Scalable Flow Control for Multicast ABR Services

Xi Zhang† Kang G. Shin† Debanjan Saha‡ Dilip Kandlur‡

†Real-Time Computing Laboratory
EECS Dept., The University of Michigan
Ann Arbor, MI 48109
{xizhang,kgshin}@eecs.umich.edu

‡Network Systems Department
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
{debanjan,kandlur}@watson.ibm.com

Abstract—**We propose a flow-control scheme for multicast ABR services in ATM networks. At the heart of the proposed scheme is an optimal second-order rate control algorithm, called the α-control, designed to deal with the variation in RM-cell round-trip time (RTT) resulting from dynamic "drift" of the bottleneck in a multicast tree. Applying two-dimensional rate control, the proposed scheme makes the rate process converge to the available bandwidth of the connection's most congested link. It also confines the buffer occupancy to a target regime bounded by a finite buffer capacity. It works well irrespective of the topology of the multicast tree. Using the fluid approximation, we model the proposed scheme and analyze the system dynamics for multicast ABR traffic. We study the convergence properties and derive the optimal-control conditions for the α-control. The analytical results show that the scheme is stable and efficient in the sense that both the source rate and bottleneck queue length rapidly converge to a small neighborhood of the designated operating point. We present simulation results which verify the analytical observations. The simulation results also demonstrate the effectiveness of the proposed scheme in dealing with RM-cell RTT and link-bandwidth variations, and in achieving fairness in both buffer and bandwidth occupancies.**

Index Terms—**ATM, ABR, flow control, multicast, scalability, feedback soft-synchronization, RTT variations, α-control, second-order rate control, rate-gain adaptation to RTT variations, buffer occupancy setpoint and fairness.**

I. Introduction

The ABR flow-control algorithm consist of two components: determining the bottleneck link bandwidth, and adjusting the source transmissions rate to match the bottleneck link bandwidth and buffer capacity. In a multicast ABR connection, determining the bottleneck link bandwidth is a daunting task. The first generation of multicast ABR algorithms [1], [2] employ a simple hop-by-hop feedback mechanism for this purpose. In these schemes, feedback RM (Resource Management) cells from downstream nodes are consolidated at the branch points. On receipt of a forward RM cell, the consolidated feedback is propagated upwards by a single hop. While hop-by-hop feedback is very simple, it does not scale well because the RM-cell RTT is proportional to the height of the multicast tree. Additionally, unless the feedback RM cells from the down stream nodes are *synchronized* at each branch point, the source may be misled by the incomplete feedback information, which can cause the *consolidation noise* problem [3].

In order to reduce the RM-cell RTT and eliminate consolidation noise, the authors of [4], [3] proposed feedback synchronization at branch points by accumulating feedback from *all* branches. The main problem with this scheme is its slow transient response since the feedback from the congested branch may have to needlessly wait for the feedback from "longer" paths, which may not be congested at all. Delayed congestion feedback can cause excessive queue build-up and cell loss at the bottleneck link. The authors of [5] proposed an improved consolidation algorithm to speed up the transient response by sending the fast overload-congestion feedback without waiting for all branches' feedback during the transient phase.

One of the critical deficiencies of the schemes described above is that they do not detect and remove non-responsive branches from the feedback synchronization process. One or more non-responsive branches may detrimentally impact end-to-end performance by providing either stale congestion information, or by stalling the entire multicast connection. We propose a *Soft-Synchronization Protocol* (SSP) which derives a consolidated RM cell at each branch point from feedback RM cells of different downstream branches that are not necessarily responses to the same forward RM cell in each synchronization cycle. The proposed SSP not only scales well with multicast-tree's height and path lengths [6] while providing efficient feedback synchronization, but also simplifies the implementation of detection and removal of non-responsive branches. A scheme similar in spirit but different in terms of implementation has been proposed independently in [4] and [3].

As clear from the above discussion, the problem of determining the bottleneck link bandwidth in a multicast ABR connection has been addressed by many researchers. Unfortunately, little attention has been paid to the problem on how to adjust the transmission rate to match the bottleneck bandwidth and buffer capacity in the multicast context. All of the schemes proposed in the literature retrofit the transmission control mechanism used for unicast ABR connections to multicast connections. Consequently, they have overlooked an important but subtle problem that is unique to multicast ABR connections. Unlike in unicast, in a multicast connection the bottleneck may shift from one path to another within the multicast tree. As a result, the RM-cell RTT in the bottleneck path may vary significantly. Since the RTT plays a critical role in determining the effectiveness of any feedback flow-control scheme, it is important to identify and handle such dynamic drifts of the bottleneck. A failure to adapt with RM-cell RTT variations may either lead to large queue build-ups at the bottleneck or slow transient response.

A key component of the scheme proposed in this paper is an optimal second-order rate control algorithm, called the *α-control*, designed to cope with RM-cell RTT variations. Specifically, besides adapting the transmission rate based on congestion feedback, the

The work reported in this paper was supported in part by the U.S. Office of Naval Research under Grant N00014-94-0229.

source also adjusts the rate-gain parameter such that flow-control performance can be adapted to the RM-cell RTT variations. Using the fluid approximation, we model the α-control with binary feedback, and study the system dynamics under the most stressful traffic condition. We develop an optimal control condition, under which the α-control guarantees the monotonic convergence of system state to the optimal regime from an arbitrary initial value. The analytical results show that the proposed scheme is efficient and stable in that both the source rate and buffer queue at the bottleneck rapidly converge to a small neighborhood of the designated operating point. The α-control is also shown to well adapt to RM-cell RTT variations in terms of buffer requirement and fairness. The analytical results are verified by simulation experiments with multiple multicast connections where the number, the location, and the bandwidth of bottlenecks vary dynamically.

The paper is organized as follows. Section II introduces the proposed scheme. Section III establishes the flow-control system model. Section IV presents the α-control algorithm and its properties. Section V derives analytical expressions for both transient and equilibrium states and evaluates the scheme's performance for the single-connection case. Section VI deals with performance analysis for multiple multicast connections, and compares the proposed scheme with the other existing schemes through simulations. The paper concludes with Section VII.

II. The Proposed Scheme

Based on the ABR flow-control framework [7], we redefine the RM-cell format by adding both the cell-rate (first-order) and the rate-gain parameter (second-order) control information in the standard RM cell to deal with RM-cell RTT variations. In particular, two new one-bit fields, BCI (Buffer Congestion Indication) and NMQ (New Maximum Queue), are defined. Our scheme distinguishes the following two types of congestion:

Bandwidth Congestion: If the queue length $Q(t)$ at a switch becomes larger than a predetermined threshold Q_h, then the switch sets the local CI (Congestion Indication) bit to 1.

Buffer Congestion: If the maximum queue length Q_{max} at a switch exceeds the target buffer occupancy Q_{goal}, where $Q_h < Q_{goal} < C_{max}$ and C_{max} is the buffer capacity, then the switch sets the local BCI to 1.

A. The Source Algorithm

A pseudocode for the source control algorithm is presented in Fig. 1. Upon receiving a feedback RM-cell, the source must first check if it is time to exercise the buffer-congestion control (the α-control). The buffer-congestion control is triggered when the source detects a transition from a rate-decrease phase to a rate-increase phase, that is, when LCI (local congestion indicator) is equal to 1, and the CI field in the RM-cell received is set to 0. The rate-gain parameter is adjusted according to the current value of the local BCI ($LBCI$) and the BCI field in the RM-cell just received. There are three different variations: (i) if BCI is set to 1 in the RM-cell received, the rate-gain parameter AIR (Additive Increase Rate) is decreased multiplicatively by a factor of q $(0 < q < 1)$; (ii) if both $LBCI$ and BCI are set to 0, the rate-gain parameter AIR is increased additively by a step of size $p > 0$; (iii) if $LBCI$ = 1 and BCI = 0, AIR is increased multiplicatively by the same factor of q. For all these three cases, the rate-decrease parameter *MDF* (Multiplicative Decrease Factor) is adjusted according to the estimated bottleneck bandwidth BW_EST. Then, the local NMQ bit is marked and the BCI bit in the RM-cell received is saved in $LBCI$ for the next α-control cycle. The source always exercises the cell-rate (first-order) control whenever an RM-cell is received. Using the same, or updated, rate-parameters, the source additively increases, or multiplicatively decreases, its ACR (Allowed Cell Rate) according to the CI bit in the RM-cell received.

```
0. **On receipt of a feedback RM cell:**
1.   if $(LCI = 1 \wedge CI = 0)$ {  !   Buffer congestion triggering conditions
2.     if $(BCI = 1)$ $\{AIR := q \times AIR\}$  !  $AIR$ multiplitcatively decreasing
3.     elseif $(BCI = 0 \wedge LBCI = 0)$ $\{AIR := p + AIR\}$  !  Linear increase
4.     elseif $(BCI = 0 \wedge LBCI = 1)$ $\{AIR := AIR/q\}$;   !  $BCI$ toggles
5.     $MDF := exp(-AIR/BW\_EST)$;  !  Update MDF
6.     $LNMQ := 1;\ LBCI := BCI;\}$;   !  Start new measurement cycle
7.   if $(CI = 0)$ $\{ACR := ACR + AIR\}$   !  $ACR$ additively increasing
8.   else $\{ACR := ACR \times MDF\}$;  !  $ACR$ multiplicatively decreasing
9.   $LCI := CI$;   !   Saved in local register for $\alpha$-control
```

Fig. 1. Pseudocode for source end system.

