PERFORMANCE COMPARISON OF BRANCH POINT ALGORITHMS FOR MULTICAST ABR FLOW CONTROL

Dong-Ho Kim*, Jang-Kyung Kim *, Byung-Chul Kim**, and You-Ze Cho**
**Network Equipment Test Center, ETRI, Taejon, KOREA*
Phone: +82-42-860-6668, Fax: +82-42-961-5404
{dhkim, jkkim}@netc.etri.re.kr
***School of Electronic & Electrical Engineering, Kyungpook National University, KOREA*
bckim@palgong.kyungpook.ac.kr, yzcho@ee.kyungpook.ac.kr

Abstract This paper compares the performance of feedback consolidation algorithms with/without a fast overload indication function at a branch point switch for multicast (or point-to-multipoint) available bit rate (ABR) flow control in ATM networks. A new backward resource management (BRM) cell-discarding policy is proposed which controls additional BRM cell overhead due to fast overload indication function. The performance of various consolidation algorithms with the proposed fast overload indication function is also compared using simulations. The simulation results show that a fast overload indication function is very effective in a severe overload situation, particularly in an initial period with a higher initial cell rate. The fast overload indication function can be also combined with any feedback consolidation algorithm. However, its performance is highly dependent on the underlying basic consolidation algorithm employed.

Keywords: Feedback Consolidation Algorithm, BRM Cell Discarding Policy, Multicast ABR Flow Control, ATM.

1. INTRODUCTION

Multicast (or point-to-multipoint) available bit rate (ABR) service capabilities are essential for asynchronous transfer mode (ATM) networks to efficiently support many data applications including IP multicasting. The current version of the ATM Forum Traffic Management Specification defines ABR multicast capabilities, but it only specifies the fundamental behaviour of branch points of a multicast tree [1]. The source and destination behaviours of a multicast ABR connection are the same as those of a unicast (or point-to-point) ABR connection, except that data cells must not be transmitted in a

D. H. K. Tsang et al. (eds.), *Broadband Communications*

backward direction. A branch point switch replicates each data cell and forward resource management (FRM) cell received from the source onto each branch that leads to a destination. Each destination returns the received FRM cells to the source, a branch point therefore receives backward RM (BRM) cells as many as the number of its downstream branches for each FRM cell. The key issue at a branch-point switch for multicast ABR services is how to consolidate BRM cells returning from downstream branches to avoid feedback implosion problem [1].

A number of feedback consolidation algorithms have been proposed in [3,7,9,10,11]. Consolidating BRM cells at a branch point, however, can cause undesirable effects such as *consolidation noise*, *consolidation delay,* and *consolidation loss* [3,7]. These problems can incur a slower response, inferior fairness, longer queue length, or lower link utilization for a multicast ABR connection compared with a unicast ABR connection. Recently, Kim et al. have proposed various solutions for resolving consolidation problems and a scalable feedback consolidation algorithm [7]. Their algorithm can eliminate consolidation noise and loss. Although, this algorithm can also limit the total consolidation delay within the longest round-trip time (RTT) of the multicast tree regardless of the number of branch points, it still exhibits a higher queue length due to consolidation delay particularly when an initial period with high initial cell rate (ICR) or a severe overload condition. Recently, fast overload indication functions have been proposed by many researchers in order to reduce consolidation delay [1,2,4,6,8]. The main idea behind this is that a severe overload situation should be reported to the source as soon as it is detected.

Most previous researches on multicast ABR flow control have focused on the development of basic consolidation algorithms and their performance evaluation [3,7,9,10,11]. Some papers have evaluated performance according to various fast overload indication functions for a particular algorithm [2,4,6,8]. However, relatively little has been reported on the performance comparison of basic consolidation algorithms with the same fast overload indication function.

The remainder of this paper is as follows: Section 2 introduces the existing feedback consolidation algorithms with/without fast overload indication function briefly. Section 3 presents feedback consolidation algorithm considered in this paper and proposes a new BRM cell discarding policy to control additional BRM cell overhead due to fast overload indication function. Section 4 compares the performance of feedback consolidation algorithms with/without a fast overload indication function by simulation. Finally, Section 5 makes conclusions.

2. RELATED WORKS

2.1 Basic Feedback Consolidation Algorithms

This paper classifies basic feedback consolidation algorithms according to two components as followings; (1) *Feedback information storing method for consolidation:* how to store feedback information extracted from BRM cells received from downstream branches in local variables and then consolidate them. (2) *BRM cell returning condition:* when to return the consolidated feedback information to the source. Table 1 shows this classification of basic feedback consolidation algorithms.

