Scalable Flow Control for Multicast ABR Services in ATM Networks

Xi Zhang, *Senior Member, IEEE*, Kang G. Shin, *Fellow, IEEE*, Debanjan Saha, and Dilip D. Kandlur, *Member, IEEE*

***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 sensed by the traffic source. It also confines the buffer occupancy to a target regime bounded by a finite buffer capacity as the system enters the equilibrium state. It works well irrespective of the topology of the multicast tree. Using the fluid analysis, 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 experiments also demonstrate the superiority of the proposed scheme to the other schemes in dealing with RM-cell RTT and link-bandwidth variations, achieving fairness in both buffer and bandwidth occupancies, and enhancing average throughput.**

***Index Terms*—α-control, ABR, ATM, buffer control, feedback-soft synchronization (SSP), flow control, multicast, multicast flow control, RTT variations, scalability, second-order rate control, target buffer occupancy.**

I. INTRODUCTION

AN ABR flow-control algorithm consists of two components: determining the bottleneck link bandwidth, and adjusting the source transmission rate to match the bottleneck link bandwidth and buffer capacity. In a multicast ABR connection, determining the bottleneck link bandwidth is a daunting task. (Note that, strictly speaking, multicast includes point-to-multipoint, multipoint-to-point, and multipoint-to-multipoint transmissions. However, for the convenience of presentation, in this paper we use the narrow-sense definition for multicast which stands for the point-to-multipoint transmission.) The first generation of multicast ABR algorithms [1]–[3] employ a simple hop-by-hop feedback mechanism for this purpose. In these algorithms, feedback Resource Management (RM) cells from downstream nodes are consolidated at 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. Moreover, unless the feedback RM cells from the downstream nodes are *synchronized* at each branch point, the source may be misled by the incomplete feedback information, which can cause the *consolidation noise* problem [4], [5].

To reduce the RM-cell RTT and eliminate consolidation noise, the authors of [5] and [6] proposed feedback synchronization at each branch point by accumulating feedback from *all* downstream 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 [7] 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 nonresponsive branches from the feedback synchronization process. One or more nonresponsive 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 nodes 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 [8] while providing efficient feedback synchronization, but also simplifies the implementation of detection and removal of nonresponsive branches. A scheme similar in spirit but different in terms of implementation was proposed independently in [5], [6].

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,

Manuscript received March 5, 1999; revised August 2, 1999, April 18, 2000, and October 29, 2000; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor R. Rom. This work was supported in part by the U.S. Office of Naval Research under Grant N00014-99-1-0465. An earlier version of this paper was presented in part at the IEEE INFOCOM'99, New York, NY.

X. Zhang was with the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: xizhang@eecs.umich.edu). He is now with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843-3259 USA (e-mail: xizhang@ee.tamu.edu).

K. G. Shin is with the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: kgshin@eecs.umich.edu).

D. Saha is with Tellium, Inc., Oceanport, NJ 07757-0901 USA (e-mail: dsaha@tellium.com).

D. D. Kandlur is with the Networking Software and Services Department, IBM T.J. Watson Research Center, Hawthorne, NY 10532 USA (e-mail: kandlur@watson.ibm.com).

Publisher Item Identifier S 1063-6692(02)01081-6.

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. 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, the proposed rate control scheme not only regulates the traffic source rate based on the congestion feedback, but also adjusts the rate-gain parameter α, which is the speed of rate increase. As will be discussed later, the maximum queue-size is an increasing function of both the RM-cell RTT and the rate-gain parameter α, and the α-control can make the flow-control performance dynamically adaptive to RM-cell RTT variations. Using the fluid analysis, we model the α-control with the binary-congestion feedback, and study the system dynamics in the scenarios of both persistent and on-off ABR traffic sources. 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 bottleneck queue length rapidly converge to a small neighborhood of the designated operating point. The α-control is also shown to adapt well to RM-cell RTT variations in terms of buffer requirements and fairness.[1] The simulation experiments also verify the analytical results and the superiority of the proposed scheme to the other schemes in RTT and link-bandwidth adaptiveness, fairness in both buffer and bandwidth usage, and average throughput.

The paper is organized as follows. Section II describes the proposed scheme. Section III establishes the flow-control system model. Section IV justifies the necessity and feasibility of the α-control, presents the α-control algorithm, and investigates its properties. Section V derives analytical expressions for both transient and equilibrium states, evaluates the scheme's performance for the single-connection case, and compare the analysis and simulation results. Section VI analyzes the flow-control performance of concurrent multiple multicast-connections, and compares the proposed scheme with the other existing schemes. The paper concludes with Section VII.

[1]The definition of fairness used throughout this paper is adopted from [9] where the fairness is achieved when all connections receive an equal share/allocation of the network resources (bandwidth or buffer capacities). This differs from the max-min fairness, which deals with more general cases where some connections' demand is smaller than an equal share/allocation of the network resources.

II. The Proposed Scheme

Based on the ABR flow-control framework in [10], we use RM cells to convey network-congestion information. A forward RM cell is sent by the root (source) node periodically or once every $N_{\rm rm}$ data-cells, and each receiver node replies by returning to the source a feedback RM cell with Congestion Indication (CI) and Explicit Rate (ER) information. We redefine the RM-cell format by adding information on the rate-gain parameter (second-order) control in the standard RM cell to deal with RM-cell RTT variations. In particular, two new one-bit fields, Buffer Congestion Indication (BCI) and New Maximum Queue (NMQ), are defined. Our scheme distinguishes the following two types of congestion.

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

Buffer Congestion: If the maximum queue length $Q_{\max}$ at a switch exceeds the target buffer occupancy $Q_{\rm goal}$, where $2Q_h < Q_{\rm goal} < C_{\max}$ [11] and $C_{\max}$ is the buffer capacity, then the switch sets the local BCI to 1.

A. The Source Algorithm

Fig. 8 in Appendix A shows the pseudocode for the source algorithm. Upon receiving a feedback RM cell, the source first checks 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 local congestion indicator (LCI) equals 1 while the CI bit in the received RM cell is 0. The rate-gain parameter is adjusted according to the current value of the local BCI (LBCI) and the BCI bit in the just received RM cell. This leads to three cases: 1) if BCI is 1 in the RM cell received, the rate-gain parameter Additive Increase Rate (AIR) is decreased multiplicatively by a factor of q $(0 < q < 1)$; 2) if both LBCI and BCI are 0, AIR is increased additively by a step of size $p > 0$; 3) if $\rm LBCI = 1$ and $\rm BCI = 0$, AIR is increased multiplicatively by the same factor of q. In all the three cases, the rate-decrease parameter Multiplicative Decrease Factor (MDF) is adjusted based on 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 Allowed Cell Rate (ACR) based on the received CI-bit. Fig. 3 in Section V shows the equilibrium dynamics of the source rate $R(t)$ (ACR) and the bottleneck queue length $Q(t)$, using the fluid functions (see Section III). Driven by feedback CI-bit, $R(t)$ fluctuates around the bottleneck bandwidth, but alternates between two different ramp-up speeds determined by the feedback BCI-bit. Consequently, the maximum queue length $Q_{\max}^{(n)}$ at the bottleneck is confined to the designated operating regime around $Q_{\rm goal}$.

B. The Switch Algorithm

At the center of switch control algorithm is a pair of connection-update vectors: 1) $conn\_patt\_vec$, the connection pattern vector where $conn\_patt\_vec(i) = 0\ (1)$ indicates the ith 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 ith downstream branch and a feedback RM cell is (not) expected from the ith downstream branch;[2] 2) $resp\_branch\_vec$, 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 a feedback RM cell is received from the ith downstream branch. The connection pattern of $conn\_patt\_vec$ is updated by $resp\_branch\_vec$ each time when the nonresponsive branch is detected or a new connection request is received from a downstream branch.

Fig. 9 of Appendix A gives the pseudocode of switch algorithm. Upon receiving a data cell, the switch multicasts it to its output ports specified by $conn\_patt\_vec$, if the corresponding output links are available, else enqueues it in its branch's queue. Mark the branch's CI (EFCI) if $Q(t) > Q_h$. Update $Q_{\max}$ for α-control (see Section IV.A) if the branch's new $Q(t)$ exceeds the old $Q_{\max}$. BCI $:= 1$ if its updated $Q_{\max} \geq Q_{\text{goal}}$. Receiving a feedback RM cell from either one of receivers or a connected downstream branch, the switch first marks its corresponding bit in $resp\_branch\_vec$ and then performs the RM-cell consolidation. If the modulo-2 addition (the soft-sychcronization operation of SSP), $conn\_patt\_vec \oplus resp\_branch\_vec = \underline{1}$, an all 1's vector, implying all feedback RM cells synchronized, then a fully-consolidated feedback RM cell is generated and sent upward. But, if the modulo-2 addition $\neq \underline{1}$, the switch awaits other feedback RM-cells for synchronization. Since the consolidated RM-cell is not required to be derived only from those 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, reset $Q_{\max} := 0$ and the buffer congestion indicator $\mathrm{MBCI} := 0$ if an NMQ request is received. The nonresponsive timer no_resp_timer, initialized to a threshold N_{nrt}, is reset to N_{nrt} if a consolidated RM-cell is sent upward. The predetermined timeout value N_{nrt} for nonresponsiveness is determined by 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 nonresponsive branches (so, it still works even if there are faults in downstream branches). If a switch receives a forward RM-cell, the multicast connection's no_resp_timer reduces by one. If $no\_resp\_timer = 0$ (timeout) and $resp\_branch\_vec \neq \underline{0}$ (i.e., there is at least one downstream responsive branch), then the switch immediately sends a partially-consolidated RM-cell upward without further awaiting feedback RM-cells. If $no\_resp\_timer = 0$, at least one nonresponsive downstream branch is detected, and is removed by the simple operation: $conn\_patt\_vec := resp\_branch\_vec \oplus \underline{1}$. The downstream branch can join the multicast tree at run-time.

C. Multicast Flow-Control Signaling and Scalability

The multicast flow-control algorithms proposed above consist of two basic components: flow-control signaling and rate control. These two components are conceptually separate from a flow-control theory viewpoint, even though they are blended together in the proposed algorithms. The flow-control signaling relies on RM cells, which deliver rate-control and congestion information between the soruce-rate controller and the network/receivers. For multicast ABR, scalability is crucial since the flow-control traffic due to RM cells and feedback delay may increase with the number of receivers. We propose SSP [8] for flow-control signaling, which scales well with the multicast session size, thanks to the following two properties: 1) the feedback delay is virtually independent of the multicast session size, and 2) the ratio of feedback RM cells to forward RM cells at each link of the multicast session is no larger than 1[4], [8].

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

III. The System Model

The proposed scheme can support both 1) CI-based rate control with a binary congestion feedback (CI-bit), and 2) ER-based rate-control with an explicit-rate feedback (ER-value). The CI-based scheme is more suitable for LANs because of its minimal multicast signaling cost and lowest implementation complexity. As compared to the CI-based scheme, the ER-based scheme is more responsive to network congestion and can better serve WAN environments where the bandwidth-delay product is large. However, the ER-based scheme is much more expensive to implement than the CI-based scheme. In this paper, we will focus only on the CI-based scheme, and the rate control and the α-control to be discussed will be only for the CI-based (not ER-based)[3] scheme. We model the CI-based flow-control system by the first-order fluid analysis [12]–[17], which uses the continuous-time functions $R(t)$ and $Q(t)$ as the fluid approximation of the source rate and bottleneck queue length, respectively. We also assume the existence of only a single bottleneck[4] on each path at a time with queue length equal to $Q(t)$ and a "persistent" source with $\mathrm{ACR} = R(t)$ for each multicast connection.

A. System Description

As shown in Fig. 1, a multicast-connection model consists of n paths with RM-cell RTT's τ_i and bottleneck bandwidths μ_i for $1 \leq i \leq n$. There is only one bottleneck on each path where $T_f^{(i)}$ is the "forward" delay from the source to the bottleneck, $T_b^{(i)} = (\tau_i - T_f^{(i)})$ the "backward" delay from the bottleneck to the source via the receiver, and $Q_i(t)$ the bottleneck queue length. We use the synchronous model by assuming that the source sends RM cells periodically with an interval Δ equal to a *fraction* of RTT. The source rate-control algorithm during the nth rate update interval can be expressed as

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

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

[3]The ER-based scheme is worth, and will be reported in, a separate paper.

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

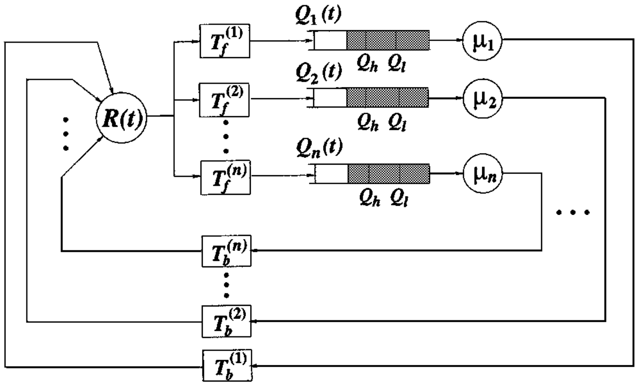


Fig. 1. System model for a multicast connection with n paths.

B. System Control Factors

In unicast ABR service, the source rate is regulated by the feedback from the most congested link/switch which has the minimum available bandwidth 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 available bandwidth share of the multicast-tree's most congested path that the traffic source has sensed. This is the key feature of ABR service, most suitable for data applications that require lossless transmission. However, the dynamics of multicast ABR flow control is more complicated than those of unicast ABR flow control, because not only the available bandwidth, but also the RTT and congestion threshold can differ from one path to another within a multicast tree. As a result, while the source rate always converges to the available bandwidth of the slowest path *perceived by the traffic source* (which is not necessarily the currently slowest path in the multicast tree), it is possible that in the transient state the dynamics of source rate is dictated by the feedback via the path with a bandwidth larger than the current minimum available bandwidth across the multicast-tree, depending on the path's RTT and congestion threshold. To explicitly model these features for the multicast flow control, we introduce the following definition.

Definition 1: The **multicast-tree bottleneck path** (also called **multicast-tree bottleneck**) is the path whose congestion feedback **currently received at the source** dictates the source rate control. The **multicast-tree RM-cell RTT** is the RM-cell RTT experienced on the multicast-tree bottleneck path. ■

Remarks on Definition 1:

R1. The multicast-tree bottleneck path is a *source flow control oriented* notion because only the congestion feedback *currently received by the source* can affect the *current* source flow control. The current congestion information detected at switches does not affect the source's flow control until it reaches the source after a certain delay. So, it is the congestion feedback currently received/perceived by the source, instead of the congestion information currently detected at the switches, that decides which path is the multicast-tree bottleneck at the current moment. Thus, at a given time instant the multicast-tree bottleneck path is ***not*** necessarily always the slowest path (with the minimum available bandwidth) in the multicast tree.