B. The Switch Algorithm

At the center of switch control algorithm is a pair of connection-update vectors: (I) $conn\_patt\_vec$, the connection pattern vector where $conn\_patt\_vec(i)$ = 0 (1) indicates the i-th output port of the switch is (not) a downstream branch of the multicast connection. Thus, $conn\_patt\_vec(i)$ = 0 (1) implies that a data copy should (not) be sent to the i-th downstream branch and a feedback RM-cell is (not) expected from the i-th downstream branch;[1] (II) $resp\_branch\_vec$, the responsive branch vector is initialized to $\underline{0}$ and reset to $\underline{0}$ whenever a consolidated RM-cell is sent upward from the switch. $resp\_branch\_vec(i)$ is set to 1 if an feedback RM-cell is received from the i-th downstream branch. The connection pattern of $conn\_patt\_vec$ is updated by $resp\_branch\_vec$ each time when the non-responsive branch is detected or a new connection request is received from a downstream branch.

A simplified pseudocode of the switch control algorithm is given in Fig. 2. Upon receiving a data cell, the switch multicasts the data cell to its output ports specified by $conn\_patt\_vec$, if the corresponding output links are available, else enqueues the data cell in its branch's queue. Mark the branch's CI $(EFCI)$, if queue length $Q(t) > Q_h$. Update Q_{max} for the α-control control (to be discussed in Section IV) if the branch's new $Q(t)$ exceeds the old Q_{max}. $BCI := 1$ if its updated $Q_{max} \geq Q_{goal}$, the target buffer occupancy.

On receipt of a feedback RM cell returned from a destination receiver or a connected downstream branch, the switch first marks its corresponding bit in the $resp\_branch\_vec$ and then conducts RM-cell consolidation operations. If the modulo-2 addition (the soft-sychcronization operation), $conn\_patt\_vec \oplus resp\_branch\_vec$ = $\underline{1}$, an all 1's vector, indicating all feedback RM-cells synchronized,

[1] Note that the negative logic is used for convenience of implementation.

```
00. **On receipt of a DATA cell:**
01.   multicast DATA cell based on $conn\_patt\_vec$; ! Multicast data cell
02.   if ($data\_qu \geq Q_h$) $\{CI := 1\}$; ! (1) Bandwidth congestion detection
03.   if ($data\_qu \geq Q_{max}$) $\{Q_{max} := data\_qu\}$; ! (2) Update $Q_{max}$
04.   if ($Q_{max} \geq Q_{goal}$) $\{BCI := 1\}$ ! (3) Buffer congestion detection
05.   else $\{BCI := 0\}$; ! (1), (2), and (3) are applied to all connected branches
06. **On receipt of a feedback RM cell from $i$-th branch:**
07.   if ($conn\_patt\_vec(i) \neq 1$) $\{$  ! Only process connected branches
08.     $resp\_branch\_vec(i) := 1$; ! Mark connected and responsive branch
09.     $MCI := MCI \vee CI$;  !  Bandwidth-congestion indicator processing
10.     $MBCI := MBCI \vee BCI$; ! Buffer-congestion indicator processing
11.     $MER := \min\{MER, ER\}$; ! ER information processing
12.     if ($conn\_patt\_vec \oplus resp\_branch\_vec = \underline{1}$) $\{$ ! soft synchronization
13.       send RM cell ($dir := back$, $ER := \min_{resp\_branches} MER$,
14.         $CI := \bigcup_{resp\_branches} MCI$, $BCI :=$
15.         $\bigcup_{resp\_branches} MBCI$); ! Send fully-consolidated RM cell up
16.       $no\_resp\_timer := N_{nrt}$; ! Reset non-responsive timer
17.       $resp\_branch\_vec := \underline{0}$; ! Reset responsive branch vector
18.       $MCI := 0$; $\}\}$; ! Reset RM-cell control variable
19. **On receipt of a forward RM cell:**
20.   multicast RM cell based on $conn\_patt\_vec$; ! Multicast RM cell
21.   if ($NMQ = 1$) $\{MBCI := 0;\ Q_{max} := 0;\}$; ! Start new measure cycle
22.   $no\_resp\_timer := no\_resp\_timer - 1$; ! No-responsive branch checking
23.   if ($no\_resp\_timer = 0$) $\{$ ! There is some non-responsive branch
24.     $conn\_patt\_vec := resp\_branch\_vec \oplus \underline{1}$; ! update connect. pattern vec.
25.     if ($resp\_branch\_vec \neq \underline{0}$) $\{$ ! There is at least one responsive branch
26.       send RM cell ($dir := back$,  $ER := \min_{resp\_branches} MER$,
27.         $CI := \bigcup_{resp\_branches} MCI$,  $BCI :=$
28.         $\bigcup_{resp\_branches} MBCI$); ! Send partially-consolidated RM cell up
29.       $no\_resp\_timer := N_{nrt}$; ! Reset non-responsive timer
30.       $resp\_branch\_vec := \underline{0}$; ! Reset responsive branch vector
31.       $MCI := 0$;  $MER := ER$; $\}\}$; ! Reset RM-cell control variables
32. **On receipt of a join request from $j$-th branch:**
33.   $conn\_patt\_vec(j) := 0$; ! Add branch in established multicast connection
```

Fig. 2. Pseudocode for intermediate switch system.

then a fully-consolidated feedback RM-cell is generated and sent upward. But, if the modulo-2 addition is not equal to $\underline{1}$, the switch needs to await other feedback RM-cells for synchronization. Since the switch control algorithm does not require that a consolidated RM-cell be derived from the feedback RM-cells corresponding to the same forward RM-cell, the feedback RM-cell consolidation is "softly-synchronized".

Upon receiving a forward RM-cell, the switch first multicasts it to all the connected branches specified by $conn\_patt\_vec$. Then, clear Q_{max} and the buffer congestion indicator $MBCI$ if an NMQ request is received. The non-responsive timer no_resp_timer is initialized to a threshold N_{nrt} and reset to N_{nrt} whenever a consolidated RM-cell is sent upward. The predetermined time-out value N_{nrt} for non-responsive checking is determined by such factors as the difference between the maximum and minimum RM-cell RTTs. We use the forward RM-cell arrival time as a natural clock for detecting/removing non-responsive branches (such that it will still work even in the presence of faults in the downstream branches). Each time a switch receives a forward RM-cell, the multicast connection's no_resp_timer is decreased by one. If $no\_resp\_timer = 0$ (time-out) and $resp\_branch\_vec \neq \underline{0}$ (i.e., there is at least one downstream branch responsive), then the switch will stop awaiting arrival of feedback RM-cells and immediately generate a partially-consolidated RM-cell, and send it upward. Whenever $no\_resp\_timer = 0$ is detected, at least one non-responsive downstream branch is detected and will be removed by the simple complementary operation: $conn\_patt\_vec := resp\_branch\_vec \oplus \underline{1}$, which updates $conn\_patt\_vec$.

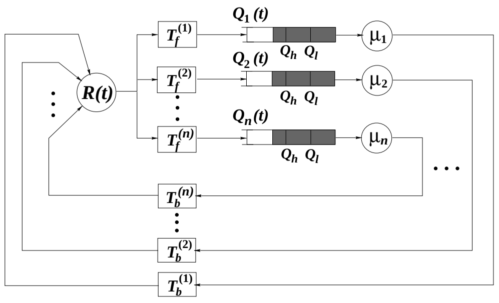

Fig. 3. The system model for a multicast connection.

Therefore, a downstream branch which has not sent any feedback RM-cell for N_{nrt} forward RM-cell time units will be removed from the multicast tree. On the other hand, a downstream node can join the multicast connection, at run-time, by submitting a join-in request to its immediate upstream branching-switch. So, our algorithm supports the dynamic reconfiguration of the multicast tree.

III. The System Model

Our proposed scheme can support both (1) CI-based rate control: a binary (CI-bit) feedback scheme; and (2) ER-based rate-control: an explicit-rate (ER-value) feedback scheme. The CI-based scheme is more suitable for LANs because of its minimum multicast signaling cost and lowest implementation complexity. Compared with CI-based scheme, the ER-based scheme is more responsive to network congestion and can better serve WANs environments where bandwidth-delay product is large. However, the ER-based scheme is much more expensive to implement as compared to the CI-based scheme. In this paper, we will focus only on the CI-based scheme modeling and analysis (and report the results on ER-based scheme in another paper). We model the CI-based flow-control system by the first-order fluid approximation method [8], [9], [10], [11], which uses the continuous-time functions $R(t)$ and $Q(t)$ as the fluid approximations of the source rate and bottleneck-queue length, respectively. We also assume the existence of only a single bottleneck[2] at a time with queue length equal to $Q(t)$ and a "persistent" source (the most stressful traffic condition) with $ACR = R(t)$ for each multicast connection.