Table 1 Classification of basic feedback consolidation algorithms

Feedback storing method for consolidation	Per-VC	Most existing algorithms
	Per-branch for each multicast VC	Cho [3], Kim [7]
BRM cell returning condition	Wait for FRM	Roberts [10], Tzeng [11]
	Wait for BRM after FRM received	Ren's the first algorithm [9],
	Wait for BRM from all branches	Ren's the second algorithm [9]
	Wait for BRM from the farthest destination	Cho [3], Kim [7]

To date, a number of basic feedback consolidation algorithms have been proposed in [3,7,9,10,11]. A simple basic feedback consolidation algorithm for a multicast ABR service was proposed by Roberts [10]. In his algorithm, for every BRM cell received, the branch point first consolidates the congestion information to local variables maintained on a per-VC basis, and then discards the received BRM cell. Also, whenever a branch point receives an FRM cell, it returns a BRM cell along with consolidated information to the upstream node. However, this BRM cell may only contain congestion information about the current switch and not include any from the branches. In order to reduce such consolidation noise, Tzeng et al. in [11] proposed a slightly more conservative scheme. Their main idea was that a branch point would return a BRM cell to its upstream node only after receiving at least one BRM cell from its branches subsequent to the previous feedback.

Both algorithms discussed above require switches to generate BRM cells at the branch points, incurring higher complexity in switch implementation. Ren et al. proposed two consolidation algorithms that do not require switches to generate BRM cells [9]. In their first algorithm, a BRM cell that is received from a branch immediately after an FRM cell has been received by the branch point is passed back to the source. However, this algorithm may exhibit consolidation noise, similar to that experienced in scheme [11], if the branch point has not received BRM cells from one or more branches. Accordingly, to avoid such consolidation noise, in their second algorithm a BRM cell is only passed when BRM cells have been received from all branches. Ren's the first and second algorithms have been also included in

the traffic management specification of the ATM Forum as sample branch-point switch algorithms [1].

However, in most existing algorithms, consolidating the BRM cells at a branch point can cause undesirable effects such as *consolidation noise*, *consolidation delay,* and *consolidation loss*. *Consolidation noise* means that the BRM cells returned to the source contain incorrect congestion information about some downstream branches. That is, this noise occurs due to the loss of congestion information about some branches and the loss of the latest feedback information which lead to a higher allowed cell rate (ACR) oscillation and a longer ramp-up delay respectively. Thus, it will introduce an inaccuracy into the computation of the ACR of the source. *Consolidation delay* indicates an additional delay at each branch point, since a branch point usually has to wait for an FRM cell from upstream node or BRM cells from other branches in order to relay the congestion information carried by a BRM cell to its upstream node. This additional delay can cause a slower response to congestion, resulting in a longer queue length or lower link utilization. *Consolidation loss* occurs when a branch point does not return as many BRM cells as the number of FRM cells received. Such loss of BRM cells can cause a slower response for a multicast VC [3,7].

Recently, Kim et al. have proposed various solutions for resolving those consolidation problems [7]. They have also proposed a scalable basic feedback consolidation algorithm by combining the proposed solutions. In their algorithm, a branch point stores feedback information on a per-branch basis for each VC and returns the BRM cells received from the farthest destination only. As a result, this algorithm can eliminate consolidation noise and loss, and also bounds the total consolidation delay within the longest RTT of the multicast tree regardless of the number of branch points. However, it still exhibits a larger queue length in some situations such as an initial period with a higher ICR or severe overload condition.

2.2 Feedback Consolidation Algorithms with Fast Overload Indication Function

Recently, a number of consolidation algorithms with fast overload indication functions have been proposed in [2,4,6,8] to facilitate an immediate warning about congestion when a severe overload condition occurs in a network. Table 2 summarises the characteristics of the existing consolidation algorithms with fast overload indication.

The fast overload indication was originally proposed by Jang et al. [6]. This algorithm uses Roberts' algorithm as its basic consolidation algorithm. All the other algorithms in [2,4,8] use Ren's the second algorithm as their basic consolidation algorithm.