R2. The multicast-tree bottleneck can be formed during the following two different types of phases, depending on feedback CI-bit in the most recently source-received RM cell:

a) **Congested phase:** where $\mathrm{CI} = 1$ consolidated from m paths with $\mathrm{CI}(j) = 1$ for $1 \leq j \leq m \leq n$. The *shortest* path (with the smallest RTT) of the m congested paths is the multicast-tree bottleneck, because it determines the RTT of multicast-tree's feedback control loop and the dynamics of the multicast-tree bottleneck.

b) **Non-congested phase:** where $\mathrm{CI} = 0$ consolidated from all paths. The shortest path of these, which will cause congestion, immediately after this noncongested phase, is the multicast-tree bottleneck due to the same reason as in the above congested phase.

R3. The multicast-tree bottleneck can change at any time instant (even within a rate-control cycle), but only at the one of the following two types of transition instants:

a) when the consolidated RM-cell's CI changes $1 \rightarrow 0$;

b) when the consolidated RM-cell's $\mathrm{CI} = 1$ remains unchanged, but $\mathrm{CI}(j)$ for the *shortest* of m congested paths changes $1 \rightarrow 0$ for $1 \leq j \leq m \leq n$; or a noncongested path P_k's $\mathrm{CI}(k)$ changes $0 \rightarrow 1$, where path P_k is shorter than all congested paths for $k \neq j$.

Thus, the location of the multicast-tree bottleneck path is a function of the bottleneck-link bandwidth (μ_i), the queue threshold $(Q_h^{(i)})$ in the bottlenecked switch, and RTT (τ_i) on path P_i, for $i = 1, \ldots, n$.

R4. At any given time instant, there exists the only one multicast-tree bottleneck path, which is the *shortest* congested path sensed by the source through the most recent-feedback RM cell. This is because at any time moment there is only one the shortest path among the congested paths perceived by the source when the congested phase starts, unless there are multiple paths that have the same RTT and become congested at the same time. In that case, albeit not very often in practice, these paths have either the same rate control parameters (μ, Q_h, and τ) or an identical feedback effect on the source rate control, and thus any one of them can be chosen as the multicast-tree bottleneck. Hence, the uniqueness of the multicast-tree bottleneck in a multicast tree for any given time instant still holds.

C. The State Equations for the Multicast-Tree Bottleneck Path

Since the multicast-tree bottleneck dictates the source rate-control, 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 current multicast-tree bottleneck defined by Definition 1, respectively. Then, the multicast-tree bottleneck state is specified by two state variables: $R(t)$ and $Q(t)$. By the rate-control defined in (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 $\beta = 1 + \log b$ (a and b are defined in (1) and Δ is the source rate update interval); t and t_0 are the current and last observation times, respectively, of the system states for the current multicast-tree bottleneck path, and t is chosen such that, during the period of $(t - t_0)$, the multicast-tree bottleneck path is *fixed* and *unique*, and also, during $(t - t_0)$, $R(t)$ is only in either an increasing or a decreasing phase; $\tau = T_f + T_b$ is the current multicast-tree RM-cell RTT; Q_h (Q_l) is the high (low) queue-threshold for the current multicast-tree bottleneck; μ is the available bandwidth of the current multicast-tree bottleneck.

Remarks on the System State Equations (2) and (3): Fluid analysis is a time-period piece-wise modeling procedure [16]. So, we can use a set of system state equations (2) and (3) of the same form to model the dynamics of different multicast-tree bottleneck paths during the different time periods, by replacing the system state variables, such as $Q(t)$, $Q(t - T_b)$, T_b, and T_f for different time periods corresponding to different multicast-tree bottleneck paths. Consequently, the system state variables $Q(t)$, $Q(t - T_b)$, T_b, and T_f given in (2) and (3) are *not* constant because they may be associated with a *different* multicast-tree bottleneck path during a *different* time period of $(t - t_0)$, depending on which path is the multicast-tree bottleneck during that time period of $(t - t_0)$.

Even though the multicast-tree bottleneck can change during any time period, the multicast-tree bottleneck path perceived by the the traffic source is *unique* because the queue-length threshold testing, $Q(t - T_b) \geq Q_h$ or $Q(t - T_b) < Q_l$, is only sampled at the time *instants*[5] which are the integer multiples of Δ. This feature of the proposed multicast flow control algorithm ensures that fluid analysis expressed by (2) and (3) can accurately capture the dynamics of multicast-tree bottleneck path under the proposed multicast flow control algorithm even when the multicast tree bottleneck path changes from one path to another, as long as we take $(t - t_0) < \Delta$ or make $(t - t_0)$ small enough such that the bottleneck path that the traffic source can perceive is always *unique*[6] during $(t - t_0)$. As a result, the system state equations (2) and (3) characterize the multicast flow-control dynamics by modeling the flow-control dynamics of the different multicast-tree bottleneck paths, one path for each time-period of $(t - t_0)$ (piece-wise modeling in terms of time period), as the multicast-tree bottleneck changes from one path during a time period, to another path during the next time period.

[5]Only at these sampling time instants, the traffic source can perceive the possible change of multicast-tree bottleneck path, and between any two consecutive sampling time instants (i.e., the RM-cell update time interval Δ) the traffic source does not have a chance to sense any change of multicast-tree bottleneck path. So, the multicast-tree bottleneck path that the traffic source can perceive remains unchanged between any two consecutive sampling time instants.

[6]The uniqueness of the multicast tree bottleneck path, which can be perceived by the traffic source, can be always achieved either by letting $(t - t_0) < \Delta$, or otherwise (if $(t - t_0) > \Delta$) by letting $(t - t_0)$ be small enough such that multicast tree bottleneck path that the traffic source can perceive is unique during $(t - t_0)$.

IV. Adaptation to Variations of Multicast-Tree RM-Cell RTT

The cross-traffic at each link may cause the multicast-tree bottleneck path to shift from one path to another. So, the multicast-tree RM-cell RTT fluctuates dynamically between $\tau_{\min} \triangleq \min_{1 \leq i \leq n}\{\tau_i\}$ and $\tau_{\max} \triangleq \max_{1 \leq i \leq n}\{\tau_i\}$. The main and direct impact of RM-cell RTT variations is on the maximum buffer requirement for the bottleneck path.

A. Maximum Buffer Requirement and Cell-Loss Control

Although SSP makes the RM-cell RTT τ for the proposed scheme much smaller than that for the hop-by-hop scheme, as shown in [8], τ's swing between $\tau_{\min}$ and $\tau_{\max}$ is still large enough to make a significant impact on $Q_{\max}$. As discussed in [15], 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 the 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 $Q_{\max}$. So, $Q_{\max}$ also depends on the source rate-gain parameter α (to be detailed in Section V). $Q_{\max}$ is analytically shown in [15] to increase with both τ and rate-gain parameter $\alpha = dR(t)/dt$ and can be written as a function, $Q_{\max}(\alpha, \tau)$, or $Q_{\max}(\alpha)$ for a given τ. In reality, the buffer capacity, $C_{\max}$, on the bottleneck path is finite, and hence, to ensure cell-lossless transmission, the condition $Q_{\max} \leq C_{\max}$ must hold. This constraint divides the two-dimensional (α, τ)-space into two regions as follows.

Definition 2: If $C_{\max} < \infty$, then the **feasible** (α, τ)-**space**, $\Omega \triangleq \{(\alpha, \tau) \,|\, \alpha > 0, \tau > 0\}$ is partitioned into two parts: **lossless transmission region**: $\mathcal{F} \triangleq \{(\alpha, \tau) \,|\, (\alpha, \tau) \in \Omega, Q_{\max}(\alpha, \tau) \leq C_{\max}\}$ and **lossy transmission region**: $\mathcal{L} \triangleq \Omega \setminus \mathcal{F}$. ■

The theorem presented below finds an upper bound for the equilibrium-state maximum queue length $Q_{\max}(\alpha, \tau)$ as a function of $(\alpha, \tau) \in \Omega$ and Q_h.

Theorem 1: Consider a **multicast-tree bottleneck** characterized by the flow-control parameters α, β, τ, μ, Δ, and Q_h. If $(\alpha, \tau) \in \Omega$ and $\beta = 1 - (\alpha/\mu)\Delta$, then the maximum queue length is upper-bounded by

$$Q_{\max}(\alpha, \tau) \leq (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2. \tag{4}$$

Proof: The proof is given in Appendix B. ■

Remarks on Theorem 1: The derived upper-bound function of $Q_{\max}(\alpha, \tau)$ described in Theorem 1 provides a closed-form expression that reveals an analytical relationship among the maximum queue size and rate-control parameters. As suggested by Theorem 1 and also analyzed in [11], [12], [15], [16], [18], $Q_{\max}(\alpha, \tau)$ is a monotonic increasing function of both α and τ, and thus can be controlled by adjusting α for given τ. The theorem given below derives an explicit relationship among α, τ, and Q_h subject to the lossless transmission and $C_{\max} < \infty$ constraints.

Theorem 2: Consider a multicast connection flow-controlled by the proposed scheme with $Q_h > 0$ and $C_{\max} < \infty$ at the multicast-tree bottleneck. If $C_{\max} > 2Q_h$, then the following claims hold.

Claim 1. $\mathcal{F} \neq \emptyset$ and $\exists\ K > 0$ such that $(\alpha, \tau) \in \mathcal{F}\, \forall(\alpha, \tau) \in \{(\alpha, \tau) \,|\, \tau\sqrt{\alpha} \leq K, (\alpha, \tau) \in \Omega\}$.

Claim 2. $\mathcal{L}$ is lower-bounded by the function $K_\ell = \tau\sqrt{\alpha}$ where $K_\ell = \sqrt{C_{\max}} - \sqrt{2Q_h}$ and $(\alpha, \tau) \in \Omega$.

Proof: The proof is provided in Appendix C. ■

Remarks on Theorem 2: **(1)** Claim 1 shows that $Q_{\max}$ is controllable, and identifies a sufficient condition $(C_{\max} > 2Q_h)$ for the feasibility of lossless transmission. Moreover, Claim 1 describes the configuration of the lossless-transmission region defined in Ω. **(2)** Claim 2 gives a lower bound of the lossy transmission region $\mathcal{L}$ for given $C_{\max}$ and Q_h, which is expressed by a continuous function defined over Ω. Since Ω is partitioned into $\mathcal{F}$ and $\mathcal{L}$, the lower bound of $\mathcal{L}$ can be used as an approximate upper bound for $\mathcal{F}$ when the lower bound for $\mathcal{L}$ is tight. Thus, for any given $C_{\max}$ and Q_h, the lower-bound function $\tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$ provides the network designer with a simple formula to estimate α without seeking its close-form expression as a function of τ and $C_{\max}$, which is impossible to obtain [due to the nonlinearity of (16)]. Furthermore, since the lower-bound function $\tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$, dividing $\mathcal{F}$ and $\mathcal{L}$, is obtained by the constraint: $Q_{\max} \leq C_{\max}$, setting $Q_{\max} = C_{\max}$ in the lower bound yields a formula: $Q_{\max} = (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2$, which can be used to estimate $Q_{\max}$ when the lower-bound of $\mathcal{L}$ is tight. **(3)** Another interesting fact revealed by Theorem 2 is that $Q_{\max}$ is virtually independent of the multicast-tree bottleneck target bandwidth μ since neither the lossless transmission condition/region nor the lower bound of $\mathcal{L}$ contains μ. This is not surprising since it is the rate mismatch between $R(t)$ and μ, instead of the absolute value of μ, that determines $Q_{\max}$.

To illustrate the tightness of the derived lower bound of $\mathcal{L}$, the exact border which partitions Ω, the lower-bound function of $\mathcal{L}$ given by $K = \tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$, and the configurations of the lossless transmission region $\mathcal{F}$ (the shaded area separated by $\tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$) and lossy transmission region $\mathcal{L}$ are plotted in Fig. 2, with $C_{\max} = 400$ cells and $Q_h = 50$ cells, which gives $K = 10$, and $\mu = 367$ cell/ms (about 155 Mb/s). The exact border between $\mathcal{F}$ and $\mathcal{L}$ is obtained numerically [by solving (16) which needs μ]. The lower-bound function of $\mathcal{L}$ (given by $K = \sqrt{C_{\max}} - \sqrt{2Q_h} = \tau\sqrt{\alpha}$) plotted in Fig. 2 is found to be very close to the exact border between $\mathcal{L}$ and $\mathcal{F}$. In addition, the smaller α, the tighter the bound is, which is consistent with the approximation $\log x \approx x - 1$ when x is close to 1 [see (44)].

B. The Second-Order Rate Control

As suggested by Theorem 2, α can be controlled to confine $Q_{\max}$ to $C_{\max}$, and as long as $C_{\max} > 2Q_h$, lossless transmission can be guaranteed by adjusting α in response to the variation of τ. The control over $\alpha = dR(t)/dt$ —which we call *α-control*— is the second-order control process which will be elaborated on below from a control-theoretic viewpoint. The original ATM recommendation for unicast (CI-based) ABR flow control is based on the Additive Increase and Multiplicative Decrease (AIMD) rate control [8]. The AIMD adapts $R(t)$ to μ based on the feedback CI-bit. Since the AIMD applies direct control over the rate $R(t)$ to match the target μ, we can call AIMD the *speed feedback control* (from a control-theoretic viewpoint). The speed feedback control system is traditionally called the first-order feedback control system (having one pole, or being represented in a one-dimensional state-space). The α-control is an acceleration feedback-control system (having two poles, or being represented in a two-dimensional state-space), which is one-order higher than the AIMD, since it exerts direct control over $\alpha = dR(t)/dt$. Thus, we call the α-control the *second-order rate control*, which provides one more dimension in state-space control over the dynamics of the proposed flow-control system.

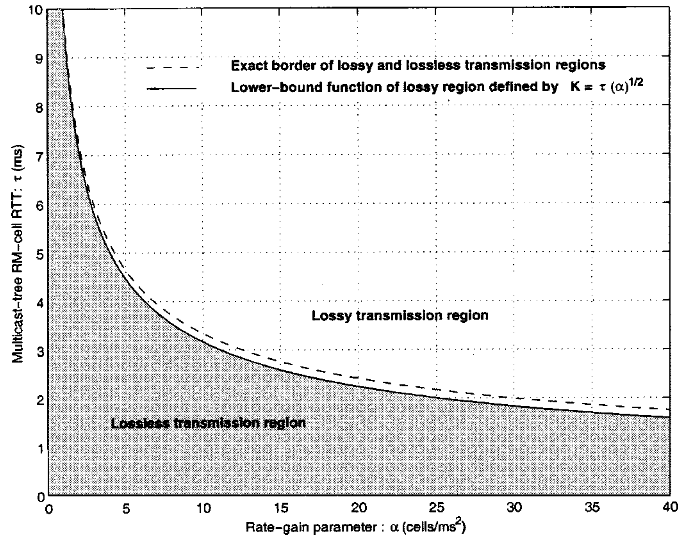


Fig. 2. Lossy and lossless transmission regions divided by the lower bound of lossy-transmission region.