A. System Description

As shown in Fig. 3, a multicast connection model consists of n paths with RM-cell RTTs equal to $\tau_1, \tau_2, \cdots, \tau_n$, and bottleneck bandwidths $\mu_1, \mu_2, \cdots, \mu_n$. There is only a single bottleneck on each path and its location may change with time. $T_f^{(i)}$ represents the "forward" delay from the source to the bottleneck, and $T_b^{(i)}$ the "backward" delay from the bottleneck to the source via the destination node of the i-th path. Clearly, $T_b^{(i)} = \tau_i - T_f^{(i)}$. Each

[2]This is not a restriction, because the bottleneck is defined as the most congested link or switch.

path's bottleneck has its own queue length function $Q_i(t), i = 1, 2, \cdots, n$. All paths in a multicast connection "interact" with each other via their "shared" source rate $R(t)$.

We use the synchronous model for rate control in which the periodic update interval Δ is usually a fraction of the RTT. The additive increase and multiplicative decrease of rate control during the n-th rate-update interval are expressed as:

$$R_n = \begin{cases} R_{n-1} + a; & \text{additively increase, } a = AIR \\ bR_{n-1}; & \text{multiplicatively decrease, } b = MDF \end{cases} \tag{1}$$

where $a > 0$ and $0 < b < 1$.

B. System Control Factors

In unicast ABR service, the source rate is regulated by the minimum bandwidth of all links along the path from source to destination. A natural extension of this strategy to multicast ABR service is to adjust the source rate to the minimum bandwidth that can be supported by all paths from the source to every receiver in the multicast tree. The minimum-bandwidth strategy is the key feature of ABR service most suitable for *reliable* data transmission. Thus, at any given time, the most congested path (with minimum bandwidth) in a multicast tree governs the dynamic behavior of the flow-control system. To explicitly model this feature, we introduce the following definition.

Definition 1: The **multicast-tree bottleneck** is the path whose feedback dictates the source rate-control actions. The **multicast-tree RM-cell RTT** is the RM-cell RTT experienced on the multicast-tree bottleneck.

Since the multicast-tree bottleneck dictates the source rate-control actions, we can analyze the multicast flow-control system by focusing on its multicast-tree bottleneck's state equations. Let $R(t)$ and $Q(t)$ be the fluid functions of the source rate and the queue length at the multicast-tree bottleneck, respectively, and $\tau = T_f + T_b$ be the multicast-tree RM-cell RTT. Then, the multicast-tree bottleneck state is specified by the two state variables, $R(t)$ and $Q(t)$. According to the rate-control algorithms described by Eq. (1), the multicast-tree bottleneck state equations in the continuous-time domain are given by:

Source-rate function:

$$R(t) = \begin{cases} R(t_0) + \alpha(t - t_0); & \text{If } Q(t - T_b) < Q_l \\ R(t_0)e^{-(1-\beta)\frac{(t-t_0)}{\Delta}}; & \text{If } Q(t - T_b) \geq Q_h \end{cases} \tag{2}$$

Multicast-tree bottleneck queue function:

$$Q(t) = \int_{t_0}^{t} [R(v - T_f) - \mu]dv + Q(t_0), \tag{3}$$

where $\alpha = a/\Delta$ and $\beta = 1 + log\ b$; t is the current time and t_0 is the time of the last rate-update; Q_h (Q_l) is the high (low) queue-threshold for the multicast-tree bottleneck's buffer, and $\mu = \min\{\mu_1, \mu_2, \cdots, \mu_n\}$ is the multicast-tree bottleneck bandwidth.

IV. The Second-Order Rate Control

As discussed in [11], increasing or decreasing $R(t)$ is not effective enough to have the maximum queue length Q_{max} upper bounded by the maximum buffer capacity C_{max} when the multicast-tree RM-cell RTT τ varies due to drift of multicast-tree bottleneck. This is because rate-increase/decrease control can only make $R(t)$ fluctuate around the designated bandwidth, but cannot adjust the rate-fluctuation amplitude that determines Q_{max}. In [11], [6], Q_{max} is analytically shown to increase with both τ and rate-gain parameter $\alpha = \frac{dR(t)}{dt}$ and can be written as a function, $Q_{max}(\tau, \alpha)$, or $Q_{max}(\alpha)$ for a given τ. Thus Q_{max} can be controlled by adjusting α in response to the variation of τ. The control over α — which we call *α-control* — is the second-order control over $R(t)$, providing one more dimension to control the dynamics of the proposed flow-control.

A. α-Control

The α-control is a discrete-time control process since it is only exercised when the source rate control is in a "decrease-to-increase" transition based on the the buffer congestion feedback signal BCI. $BCI(n) := 0$ (or 1) if $Q_{max}^{(n)} \leq Q_{goal}$ (or $Q_{max}^{(n)} > Q_{goal}$), where Q_{goal} ($Q_h < Q_{goal} < C_{max}$) is the target buffer occupancy (also called a *setpoint*) in the equilibrium state. If the multicast-tree bottleneck shifts from a shorter path to a longer one, then τ will increase, making Q_{max} larger. When Q_{max} eventually grows beyond Q_{goal}, buffer will tend to overflow, implying that the current α is too large for the increased τ. The source must reduce α to prevent cell losses. On the other hand, if τ decreases from its current value due to the shift of the multicast-tree bottleneck from a longer path to a shorter one, then Q_{max} will decrease. When $Q_{max} < Q_{goal}$, only a small portion of buffer space will be utilized, implying that the current α is too small for the decreased τ. The source should increase α to avoid buffer under-utilization and improve responsiveness in grabbing available bandwidth. So, feedback BCI contains the information on RM-cell RTT variation. Keeping $Q_h < Q_{goal} < C_{max}$ has two benefits: (1) the source can quickly grab available bandwidth; (2) it can achieve high throughput and network resource utilization.

The main purpose of α-control is to handle the buffer congestion resulting from the variation of τ. We set three goals for α-control: (1) ensure that $Q_{max}^{(n)}$ quickly converges to, and stays within, the neighborhood of Q_{goal}, which is upper-bounded by C_{max}, from an arbitrary initial value by driving their corresponding rate-gain parameters α_n to the neighborhood of α_{goal} for a given τ; (2) maintain statistical fairness on the buffer occupancy among multiple multicast connections which share a common multicast-tree bottleneck; (3) minimize the extra cost incurred by the α-control algorithm. To achieve these goals, we propose a "converge and lock" α-control law in which the new value α_{n+1} is determined by α_n, and the feedback information BCI on Q_{max}'s current and one-step-old values, $Q_{max}^{(n)}$ and $Q_{max}^{(n-1)}$. The α-control law can be expressed by the following equations:

$$\alpha_{n+1} = \begin{cases} \alpha_n + p; & \text{if } BCI(n-1, n) = (0, 0), \\ q\alpha_n; & \text{if } BCI(n) = 1, \\ \alpha_n/q; & \text{if } BCI(n-1, n) = (1, 0), \end{cases} \tag{4}$$

where q is the α-decrease factor such that $0 < q < 1$ and p is the

α-increase step-size, whose values will be discussed next.

B. The Properties of the α-Control

To characterize the α-control convergence, we first introduce the following two definitions.

Definition 2: The **neighborhood** of target buffer occupancy Q_{goal} is specified by $\{Q^l_{goal}, Q^h_{goal}\}$ with

$$Q^l_{goal} \triangleq \max_{n\in\{0,1,2,\cdots\}} \{Q^{(n)}_{max} \mid Q^{(n)}_{max} \le Q_{goal}\} \quad (5)$$

$$Q^h_{goal} \triangleq \min_{n\in\{0,1,2,\cdots\}} \{Q^{(n)}_{max} \mid Q^{(n)}_{max} \ge Q_{goal}\} \quad (6)$$

where $Q^{(n)}_{max}$ is governed by the proposed α-control law.

Definition 3: $\{Q^{(n)}_{max}\} \triangleq \{Q_{max}(\alpha_n)\}$ is said to **monotonically** converge to Q_{goal}'s neighborhood at time $n = n^*$ from its initial value $Q^{(0)}_{max} = Q_{max}(\alpha_0)$, if $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (0,0,0,0,\cdots,0,1,0,1,0,\cdots)$ for $\alpha_0 < \alpha_{goal}$; and $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (1,1,1,1,\cdots,1,0,1,0,1,\cdots)$ for $\alpha_0 > \alpha_{goal}$.

The α-control is applied either in *transient* state, during which $Q^{(n)}_{max}$ has not yet reached Q_{goal}'s neighborhood, or in *equilibrium* state, in which $Q^{(n)}_{max}$ fluctuates within Q_{goal}'s neighborhood periodically. The α-control aims at making $Q^{(n)}_{max}$ converge rapidly in transient state and staying steadily within its neighborhood in equilibrium state. The following theorem summarizes the α-control law's convergence properties, optimal control conditions, and the method of computing the α-control parameters in both the transient and equilibrium states. Note that Q^l_{goal} and Q^h_{goal} are the closest attainable points around Q_{goal}, but Q_{goal} may not necessarily be the midpoint between Q^l_{goal} and Q^h_{goal}. The actual location of Q_{goal} between Q^l_{goal} and Q^h_{goal} depends on all rate-control parameters and the initial value of α_0.