Table 2 Feedback consolidation algorithms with fast overload indication function

Algorithms	Jang [6]	Moh [8]	Fahmy [4]	Chen [2]
Basic consolidation algorithm	Roberts' algorithm	Ren's the second algorithm	Ren's the second algorithm	Ren's the second algorithm
Overload condition detection	Last_ER, Dynamic threshold	Last_ER, Static threshold, Timer	Last_ER, Static threshold	Last_ER, Static threshold, Probability

In all the algorithms, a branch point detects an overload situation using a rate comparison with the last returned ER, Last_ER, and the current received ER, ER. If the ratio of the Last_ER and ER is larger than a specified threshold, a branch point will immediately send an extra BRM cell to the upstream node. Jang's algorithm uses a dynamic threshold using a load factor and BN (backward notification) bit in the BRM cells to detect the condition for sending an extra BRM cell. All the other algorithms use a static threshold, a timer, or a probability. However, their differences in performance are marginal, particularly when the threshold value is high [2].

Table 3 Characteristics of consolidation algorithms considered

Algorithms	1	2	3	4
Feedback storing method	Per-VC	Per-branch for each multicast VC	Per-VC	Per-branch for each multicast VC
BRM cell returning condition	Wait for all feedback	Wait for BRM from the farthest destination	Algorithm 1 + fast overload indication	Algorithm 2 + fast overload indication
Comments	Ren's the second algorithm [9]	Kim [7]	Modified from Algorithm 1	Modified from Algorithm 2

3. CONSOLIDATION ALGORITHMS CONSIDERED

Table 3 summarises the characteristics of the consolidation algorithms considered for comparison in this paper. Four consolidation algorithms are considered. Algorithms 1 and 2 are basic consolidation algorithms without fast overload indication. Algorithm 1 is Ren's the second algorithm, which has been widely used, as a basic consolidation algorithm in many papers [2,4,8,9]. Algorithm 2 is a counter-based basic feedback consolidation algorithm proposed by the authors in [7]. Algorithms 3 and 4 are the improved versions of Algorithms 1 and 2, respectively, with an added fast overload indication for reducing consolidation delay. In Algorithms 1 and 3, a branch point stores feedback information received from downstream branches on a per-VC basis. In contrast, in Algorithms 2 and 4, a branch point stores feedback information on a per-branch basis for each multicast VC.

3.1 Algorithm 1

The main idea of this algorithm is that a BRM cell is passed to the source

only when BRM cells have been received from all branches [9]. The pseudo code is as follows.

A branch point switch maintains resisters MER, XER, flags MCI, MNI, XNI, and M flags for each multicast VC.

On the receipt of an FRM (ER, CI, NI) cell:
    Let XER = ER, XNI = NI;
    Multicast this FRM cell to all participating branches;

On the receipt of a BRM (ER, CI, NI) cell from branch i:
    Set M(i) = 1 for branch i;
    Let MER = min(ER, MER), MCI = OR(CI, MCI), and MNI = OR(NI, MNI);
    If (M(j) = 1 for all the other branches j, $j \neq i$) then
        Send this BRM (MER, MCI, MNI) cell back to the source;
        Reset M(j) = 0 for all participating branches j;
        Reset MER = XER, MCI = 0, and MNI = XNI;
    Else
        Discard this BRM cell;

3.2 Algorithm 2

The main ideas of this algorithm are that each branch point stores the feedback information on a per-branch basis for each VC and only passes the BRM cells returning from the farthest destination among its all branches [7]. The pseudo code of Algorithm 2 is presented below.

A branch point switch maintains a resister MER, flags MCI, MNI, and a counter CTR for each branch.

On the receipt of an FRM (ER, CI, NI) cell:
    Multicast this FRM cell to all participating branches;

On the receipt of a BRM (ER, CI, NI) cell from branch i:
    Let CTR(i) = CTR(i)+1 for the corresponding branch i;
    Let MNI(i) = NI, MCI(i) = CI, and MER(i) = ER for the corresponding branch i;
    If (CTR(j) > 0 for all other branches j, $j \neq i$) then
        Let ER = min(MER(j)), CI = OR(MCI(j)), NI = OR(MNI(j)) for all participating branches j;
        Let CTR(j) = CTR(j)-1 for all participating branches j;
        Send this BRM (ER, CI, NI) cell back to the source;
    Else
        Discard this BRM cell;

3.3 Algorithm 3

Some variations of this algorithm were presented in [2,4,8]. The normal operation of Algorithm 3 is the same as in Algorithm 1. That is, in an underload situation a branch point can pass a BRM cell to the upstream node only when it has received feedback from all branches. A difference occurs only

when an overload situation has been detected. An overload situation is detected when the current feedback, ER, received from a branch is *much less* than the last feedback, Last_ER, returned by the branch point to the source [2,4,6,8]. Normally, a threshold is used for detecting the "much less" condition as follows:

$$\frac{ER}{Last\ \_\ ER} \leq \alpha \ ,$$

where α is a threshold and $0 < \alpha < 1$. This threshold can be determined as a static value [2,4,8] or a dynamic parameter considering the load factor [6].