C. The α-Control

The α-control is a discrete-time control since it is only exercised when the source rate control is in a "decrease-to-increase" transition based on the buffer congestion feedback BCI. $\mathrm{BCI}(n) := 0$ (or 1) if $Q_{\max}^{(n)} \leq Q_{\text{goal}}$ (or $Q_{\max}^{(n)} > Q_{\text{goal}}$), where Q_{goal} $(2Q_h < Q_{\text{goal}} < C_{\max})$ is the target buffer occupancy (i.e., *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}, the buffer tends 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 to a shorter path, then $Q_{\max}$ will decrease. When $Q_{\max} < Q_{\text{goal}}$, only a small portion of buffer is used, 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 variations. Keeping $2Q_h < Q_{\text{goal}} < C_{\max}$ has two benefits: (1) the source can quickly grab available bandwidth, and (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 given τ; (2) maintain statistical fairness on the buffer occupancy among multiple multicast connections which share a common multicast-tree bottleneck; and (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 BCI bit 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:

$$\alpha_{n+1} = \begin{cases} \alpha_n + p, & \text{if } \mathrm{BCI}(n-1,n) = (0,0) \\ q\alpha_n, & \text{if } \mathrm{BCI}(n) = 1 \\ \alpha_n / q, & \text{if } \mathrm{BCI}(n-1,n) = (1,0) \end{cases} \tag{5}$$

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.

D. The Convergence Properties of the α-Control

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

Definition 3: The **neighborhood** of target buffer occupancy Q_{goal} is specified by $\{Q_{\text{goal}}^l, Q_{\text{goal}}^h\}$ with

$$Q_{\text{goal}}^l \triangleq \max_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} \le Q_{\text{goal}} \right\} \tag{6}$$

$$Q_{\text{goal}}^h \triangleq \min_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} \ge Q_{\text{goal}} \right\} \tag{7}$$

where $Q_{\max}^{(n)}$ is governed by the proposed α-control law. ■

Definition 4: $\{Q_{\max}^{(n)}\} \triangleq \{Q_{\max}(\alpha_n)\}$ is said to **monotonically** converge to Q_{goal}'s neighborhood at time $n = n^*$ from its initial value $Q_{\max}^{(0)} = Q_{\max}(\alpha_0)$, if $\mathrm{BCI}(0,1,2,3,\ldots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\ldots) = (0,0,0,0,\ldots,0,1,0,1,0,\ldots)$, $\forall \alpha_0 < \alpha_{\text{goal}}$; and $\mathrm{BCI}(0,1,2,3,\ldots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\ldots) = (1,1,1,1,\ldots,1,0,1,0,1,\ldots)$, $\forall \alpha_0 > \alpha_{\text{goal}}$. ■

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

Theorem 3: Consider the proposed α-control law (5) 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) $p \le ((1-q)/q)((\sqrt{Q_{\text{goal}}} - \sqrt{2Q_h})/\tau)^2$, then the following claims hold:

Claim 1. During the **transient state**, the α-control law guarantees $Q_{\max}^{(n)}$ to **monotonically** converge to Q_{goal}'s neighborhood $\{Q_{\text{goal}}^l, Q_{\text{goal}}^h\} = \{Q_{\max}(\alpha_{\text{goal}}^l), Q_{\max}(\alpha_{\text{goal}}^h)\}$, which are determined by

$$Q_{\text{goal}}^l = \begin{cases} Q_{\max}\left(q^{n^*}\alpha_0\right), & \text{if } \alpha_0 > \alpha_{\text{goal}} \\ Q_{\max}(q(n^*p + \alpha_0)), & \text{if } \alpha_0 \le \alpha_{\text{goal}} \end{cases} \tag{8}$$

$$Q_{\text{goal}}^h = \begin{cases} Q_{\max}\left(q^{(n^*-1)}\alpha_0\right), & \text{if } \alpha_0 > \alpha_{\text{goal}} \\ Q_{\max}(n^*p + \alpha_0), & \text{if } \alpha_0 \le \alpha_{\text{goal}} \end{cases} \tag{9}$$

where n^* is defined in Definition 4.

Claim 2. During the **equilibrium state**, the fluctuation amplitudes of $Q_{\max}^{(n)}$ around Q_{goal} are upper-bounded by

$$Q_{\text{goal}}^h - Q_{\text{goal}} \le \tau^2\alpha_{\text{goal}}\left(\frac{1}{q} - 1\right) + \tau\sqrt{8\alpha_{\text{goal}}Q_h}\left(\frac{1}{\sqrt{q}} - 1\right) \tag{10}$$

$$Q_{\text{goal}} - Q_{\text{goal}}^l \le \tau^2\alpha_{\text{goal}}(1-q) + \tau\sqrt{8\alpha_{\text{goal}}Q_h}(1-\sqrt{q}) \tag{11}$$

and the diameter of neighborhood for the target buffer occupancy Q_{goal} is upper-bounded as follows:

$$Q_{\text{goal}}^h - Q_{\text{goal}}^l \le \tau^2\alpha_{\text{goal}}\left(\frac{1}{q} - q\right) + \tau\sqrt{8\alpha_{\text{goal}}Q_h}\left(\frac{1}{\sqrt{q}} - \sqrt{q}\right) \tag{12}$$

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

Proof: The proof is detailed in Appendix D. ■

Remarks on Theorem 3: The α-control law is similar to, but differs from, the AIMD algorithm [9] in the following senses. In the transient state, the α-control behaves like AIMD, accommodating statistical convergence to fairness of buffer usage among the multicast connections sharing a multicast-tree bottleneck. On the other hand, in equilibrium state, the α-control ensures buffer occupancy to be locked within its setpoint region at the first time when $Q_{\max}^{(n)}$ reaches Q_{goal}'s neighborhood, regardless of the initial value α_0. In contrast, AIMD does not guarantee this monotonic convergence since α-control is a discrete-time control and its convergence is dependent on α_0. The monotonic convergence ensures that $Q_{\max}^{(n)}$ quickly converges to, and stays within, the neighborhood of Q_{goal}. The extra cost paid for achieving these benefits is minimized since only a single binary bit, BCI, is conveyed from the network and two bits are used to store the current and one-step-old feedback $\mathrm{BCI}(n-1)$ and $\mathrm{BCI}(n)$ bits at the source. The α-increase step-size p specified by condition (3) in Theorem 3 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 3, if $q \to 1$, then $p \to 0$, which is expected since for a stable convergent system, a zero decrease corresponds to a zero increase in system state. Based on (10), (11), and (12), when $q \to 1$, both Q_{goal}^l and $Q_{\text{goal}}^h \to Q_{\text{goal}}$, i.e., $Q_{\max}^{(n)}$'s fluctuation amplitude approaches zero, which also makes sense since $q \to 1$ implies $p \to 0$, and thus $Q_{\max}^{(n)}$ approaches a constant for all n.

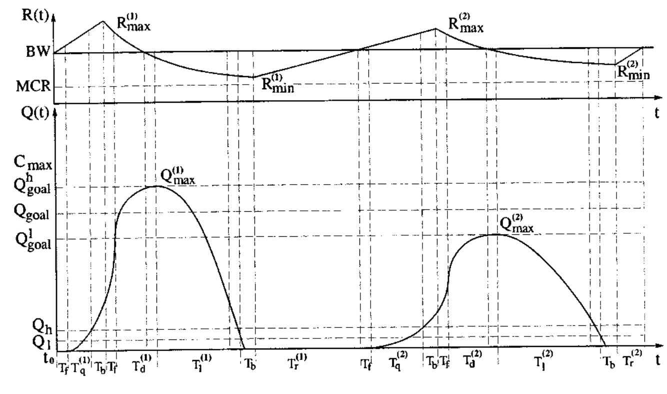


Fig. 3. Dynamic behavior of $R(t)$ and $Q(t)$ for a single multicast connection.

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 (5), the proposed algorithm also updates the rate-decrease factor by $\beta_n = 1 - (\alpha_n/\mu)\Delta$ 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 converged to the certain regime, oscillating with a fixed frequency and average amplitude. In this state, $R(t)$ fluctuates around μ, and $Q_{\max}^{(n)}$ around Q_{goal}. The fluctuation amplitudes and periods are determined by the rate-control parameters α, β; bandwidth μ; target buffer occupancy Q_{goal}; α-control parameters p, q; queue thresholds Q_h, Q_l; and delays T_b, T_f. The equilibrium-state analysis is mainly used to characterize the dynamics of the *multicast-tree bottleneck* after it has converged to a particular path and become relatively steady. For simplicity, we assume that α-control parameters—$\alpha_0, Q_{\text{goal}}, p$, and q—are properly chosen based on the conditions given in Theorem 3, such that $Q_{\max}^{(n)}$ converges to the midpoint of the neighborhood: $Q_{\text{goal}} = (1/2)(Q_{\text{goal}}^l + Q_{\text{goal}}^h)$ and $Q_{\text{goal}}^h < C_{\max}$.

Fig. 3 illustrates the first two cycles of rate fluctuation and the associated queue-length function at the bottleneck link in equilibrium state with $\alpha_1 = \alpha_{\text{goal}}^h$. At time t_0, $R(t)$ reaches μ(BW) and $Q(t)$ starts to build up after a delay T_f. At time $t_0 + T_b + T_q^{(1)}$, $Q(t)$ reaches Q_h and bandwidth congestion is detected. After a delay T_b, the source receives CI $= 1$ feedback and $R(t)$ begins to decrease exponentially. $Q(t)$ reaches the peak as $R(t)$ drops back to μ. When $R(t)$ falls below μ, $Q(t)$ starts to decrease. After a period T_l elapsed, $Q(t)$ reaches Q_l, then the noncongestion status $(\text{CI} = 0)$ is detected and fedback to the source. After a delay T_b, the $(\text{CI} = 0)$ feedback arrives at the source, then the "rate-decrease to rate-increase" transition condition $(\text{local\_CI} = 1 \wedge \text{CI} = 0)$ is detected at the source. Subsequently, the source adjusts the next rate-gain parameter α_2 to a smaller value, $q\alpha_1$ (β_2 is also adjusted by $\beta_2 = 1 - (\alpha_2/\mu)\Delta$) since $\text{BCI}(1) = 1$ (due to $Q_{\max}^{(1)} > Q_{\text{goal}}$) is received in the feedback RM cell. Then, $R(t)$ increases linearly with the newly updated rate-gain parameter $\alpha_2 = q\alpha_1 = \alpha_{\text{goal}}^l$. When $R(t)$ reaches μ after a period $T_r^{(1)}$, the system starts the second fluctuation cycle.

The dynamics of the second fluctuation cycle is similar to the first cycle except for the reduced α_2 and increased β_2, leading to a longer cycle length. 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 $Q_{\max}^{(2)} < Q_{\text{goal}}$, i.e., $\text{BCI}(2) = 0$, hence $\text{BCI}(1,2) = (1,0)$. But $\alpha_3 = \alpha_2/q = (q\alpha_1)/q = \alpha_1$ since α_n has already converged to $\{\alpha_{\text{goal}}^l, \alpha_{\text{goal}}^h\}$ in equilibrium state. Thus, 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 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 $T \stackrel{\Delta}{=} T_1 + T_2$.

In the ith fluctuation cycle $(i = 1, 2)$, let $R_{\max}^{(i)}$ and $R_{\min}^{(i)}$ be its maximum and minimum rates, respectively. Then we have

$$R_{\max}^{(i)} = \mu + \alpha_i \left(T_q^{(i)} + T_b + T_f \right) \tag{13}$$

where $T_q^{(i)} = \sqrt{2Q_h/\alpha_i}$ is the time for $Q(t)$ to grow from 0 to Q_h, $\alpha_1 = \alpha_{\text{goal}}^h = \alpha_{\text{goal}}^l/q$ and $\alpha_2 = q\alpha_1 = \alpha_{\text{goal}}^l$. We define

$$T_{\max}^{(i)} \stackrel{\Delta}{=} T_b + T_q^{(i)} + T_f = T_b + \sqrt{\frac{2Q_h}{\alpha_i}} + T_f \tag{14}$$

during which $R(t)$ increases from μ to $R_{\max}^{(i)}$ under linear rate-increase control. Then, the maximum queue length is given by

$$Q_{\max}^{(i)} = \int_0^{T_{\max}^{(i)}} \alpha_i t \, dt + \int_0^{T_d^{(i)}} \left(R_{\max}^{(i)} e^{-(1-\beta_i)\frac{t}{\Delta}} - \mu \right) dt \tag{15}$$

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

$$Q_{\max}^{(i)} = \frac{\alpha_i}{2} \left[T_{\max}^{(i)} \right]^2 + \frac{\Delta}{1-\beta_i} \left[\alpha_i T_{\max}^{(i)} + \mu \log \frac{\mu}{R_{\max}^{(i)}} \right]. \tag{16}$$

Letting $T_l^{(i)}$ be the time for $Q(t)$ to drop from $Q_{\max}^{(i)}$ to Q_l yields

$$Q_{\max}^{(i)} - Q_l = \int_0^{T_l^{(i)}} \mu \left(1 - e^{-(1-\beta_i)\frac{t}{\Delta}} \right) dt. \tag{17}$$

So, $T_l^{(i)}$ is the nonnegative real root of nonlinear equation

$$e^{-(1-\beta_i)\frac{T_l^{(i)}}{\Delta}} + \frac{1-\beta_i}{\Delta} \left[T_l^{(i)} - \frac{Q_{\max}^{(i)} - Q_l}{\mu} \right] - 1 = 0. \tag{18}$$

Then, the minimum rate is given by

$$R_{\min}^{(i)} = \mu e^{-(1-\beta_i)\frac{T_l^{(i)} + T_b + T_f}{\Delta}}. \tag{19}$$

The control period is determined by

$$T = \sum_{i=1}^{2} T_i = \sum_{i=1}^{2} \left[T_q^{(i)} + T_d^{(i)} + T_l^{(i)} + 2\tau + T_r^{(i)} \right] \tag{20}$$

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 α_{i+1} $(\alpha_3 = \alpha_1)$.