Theorem 1: Consider the proposed α-control law Eq. (4) which is applied to a multicast connection with its multicast-tree bottleneck characterized by Q_{goal}, Q_h, and τ. **If** (1) $\alpha = \alpha_0$, an arbitrary initial value at time $n = 0$, (2) $0 < q < 1$, and (3) $p \le \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2$, **then** (1) in transient state the α-control law guarantees $Q^{(n)}_{max}$ to **monotonically** converge to Q_{goal}'s neighborhood, and (2) in equilibrium state the fluctuation amplitudes of $Q^{(n)}_{max}$ around Q_{goal} are bounded as follows:

$$Q^h_{gaol} - Q_{goal} \le \tau^2\alpha_{goal}\left(\frac{1}{q}-1\right) + \tau\sqrt{8\alpha_{goal}Q_h}\left(\frac{1}{\sqrt{q}}-1\right) \quad (7)$$

$$Q_{gaol} - Q^l_{goal} \le \tau^2\alpha_{goal}(1-q) + \tau\sqrt{8\alpha_{goal}Q_h}(1-\sqrt{q}) \quad (8)$$

and the diameter of neighborhood for the target buffer occupancy Q_{goal} is bounded by

$$Q^h_{gaol} - Q^l_{goal} \le \tau^2\alpha_{goal}\left(\frac{1}{q}-q\right) + \tau\sqrt{8\alpha_{goal}Q_h}\left(\frac{1}{\sqrt{q}}-\sqrt{q}\right) \quad (9)$$

where α_{goal} is the rate-gain parameter corresponding to Q_{goal}.

Proof: The proof is available in [6]. □

Remarks: The α-control law is similar to, but differs from, additive-increase/multiplicative-decrease algorithm in the following senses. In the transient state, the α-control law behaves like an additive-increase/multiplicative-decrease algorithm, which accommodates statistical convergence-to-fairness of buffer utilization among the multiple multicast connections sharing a common multicast-tree bottleneck. On the other hand, in equilibrium state, the α-control law guarantees buffer occupancy to be locked with its setpoint region at the first time when $Q^{(n)}_{max}$ reaches Q_{goal}'s neighborhood, regardless of the initial value α_0. In contrast, the additive-increase/multiplicative-decrease does not guarantee this monotonic convergence since α-control is a discrete-time control process and its convergence is dependent on α_0. The monotonic convergence ensures that $Q^{(n)}_{max}$ quickly converges to, and stays within, the neighborhood of its target value Q_{goal}. The extra cost paid for achieving these benefits is minimized since only a binary bit, BCI, is conveyed from the network bottleneck and two bits are used to store the current, and one-step-old feedback information, $BCI(n-1)$ and $BCI(n)$, at the source. The α-increase step-size p specified by condition (3) in *Theorem 1* is a function of α-decrease factor q. A large q (small decrease step-size) requests a small p for the monotonic convergence. By the condition (3) of *Theorem 1*, if $q \to 1$, then $p \to 0$, which is expected since for a stable convergent system, zero decrease corresponds to zero increase in system state. According to Eqs. (7), (8), and (9), when $q \to 1$, $Q^l_{goal}, Q^h_{goal} \to Q_{goal}$, i.e., $Q^{(n)}_{max}$'s fluctuation amplitude approaches zero, which also makes sense since $q \to 1$ implies $p \to 0$, thus $Q^{(n)}_{max}$ approaches a constant for all n.

To balance $R(t)$'s increase and decrease rates and to ensure the average of the offered traffic load not to exceed the bottleneck bandwidth, each time when α_n is updated by the α-control law specified by Eq. (4), the proposed algorithm also updates the rate decrease factor by $\beta_n = 1 - \frac{\alpha_n}{\mu}\Delta$ accordingly.

V. Single Connection Bottleneck Dynamics

A. Equilibrium-State Analysis

The system is said to be in the equilibrium state if $R(t)$ and $Q(t)$ have already converged to a certain regime and oscillate with a constant frequency and a steady average amplitude. In this state, $R(t)$ fluctuates around μ, and $Q^{(n)}_{max}$ around Q_{goal}. The fluctuation amplitudes and periods are determined by the rate-control parameters α, β; link bandwidth μ; target buffer occupancy Q_{goal}; α-control parameters p, q; congestion detection thresholds Q_h, Q_l; and delays T_b, T_f. To simplify the analysis of equilibrium state, we assume that the α-control parameters (i.e., α_0, Q_{goal}, p, and q)

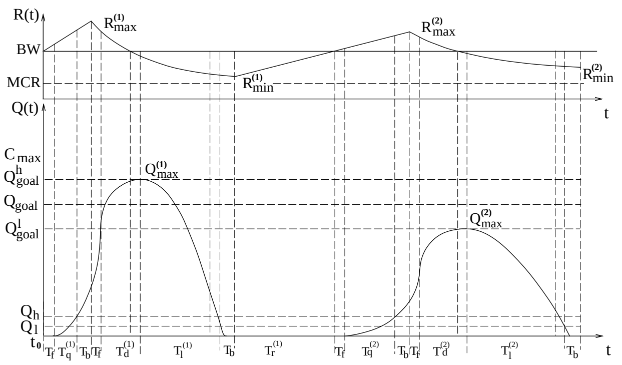

Fig. 4. Dynamic of $R(t)$ and $Q(t)$ for a multicast VC.

are properly selected according to the conditions specified in *Theorem 1*, such that $Q_{max}^{(n)}$ converges to a symmetric neighborhood of Q_{goal} where $Q_{goal} = \frac{1}{2}(Q_{goal}^{l} + Q_{goal}^{h})$ and $Q_{goal}^{h} < C_{max}$.

Fig. 4 illustrates the first two cycles of rate fluctuation and the associated queue-length function at the bottleneck link in equilibrium state with $\alpha_1 = \alpha_{goal}^{h}$. At time t_0, the rate reaches the link bandwidth μ and the queue starts to build up after a delay of T_f. At time $t_0 + T_b + T_q^{(1)}$, $Q(t)$ reaches Q_h and bandwidth congestion is detected. After a backward delay of T_b, the source receives $CI = 1$ feedback and its rate begins to decrease exponentially. $Q(t)$ reaches the peak as $R(t)$ drops back to the link bandwidth μ. When the rate falls below the link bandwidth, $Q(t)$ starts to decrease. After a time period of T_l elapsed, $Q(t)$ reaches Q_l, then the non-congestion condition $(CI = 0)$ is detected and sent backward to the source. After a delay of T_b, the $(CI = 0)$ feedback arrives at the source, then the "rate-decrease to rate-increase" transition condition $(local\_CI = 1 \wedge CI = 0)$ is detected at the source. Subsequently, the source adjusts the next rate-gain parameter α_2 to a smaller value of $q\alpha_1$ (β_2 is also adjusted accordingly by $\beta_2 = 1 - \frac{\alpha_2}{\mu}\Delta$) since $BCI(1) = 1$ (due to $Q_{max}^{(1)} > Q_{goal}$) is received in the feedback RM cell. Then, the source rate increases linearly with the newly updated rate-gain parameter $\alpha_2 = q\alpha_1 = \alpha_{goal}^{l}$. When $R(t)$ reaches μ after a time period of $T_r^{(1)}$, the system starts the second fluctuation cycle.

The dynamic behavior of the second cycle of fluctuation follows a similar pattern to that in the first cycle except for the adjusted rate-control parameters α_2 and β_2 resulting in a longer cycle length due to smaller increase/decrease rates. When the transition from rate-decrease to rate-increase is detected again for the second fluctuation cycle, the source sets $\alpha_3 = \alpha_2/q$ because $Q_{max}^{(2)} < Q_{goal}$, i.e., $BCI(2) = 0$, hence $BCI(1,2) = (1,0)$. But $\alpha_3 = \alpha_2/q = (q\alpha_1)/q = \alpha_1$ since α_n has already converged to $\{\alpha_{goal}^{l}, \alpha_{goal}^{h}\}$ in equilibrium state. Thus, the dynamic behavior of the third fluctuation cycle is exactly the same as the first cycle. Likewise, the fourth cycle is the same as the second one, and so on. So, we can only focus on the dynamic behavior of the first fluctuation cycle $T_1 = 2(T_f + T_b) + T_q^{(1)} + T_d^{(1)} + T_l^{(1)} + T_r^{(1)}$ and the second fluctuation cycle $T_2 = 2(T_f + T_b) + T_q^{(2)} + T_d^{(2)} + T_l^{(2)} + T_r^{(2)}$. We define the *control period* to be $T = T_1 + T_2$.