In order to avoid an additional increase of BRM overhead due to fast overload indication function, a Skip_Num counter is used for each multicast VC and initialised to zero [4]. The Skip_Num is increased whenever a BRM cell is sent before the normal BRM cell returning condition is satisfied. When a normal BRM cell returning condition is satisfied and the value of the Skip_Num is greater than zero, this particular feedback can be ignored and the Skip_Num is decreased by one if the following BRM cell discarding condition is satisfied:

$$Last\ \_\ ER \leq MER \leq \beta \cdot Last\ \_\ ER \ ,$$

where β is a scaling factor and $\beta > 1$, and MER represents a consolidated ER value when the normal BRM cell returning condition is satisfied. The main idea of the proposed discarding policy is that a sudden change to an underload in the network status should be notified to the source even though the Skip_Num value is greater than zero and the normal BRM cell returning condition is satisfied. That is, the additional BRM overhead due to fast overload indication function are only controlled when the network is in a steady state so as to avoid any degradation in utilization by the indiscriminate discarding of a BRM cell indicating an underload. The pseudo code of Algorithm 3 is presented below.

A branch point maintains resisters MER, XER, Last_ER, flags MCI, XNI, MNI, Send_Flag, Reset_Flag, a counter Skip_Num, and M flags for each multicast VC.

On the receipt of an FRM (ER, CI, NI) cell:
    Let XER = ER, XNI = NI;
    Multicast this FRM cell to all participating branches;

On the receipt of a BRM (ER, CI, NI) cell from branch i:
    Let Send_Flag = 0;
    Let Reset_Flag = 0;
    Set M(i) = 1 for branch i;
    Let MER = min(ER, MER), MCI = OR(CI, MCI), and MNI = OR(NI, MNI);
    If (M(j) = 1 for all the other branches j, $j \neq i$) then
        Set Send_Flag = 1;

```
        Set Reset_Flag = 1;
If ( β *Last_ER ≥ MER ≥ Last_ER AND Skip_Num>0 AND Send_ Flag=1) then
        Let Skip_Num = Skip_Num-1;      /* β =1.1*/
        Reset Send_ Flag = 0;
If (MER < α *Last_ER) then              /* Threshold α = 0.9 */
        If (Send_Flag = 0) then
                Let Skip_Num = Skip_Num+1;
                Set Send_Flag = 1;
If (Send_Flag = 1) then
        Let Last_ER = MER;              /* Initially set to ICR */
        Send this BRM (MER, MCI, MNI) cell back to the source;
Else
        Discard this BRM cell;
If (Reset_Flag = 1) then
        Reset M(j) = 0 for all participating branches j;
        Reset MER = XER, MCI = 0, and MNI = XNI;
```

3.4 Algorithm 4

The normal operation for Algorithm 4 is the same as in Algorithm 2. That is, in an underload situation a branch point passes a BRM cell to the upstream node only when it receives feedback from the farthest destination among all its branches. The abnormal operation for Algorithm 4 in an overload situation is the same as in Algorithm 3. The pseudo code of Algorithm 4 is presented below.

A branch point maintains a resister MER, flags MCI, MNI, and a counter CTR for each branch. And it also maintains a resister Last_ER, a flag Send_Flag, and a counter Skip_Num for multicast VC.

```
On the receipt of an FRM (ER, CI, NI) cell:
        Multicast this FRM cell to all participating branches;

On the receipt of a BRM (ER, CI, NI) cell from branch i:
        Let Send_Flag = 0;
        Let ER(i) = ER, CI(i) = CI, and NI(i) = NI for branch i;
        Let CTR(i) = CTR(i)+1 for branch i;
        If (CTR(j) > 0 for all the other branches j, j ≠ i ) then
                Set Send_Flag = 1;
                Let CTR(j) = CTR(j)-1 for all participating branches j;
                Let MER = min(ER(j)), MCI = OR(CI(j)), and MNI = OR(NI(j)) for all
                    participating branches j;
        If ( β *Last_ER ≥ MER ≥ Last_ER AND Skip_Num>0 AND Send_Flag=1) then
                Let Skip_Num = Skip_Num-1;      /* β =1.1*/
                Let Send_Flag = 0;
        If (ER(i) < α *Last_ER) then            /* Threshold α = 0.9 */
                If (Send_Flag = 0) then
                        Let Skip_Num = Skip_Num+1;
                        Set Send_Flag = 1;
        If (Send_Flag = 1) then
```

```
        Let MER = min(ER(j)), MCI = OR(CI(j)), and MNI = OR(NI(j)) for all
            participating branches j;
        Send this BRM (MER, MCI, MNI) cell back to the source;
        Let Last_ER = MER;                    /* Initially set to ICR */
Else
        Discard this BRM cell;
```