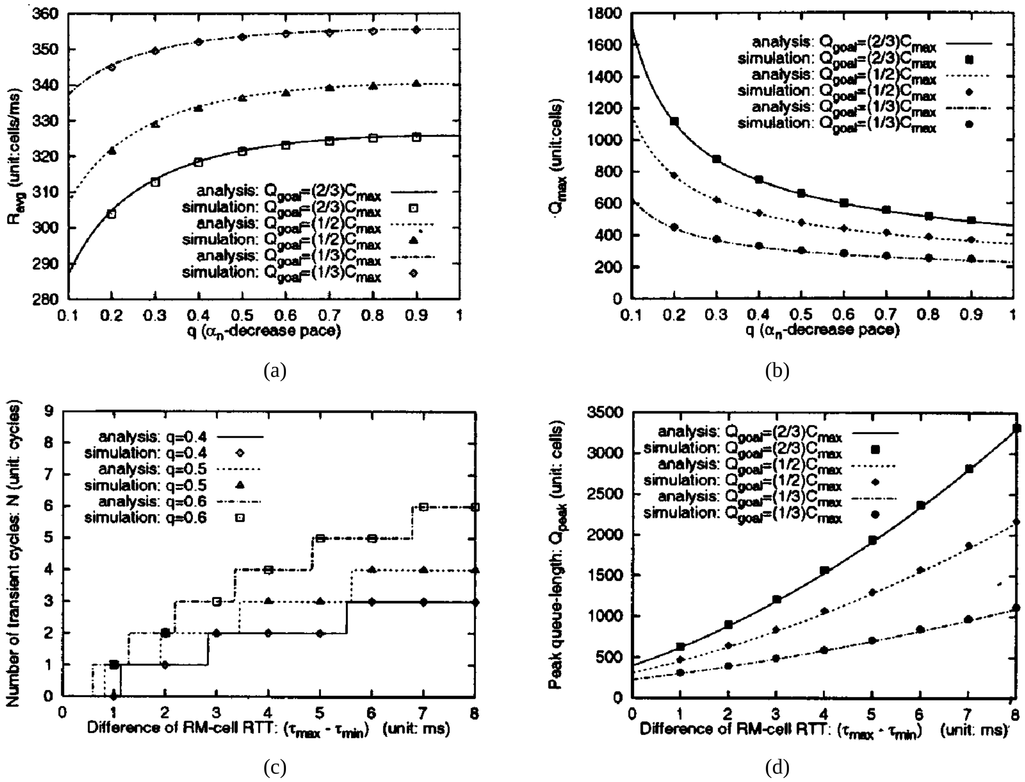


Fig. 4. Equilibrium and transient state performance evaluation. (a) $\overline{R}$ versus q. (b) $Q_{\max}$ versus q. (c) N versus $(\tau_{\max} - \tau_{\min})$. (d) $Q_{\rm peak}$ versus $(\tau_{\max} - \tau_{\min})$.

The average equilibrium throughput, $\overline{R}$, can be calculated by

$$\overline{R} = \frac{1}{T}\sum_{i=1}^{2}\left[\int_0^{T_{\max}^{(i)}}(\mu+\alpha_i t)\,dt + \int_0^{T_e^{(i)}} R_{\max}^{(i)} e^{-(1-\beta_i)\frac{t}{\Delta}}\,dt + \int_0^{T_r^{(i)}}\left(R_{\min}^{(i)}+\alpha_{i+1}t\right)dt\right] \qquad (21)$$

where $T_e^{(i)} = T_d^{(i)} + T_l^{(i)} + \tau$ is the time spent on exponential-decrease rate control within the ith cycle. Equation (20) reduces to

$$\begin{aligned}\overline{R} = \frac{1}{T}\sum_{i=1}^{2}\Bigg\{&\mu T_{\max}^{(i)} + \frac{\alpha_i}{2}\left[T_{\max}^{(i)}\right]^2 \\ &+ R_{\max}^{(i)}\left(\frac{\Delta}{1-\beta_i}\right)\left(1-e^{-(1-\beta_i)\frac{T_e^{(i)}}{\Delta}}\right) \\ &+ T_r^{(i)}R_{\min}^{(i)} + \frac{\alpha_{i+1}}{2}\left[T_r^{(i)}\right]^2\Bigg\}. \end{aligned} \qquad (22)$$

B. Equilibrium-State Performance Evaluation

Let the bottleneck link bandwidth $\mu = 155$ Mb/s (367 cells/ms) and $C_{\max} = 750$ cells. Assume $T_b = T_f = 1$ ms and $\tau = T_b + T_f = 2$ ms. Also, set $\Delta = 0.5\tau = 1$ ms, $Q_h = 50$, $Q_l = 25$ cells, and the initial source rate $R_0 = \mu$ for the equilibrium state.

Fig. 4(a) plots $\overline{R}$ versus q for different $Q_{\rm goal}$'s obtained from the analysis and the simulations[7] for the ideal case where $Q_{\rm goal} = (1/2)(Q_{\rm goal}^h + Q_{\rm goal}^l)$. Fig. 4(a) shows that $\overline{R}$ monotonically increases as q grows. This is expected since a smaller q leads to a larger fluctuation of $R_{\max}^{(n)}$ and $Q_{\max}^{(n)}$, degrading $\overline{R}$ in the equilibrium state. When q gets larger, the fluctuation amplitudes of $Q_{\max}^{(n)}$ and $R_{\max}^{(n)}$ get smaller, as shown in Theorem 3. In the extreme case when $q \to 1$ ($q \neq 1$ since $q = 1$ means that α-control is shut down), $R_{\max}^{(n)}$ approaches a constant. Fig. 4(a) also indicates that for the same q, a smaller $Q_{\rm goal} = kC_{\max}$, $0 < k < 1$, gives a larger $\overline{R}$ in equilibrium, also confirming our observations in [15], as a smaller $Q_{\rm goal}$ implies a smaller $\alpha_{\rm goal}$. Moreover, Fig. 4(a) shows: 1) $\overline{R}$ drops more quickly if $q \le 0.4$ and 2) $\overline{R}$ gains slowly if $q \ge 0.6$, providing network designers with the optimal range of α-control parameter q. Fig. 4(a) also shows that the analytical results based on the fluid modeling match the simulated results well. The slight discrepancy is due to the RM-cell processing and queueing delays, and fluid analysis approximations.

While $Q_{\rm goal}$ can be anywhere between $Q_{\rm goal}^l$ and $Q_{\rm goal}^h$ depending on α_0, to analyze how q affects $Q_{\max}$ in the worst case, Fig. 4(b) plots $Q_{\max}$ versus q in the worst case where $Q_{\rm goal} \overset{\sim}{\ge} Q_{\rm goal}^l$. $Q_{\max}$ is observed to increase as q decreases, which makes sense since a smaller q implies a larger fluctuation amplitude of $Q_{\max}^{(n)}$. Fig. 4(b) also shows that $Q_{\max}$ shoots up quickly when q is below the range of 0.4–0.6 while $Q_{\max}$ drops slowly when q is above the range of 0.4–0.6, giving the same optimal range of q as observed in Fig. 4(a). Again, the analytical results are verified by the simulated results as shown in Fig. 4(b).

C. Transient-State Analysis

The system can enter the transient state due to the variation of τ and bandwidth in two different cases: 1) $\alpha_0 > \alpha_{\rm goal}^h$, the rate

[7]We used the NetSim package [19] for the simulations, and set the simulation parameters exactly the same as those used the analysis for comparison purposes.

convergence is underdamped, and 2) $\alpha_0 < \alpha^l_{\rm goal}$, the rate convergence is overdamped, where $\alpha^h_{\rm goal}$ and $\alpha^l_{\rm goal}$ are functions of $Q_{\rm goal}, p, q, \tau$, and μ. The transient-state analysis aims at characterizing the system dynamics while the multicast-tree bottleneck path is still in progress converging from one to another equilibrium state. Denote the transient-state initial rate-gain by α_0, and the new bottleneck's target rate-gain by $\widetilde{\alpha_{\rm goal}}$ corresponding to the new bottleneck path's RTT $\widetilde{\tau}$ and target bandwidth $\widetilde{\mu}$. The following theorem calculates the number of transient cycles.

Theorem 4: Consider a multicast-tree bottleneck characterized by $Q_{\rm goal}, Q_h, p$, and q. If the initial rate gain $\alpha = \alpha_0$, 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

$$N = \begin{cases} \left\lfloor \log\left(\dfrac{\widetilde{\alpha_{\rm goal}}}{\alpha_0}\right) \Big/ \log q \right\rfloor, & \text{if } \alpha_0 > \widetilde{\alpha_{\rm goal}} \\ \lceil (\widetilde{\alpha_{\rm goal}} - \alpha_0)/p \rceil, & \text{if } \alpha_0 \le \widetilde{\alpha_{\rm goal}} \end{cases} \tag{23}$$

where $\widetilde{\alpha_{\rm goal}}$ is the nonnegative real root of nonlinear equation

$$\frac{\widetilde{\alpha_{\rm goal}}\widetilde{\rho}^2}{2} + \widetilde{\mu}\widetilde{\rho} + \frac{\widetilde{\mu}^2}{\widetilde{\alpha_{\rm goal}}} \log \frac{\widetilde{\mu}}{\widetilde{\mu} + \widetilde{\alpha_{\rm goal}}\widetilde{\rho}} - Q_{\rm goal} = 0 \tag{24}$$

where $\widetilde{\rho} = \widetilde{\tau} + \sqrt{2Q_h/\widetilde{\alpha_{\rm goal}}}$, and $\widetilde{\alpha_{\rm goal}}$ can be approximated by

$$\widetilde{\alpha_{\rm goal}} \approx \left(\frac{\sqrt{Q_{\rm goal}} - \sqrt{2Q_h}}{\widetilde{\tau}}\right)^2 \tag{25}$$

if $Q_{\rm goal}$ is small.

Proof: The proof is presented in Appendix F. ■

Let $R^{(i)}_{\rm peak}$ and $Q^{(i)}_{\rm peak}$ be the peak source rate and queue length, respectively, in the ith transient cycle, $i = 1, \ldots, N$ ($\ge$1) (assuming $\alpha_0 \ge (1/q)\widetilde{\alpha_{\rm goal}}$ or $\alpha_0 \le \widetilde{\alpha_{\rm goal}} - p$). Let us start from the first ($i = 1$) transient cycle. Since $R(t) = R_0 + \alpha_0 t$, we have

$$R^{(1)}_{\rm peak} = R_0 + \alpha_0\left(T^{(1)}_q + \widetilde{\tau}\right) \tag{26}$$

where $T^{(1)}_q = (1/\alpha_0)[-(R_0 - \widetilde{\mu}) + \sqrt{(R_0 - \widetilde{\mu})^2 + 2\alpha_0 Q_h}]$ is obtained by solving following equation:

$$Q_h = \int_0^{T^{(1)}_q} (R(t) - \widetilde{\mu})\, dt. \tag{27}$$

Define $T^{(1)}_{\rm peak} \triangleq T^{(1)}_q + \widetilde{\tau}$ as the time for $R(t)$ to increase from R_0 to $R^{(1)}_{\rm peak}$, the peak queue length can be obtained by

$$Q^{(1)}_{\rm peak} = \int_0^{T^{(1)}_{\rm peak}} (R_0 + \alpha_0 t - \widetilde{\mu})\, dt + \int_0^{T^{(1)}_d} \left(R^{(1)}_{\rm peak} e^{-(1-\beta_0)\frac{t}{\Delta}} - \widetilde{\mu}\right) dt \tag{28}$$

where $T^{(1)}_d = -(\Delta/(1-\beta_0))\log(\widetilde{\mu}/R^{(1)}_{\rm peak})$ is the time for $R(t)$ to drop from $R_{\rm peak}$ back to $\widetilde{\mu}$. Reducing (28) gives

$$Q^{(1)}_{\rm peak} = (R_0 - \widetilde{\mu})T^{(1)}_{\rm peak} + \frac{\alpha_0}{2}\left[T^{(1)}_{\rm peak}\right]^2 + \frac{\Delta}{1-\beta_0} \cdot \left[\alpha_0 T^{(1)}_{\rm peak} + (R_0 - \widetilde{\mu}) + \widetilde{\mu}\log\frac{\widetilde{\mu}}{R^{(1)}_{\rm peak}}\right]. \tag{29}$$

If $R_0 = \widetilde{\mu}$, (29) reduces to (16), which is consistent with the fact that $Q^{(i)}_{\max}$ is the special case of $Q^{(1)}_{\rm peak}$ with $R_0 = \widetilde{\mu}$.

To compute the first transient-state cycle, we need to find $T^{(1)}_l$ which is the nonnegative real root of nonlinear equation

$$e^{-(1-\beta_0)\frac{T^{(1)}_l}{\Delta}} + \frac{1-\beta_0}{\Delta}T^{(1)}_l - \left[\frac{Q^{(1)}_{\rm peak} - Q_l}{\widetilde{\mu}}\left(\frac{1-\beta_0}{\Delta}\right) + 1\right] = 0. \tag{30}$$

This transient-state cycle is $T^{(1)} = T^{(1)}_q + T^{(1)}_d + T^{(1)}_l + 2\widetilde{\tau} + T^{(1)}_r$, where $T^{(1)}_r = (\widetilde{\mu}/\alpha_1)(1 - e^{-(1-\beta_0)(T^{(1)}_l + \widetilde{\tau})/\Delta})$ is the time for $R(t)$ to reach $\widetilde{\mu}$ from its lowest value in the first transient cycle. The average throughput in the first transient-state cycle is given by

$$\overline{R}^{(1)} = \frac{1}{T^{(1)}}\left[R_0 T^{(1)}_{\rm peak} + \frac{\alpha_0}{2}\left[T^{(1)}_{\rm peak}\right]^2 + R^{(1)}_{\rm peak} \cdot \left(\frac{\Delta}{1-\beta_0}\right)\left(1 - e^{-(1-\beta_0)\frac{T^{(1)}_d + T^{(1)}_l + \widetilde{\tau}}{\Delta}}\right) + T^{(1)}_r\left(\widetilde{\mu}e^{-(1-\beta_0)\frac{T^{(1)}_l + \widetilde{\tau}}{\Delta}}\right) + \frac{\alpha_1}{2}\left[T^{(1)}_r\right]^2\right].$$