In the i-th fluctuation cycle $(i = 1, 2)$, let $R_{max}^{(i)}$ and $R_{min}^{(i)}$ be its maximum and minimum rates, respectively, and $Q_{max}^{(i)}$ be its maximum queue length, then we have

$$R_{max}^{(i)} = \mu + \alpha_i(T_q^{(i)} + T_b + T_f) \qquad (10)$$

where $T_q^{(i)} = \sqrt{\frac{2Q_h}{\alpha_i}}$ is the time for the queue length to grow from 0 to Q_h, $\alpha_1 = \alpha_{goal}^{h} = \alpha_{goal}^{l}/q$ and $\alpha_2 = q\alpha_1 = \alpha_{goal}^{l}$. For convenience of presentation, we define

$$T_{max}^{(i)} \triangleq T_b + T_q^{(i)} + T_f = T_b + \sqrt{\frac{2Q_h}{\alpha_i}} + T_f \qquad (11)$$

which is the time for $R(t)$ to increase from μ to its maximum $R_{max}^{(i)}$ by exercising linear rate-increase control. Then, the maximum queue length is expressed as

$$Q_{max}^{(i)} = \int_0^{T_{max}^{(i)}} \alpha_i t \, dt + \int_0^{T_d^{(i)}} (R_{max}^{(i)} e^{-(1-\beta_i)\frac{t}{\Delta}} - \mu) dt$$

where $T_d^{(i)} = -\frac{\Delta}{(1-\beta_i)} \, log \, \frac{\mu}{R_{max}^{(i)}}$. Thus, we obtain

$$Q_{max}^{(i)} = \frac{\alpha_i}{2}[T_{max}^{(i)}]^2 + \frac{\Delta}{1-\beta_i}\left[\alpha_i T_{max}^{(i)} + \mu \, log \, \frac{\mu}{R_{max}^{(i)}}\right]. \qquad (12)$$

Letting $T_l^{(i)}$ be the period for $Q(t)$ to decrease from $Q_{max}^{(i)}$ to Q_l, we have

$$Q_{max}^{(i)} - Q_l = \int_0^{T_l^{(i)}} \mu(1 - e^{-(1-\beta_i)\frac{t}{\Delta}}) dt. \qquad (13)$$

So, $T_l^{(i)}$ is the non-negative real root of non-linear equation:

$$e^{-(1-\beta_i)\frac{T_l^{(i)}}{\Delta}} + \frac{1-\beta_i}{\Delta}\left[T_l^{(i)} - \frac{Q_{max}^{(i)} - Q_l}{\mu}\right] - 1 = 0. \qquad (14)$$

The minimum rate is given by $R_{min}^{(i)} = \mu e^{-(1-\beta_i)\frac{(T_l^{(i)} + T_b + T_f)}{\Delta}}$.

The control period is determined by

$$T = \sum_{i=1}^{2} T_i = \sum_{i=1}^{2}\left[T_q^{(i)} + T_d^{(i)} + T_l^{(i)} + 2\tau + T_r^{(i)}\right] \qquad (15)$$

where $T_r^{(i)} = (\mu - R_{min}^{(i)})/\alpha_{i+1}$ is the time for $R(t)$ to grow from $R_{min}^{(i)}$ to μ with the increase-rate parameter α_{i+1} $(\alpha_3 = \alpha_1)$. Note that each T_i contains two RTTs, which correspond to the two transitions of $R(t)$ (from linear to exponential and then back to linear).

The average equilibrium throughput, denoted by $\overline{R}$, can be calculated by averaging $R(t)$ over control period T, and thus we get

$$\overline{R} = \frac{1}{T}\sum_{i=1}^{2}\left[\mu T_{max}^{(i)} + \frac{\alpha_i}{2}[T_{max}^{(i)}]^2 + R_{max}^{(i)}\left(\frac{\Delta}{1-\beta_i}\right)\right.$$
$$\left.\cdot\left(1 - e^{-\left[T_d^{(i)} + T_l^{(i)} + \tau\right]\frac{1-\beta_i}{\Delta}}\right) + T_r^{(i)} R_{min}^{(i)} + \frac{\alpha_{i+1}}{2}[T_r^{(i)}]^2\right].$$

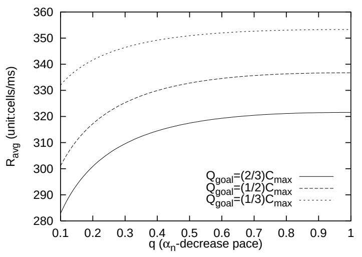

Fig. 5. $\overline{R}$ vs. q

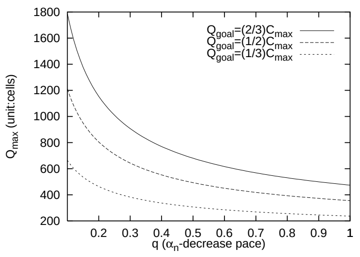

Fig. 6. Q_{max} vs. q

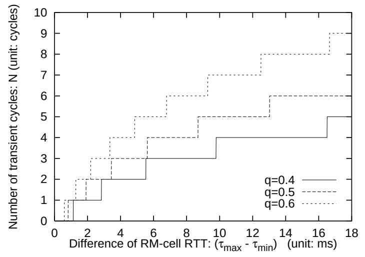

Fig. 7. N vs. $\tau_{max} - \tau_{min}$

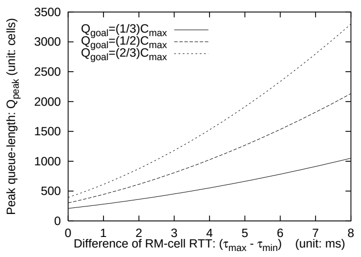

Fig. 8. Q_{peak} vs. $\tau_{max} - \tau_{min}$

B. Equilibrium-State Performance Evaluation

Assume (i) the bottleneck link bandwidth $\mu = 155$ Mbps (367 cells/ms) and $C_{max} = 750$ cells, and (ii) the bottleneck is detected at a node farthest away from the source, so, $T_b = T_f = 1$ ms and $\tau = T_b + T_f = 2$ ms. Also, we use $\Delta = 0.5\tau = 1$ ms, $Q_h = 50$ cells, $Q_l = 25$ cells, and the initial source rate $R_0 = \mu$ as we are dealing with equilibrium state.

Fig. 5 plots the average throughput $\overline{R}$ vs. q for different values of Q_{goal}. We first focus on the ideal case where $Q_{goal} = \frac{1}{2}(Q^h_{goal} + Q^l_{goal})$, i.e., $Q^{(n)}_{max}$ fluctuates symmetrically above and below Q_{goal}. Fig. 5 shows that $\overline{R}$ monotonically increases as q grows from 0.1 to 1.0. This is expected since a smaller q leads to a larger fluctuation of $R^{(n)}_{max}$ and $Q^{(n)}_{max}$, which defeats the equilibrium-state performance of $\overline{R}$. When q gets larger, the fluctuation amplitudes of $Q^{(n)}_{max}$ and $R^{(n)}_{max}$ get smaller, as shown in *Theorem 1*. In the extreme case when $q \to 1$ (q cannot be equal to 1 since $q = 1$ means that the α-control is shut down), $R^{(n)}_{max}$ approaches a constant value, and the equilibrium-state performance of $\overline{R}$ attains its maximum. Fig. 5 also indicates that for the same value of q, a smaller value of $Q_{goal} = kC_{max}$, $0 < k < 1$, leads to a larger $\overline{R}$ in equilibrium state, which is also consistent with our observations in [11], since a smaller Q_{goal} implies a smaller α_{goal}. In summary, Fig. 5 shows (i) an increasingly sharp drop in $\overline{R}$ when q gets smaller than 0.4, and (ii) a slow gain in $\overline{R}$ when $q > 0.6$, providing information on how to select q for the α-control to operate in a balanced region within which an optimal balance between average throughput and response speed is achieved.

Although Q_{goal} can be anywhere between Q^l_{goal} and Q^h_{goal}, depending on α_0, in order to analyze how q affects the maximum buffer requirement, we consider the worst case when $Q_{goal} \overset{\sim}{\geq} Q^l_{goal}$. Fig. 6 plots Q_{max} vs. q in the worst case of buffer requirement. Q_{max} is observed to increase as q decreases, which makes sense since a smaller q implies a larger fluctuation amplitude of $Q^{(n)}_{max}$. Moreover, when q is very small, particularly below the range of 0.4–0.6, Q_{max} shoots up quickly. Also, when q is beyond the range of 0.4–0.6, Q_{max} drops slowly as q increases.

C. Transient-State Analysis and Evaluation

An equilibrium state can be broken by either the variation of τ due to the change of bottleneck location, or the change of available bandwidth due to the variation of the cross traffic or the number of active VCs. The transient state can be caused by the variation of τ in two different cases: (I) $\alpha_0 > \alpha^h_{goal}$, the rate convergence is underdamped, and (II) $\alpha_0 < \alpha^l_{goal}$, the rate convergence is overdamped, where α^h_{goal} and α^l_{goal} are functions of Q_{goal}, p, q, τ, and μ. Denote the rate-gain parameter at the beginning of transient state by α_0. Let the new bottleneck's target rate-gain parameter be $\widetilde{\alpha}_{goal}$ which corresponds to the new bottleneck path's RM-cell RTT $\widetilde{\tau}$ and target bandwidth $\widetilde{\mu}$. The following theorem gives a formula to calculate the number of transient cycles.