4. PERFORMANCE EVALUATION

4.1 Parameter Values and Network Model

Table 4 presents the ABR parameter values used for performance evaluation in this paper. All ATM switches are assumed to be ERICA switches with a target link utilization of 90% [5]. It is also assumed that all links had a bandwidth of 155.52 Mbps.

Table 4 ABR parameter values considered

Parameter	Description	Value
PCR	Peak cell rate	155.52 Mbps
ICR	Initial cell rate	4 Mbps
MCR	Minimum cell rate	0.1 Mbps
Nrm	Number of cells between FRM cells	32
RIF	Rate increase factor	1
RDF	Rate decrease factor	1/32,768
CRM	Missing RM cell count	32
CDF	Cutoff decrease factor	1/16
ADTF	ACR decrease time factor	0.5 sec

A simple network configuration is used to investigate the scalability of consolidation algorithms with and without fast overload indication function. Figure 1 illustrates the network model. In this model, the link distance between the end stations and their access switches is assumed to be 1 km, except for the distance between the switch S_N and destination A_N that determined as 3,000 km. The link distance between the switches S_1 and S_N is maintained at 1,000 km regardless of the number of branch points N. The source A is a multicast ABR source passing through N branch points, and source B is a unicast variable bit rate (VBR) source as background traffic. It is assumed that the ABR source A is a persistent source with 20 independent VCs and the VBR source B is a periodic high(120 Mbps)/low(30 Mbps) source with a 100 ms duration for each. In this configuration, the link between S_N and S_{N+1} is always a bottleneck point.

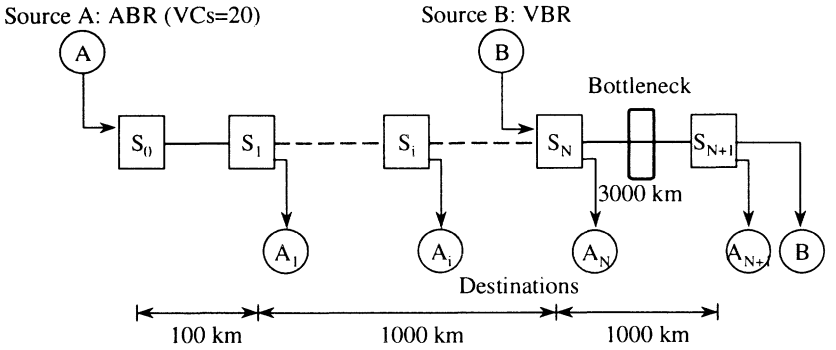


Figure 1 Network model.

4.2 Simulation Results

Figure 2 shows the ACR changes of source A according to the rate changes of the VBR source B for a different number of branch points N. All algorithms considered can eliminate ACR oscillation problem. However, Algorithms 1 and 3 suffer from ramp-up delay problem due to consolidation noise since they sometimes lose the latest feedback information due to storing it on a per-VC basis. This consolidation noise can be eliminated if each branch point stores feedback information on a per-branch basis for each multicast VC [3,7].

From this figure, the impact of the number of branch points N on ACR changes can also be observed. In the case of an overload situation, Algorithms 3 and 4 exhibit a faster ramp-down than Algorithms 1 and 2 by virtue of the fast overload indication function, regardless of N. In the case of an underload situation, Algorithms 1 and 3 exhibit a much slower ramp-up of the ACR when N=10, compared with their behaviours when N=2. In contrast, Algorithms 2 and 4 demonstrate their insensitiveness to an increase of N in terms of the ACR changes of source A. This is due to the fact that, in Algorithms 1 and 3, the total accumulated consolidation delay increases proportionally with N, whereas, in Algorithms 2 and 4 the total accumulated consolidation delay is bounded within the longest RTT in the multicast tree, regardless of N.