Now, for the cases of $2 \le i \le N$ (N is given by (23) of Theorem 4), since the performance metrics are derived similarly to the case for $i = 1$, we only give the final results for the average throughput, peak queue length, and the length of the ith transient cycle:

$$\overline{R}^{(i)} = \frac{1}{T^{(i)}}\left[\widetilde{\mu}T^{(i)}_{\rm peak} + \frac{\alpha_{i-1}}{2}\left[T^{(i)}_{\rm peak}\right]^2 + R^{(i)}_{\rm peak} \cdot \left(\frac{\Delta}{1-\beta_{i-1}}\right)\left(1 - e^{-(1-\beta_{i-1})\frac{T^{(i)}_d + T^{(i)}_l + \widetilde{\tau}}{\Delta}}\right) + T^{(i)}_r\left(\widetilde{\mu}e^{-(1-\beta_{i-1})\frac{T^{(i)}_l + \widetilde{\tau}}{\Delta}}\right) + \frac{\alpha_i}{2}\left[T^{(i)}_r\right]^2\right] \tag{31}$$

$$Q^{(i)}_{\rm peak} = \frac{\alpha_{i-1}}{2}\left[T^{(i)}_{\rm peak}\right]^2 + \alpha_{i-1}\frac{\Delta}{(1-\beta_{i-1})}T^{(i)}_{\rm peak} + \widetilde{\mu}\frac{\Delta}{(1-\beta_{i-1})}\log\frac{\widetilde{\mu}}{R^{(i)}_{\rm peak}} \tag{32}$$

$$T^{(i)} = \sqrt{\frac{2Q_h}{\alpha_{i-1}}} + T^{(i)}_d + T^{(i)}_l + T^{(i)}_r + 2\widetilde{\tau} \tag{33}$$

where

$$T^{(i)}_{\rm peak} = \widetilde{\tau} + \sqrt{\frac{2Q_h}{\alpha_{i-1}}} \tag{34}$$

$$R^{(i)}_{\rm peak} = \widetilde{\mu} + \alpha_{i-1}T^{(i)}_{\rm peak} \tag{35}$$

$$T^{(i)}_d = -\frac{\Delta}{1-\beta_{i-1}}\log\frac{\widetilde{\mu}}{R^{(i)}_{\rm peak}} \tag{36}$$

$$T^{(i)}_r = \frac{\widetilde{\mu}}{\alpha_i}\left[1 - e^{-(1-\beta_{i-1})\frac{T^{(i)}_l + \widetilde{\tau}}{\Delta}}\right] \tag{37}$$

and $T_l^{(i)}$ is the nonnegative real root of the nonlinear equation:

$$e^{-(1-\beta_{i-1})\frac{T_l^{(i)}}{\Delta}} + \frac{1-\beta_{i-1}}{\Delta}\left(T_l^{(i)} - \frac{Q_{\text{peak}}^{(i)} - Q_l}{\widetilde{\mu}}\right) - 1 = 0 \tag{38}$$

where $2 \leq i \leq N$. The entire transient-state period is then $T_{\text{tran}} = \sum_{i=1}^{N} T^{(i)}$, and its average throughput is expressed by

$$\overline{R}_{\text{tran}} = \frac{1}{T_{\text{tran}}} \sum_{i=1}^{N} \overline{R}^{(i)} T^{(i)}. \tag{39}$$

The peak queue length for the case of $\alpha_0 > \alpha_{\text{goal}}^h$ is $Q_{\text{peak}} = Q_{\text{peak}}^{(1)}$, and α_i is determined by the α-control given by (5).

D. Transient-State Performance Evaluation

For the transient-state performance analysis, the same flow-control parameters are used as in the equilibrium-state analysis, except that $C_{\max} = 700$ cells, $Q_{\text{goal}} = (1/2)C_{\max} = 350$ cells, and α_0 is set by $\mu_0 = 367$ cells/ms and $\tau_0 = 2$ ms. To study the worst case, set $\tau_0 = \tau_{\min} \triangleq \min_{i \in \{1,\ldots,n\}}\{\tau_i\}$ and $\widetilde{\tau} = \tau_{\max} \triangleq \max_{i \in \{1,\ldots,n\}}\{\tau_i\}$ of a multicast VC (Virtual Circuit) with n paths, and assume $\widetilde{\mu} = 267$ cells/ms. Fig. 4(c) plots N, obtained numerically by (23) and simulations by NetSim [19], versus $(\tau_{\max} - \tau_{\min})$ for different 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. These observations have been exactly duplicated by simulations, thus verifying Theorem 4. Fig. 4(d) shows the numerical and simulation results for Q_{peak} versus $(\tau_{\max} - \tau_{\min})$ with Q_{goal} varying, where we set $R_0 = 367$ cells/ms, $\widetilde{\mu} = 347$ cells/ms, $\tau_{\min} = \tau_0 = 2$ ms, and $C_{\max} = 700$ cells. Q_{peak} is observed to shoot up quickly with $(\tau_{\max} - \tau_{\min})$, further justifying the necessity of α-control, and a larger target Q_{goal} is found to result in a faster increase of Q_{peak}. The simulation results closely match the analytical results as shown in Fig. 4(d).

VI. Multiple Multicast Connections

A. Analytical Analysis

$M\ (>1)$ concurrent flow controlled VCs with a common multicast-tree bottleneck are modeled by a single buffer and a server shared by M source rates $R_i(t)$. At time t the aggregate arrival rate at the multicast-tree bottleneck is $\sum_{i=1}^{M} R_i(t - T_f^{\langle i \rangle})$. So, the bottleneck's buffer queue length function at time t is

$$Q(t) = \int_{t_0}^{t} \left\{ \sum_{i=1}^{M} R_i\left(v - T_f^{\langle i \rangle}\right) - \mu \right\} dv + Q(t_0) \tag{40}$$

where $T_f^{\langle i \rangle}$ is forward delay for the ith VC. Applying the same rate-control proposed in Section II, for $i = 1, \ldots, M$, we have

$$R_i(t) = \begin{cases} R_i(t_0) + \alpha^{\langle i \rangle}(t - t_0), & \text{if } Q\left(t - T_b^{\langle i \rangle}\right) < Q_l \\ R_i(t_0) e^{-(1-\beta^{\langle i \rangle})\frac{(t-t_0)}{\Delta_i}}, & \text{if } Q\left(t - T_b^{\langle i \rangle}\right) \geq Q_h. \end{cases} \tag{41}$$

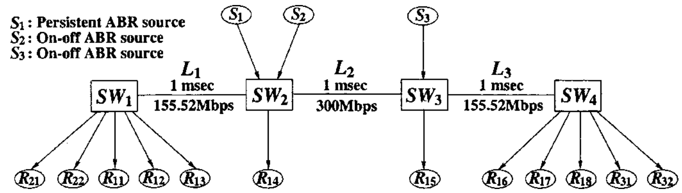


Fig. 5. Simulation model for multiple multicast VCs.

The α-control is applied in the same way as in the single multicast VC case, but $Q_{\max}^{(n)}$ is contributed, and Q_{goal} is shared, by all M VCs. The analytical results for multiple concurrent multicast VCs are omitted for lack of space. Instead, we present the simulation results below to (1) verify the analytical results and (2) analyze the performance of the proposed scheme for more general cases where locations, number, and bandwidth of multicast-tree bottlenecks vary with time.

B. Simulation Results

Using the NetSim simulator [19], we conducted extensive simulations for 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 modeling analysis, the simulation experiments 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 queueing delays, instantaneous variations of bottleneck bandwidths, which are very difficult to deal with analytically.

The simulated network is shown in Fig. 5, which consists of three multicast VCs running through four switches $\text{SW}_1, \ldots, \text{SW}_4$ connected by three links L_1, L_2, L_3. S_i is the source of $\text{VC}_i, i = 1, 2, 3$, and R_{ij} is S_i's jth 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) $\mu_1 = \mu_3 = 155.52$ Mb/s and (2) $\mu_2 = 300$ Mb/s, 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 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}. 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, $Q_{\text{goal}} = 400$ 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 any overlap in time. Consequently, as shown in Fig. 6(a)–(f), 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

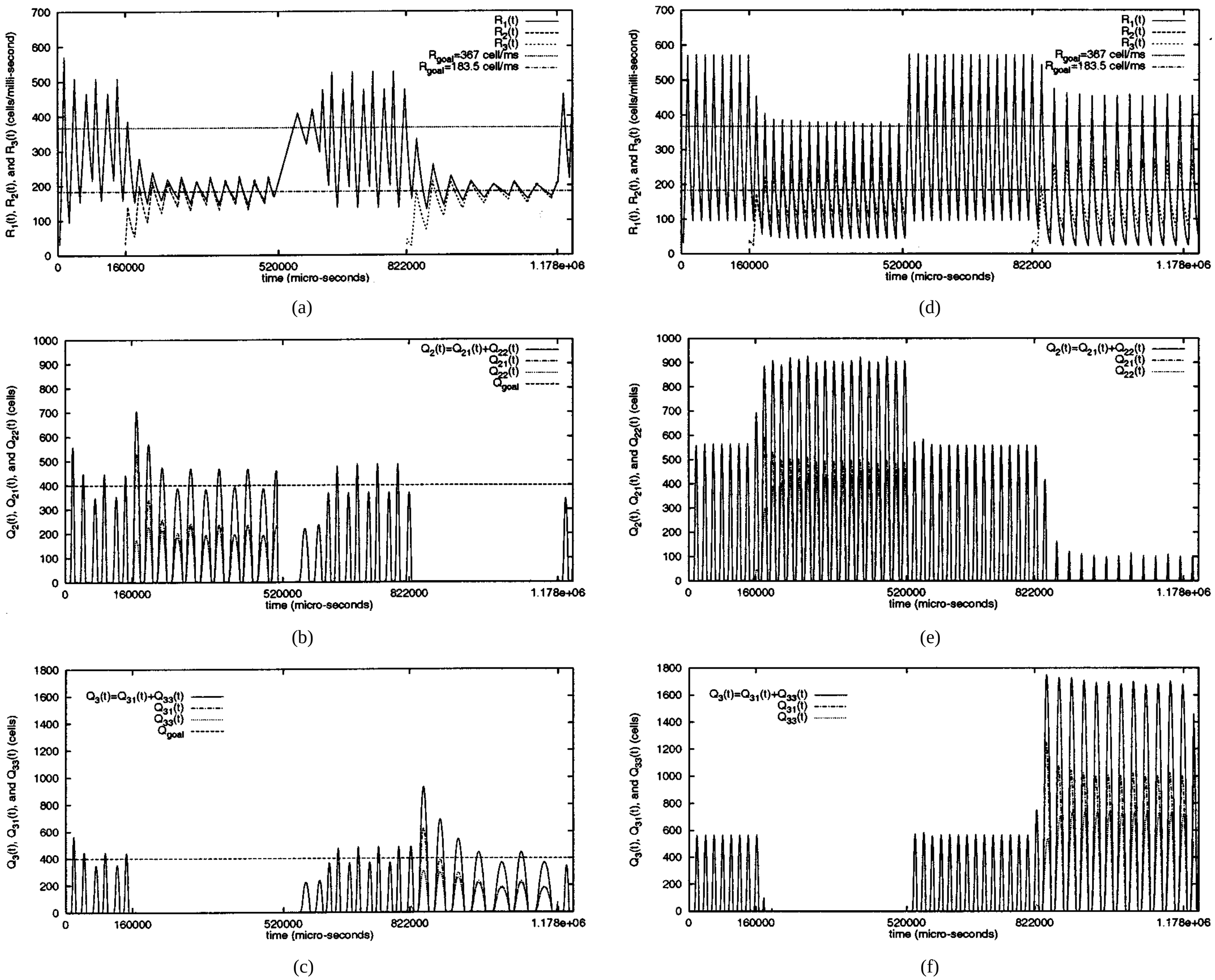


Fig. 6. Dynamics performance comparison between schemes with α-control and without α-control. (a) $R(t)$'s converge to bottleneck bandwidth with α-control. (b) SW_2: Total $Q_{\max}$ converges to Q_{goal} with α-control. (c) SW_3: Total $Q_{\max}$ converges to Q_{goal} with α-control. (d) $R(t)$'s converge to bottleneck bandwidth without α-control. (e) SW_2: Total $Q_{\max}$ does not converge to Q_{goal} without α-control. (f) SW_3: Total $Q_{\max}$ does not converge to Q_{goal} without α-control.

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. 6(a)–(f) and 7(a)–(d), where all results with α-control are plotted in Figs. 6(a)–(c) and 7(a)–(b) on the left, while those without α-control are shown in Figs. 6(d)–(f) and 7(c)–(d) 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. 6(a) shows that VC_1's rate $R_1(t)$ converges to L_1 and L_3's capacity 367 cells/ms (155.52 Mb/s) 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 $Q_{\max}^{\langle 3\rangle}$, respectively. Fig. 6(a)–(c) shows that after experiencing one transient cycle due to $Q_{\max}^{\langle 2\rangle} = Q_{\max}^{\langle 3\rangle} = 560 > Q_{\mathrm{goal}}$, $Q_{\max}^{\langle 2\rangle}$ and $Q_{\max}^{\langle 3\rangle}$ converge to Q_{goal}'s neighborhood [350, 446] by α-control. So, 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, Fig. 6(d)–(f) 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 went down to $Q_{\mathrm{goal}} = 400$.

