0, 188)	289 (13.5, 429)	112 (2.0, 246)
	$D_{ISB}$ (Min, Max)	N/A	986 (7.0, 1667)	43 (1.0, 147)	N/A	267 (1.0, 375)	1.1 (1.0, 10)
	$S_{OSB}$ (Min, Max)	N/A	62 (5.5, 87)	2580 (2160, 2980)	N/A	112 (64, 178)	3175 (2048, 4738)
$\rho = 0.70$	Throughput	0.70	0.12984	0.69988	0.70	0.54085	0.69992
	$D_{E-to-E}$ (Min, Max)	26 (1.0, 75)	853 (30, 1457)	47 (2.0, 97)	15.4 (1.0, 38.2)	97 (11.7, 189)	26.3 (2.0, 57.3)
	$D_{ISB}$ (Min, Max)	N/A	837 (5.0, 1412)	19 (1.0, 65)	N/A	88 (1.1, 133)	1.1 (1.0, 7.0)
	$S_{OSB}$ (Min, Max)	N/A	12.0 (5.5, 21)	597 (436, 770)	N/A	110 (68, 203)	786 (130, 1289)
$\rho = 0.50$	Throughput	0.50	0.12746	0.49993	0.50	0.49925	0.49995
	$D_{E-to-E}$ (Min, Max)	11 (1.0, 23)	553 (2.0, 718)	17.8 (2.0, 37)	6.0 (1.0, 13.4)	13.2 (10.0, 22)	10.7 (1.0, 21.3)
	$D_{ISB}$ (Min, Max)	N/A	539 (4.0, 694)	8.7 (1.0, 17)	N/A	8.2 (2.0, 186)	1.0 (1.0, 6.0)
	$S_{OSB}$ (Min, Max)	N/A	8.0 (6.5, 9.3)	277 (120, 573)	N/A	106 (45, 178)	287 (21.3, 379)

**Table 2.** Performance Comparison under Non-uniform Traffic with Different Input Load  $\rho$ . We apply “1 ISB - 1 OSB HotSpot” traffic with two patterns — unicast traffic, and multicast traffic.

ISBs have no authority to send cells to the starved OSB. Under “1 ISB  $\rightarrow$  1 OSB HotSpot Traffic”, an ISB only has cells to be delivered in 1 out of every  $K$  cell slots. Hence, performance of the switch in [9] is very poor. It is observed that, when input load  $\rho = 0.99$ , the switch in [9] gains no more than 14% throughput under unicast traffic and obtains 54% throughput under multicast traffic. GVOQ scheme is effective in the multicast traffic so that it results in a higher throughput than unicast traffic. Since more and more cells are backlogged in ISBs, the Switch in [9] suffers an increasing cell delay.

The proposed enhanced switch, unlike the Switch in [9], utilizes dual round robin dynamic link reservation which can adapt the traffic loading to perform an efficient link resources allocation among ISBs. Starvation of OSBs is relaxed. It shows that the proposed switch can approach to a very similar performance as the OQ switch on throughput, end-to-end cell delay  $D_{E-to-E}$ , and delay jitter (Min, Max) of  $D_{E-to-E}$ . In the proposed switch, most cells are queued in OSBs,

and  $D_{E-to-E}$  is mainly resulted from the cell delay in OSBs. As we mentioned, it is a good feature of the proposed switch because OSBs may incorporate per VC queueing with appropriate cell schedulers to provide QoS guarantees. In summary, the proposed switch demonstrates a promising capability to achieve a high performance like the OQ switch under both uniform and non-uniform traffic scenarios. Compared with the OQ switch, the proposed enhanced switch eliminates  $N$  times speedup which, however, is necessary in the OQ switch.

## 5 Conclusion

In this paper, we present a scalable multicast packet switch with a modular switch architecture and a distributed dynamic link reservation algorithm. The switch benefits from input and output link sharing. It resolves output contention and virtually eliminates speedup requirement for the central switch fabric. QOBDLR is a distributed algorithm in which an ISB can dynamically increase/decrease its link reservation for a specific OSB according to its local available information. Arbitration complexity is only  $O(1)$ . QOBDLR can achieve a fast and fair link resource allocation among ISBs. Simulations on an  $256 \times 256$  switch demonstrate that the proposed switch can achieve a comparable performance as the output queued switch under any traffic pattern.

Apart from the preliminary results discussed in this paper, theoretical work on the choice of HT and LT for QOBDLR algorithm is our ongoing work.

## References

1. M. H. Guo, R. S. Chang, *Multicast ATM Switches : Survey and Performance Evaluation*, Computer Communication Review, Vol 28, No. 2, April 1998, pp. 98-131.
2. J. Turner, N. Yamanaka, *Architecture Choices in Large Scale ATM Switches*, WUCS 97-21, May 1997.
3. H. J. Chao, B. S. Choe, *Design and Analysis of A Large-Scale Multicast Output Buffered ATM Switch* IEEE/ACM Trans. on Networking, Vol. 3, No. 2, April 1995, pp. 126-138.
4. H. J. Chao, B. S. Choe, J. S. Park, N. Uzun, *Design and Implementation of Abacus Switch : A Scalable Multicast ATM Switch*, IEEE J. on Select. Areas in Commun., Vol. 15, No. 5, June 1997, pp. 830-843.
5. N. Mckeown, V. Anantharam, J. Walrand, *Achieving 100% Throughput in an Input-Queued Switch*, Proc. of IEEE Infocom96, March 1996.
6. A. Mekittikul, N. Mckeown, *A Practical Scheduling Algorithm to Achieve 100% Throughput in Input-Queued Switches*, Proc. of IEEE Infocom98, April 1998.
7. S-T. Chuang, A. Goel, N. Mckeown, B. Prabhakar, *Matching Output Queueing with Combined Input and Output Queueing*, Proc. of IEEE Infocom99, March, 1999.
8. B. Prabhakar, N. Mckeown, *Designing A Multicast Switch Scheduler*, Proc. of the 33<sup>th</sup> Annual Allerton Conference on Communication, Control and Computing, October 1995.
9. F. Chen, N. Uzun, A.N. Akansu, *A Large Scale Multicast ATM Switch With Input and Output Link Sharing*, Proc. of IEEE Globecom'99, Rio de Janeiro, Brazil, December 1999, pp. 1251-1255.
10. F. Chen, N. Uzun, A.N. Akansu, *A Scalable Multicast ATM Switch using Link Sharing and Prioritized Link Reservation*, Proc. of IEEE ICCCN'99, Boston, Massachusetts, October 1999, pp. 218-222.
11. OPNET by MIL3 Inc., Washington, DC 20008.