Figure 3 illustrates the impact of the number of branch points N on the queue length of the bottleneck switch S_N. Algorithms 3 and 4 exhibit a smaller queue length than Algorithms 1 and 2.

Table 5 compares the utilization of the bottleneck link between switches S_N and S_{N+1}. Algorithm 2 exhibits a higher link utilization than the other algorithms. Among algorithms with a fast overload indication function, Algorithm 4 produces a better performance than Algorithm 3, and is superior to Algorithm 1 which does not use a fast overload indication function. Algo-

rithms 1 and 3 are both sensitive in link utilization to the number of branch points N. On the other hand, Algorithms 2 and 4 consistently provide a higher link utilization regardless of the number of branch points.

Table 5 Link utilization at the bottleneck link

Algorithms	1	2	3	4
$N = 2$	0.85	0.88	0.82	0.85
$N = 10$	0.78	0.88	0.76	0.85

Table 6 compares the fairness index among the multicast VCs at the bottleneck link. All of the algorithms seem to be fair when N=2, however, this is not true for Algorithms 1 and 3 when N=10 due to consolidation problems. Algorithms 2 and 4 show a good fairness regardless of N.

Table 6 Fairness index at the bottleneck link

Algorithms	1	2	3	4
$N = 2$	0.99	0.99	0.98	0.99
$N = 10$	0.90	0.98	0.89	0.98

Table 7 shows the ratio of BRM cells to FRM cells at the source. Algorithms 1 and 3 suffer from a severe consolidation loss, thereby the number of BRM cells returned to the source is much lower than that of FRM cells generated by the source. However, this ratio in Algorithms 2 is consistently maintained at one without any consolidation loss. In addition, the overhead of BRM cells of Algorithms 3 and 4 are almost the same as those of Algorithms 1 and 2, respectively, since they control the additional increase of BRM cells due to a fast overload indication function.

Table 7 Ratio of BRM cells to FRM cells at the source

Algorithms	1	2	3	4
$N = 2$	0.69	1	0.69	1
$N = 10$	0.67	1	0.68	1

From these results, it can be concluded that Algorithms 3 and 4 with fast overload indication function are very effective in terms of queue length at the expense of a lower link utilization than Algorithms 1 and 2, respectively. However, in an underload situation, the behaviour of consolidation algorithms with fast overload indication function is the same as that of their basic consolidation algorithms without fast overload indication function, since fast overload indication function only works when a severe overload situation has been detected. Therefore, the overall performances are highly dependent on the underlying basic consolidation algorithms.

5. CONCLUSIONS

This paper has proposed a new discarding policy for controlling excessive BRM cell overhead due to fast overload indication function. In addition, the performance of consolidation algorithms with/without fast overload indication function were compared using simulations.

Simulation results showed that fast overload indication function was very effective in a severe overload situation, particularly in an initial period with a higher ICR. The queue length at a bottleneck switch can be significantly decreased at the expense of the link utilization. In an underload situation, however, the behaviour of consolidation algorithms with fast overload indication function is the same as that of their basic consolidation algorithms without fast overload indication function, since fast overload indication function only works when a severe overload situation has been detected. Accordingly, the overall performance of consolidation algorithms with fast overload indication function is highly dependent on their underlying basic consolidation algorithms employed.

References

[1] ATM Forum Technical Committee, "The ATM traffic management specification draft version 4.1," *STR-TM-41.01 Straw Ballot*, Mar. 1999.

[2] H-S. Chen and K. Nahrstedt, "Feedback consolidation and timeout algorithms for point-to-multipoint ABR service," *In Proc. of IEEE ICC'99*, June 1999.

[3] Y-Z. Cho, S-M. Lee, and M-Y. Lee, "An efficient rate-based algorithm for point-to-multipoint ABR service," *In Proc. of IEEE GLOBECOM'97*, Dec. 1997.

[4] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, S. Kalyanaraman, S. Kota, and P. Samudra, "Feedback consolidation algorithms for ABR point-to-multipoint connections," *In Proc. of the IEEE INFOCOM'98*, Apr. 1998.

[5] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and F. Lu, "ERICA+: Extensions to the ERICA switch algorithm," *ATM Forum/95-1346*, Oct. 1998.

[6] W-S. Jang, Y-Z. Cho, and M-Y. Lee, "Performance analysis of branch-point switch behaviors for point-to-multipoint ABR flow control in ATM networks," *In Proc. of the 23rd Korea Information Science Society Fall Conference (KISS-FC)*, Sept. 1996.