2) **During** T_2. VC_2 starts transmission, competing 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 bottleneck with $\mathrm{RTT} = 2$ ms, target bandwidth $= (1/2)\mu_1$ for each of VC_1 and VC_2. For the α-controlled scheme, Fig. 6(a) shows that the source rates $R_1(t)$ and $R_2(t)$ experience two transient cycles during which $R_1(t)$ gives up $(1/2)\mu_1$ to $R_2(t)$ until they reach a new equilibrium. Fig. 6(b) shows that a large queue build-up $Q_{\max}^{\langle 2\rangle} = 704$ as a result of the superposed rate-gain parameter from $R_1(t)$ and $R_2(t)$, and the reduced bottleneck bandwidth. With α-control, $Q_{\max}^{\langle 2\rangle}$ is driven down to Q_{goal}'s neighborhood of [385, 468]. Fig. 6(c) shows $Q_3(t) = 0$, ver-

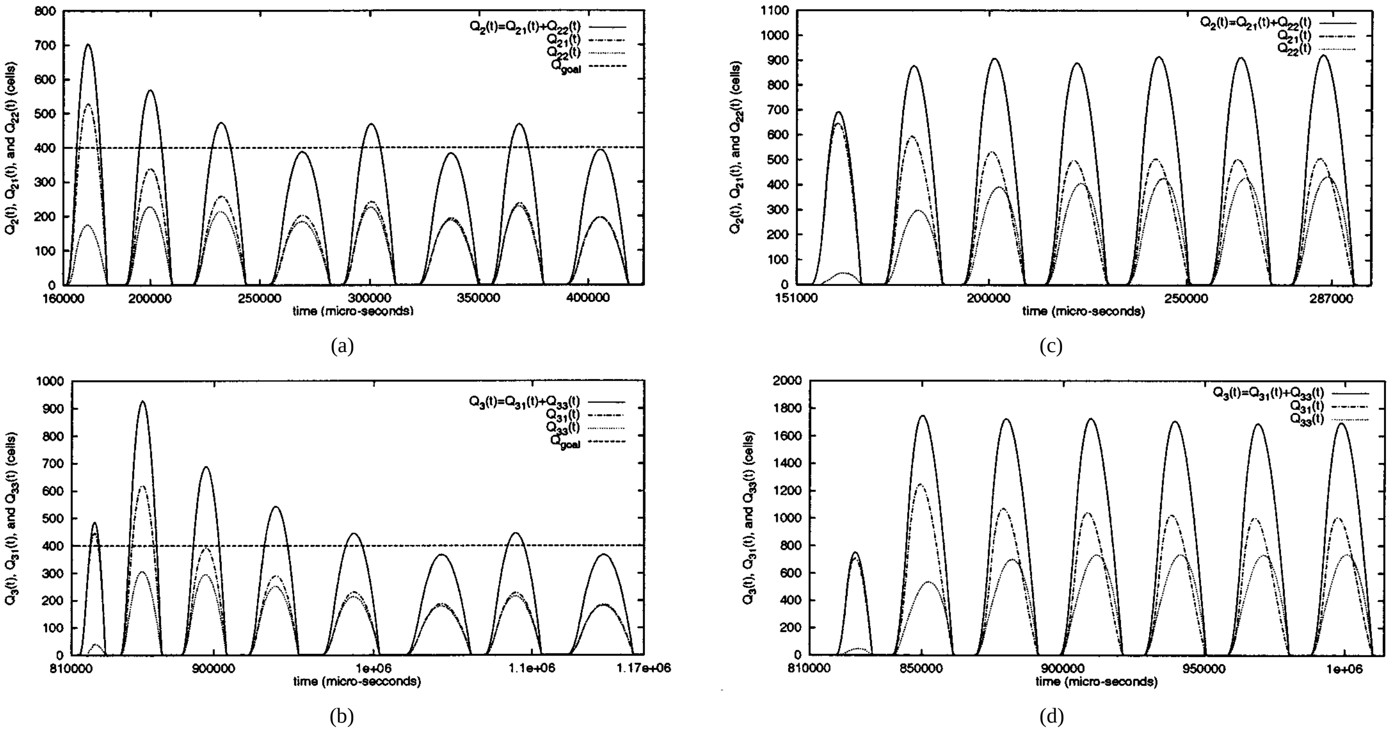


Fig. 7. Buffer occupancy fairness comparison between schemes with and without α-control. (a) SW_2: $Q_{21}(t)$, $Q_{22}(t)$ converge to fairness with α-control. (b) SW_3: $Q_{31}(t)$, $Q_{33}(t)$ converge to fairness with α-control. (c) SW_2: $Q_{21}(t)$, $Q_{22}(t)$ do not converge to fairness without α-control. (d) SW_3: $Q_{31}(t)$, $Q_{33}(t)$ do not converge to fairness without α-control.

ifying that the bottleneck at L_3 vanished. Fig. 7(a) is a zoom-in picture of $Q_2(t) = Q_{21}(t) + Q_{22}(t)$ of Fig. 6(b), 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. 7(a) indicates that in the first transient cycle, $Q_{21}(t)$'s maximum $Q_{\max}^{\langle 21\rangle} = 528$, which is more than 3 times of $Q_{22}(t)$'s maximum $Q_{\max}^{\langle 22\rangle} = 175$. 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. 6(e) illustrates that $Q_{\max}^{\langle 2\rangle}$ jumps up to as high as 900 and stays at 900 even after the transient state. Fig. 7(c), the zoom-in picture of Fig. 6(e), 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 enters an off-period, $R_1(t)$ grabs all μ_1 again. After $R_1(t)$ reaches μ_1, 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 enters an off-period, making $Q_{\max}^{\langle 2\rangle} \ll Q_{\text{goal}}$, 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 $Q_{\max}^{\langle 2\rangle}$ converges to Q_{goal}'s neighborhood [367, 483] within 3 transient cycles. Note that $Q_{\max}^{\langle 2\rangle}$ *monotonically* converges to [367, 483] as shown in Fig. 6(b). This is expected since $p = 16.67 \leq ((1-q)/q)((\sqrt{Q_{\text{goal}}} - \sqrt{2Q_h})/\tau)^2$, satisfying the condition (3) in Theorem 3. This observation further verifies the correctness of the optimal monotonic convergence condition derived in Theorem 3. In Fig. 6(d)–(e) 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 bottleneck is now at L_3 with a new target bandwidth $= (1/2)\mu_3$ and a longer $\mathrm{RTT} = 4$ ms. For the α-controlled scheme, Fig. 6(b) shows $Q_2(t) = 0$, i.e., the bottleneck at L_1 disappeared and L_3 is the only bottleneck. Fig. 6(c) shows that $Q_{\max}^{\langle 3\rangle}$ shoots up to 928, as a result of the doubled RTT (4 ms) via L_3. Within 3 transient cycles, $Q_{\max}^{\langle 3\rangle}$ converges to Q_{goal}'s neighborhood of [367, 445] in equilibrium state. Fig. 7(b), a zoom-in picture of Fig. 6(c), 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, for the scheme without α-control, Fig. 6(e)–(f) shows 2 bottlenecks: 1) a bandwidth-congestion bottleneck at L_1 and 2) a buffer-congestion bottleneck at L_3. Fig. 6(f) shows that $Q_{\max}^{\langle 3\rangle} = 1740$, almost 2 times of that under the α-controlled scheme. More importantly, $Q_{\max}^{\langle 3\rangle}$ stays around 1740 even after the transient state. Moreover, Fig. 7(d), a zoom-in picture of Fig. 6(f), demonstrates that buffer occupancy is not fair because $Q_{\max}^{\langle 31\rangle} = 1000$ but $Q_{\max}^{\langle 33\rangle} = 740$.

The three VCs average throughputs $\overline{R}_1, \overline{R}_2, \overline{R}_3$ (cells/ms) (for on-off sources averaging over the on-period only) obtained by the simulation are compared for the two types of schemes in Table I. In all the three VC cases, the proposed scheme with α-control is observed to outperform the scheme without α-control in terms of average throughput.

VII. Summary and Remarks

A. Summary

We proposed and analyzed a flow-control scheme for multicast ATM ABR services, which scales well and is efficient in

TABLE I
AVERAGE THROUGHPUTS COMPARISON OF THE TWO TYPES OF SCHEMES.

scheme type	$\overline{R}_1$ of VC_1	$\overline{R}_2$ of VC_2	$\overline{R}_3$ of VC_3
with α-control	234.448	150.671	147.709
without α-control	209.367	143.672	137.655

dealing with the variations in the multicast-tree structure. We developed the α-control, the second-order rate control, to handle the variation of RM-cell RTT. Under the α-control, the proposed scheme not only adapts the source rate to the available bandwidth of the multicast tree's most congested path, but also brings the buffer occupancy to a small neighborhood of the target setpoint bounded by buffer size. Although the second-order rate control was proposed for multicast flow control in [17], it is also applicable to unicast flow control as shown in [11], [15].

Applying the fluid analysis, we modeled the proposed multicast flow-control scheme and derived expressions for queue length, average throughput, and other performance measures in both transient and equilibrium states. We also derived an analytical relationship between the rate-gain parameter and RM-cell RTT, ensuring the feasibility of the α-control in dealing with RM-cell RTT variations. We developed an optimal control condition, under which the α-control guarantees the monotonic convergence of system states to the optimal regime from any initial values. The simulation results verified the analytical results in both transient and equilibrium states.

B. Remarks

Although a synchronous model is employed to control the RM-cell interval in the analysis, we also simulated our scheme under the asynchronous model where an RM cell is sent once every $N_{\rm rm} = 32$ [1] data cells. The asynchronous model turns out to have little effect on performance if $N_{\rm rm}$ is not too large. The throughput may degrade due to RM-cell starvation if $N_{\rm rm}$ is too large when RTT is large while keeping MCR low. On the other hand, too small an $N_{\rm rm}$ will cost too much bandwidth in signaling, and may also result in a high rate oscillation. Moreover, the asynchronous model is also applicable to the connections with different RTTs. The simulated results in Figs. 6 and 7 show that the α-control still converges to both bandwidth and buffer's efficiency and fairness even for connections with different RTTs.

While the infinite source, an assumption used in our fluid modeling, represents many typical network applications (e.g., file or image transmissions), there are also some finite sources, such as the on-off ABR sources. It is possible that a large number of on-off ABR sources sharing the same bottleneck enter an on-state from an off-state simultaneously, causing a severe congestion during the transient state. The simulation results in Figs. 6 and 7 show a large queue size when on-off ABR sources enter an on-state from an off-state. However, the congestion due to the on-off ABR source lasts only for a very limited time period during the transient state, and is quickly overcome under the α-control as the system enters the equilibrium state.

```
On receipt of a feedback RM cell:
  if (LCI = 1 ∧ CI = 0) {  !   Buffer congestion triggering conditions;
    if (BCI = 1) {AIR := q × AIR} ! AIR multiplitcatively decrease;
    elseif (BCI = 0 ∧ LBCI = 0) {AIR := p + AIR} ! Linear increase;
    elseif (BCI = 0 ∧ LBCI = 1) {AIR := AIR/q};  ! BCI toggles;
    MDF := exp(−AIR/BW_EST);  ! Update MDF;
    LNMQ := 1; LBCI := BCI;}; ! Start the new measurement cycle;
  if (CI = 0) {ACR := ACR + AIR}   ! ACR additively increasing;
  else {ACR := ACR × MDF};  ! ACR multiplicatively decreasing;
  LCI := CI;  !  Saved in local register for α-control in the next cycle.
```

Fig. 8. Pseudocode for source end system.

```
On receipt of a DATA cell:
  multicast DATA cell based on conn_patt_vec; ! Multicast data cell;
  if (data_qu ≥ Q_h) {CI := 1}; ! (1) Bandwidth congestion detection;
  if (data_qu ≥ Q_max) {Q_max := data_qu}; ! (2) Update Q_max;
  if (Q_max ≥ Q_goal) {BCI := 1} ! (3) Buffer congestion detection;
  else {BCI := 0}; ! (1), (2), and (3) applied to all connected branches;
On receipt of a feedback RM cell from i-th branch:
  if (conn_patt_vec(i) ≠ 1) {  ! Only process connected branches;
    resp_branch_vec(i) := 1; ! Mark connected and responsive branch;
    MCI := MCI ∨ CI;  !  Bandwidth-congestion indicator process;
    MBCI := MBCI ∨ BCI; ! Buffer-congestion indicator process;
    MER := min{MER, ER}; ! ER information processing;
    if (conn_patt_vec⊕resp_branch_vec=1) { ! Soft synchronization;
      send RM cell (dir := back, ER := min_resp_branches MER,
        CI := ⋃_resp_branches MCI, BCI :=
        ⋃_resp_branches MBCI); ! Send fully-consolidated RM cell;
      no_resp_timer := N_nrt; ! Reset the non-responsive timer;
      resp_branch_vec := 0; ! Reset the responsive branch vector;
      MCI := 0; }}; ! Reset RM-cell control variable;
On receipt of a forward RM cell:
  multicast RM cell based on conn_patt_vec; ! Multicast RM cell;
  if (NMQ=1) {MBCI := 0; Q_max := 0;}; ! Start new measure cyc
  no_resp_timer := no_resp_timer − 1; ! No-responsive branch check
  if (no_resp_timer = 0) { ! There is some non-responsive branch
    conn_patt_vec := resp_branch_vec ⊕ 1; ! Update connect pttn vec
    if (resp_branch_vec ≠ 0) { ! There is at least one responsive branch
      send RM cell (dir := back,  ER := min_resp_branches MER,
        CI := ⋃_resp_branches MCI,  BCI :=
        ⋃_resp_branches MBCI); ! Send partially-consolidated RM cell
      no_resp_timer := N_nrt; ! Reset non-responsive timer
      resp_branch_vec := 0; ! Reset responsive branch vector
      MCI := 0;  MER := ER; }}; ! Reset RM-cell control variables
On receipt of a join request from j-th branch:
  conn_patt_vec(j) := 0; ! Add branch in existing multicast connection.
```

Fig. 9. Pseudocode for intermediate switch system.

APPENDIX A
THE PSEUDOCODES FOR SOURCE-END AND INTERMEDIATE NODES

Figs. 8 and 9 give source and switch algorithms, respectively.

APPENDIX B
PROOF OF THEOREM 1

Proof: Using the fluid-modeling results on the multicast-tree bottleneck described in Section V, for $(\alpha, \tau) \in \Omega$ we have [see the derivations of (15) and (15)]

$$
\begin{aligned}
&Q_{\max}(\alpha, \tau) \\
&\quad = \int_0^{T_{\max}} \alpha t \, dt + \int_0^{T_d} \left(R_{\max} e^{-(1-\beta)\frac{t}{\Delta}} - \mu \right) dt \\
&\quad = \frac{\alpha}{2} T_{\max}^2 + \frac{\Delta}{1-\beta} \left(\alpha T_{\max} + \mu \log \frac{\mu}{R_{\max}} \right) \qquad (42)
\end{aligned}
$$

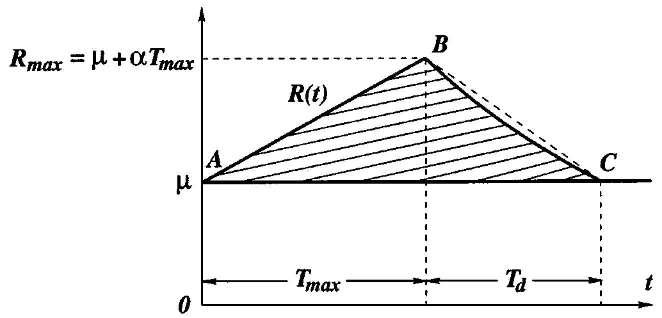


Fig. 10. $Q_{\max}$ (shaded area) is upper-bounded by the area of $\triangle ABC$.

where μ is the multicast-tree bottleneck target bandwidth, and

$$T_{\max} = \tau + \sqrt{\frac{2Q_h}{\alpha}}$$
$$R_{\max} = \mu + \alpha\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)$$
$$T_d = \frac{\Delta}{(1-\beta)} \log\left[1 + \frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right].$$