Theorem 2: If the initial rate-control parameter $\alpha = \alpha_0$, the new RM-cell RTT $\tau = \widetilde{\tau}$, and new target bandwidth $\mu = \widetilde{\mu}$, then the number of transient cycles, N, is determined by

$$N = \begin{cases} \lceil \frac{log\ [\frac{\widetilde{\alpha}_{goal}}{\alpha_0}]}{log\ q} \rceil; & \text{if } \alpha_0 > \widetilde{\alpha}_{goal} \\ \lceil \frac{\widetilde{\alpha}_{goal} - \alpha_0}{p} \rceil; & \text{if } \alpha_0 \leq \widetilde{\alpha}_{goal} \end{cases} \qquad (16)$$

where $\widetilde{\alpha}_{goal}$ is the non-negative real root of non-linear equation:

$$\frac{\widetilde{\alpha}_{goal}\widetilde{\rho}^2}{2} + \widetilde{\mu}\widetilde{\rho} + \frac{\widetilde{\mu}^2}{\widetilde{\alpha}_{goal}}\ log\ \frac{\widetilde{\mu}}{\widetilde{\mu} + \widetilde{\alpha}_{goal}\widetilde{\rho}} - Q_{goal} = 0$$

where $\widetilde{\rho} = \widetilde{\tau} + \sqrt{\frac{2Q_h}{\widetilde{\alpha}_{goal}}}$, and can be approximated by $\widetilde{\alpha}_{goal} \approx \left(\frac{\sqrt{Q_{goal}} - \sqrt{2Q_h}}{\widetilde{\tau}}\right)^2$, if Q_{goal} is small.

Proof: The proof is provided in [6]. □

We derived peak source rate $R^{(i)}_{peak}$, bottleneck queue length $Q^{(i)}_{peak}$, and transient-state cycle $T^{(i)}$ with initial rate $R_0 > 0$ for the i-th transient cycle, $1 \leq i \leq N$. We omit these expressions here due to space limitation, and the interested readers are referred to [6] for details. Using the analytical results, we run numerical solutions to evaluate transient-state performance. Assume the same flow-control parameter settings as in the equilibrium-state analysis, except that $C_{max} = 700$ cells and $Q_{goal} = \frac{1}{2}C_{max} = 350$ cells, and α_0 is specified by $\mu_0 = 367$ cells/ms and $\tau_0 = 2$ ms. For worst case study, we let the initial $\tau_0 = \tau_{min} \stackrel{\triangle}{=} \min_{i \in \{1,2,\cdots,n\}}\{\tau_i\}$ and $\widetilde{\tau} = \tau_{max} \stackrel{\triangle}{=} \max_{i \in \{1,2,\cdots,n\}}\{\tau_i\}$ of a multicast VC with n paths. Also, assume $\widetilde{\mu} = 267$ cells/ms. Fig. 7 plots N given by Eq. (16), vs. $(\tau_{max} - \tau_{min})$ for different values of q. N is found to increase stepwise monotonically with $(\tau_{max} - \tau_{min})$. This is expected since a large variation in RM-cell RTT requires more transient cycles to converge to the new optimal equilibrium state. A smaller q results in a fewer number of transient cycles. Thus, q measures the speed of convergence. In

Fig. 8, Q_{peak} is observed to shoot up quickly with $(\tau_{max} - \tau_{min})$, further justifying the necessity of α-control, and a larger target buffer occupancy is found to result in a faster increase of Q_{peak}.

VI. Multiple Multicast Connections

A. Analytical Results

M (> 1) concurrent flow-controlled connections with a common multicast-tree bottleneck are modeled by a single buffer and a server shared by M source rates $R_i(t)$. So, the bottleneck's buffer queue function at time t is

$$Q(t) = \int_{t_0}^{t} \left\{ \sum_{i=1}^{M} R_i(v - T_f^{\langle i \rangle}) - \mu \right\} dv + Q(t_0) \tag{17}$$

where $T_f^{\langle i \rangle}$ is the forward delay for the i-th connection. The rate control function remains the same as in Eq. (2), but with different rate parameters for the respective connections. Applying the derived analytical results to some simple multiple connection cases, we have already shown in [6] that our proposed scheme based on α-control is stable and efficient, and outperforms the schemes without α-control in dealing with RM-cell RTT and bandwidth variations, and achieving fairness in both buffer and bandwidth occupancies. For lack of space, we omit the analytical evaluations in this paper and refer the interested readers to [6] for more details. Instead, in the next section we present the simulation results to (1) verify the obtained analytical results; (2) analyze the performance of our proposed scheme for more general cases where the locations, the number, and the bandwidth of multicast-tree bottlenecks vary with time.

B. Simulation Results

We carried out extensive simulations for the scenarios of concurrent multiple multicast VCs with multiple bottlenecks to study the performance of the proposed scheme with α-control, and compare it with schemes without α-control. By removing the assumptions made for the analytical analysis, the simulation results accurately capture the dynamics of real networks, such as the noise-effect of RM-cell RTT due to the randomness of network environments, and RM-cell processing and queuing delays, instantaneous variations of bottleneck bandwidths, which are very difficult to deal with analytically.

The simulated network is shown in Fig. 9, which consists of 3 multicast VCs running through 4 switches $SW_1, SW_2, \cdots, SW_4$ connected by 3 links L_1, L_2, L_3. S_i is the source of VC$_i$, $i = 1, 2, 3$, and R_{ij} is S_i's j-th receiver. So, VC$_2$ and VC$_3$ share L_1 and L_3, respectively, with VC$_1$. S_1 is a persistent ABR source which generates the main data traffic flow. S_2 and S_3 are two periodic on-off ABR sources with on-period $= 360$ ms and off-period $= 1011$ ms, respectively, which mimic cross-traffic noises, causing the bandwidth to vary dynamically at the bottlenecks. We set L_i's bandwidth capacity μ_i to (1) $\mu_1 = \mu_3 = 155.52$ Mbps; (2) μ_2 = 300 Mbps, forcing the potential bottlenecks L_1 and L_3 to show up. Letting all links' delays be 1 ms, S_1's RM-cell RTTs via

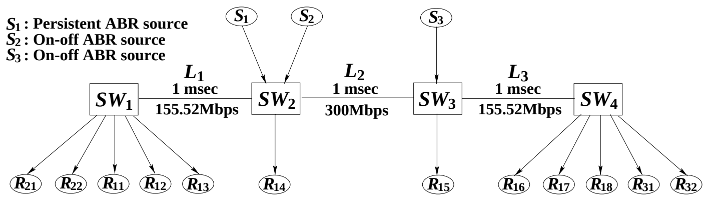

Fig. 9. Simulation model for multiple multicast VCs.

R_{16}, R_{17}, R_{18} equal 4 ms which is 2 times of S_1's RM-cell RTTs via R_{11}, R_{12}, R_{13}.

We implemented the simulation model by using the NetSim event-driven simulator. The flow-control parameters used in the simulation remain the same as those used in the analytical solutions for comparison purposes. Specifically, $Q_h = 50$ cells, $Q_{goal} = 400$ cells, $\Delta = 0.4$ ms, $q = 0.6$, $p = 16.67$ cells/ms^2, and R_0 = 30 cells/ms; VC$_1$'s $\alpha_0 = 57.8$ cells/ms^2, VC$_2$ and VC$_3$'s $\alpha_0 = 22.9$ cells/ms^2. We let S_1 start at $t = 0$, S_2 at $t = 160$ ms, and S_3 at $t = 822$ ms such that S_2 and S_3 generate the cross-traffic noises against the main data traffic flow at the potential bottlenecks L_1 and L_3 with the respective on-periods appearing alternately without overlapping in time. Consequently, as shown in Figs. 10–19, the first two on-periods of VC$_2$ and VC$_3$ divide the first 1178 ms simulation time axis into the following 4 time periods (ms). $T_1 = [0, 160]$ where only VC$_1$ is active; $T_2 = [160, 520]$ where both VC$_1$ and VC$_2$ are active; $T_3 = [520, 822]$ where only VC$_1$ is active; $T_4 = [822, 1178]$ where both VC$_1$ and VC$_3$ are active. The simulation results for the two different schemes are summarized in Figs. 10–19, where all results with α-control are depicted in Figs. 10–14 on the left, while those without α-control are shown in Figs. 15–19 on the right. Each individual performance measure with α-control is compared with its counterpart without α-control listed in the same row.

(1) During T_1. For the α-controlled scheme, Fig. 10 shows that VC_1's rate $R_1(t)$ converges to L_1 and L_3's capacity 367 cells/ms (155.52 Mbps) since VC$_1$ is the only active VC and it grabs all the bandwidth available. Thus, during T_1, there exist 2 bottlenecks located at L_1 and L_3 with RTT equal to 2 ms and 4 ms, respectively. Denote these two bottlenecks' total queue lengths at SW_2 and SW_3 by $Q_2(t)$ and $Q_3(t)$ and their maximum by $Q_{max}^{\langle 2 \rangle}$ and $Q_{max}^{\langle 3 \rangle}$. From Figs. 10–12 we observe that experiencing one transient cycle due to $Q_{max}^{\langle 2 \rangle} = Q_{max}^{\langle 3 \rangle} = 560 > Q_{goal}$, $Q_{max}^{\langle 2 \rangle}$ and $Q_{max}^{\langle 3 \rangle}$ converge to Q_{goal}'s neighborhood $[350, 446]$ by the α-control. So, the α-control not only drives $R_1(t)$ to its target bandwidth, but also confines the maximum queue lengths at the bottlenecks to Q_{goal}'s neighborhood. In contrast, for the schemes without α-control, Figs. 15–17 show that $R_1(t)$ converges to $\mu_1 = \mu_3 = 367$, but $Q_{max}^{\langle 2 \rangle} = Q_{max}^{\langle 3 \rangle} = 560$ and never go down to $Q_{goal} = 400$.