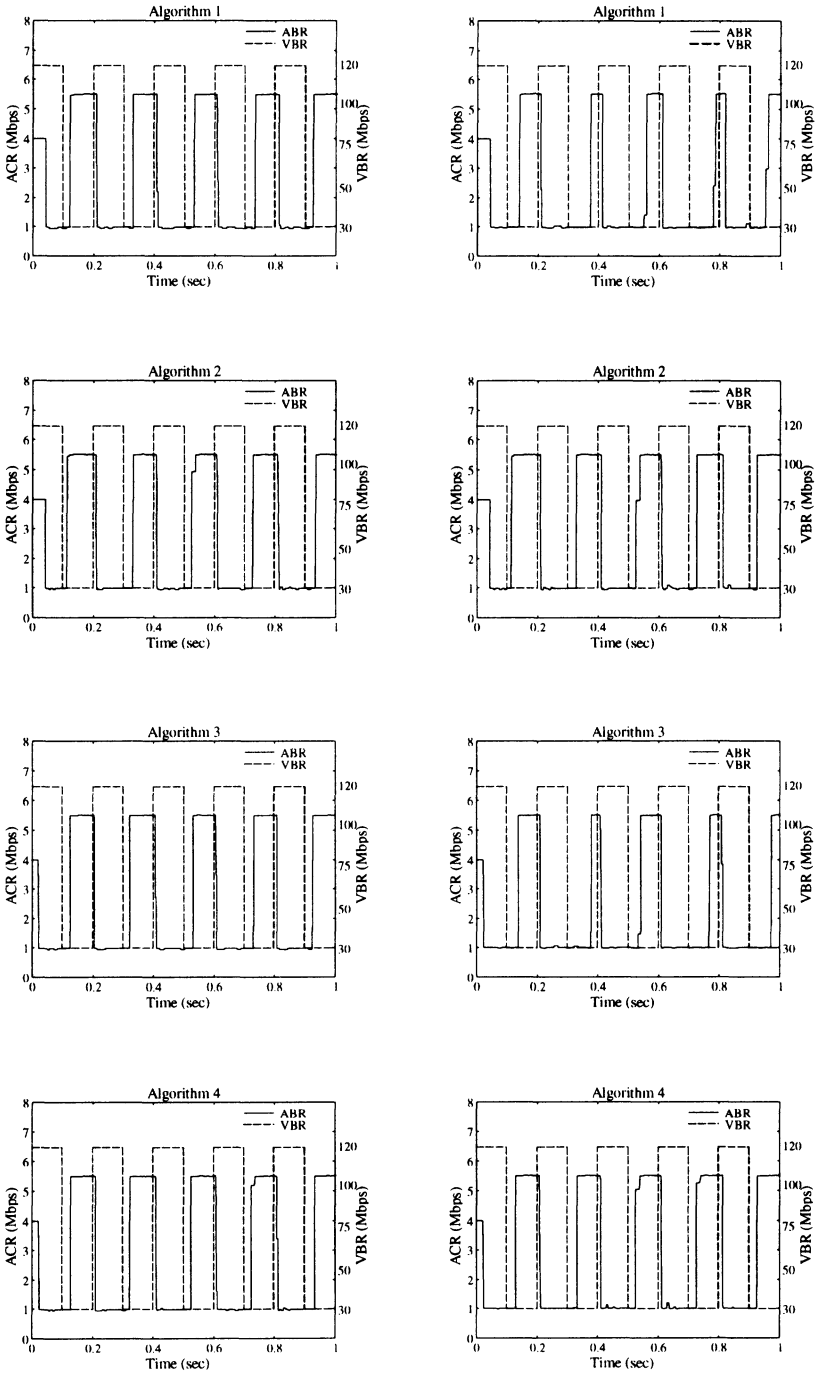
[7] D-H. Kim, Y-Z. Cho, Y-Y. An, and Y. Kwon, "A scalable consolidation algorithm for point-to-multipoint ABR flow control in ATM networks," *In Proc. of IEEE ICC'99*, June 1999.

[8] W. M. Moh, "On multicasting ABR protocols for wireless ATM networks," *In Proc. of the International Conference on Network Protocols (ICNP) '97*, Oct. 1997.