On the other hand, $Q_{\max}$ is also equal to the area between $R(t)$ and μ over the time interval of $T_{\max}+T_d$, and is upper-bounded by the area of its circumscribed triangle $\triangle ABC$ as shown in Fig. 10. Thus, we have

$$\begin{aligned}
&Q_{\max}(\alpha,\tau)\\
&\le \frac{1}{2}(\alpha T_{\max})(T_{\max}+T_d)\\
&= \frac{1}{2}\Bigg\{\alpha\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\\
&\quad \cdot \left(\frac{\alpha\Delta}{1-\beta}\log\left[1+\frac{\alpha}{\mu}\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\Bigg\}\\
&= \frac{1}{2}\Bigg\{\alpha\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\\
&\quad \cdot \left(\mu\log\left[1+\frac{\alpha}{\mu}\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\Bigg\} && (43)\\
&\le \frac{1}{2}\Bigg\{\alpha\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\\
&\quad \cdot \left(\mu\left[\frac{\alpha}{\mu}\left(\tau+\sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\Bigg\} && (44)\\
&= (\tau\sqrt{\alpha}+\sqrt{2Q_h})^2. && (45)
\end{aligned}$$

Since $\alpha > 0$ due to $(\alpha,\tau) \in \Omega$, equation (43) above holds because of the given constraint condition $\beta = 1 - (\alpha/\mu)\Delta$ (see the end of Section IV for the details). Equation (44) follows due to the fact that $\log x \le x - 1$ (Note: $\log x \approx x - 1$ for x close to 1. So, the bound gets tighter if $[1 + (\alpha/\mu)(\tau + \sqrt{2Q_h/\alpha})] = (1/\mu)R_{\max}$ is close to 1, i.e., $\mu < R_{\max} = \mu + \alpha(\tau + \sqrt{2Q_h/\alpha}) \ll 2\mu$, or equivalently $1 < (1/\mu)R_{\max} \ll 2$, which is the typical operating regime for the proposed scheme since α is small under the α-control for the given finite buffer capacity $C_{\max}$). Equation (45) yields the upper bound derived in (4), completing the proof. ■

APPENDIX C
PROOF OF THEOREM 2

Proof: Claim 1: Let $K \triangleq \tau\sqrt{\alpha}$, a positive real-valued number for $(\alpha,\tau) \in \Omega$. Define a real-valued function $\zeta(K) = \zeta(\tau\sqrt{\alpha}) \triangleq (K + \sqrt{2Q_h})^2$, which is the upper-bound function of $Q_{\max}(\alpha,\tau)$ obtained from (45). Thus, by Theorem 1 we have $\zeta(K) \ge Q_{\max}(\alpha,\tau)$ for $(\alpha,\tau) \in \Omega$, and further

$$\begin{aligned}
Q_{\max}(\alpha,\tau) &\le \zeta(K)\\
&= [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{\max})] + C_{\max}. && (46)
\end{aligned}$$

Since $C_{\max} > 2Q_h$ and $\zeta(K)$ is a continuous and monotonically increasing function of K, $\exists K > 0$ such that

$$K^2 + 2\sqrt{2Q_h}K < (C_{\max} - 2Q_h) \quad (47)$$
$$\Longrightarrow [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{\max})] < 0, \quad (48)$$

and $\{(\alpha,\tau)\,|\,\tau\sqrt{\alpha} \le K, (\alpha,\tau) \in \Omega\} \ne \emptyset$. Thus, $\forall(\alpha,\tau) \in \{(\alpha,\tau)\,|\,\tau\sqrt{\alpha} \le K, (\alpha,\tau) \in \Omega\}$ where K is specified by (48), by (46) and (48), we obtain

$$\begin{aligned}
Q_{\max}(\alpha,\tau) &\le [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{\max})] + C_{\max}\\
&< C_{\max} && (49)
\end{aligned}$$

which implies $(\alpha,\tau) \in \mathcal{F}$, thus $\mathcal{F} \ne \emptyset$.

Claim 2: To obtain a tight lower bound for $\mathcal{L}$, we set $Q_{\max}(\alpha,\tau)$'s upper-bound function $\zeta(K)$ equal to $C_{\max}$, i.e.,

$$Q_{\max}(\alpha,\tau) \le \zeta(K) = K^2 + 2\sqrt{2Q_h}K + 2Q_h = C_{\max} \quad (50)$$

which reduces to a quadratic equation: $K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{\max}) = 0$. Solving this quadratic equation for K and taking the positive root, we obtain $K_\ell = \sqrt{C_{\max}} - \sqrt{2Q_h} > 0$ since $C_{\max} > 2Q_h$. By (50), we have $(\alpha,\tau) \in \mathcal{F}, \forall(\alpha,\tau) \in \{(\alpha,\tau)\,|\,\tau\sqrt{\alpha} \le K_\ell, (\alpha,\tau) \in \Omega\}$, implying that all points located below or on the curve of function $K_\ell = \tau\sqrt{\alpha} \notin \mathcal{L}$. Thus, $\mathcal{L}$ is lower bounded by the function of $\zeta(K) = C_{\max}$ or $K_\ell = \tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$. This completes the proof. ■

APPENDIX D
PROOF OF THEOREM 3

Proof: Claim 1: We prove this claim by considering the following two cases, depending upon the range of α_0.

Case 1) $\alpha_0 \le \alpha_{\text{goal}}$: $Q_{\max}(\alpha)$ is a monotonically-increasing function of α and $\alpha_0 \le \alpha_{\text{goal}} \Rightarrow Q_{\max}^{(0)} = Q_{\max}(\alpha_0) \le Q_{\text{goal}} = Q_{\max}(\alpha_{\text{goal}})$. Applying α-control with an increase-step size p, $Q_{\max}^{(n)}$ monotonically approaches Q_{goal} from below at $Q_{\max}^{(0)}$. When the first time $Q_{\max}^{(n)} > Q_{\text{goal}}$ holds at $n = n^*$, i.e., $\alpha_0 + n^*p = \alpha_{n^*} > \alpha_{\text{goal}}$, the source detects BCI$(n^*-1, n^*) = (0,1)$, and then reduces α_n exponentially by setting $\alpha_{n^*+1} = q\alpha_{n^*}$. We want to prove the following:

$$Q_{\max}(\alpha_{n^*+1}) = Q_{\max}(q\alpha_{n^*}) \le Q_{\text{goal}}. \quad (51)$$

Since $(\tau\sqrt{\alpha_{\rm goal}} + \sqrt{2Q_h})^2 \geq Q_{\max}(\alpha_{\rm goal}) = Q_{\rm goal}$ by Theorem 1, we have $((\sqrt{Q_{\rm goal}} - \sqrt{2Q_h})/\tau)^2 \leq \alpha_{\rm goal}$. But, since $p \leq ((1-q)/q)((\sqrt{Q_{\rm goal}} - \sqrt{2Q_h})/\tau)^2$, we get $p \leq ((1-q)/q)\alpha_{\rm goal}$, which reduces to $q(\alpha_{\rm goal}+p) \leq \alpha_{\rm goal}$. On the other hand, due to $\alpha_{n^*-1} \leq \alpha_{\rm goal}$, we have $q(\alpha_{n^*-1}+p) \leq q(\alpha_{\rm goal}+p)$. Because $\alpha_{n^*} = \alpha_{n^*-1}+p$, $q(\alpha_{n^*-1}+p) \leq q(\alpha_{\rm goal}+p)$, and $q(\alpha_{\rm goal}+p) \leq \alpha_{\rm goal}$, we obtain

$$\alpha_{n^*+1} = q\alpha_{n^*} = q(\alpha_{n^*-1}+p) \leq q(\alpha_{\rm goal}+p) \leq \alpha_{\rm goal}. \tag{52}$$

Thus, $Q_{\max}(\alpha_{n^*+1}) = Q_{\max}(q\alpha_{n^*}) \leq Q_{\max}(\alpha_{\rm goal}) = Q_{\rm goal}$, which is (51). By (51), $\mathrm{BCI}(n^*, n^*+1) = (1,0)$. Applying α-control, we get $\alpha_{n^*+2} = \alpha_{n^*+1}/q$. But $\alpha_{n^*+1} = q\alpha_{n^*}$, giving $\alpha_{n^*+2} = q\alpha_{n^*}/q = \alpha_{n^*} > \alpha_{\rm goal}$; thus, $\mathrm{BCI}(n^*+1, n^*+2) = \mathrm{BCI}(0,1)$. Applying α-control again, $\alpha_{n^*+3} = q\alpha_{n^*+2} = q\alpha_{n^*} = \alpha_{n^*+1}$. But by (52), $\alpha_{n^*+3} = q\alpha_{n^*} \leq \alpha_{\rm goal}$, and thus $\mathrm{BCI}(n^*+2, n^*+3) = (1,0)$. Repeating the above procedure, we get $\forall k \in \{0,1,2,\ldots,\}$

$$\begin{cases} \alpha_{n^*+(2k+1)} = \alpha_{n^*+1} = q\alpha_{n^*} \leq \alpha_{\rm goal} \\ \alpha_{n^*+2k} = \alpha_{n^*} > \alpha_{\rm goal} \end{cases} \tag{53}$$

implying that $\mathrm{BCI}(0,1,2,3,\ldots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\ldots) = (0,0,0,0,\ldots,0,1,0,1,0,\ldots)$. By Definition 4, $Q_{\max}^{(n)}$ monotonically converges to $Q_{\rm goal}$'s neighborhood $\{Q_{\rm goal}^l, Q_{\rm goal}^h\}$. In addition, in the equilibrium state

$$\begin{cases} Q_{\max}(q\alpha_{n^*}) = Q_{\max}(q(n^*p+\alpha_0)) \\ \quad = \max_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} \leq Q_{\rm goal} \right\} \\ Q_{\max}(\alpha_{n^*}) = Q_{\max}(n^*p+\alpha_0) \\ \quad = \min_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} > Q_{\rm goal} \right\}. \end{cases} \tag{54}$$

Thus, by Definition 3, $Q_{\rm goal}^l = Q_{\max}(q(n^*p+\alpha_0))$ and $Q_{\rm goal}^h = Q_{\max}(n^*p+\alpha_0)$.

Case 2) $\alpha_0 > \alpha_{\rm goal}$: Since $Q_{\max}^{(0)} = Q_{\max}(\alpha_0) > Q_{\rm goal} = Q_{\max}(\alpha_{\rm goal})$, applying the α-control with a factor q $(0<q<1)$, $Q_{\max}^{(n)}$ monotonically decreases from $Q_{\max}^{(0)}$ toward $Q_{\rm goal}$. When $Q_{\max}^{(n)} \leq Q_{\rm goal}$ for the first time at $n = n^*$, i.e., $q^{n^*}\alpha_0 = \alpha_{n^*} \leq \alpha_{\rm goal}$, the source detects $\mathrm{BCI}(n^*-1, n^*) = (1,0)$. Applying α-control, we get $\alpha_{n^*+1} = \alpha_{n^*}/q = \alpha_{n^*-1} > \alpha_{\rm goal}$, and thus $\mathrm{BCI}(n^*, n^*+1) = (0,1)$. By α-control, $\alpha_{n^*+2} = q\alpha_{n^*+1} = q(\alpha_{n^*}/q) = \alpha_{n^*} \leq \alpha_{\rm goal}$, and hence $\mathrm{BCI}(n^*+1, n^*+2) = (1,0)$. Applying α-control again, we get $\alpha_{n^*+3} = \alpha_{n^*+2}/q = \alpha_{n^*+1} > \alpha_{\rm goal}$, and thus $\mathrm{BCI}(n^*+2, n^*+3) = (0,1)$. Repeat the above deducing procedure, we have $\forall k \in \{0,1,2,\ldots,\}$

$$\begin{cases} \alpha_{n^*+(2k+1)} = \alpha_{n^*+1} = \alpha_{n^*}/q > \alpha_{\rm goal} \\ \alpha_{n^*+2k} = \alpha_{n^*} \leq \alpha_{\rm goal} \end{cases} \tag{55}$$

implying that $\mathrm{BCI}(0,1,2,3,\ldots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\ldots) = (1,1,1,1,\ldots,1,0,1,0,1,\ldots)$. Therefore, by Definition 4, $Q_{\max}^{(n)}$ monotonically converges to $Q_{\rm goal}$'s neighborhood $\{Q_{\rm goal}^l, Q_{\rm goal}^h\}$. In addition, in the equilibrium state

$$\begin{cases} Q_{\max}(\alpha_{n^*}) = Q_{\max}(q^{n^*}\alpha_0) \\ \quad = \max_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} \leq Q_{\rm goal} \right\} \\ Q_{\max}(\alpha_{n^*}/q) = Q_{\max}\left(q^{(n^*-1)}\alpha_0\right) \\ \quad = \min_{n\in\{0,1,2,\ldots\}} \left\{ Q_{\max}^{(n)} \,\middle|\, Q_{\max}^{(n)} > Q_{\rm goal} \right\}. \end{cases} \tag{56}$$

Thus, by Definition 3, $Q_{\rm goal}^l = Q_{\max}(q^{n^*}\alpha_0)$ and $Q_{\rm goal}^h = Q_{\max}(q^{(n^*-1)}\alpha_0)$.