(2) During T_2. VC$_2$ starts transmission, and competes for bandwidth and buffer space with VC$_1$. The bottleneck at L_3 is expected to disappear since $R_1(t)$'s new target bandwidth along path via L_1 is only a half of that via L_3. So, L_1 is the only one bottleneck with RTT = 2 ms, target bandwidth = $\frac{1}{2}\mu_1$ for each of VC$_1$ and VC$_2$. For the α-controlled scheme, Fig. 10 shows that the source rates $R_1(t)$ and $R_2(t)$ experience two transient cycles during which

$R_1(t)$ gives up $\frac{1}{2}\mu_1$ to $R_2(t)$ until they reach the new equilibrium. Fig. 11 shows that a big queue build-up $Q^{\langle 2\rangle}_{max} = 704$ as a result of the superposed rate-gain parameter from $R_1(t)$ and $R_2(t)$, and the reduced bottleneck bandwidth. By α-control, $Q^{\langle 2\rangle}_{max}$ is driven down to Q_{goal}'s neighborhood of $[385, 468]$. Fig. 12 shows $Q_3(t) = 0$, verifying that the bottleneck at L_3 vanished. Fig. 13 is a zoom-in picture of $Q_2(t) = Q_{21}(t) + Q_{22}(t)$ of Fig. 11, where $Q_{21}(t)$ is the per-VC queue of VC$_1$ and $Q_{22}(t)$ is the per-VC queue of VC$_2$ at SW_2, respectively. Fig. 13 indicates that in the first transient cycle, $Q_{21}(t)$'s maximum $Q^{\langle 21\rangle}_{max} = 528$, which is more than 3 times of $Q_{22}(t)$'s maximum $Q^{\langle 22\rangle}_{max} = 175$. Under α-control, $Q_{21}(t)$ and $Q_{22}(t)$ converge to each other quickly and become identical from $t = 391$ ms. This verifies that the α-control law can ensure the fairness in buffer occupancy between the competing VCs. By contrast, for the scheme without α-control, Fig. 16 illustrates that $Q^{\langle 2\rangle}_{max}$ jumps up to as high as 900 and stays at 900 even after the transient state. Fig. 18, the zoom-in picture of Fig. 16, shows that $Q_{21}(t)$ never converges to $Q_{22}(t)$ even after the transient state, and thus the buffer space is not fairly occupied.

(3) During T_3. After VC$_2$ goes into an off-period, $R_1(t)$ grabs all the bandwidth of μ_1 again. After $R_1(t)$ reaches the L_1's bandwidth capacity, the bottleneck at L_3 also shows up due to $\mu_1 = \mu_3$, and then the total number of bottlenecks becomes 2 again. For the scheme with α-control, because $Q_{22}(t)$ suddenly drops to zero as VC$_2$ goes into an off-period, making $Q^{\langle 2\rangle}_{max} \ll Q_{goal}$, which generates 3 consecutive $BCI = 0$, the α-control's additive-increase operation $\alpha_n = \alpha_{n-1} + p$ is executed twice during the transient cycles until $Q^{\langle 2\rangle}_{max}$ converges to Q_{goal}'s neighborhood $[367, 483]$ within 3 transient cycles. Note that $Q^{\langle 2\rangle}_{max}$ *monotonically* converges to $[367, 483]$ as shown in Fig. 11. This is expected since $p = 16.67 \le \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}} - \sqrt{2Q_h}}{\tau}\right)^2$, satisfying the condition (3) in *Theorem 1*. This observation further verifies the correctness of the optimal monotonic convergency condition derived in *Theorem 1*. In Figs. 15–16 for schemes without α-control, the queue and rate dynamics simply repeat their dynamics in T_1, suffering from a large buffer requirement.

(4) During T_4. The rate and queue dynamics are similar to T_2's, except that the new bottleneck is now located at L_3 with a new target bandwidth = $\frac{1}{2}\mu_3$ and a longer RTT = 4 ms. For the α-controlled scheme, Fig. 11 shows $Q_2(t) = 0$, indicating that the bottleneck at L_1 disappeared and L_3 is the only bottleneck. Fig. 12 shows that $Q^{\langle 3\rangle}_{max}$ shoots up to 928, as a result of the doubled RTT (4 ms) via L_3. Within 3 transient cycles, $Q^{\langle 3\rangle}_{max}$ converges to Q_{goal}'s neighborhood of $[367, 445]$ in equilibrium state. Fig. 14, the zoom-in picture of Fig. 12, shows the buffer-occupancy fairness ensured by α-control. These observations verify that α-control can efficiently adapt to RM-cell RTT variations in terms of buffer requirement and fairness. By contrast, in the scheme without α-control, Figs. 16 and 17 show 2 bottlenecks: (1) a bandwidth-congestion bottleneck at L_1; (2) a buffer-congestion bottleneck at L_3. Fig. 17 shows that $Q^{\langle 3\rangle}_{max} = 1740$, almost 2 times of that under the α-controlled scheme. More importantly, $Q^{\langle 3\rangle}_{max}$ stays that high (around 1740) even after the transient state. Moreover, Fig. 19, the zoom-in picture of Fig. 17, demonstrates that buffer occupancy is not fair as $Q^{\langle 31\rangle}_{max} = 1000$, but $Q^{\langle 33\rangle}_{max} = 740$.

scheme type	$\bar{R}_1$ of VC$_1$	$\bar{R}_2$ of VC$_2$	$\bar{R}_3$ of VC$_3$
with α-control	234.448	150.671	147.709
without α-control	209.367	143.672	137.655

TABLE I
Average throughputs (cells/ms) of schemes with and without α-control.

The 3 VCs' average throughputs (for on-off sources averaging over the on-period only) obtained by the simulation are compared between the two types of schemes in Table I. We observe that in all the 3 VC cases the proposed scheme with α-control outperforms the scheme without α-control in terms of average throughput.

VII. Conclusion

We proposed and analyzed a flow-control scheme for ATM ABR multicast services, which scales well and is efficient in dealing with the variations in the multicast-tree structure and RM-cell RTT. We developed the α-control, the second-order rate control, algorithm to handle the variation of RM-cell RTT. Analytical results show our scheme based on α-control to be stable and efficient in that both the source rate and bottleneck queue length rapidly converge to a small neighborhood of the designated operating point. The simulation experiments verify the derived analytical results, and demonstrate the superiority of the proposed scheme to the other schemes in dealing with the variations of RM-cell RTT and link bandwidth, and achieving fairness in both buffer and bandwidth occupancies.

References

[1] L. Roberts, *Rate Based Algorithm for Point to Multipoint ABR Service*, ATM Forum contribution 94-0772, September 1994.

[2] K.-Y. Siu and H.-Y. Tzeng, "On max-min fair congestion control for multicast ABR services in ATM," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 545–556, April 1997.

[3] Y.-Z. Cho and M.-Y. Lee, "An efficient rate-based algorithm for point-to-multipoint ABR service," in *Proc. of GLOBECOM*, November 1997.

[4] W. Ren, K.-Y. Siu, and H. Suzuki, "On the performance of congestion control algorithms fo multicast ABR in ATM," *Proc. of IEEE ATM WORKSHOP*, August 1996.

[5] S. Fahmy, R. Jain, R. Goyal, B. Vandalor, and S. Kalyanaraman, "Feedbackback consolidation algorithms for ABR point-to-mulipoint connections in ATM networks," in *Proc. of IEEE INFOCOM*, April 1998.

[6] X. Zhang, K. G. Shin, D. Saha, and D. Kandlur, "A scalable flow control scheme for multicast ABR service in ATM networks," *submitted to IEEE/ACM Trans. on Networking for publication*, URL http://www.eecs.umich.edu/∼xizhang/papers/mcast.ps, December 1998.

[7] S. Sathaye, *ATM Forum traffic management specifications Version 4.0*, ATM Forum contribution 95-0013R7.1, August 1995.

[8] N. Yin and M. G. Hluchyj, "On closed-loop rate control for ATM cell relay networks," in *Proc. of IEEE INFOCOM*, pp. 99–109, June 1994.

[9] H. Ohsaki, M. Murata, H. Suzuki, C. Ikeda, and H. Miyahara, "Analysis of rate-based congestion control for ATM networks," *ACM SIGCOMM Computer Communication Review*, vol. 25, pp. 60–72, April 1995.

[10] F. Bonomi, D. Mitra, and J. Seery, "Adaptive algorithms for feedback-based flow control in high-speed, wide-area ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, pp. 1267–1283, Sept. 1995.

[11] X. Zhang, K. G. Shin, and Q. Zheng, "Integrated rate and credit feedback control for ABR services in ATM networks," in *Proc. of IEEE INFOCOM*, pp. 1297–1305, April 1997.