[9] W. Ren, K-Y Siu, and H. Suzuki, "On the performance of congestion control algorithms for multicast ABR service in ATM," *In Proc. of IEEE ATM'96*, Aug. 1996.

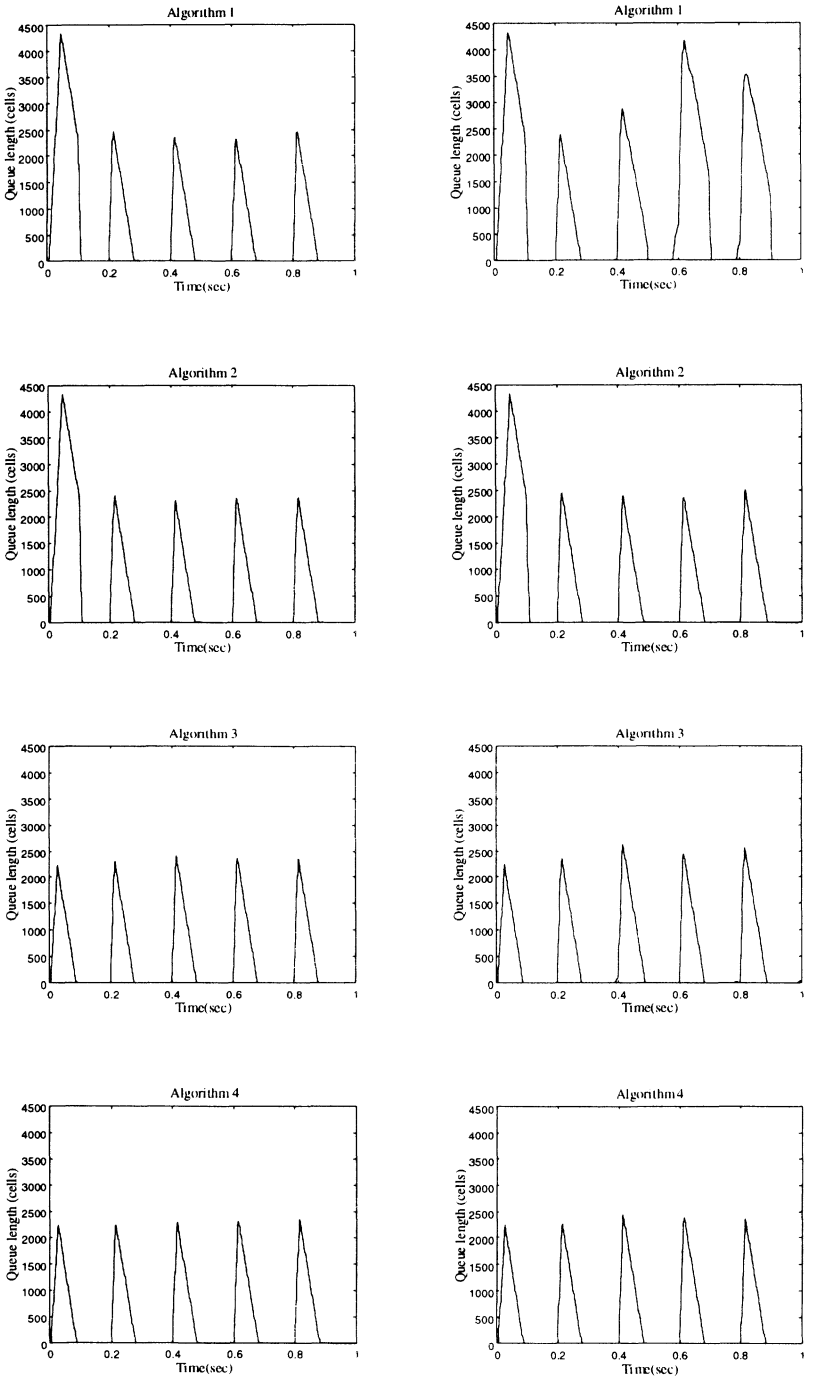
[10] L. Roberts, "Rate based algorithm for point to multipoint ABR service," *ATM Forum/94-0772R1*, Nov. 1994.

[11] H-Y. Tzeng and K-Y. Siu, "On max-min fair congestion control for multicast ABR service in ATM," *IEEE JSAC*, Vol. 15, No. 3, pp. 545-555, Apr. 1997.



2.A N=2 *2.B N=10*

Figure 2 ACR changes according to the number of branch point *N*.



3.A N=2 *3.B N=10*

Figure 3 Queue lengths according to the number of branch point *N*.