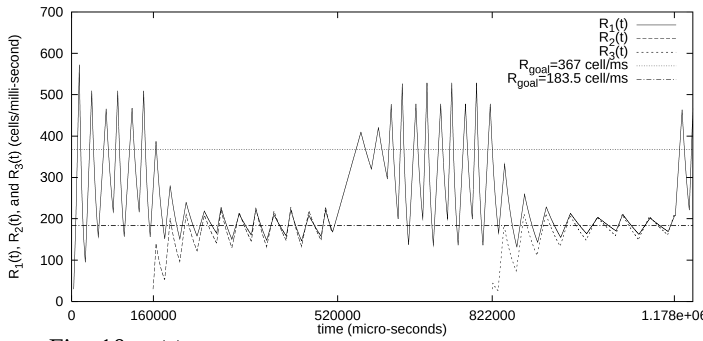

Fig. 10. $R(t)$'s converge to bottleneck bandwidth with α-control

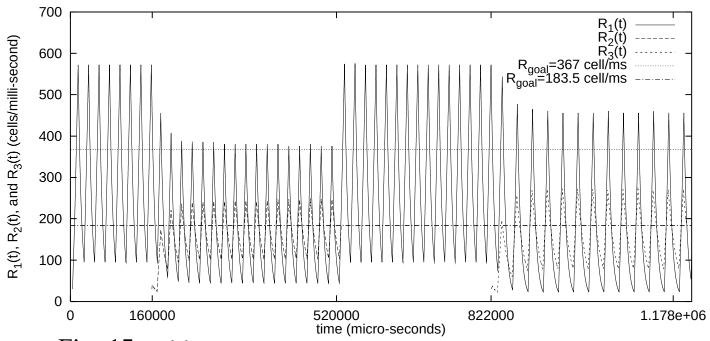

Fig. 15. $R(t)$'s converge to bottleneck bandwidth without α-control

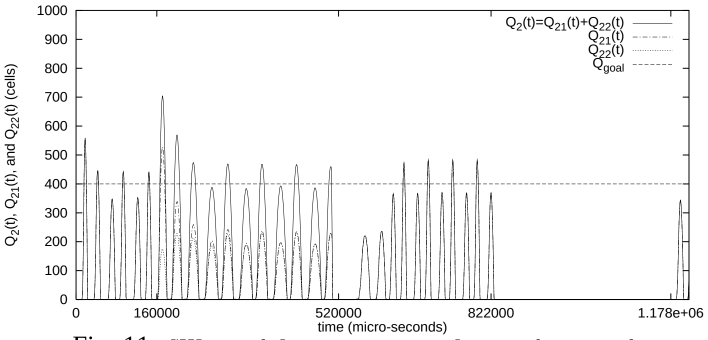

Fig. 11. SW_2: Total Q_{max} converges to Q_{goal} with α-control

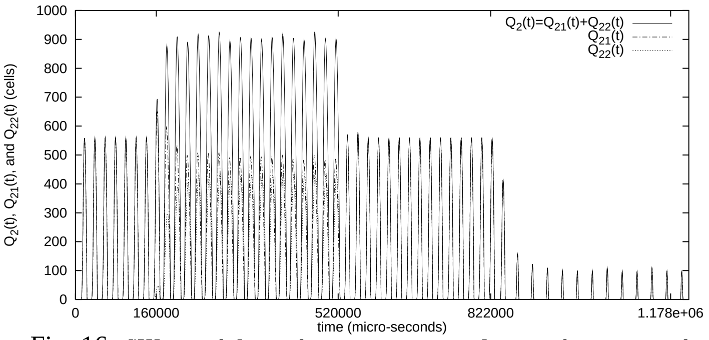

Fig. 16. SW_2: Total Q_{max} does not converge to Q_{goal} without α-control

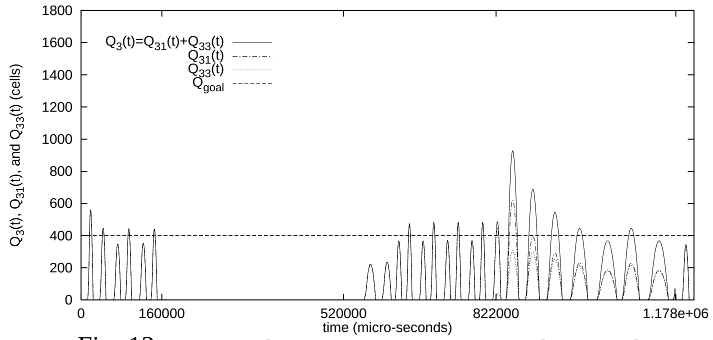

Fig. 12. SW_3: Total Q_{max} converges to Q_{goal} with α-control

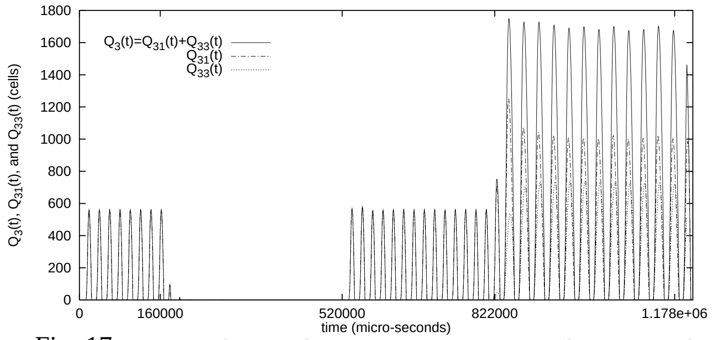

Fig. 17. SW_3: Total Q_{max} does not converge to Q_{goal} without α-control

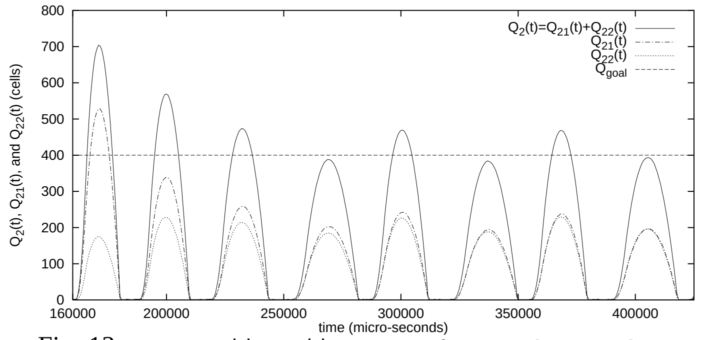

Fig. 13. SW_2: $Q_{21}(t)$, $Q_{22}(t)$ converge to fairness with α-control

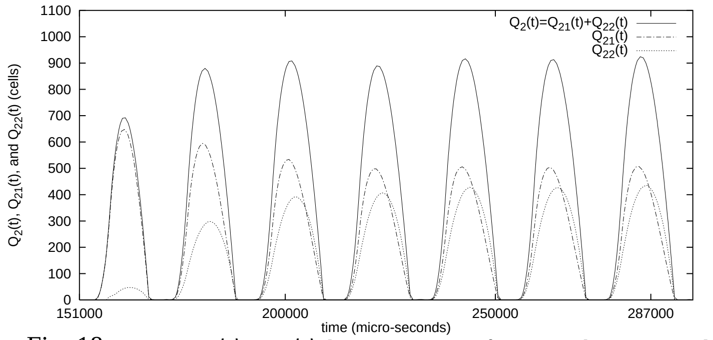

Fig. 18. SW_2: $Q_{21}(t)$, $Q_{22}(t)$ do not converge to fairness without α-control

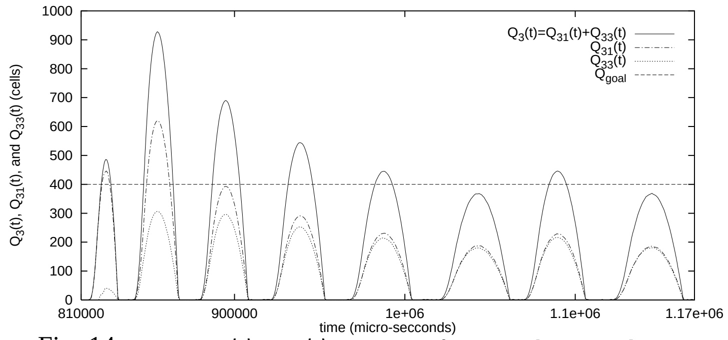

Fig. 14. SW_3: $Q_{31}(t)$, $Q_{33}(t)$ converge to fairness with α-control

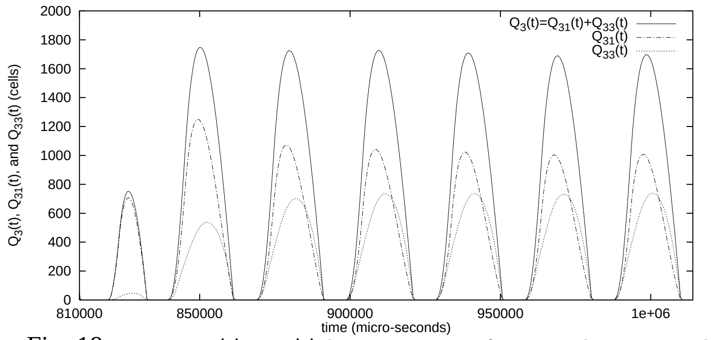

Fig. 19. SW_3: $Q_{31}(t)$, $Q_{33}(t)$ do not converge to fairness without α-control