<u>Claim 2:</u> Since $p \leq ((1-q)/q)((\sqrt{Q_{\rm goal}} - \sqrt{2Q_h})/\tau)^2$ and $0<q<1$, by Claim 1 of Theorem 3, $Q_{\max}^{(n)}$ is guaranteed to converge to $Q_{\rm goal}$'s neighborhood in the equilibrium state. Define maximum-queue-length upper-bound error function for $(\alpha,\tau) \in \mathcal{F}$ by

$$\begin{aligned} \gamma(\alpha,\tau) &\triangleq \zeta(\tau\sqrt{\alpha}) - Q_{\max}(\alpha,\tau) \\ &= (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2 - Q_{\max}(\alpha,\tau), \quad (\alpha,\tau)\in\mathcal{F} \end{aligned} \tag{57}$$

which is a nonnegative real-valued function since $Q_{\max}(\alpha,\tau) \leq \zeta(\tau\sqrt{\alpha})$. According to Lemma 1, see Appendix E, which is also verified in Fig. 2, and because $\alpha_{\rm goal}^h \geq \alpha_{\rm goal}$, we have $\gamma(\alpha_{\rm goal}^h,\tau) - \gamma(\alpha_{\rm goal},\tau) \geq 0$, leading to

$$\begin{aligned} &Q_{\rm goal}^h - Q_{\rm goal} \\ &\quad \leq \left[Q_{\rm goal}^h - Q_{\rm goal}\right] + \left[\gamma\left(\alpha_{\rm goal}^h,\tau\right) - \gamma(\alpha_{\rm goal},\tau)\right] \\ &\quad = Q_{\rm goal}^h - Q_{\rm goal} + \left[\left(\tau\sqrt{\alpha_{\rm goal}^h} + \sqrt{2Q_h}\right)^2 \right. \\ &\qquad \left. - Q_{\rm goal}^h\right] - \left[(\tau\sqrt{\alpha_{\rm goal}} + \sqrt{2Q_h})^2 - Q_{\rm goal}\right] \\ &\quad = \tau^2\left(\alpha_{\rm goal}^h - \alpha_{\rm goal}\right) + \tau\sqrt{8Q_h}\left(\sqrt{\alpha_{\rm goal}^h} - \sqrt{\alpha_{\rm goal}}\right) \\ &\quad \leq \tau^2\left(\frac{1}{q}\alpha_{\rm goal} - \alpha_{\rm goal}\right) + \tau\sqrt{8Q_h} \\ &\qquad \cdot \left(\sqrt{\frac{1}{q}\alpha_{\rm goal}} - \sqrt{\alpha_{\rm goal}}\right) \end{aligned} \tag{58}$$

$$= \tau^2\alpha_{\rm goal}\left(\frac{1}{q} - 1\right) + \tau\sqrt{8\alpha_{\rm goal}Q_h}\left(\frac{1}{\sqrt{q}} - 1\right) \tag{59}$$

where (58) is due to the inequality $\alpha_{\rm goal}^h \leq (1/q)\alpha_{\rm goal}$ that resulted from the α-control law. This proves (10). Likewise, because $\alpha_{\rm goal} \geq \alpha_{\rm goal}^l$, which results in $\gamma(\alpha_{\rm goal},\tau) - \gamma(\alpha_{\rm goal}^l,\tau) \geq 0$ due to Lemma 1 (see Appendix E), we obtain

$$\begin{aligned} &Q_{\rm goal} - Q_{\rm goal}^l \\ &\quad \leq \left[Q_{\rm goal} - Q_{\rm goal}^l\right] + \left[\gamma(\alpha_{\rm goal},\tau) - \gamma\left(\alpha_{\rm goal}^l,\tau\right)\right] \\ &\quad = Q_{\rm goal} - Q_{\rm goal}^l + \left[(\tau\sqrt{\alpha_{\rm goal}} + \sqrt{2Q_h})^2\right. \\ &\qquad \left. - Q_{\rm goal}\right] - \left[\left(\tau\sqrt{\alpha_{\rm goal}^l} + \sqrt{2Q_h}\right)^2 - Q_{\rm goal}^l\right] \\ &\quad = \tau^2\left(\alpha_{\rm goal} - \alpha_{\rm goal}^l\right) + \tau\sqrt{8Q_h} \\ &\qquad \cdot \left(\sqrt{\alpha_{\rm goal}} - \sqrt{\alpha_{\rm goal}^l}\right) \\ &\quad \leq \tau^2\left(\alpha_{\rm goal} - q\alpha_{\rm goal}\right) + \tau\sqrt{8Q_h} \\ &\qquad \cdot (\sqrt{\alpha_{\rm goal}} - \sqrt{q\alpha_{\rm goal}}) \end{aligned} \tag{60}$$

$$= \tau^2\alpha_{\rm goal}(1-q) + \tau\sqrt{8\alpha_{\rm goal}Q_h}(1-\sqrt{q}) \tag{61}$$

where (60) is due to the fact that $\alpha_{\rm goal}^l \geq q\alpha_{\rm goal}$ resulting from the α-control law. This proves (11). Adding both sides of (59) and those of (61), (12) follows. ■

APPENDIX E
MAXIMUM QUEUE-LENGTH UPPER-BOUND ERROR FUNCTION MONOTONICITY LEMMA

Lemma 1: The maximum queue length upper-bound error function $\gamma(\alpha,\tau) = \zeta(\tau\sqrt{\alpha}) - Q_{\max}(\alpha,\tau) = (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2 - Q_{\max}(\alpha,\tau)$ defined in (57), is a **strictly monotonic-increasing** function of α for $\forall\alpha > 0$ and $(\alpha,\tau) \in \mathcal{F}$.

Proof: Since $\gamma(\alpha,\tau)$ is defined for $(\alpha,\tau) \in \mathcal{F}$, we only need to consider $(\alpha,\tau) \in \mathcal{F} \subset \Omega$, where $\gamma(\alpha,\tau)$ is differentiable, and thus we can take the partial derivative on α as follows

$$\frac{\partial\gamma(\alpha,\tau)}{\partial\alpha} = \frac{\partial\zeta(\alpha,\tau)}{\partial\alpha} - \frac{\partial Q_{\max}(\alpha,\tau)}{\partial\alpha} \tag{62}$$

where

$$\frac{\partial\zeta(\alpha,\tau)}{\partial\alpha} = \tau^2 + \tau\sqrt{\frac{2Q_h}{\alpha}} \tag{63}$$

$$\begin{aligned}\frac{\partial Q_{\max}(\alpha,\tau)}{\partial\alpha} &= \frac{1}{2}\tau^2 + \tau\sqrt{\frac{2Q_h}{\alpha}} - \tau\sqrt{\frac{Q_h}{2\alpha}} - \mu\sqrt{\frac{Q_h}{2}}\alpha^{-\frac{3}{2}} \\ &\quad + \frac{\mu^2}{\alpha^2}\log\left[1 + \frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right] \\ &\quad - \frac{\mu^2\left(\tau + \sqrt{\frac{Q_h}{2\alpha}}\right)}{\mu\alpha + \alpha^2\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)}.\end{aligned} \tag{64}$$

Note that again, we use the fact that $\mu = (\Delta\alpha)/(1-\beta)$ in derivations of $\partial Q_{\max}(\alpha,\tau)/\partial\alpha$ in (64). Thus, we obtain

$$\begin{aligned}\frac{\partial\gamma(\alpha,\tau)}{\partial\alpha} &= \frac{\tau^2}{2} + \tau\sqrt{\frac{Q_h}{2\alpha}} + \mu\sqrt{\frac{Q_h}{2}}\alpha^{-\frac{3}{2}} \\ &\quad - \frac{\mu^2}{\alpha^2}\log\left[1 + \frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right] \\ &\quad + \frac{\mu^2\left(\tau + \sqrt{\frac{Q_h}{2\alpha}}\right)}{\mu\alpha + \alpha^2\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)}.\end{aligned} \tag{65}$$

Using (65), we define a new real-valued function $\varphi(\alpha,\tau)$

$$\begin{aligned}\varphi(\alpha,\tau) &\triangleq \left(\frac{\alpha^2}{\mu^2}\right)\frac{\partial\gamma(\alpha,\tau)}{\partial\alpha} \\ &= \frac{1}{2}\left(\frac{\tau\alpha}{\mu}\right)^2 + \frac{\tau}{\mu^2}\sqrt{\frac{Q_h}{2}}\alpha^{\frac{3}{2}} + \frac{1}{\mu}\sqrt{\frac{Q_h\alpha}{2}} \\ &\quad - \log\frac{\mu + \tau\alpha + \sqrt{2Q_h\alpha}}{\mu} + \left(\tau + \sqrt{\frac{Q_h}{2\alpha}}\right) \\ &\quad \cdot \frac{\alpha}{\mu + \alpha\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)}.\end{aligned} \tag{66}$$

Taking the partial derivative on α over both sides of (66), we obtain

$$\begin{aligned}\frac{\partial\varphi(\alpha,\tau)}{\partial\alpha} &= \frac{1}{\left(\mu + \sqrt{2Q_h\alpha} + \tau\alpha\right)^2}\left[\frac{Q_h^{\frac{3}{2}}}{\mu}\sqrt{\frac{\alpha}{2}}\right. \\ &\quad + \frac{4\tau Q_h}{\mu}\alpha + \alpha^{\frac{3}{2}}\left(\frac{2\tau^2}{\mu}\sqrt{2Q_h} + \frac{3\tau^3}{\mu}\sqrt{\frac{Q_h}{2}}\right. \\ &\quad \left. + \frac{3\tau Q_h^{\frac{3}{2}}}{\sqrt{2}\mu^2} + \frac{\tau^2}{2\mu}\sqrt{\frac{Q_h}{2}}\right) \\ &\quad + \alpha^2\left(\frac{2\tau^3}{\mu} + \frac{5\tau^2 Q_h}{\mu^2}\right) + \alpha^{\frac{5}{2}} \\ &\quad \left. \cdot \left(\frac{2\tau^3}{\mu^2}\sqrt{2Q_h} + \frac{3\tau^3}{2\mu^2}\sqrt{\frac{Q_h}{2}}\right) + \frac{\tau^4}{\mu^2}\alpha^3\right] > 0.\end{aligned} \tag{67}$$

That is, (67) proves the following:

$$\frac{\partial\varphi(\alpha,\tau)}{\partial\alpha} > 0, \quad \forall\alpha > 0, \quad (\alpha,\tau) \in \mathcal{F} \tag{68}$$

which implies that $\varphi(\alpha,\tau)$ is a strictly monotonic-increasing function with respect to $\alpha, \forall\alpha > 0$ and $(\alpha,\tau) \in \mathcal{F}$. Notice

$$\varphi(\alpha,\tau)\mid_{\alpha=0} = 0. \tag{69}$$

Combining (68) and (69), it follows that $\varphi(\alpha,\tau) > 0, \forall\alpha > 0$ and $(\alpha,\tau) \in \mathcal{F}$, and that is for $\forall\alpha > 0$ and $(\alpha,\tau) \in \mathcal{F}$,

$$\varphi(\alpha,\tau) = \left(\frac{\alpha^2}{\mu^2}\right)\frac{\partial\gamma(\alpha,\tau)}{\partial\alpha} > 0. \tag{70}$$

Reducing (70), we obtain

$$\frac{\partial\gamma(\alpha,\tau)}{\partial\alpha} > 0, \quad \forall\alpha > 0 \quad \text{and} \quad (\alpha,\tau) \in \mathcal{F} \tag{71}$$

which completes the proof. ■

APPENDIX F
PROOF OF THEOREM 4

Proof: We also need to prove this theorem by considering the following two cases, which correspond to the first and second parts of (23), respectively.

Case 1) $\alpha_0 > \widetilde{\alpha_{\rm goal}}$: Let $\widetilde{\alpha_{\rm goal}^l}$ correspond to the new $\widetilde{Q_{\rm goal}^l} = Q_{\max}(\widetilde{\alpha_{\rm goal}^l})$. By (9), we have $\widetilde{\alpha_{\rm goal}^l} = q^{n^*}\alpha_0$, leading to

$$n^* = \frac{\log\frac{\alpha_0}{\widetilde{\alpha_{\rm goal}^l}}}{\log\frac{1}{q}} \geq \frac{\log\frac{\alpha_0}{\widetilde{\alpha_{\rm goal}}}}{\log\frac{1}{q}} = \frac{\log\frac{\widetilde{\alpha_{\rm goal}}}{\alpha_0}}{\log q} \tag{72}$$

where the inequality in (72) is due to $\widetilde{\alpha_{\rm goal}} \geq \widetilde{\alpha_{\rm goal}^l}$. But since $q\widetilde{\alpha_{\rm goal}} < \widetilde{\alpha_{\rm goal}^l}$, that is

$$\frac{\log\frac{\alpha_0}{\widetilde{\alpha_{\rm goal}^l}}}{\log\frac{1}{q}} - \frac{\log\frac{\alpha_0}{\widetilde{\alpha_{\rm goal}}}}{\log\frac{1}{q}} < 1 \tag{73}$$

we have

$$\frac{\log\frac{\widetilde{\alpha_{\rm goal}}}{\alpha_0}}{\log q} \leq n^* = \frac{\log\frac{\alpha_0}{\widetilde{\alpha_{\rm goal}^l}}}{\log\frac{1}{q}} < 1 + \frac{\log\frac{\widetilde{\alpha_{\rm goal}}}{\alpha_0}}{\log q} \tag{74}$$

which implies $n^* = \lceil \log(\widetilde{\alpha_{\rm goal}}/\alpha_0)/\log q \rceil$, because n^* must be an integer. By Definition 4, $N = n^* - 1$ for $\alpha > \widetilde{\alpha_{\rm goal}}$, and thus

$$N = n^* - 1 = \left\lfloor \log\left(\frac{\widetilde{\alpha_{\rm goal}}}{\alpha_0}\right) \Big/ \log q \right\rfloor. \tag{75}$$

Case 2) $\alpha_0 \le \widetilde{\alpha_{\rm goal}}$: Let $\widetilde{\alpha_{\rm goal}^h}$ correspond to the new $\widetilde{Q_{\rm goal}^h} = Q_{\max}(\widetilde{\alpha_{\rm goal}^h})$. By (9), we get $\widetilde{\alpha_{\rm goal}^h} = n^* p + \alpha_0$, leading to

$$n^* = \frac{\widetilde{\alpha_{\rm goal}^h} - \alpha_0}{p} \ge \frac{\widetilde{\alpha_{\rm goal}} - \alpha_0}{p} \tag{76}$$

where the inequality in (76) is due to $\widetilde{\alpha_{\rm goal}} \le \widetilde{\alpha_{\rm goal}^h}$. Since $\widetilde{\alpha_{\rm goal}^h} - \widetilde{\alpha_{\rm goal}} < p$, i.e., $(\widetilde{\alpha_{\rm goal}^h} - \alpha_0)/p - (\widetilde{\alpha_{\rm goal}} - \alpha_0)/p < 1$, we have

$$\frac{\widetilde{\alpha_{\rm goal}} - \alpha_0}{p} \le n^* = \frac{\widetilde{\alpha_{\rm goal}^h} - \alpha_0}{p} < 1 + \frac{\widetilde{\alpha_{\rm goal}} - \alpha_0}{p} \tag{77}$$

implying $n^* = \lceil (\widetilde{\alpha_{\rm goal}} - \alpha_0)/p \rceil$, because n^* must be an integer. By Definition 4, $N = n^*$ for $\alpha \le \widetilde{\alpha_{\rm goal}}$, and thus

$$N = n^* = \lceil (\widetilde{\alpha_{\rm goal}} - \alpha_0)/p \rceil. \tag{78}$$

Since $\widetilde{\alpha_{\rm goal}}$ corresponds to $Q_{\rm goal} = Q_{\max}(\widetilde{\alpha_{\rm goal}})$, we can solve (42) for $\widetilde{\alpha_{\rm goal}}$ by letting $Q_{\max} = Q_{\rm goal}$ and $\alpha(\Delta/(1-\beta)) = \mu$, which yields (24). Since $Q_{\rm goal}$ is small, implying $\widetilde{\alpha_{\rm goal}}$ is small, the lower-bound function $\tau\sqrt{\alpha} = \sqrt{C_{\max}} - \sqrt{2Q_h}$ given in Theorem 2 is tight, we can use

$$Q_{\max}(\alpha, \tau) \approx (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2 \tag{79}$$

to estimate $Q_{\max}$ as discussed in **(2)** (about Claim 2) of **Remarks on Theorem 2**. Substituting α, τ, and $Q_{\max}(\alpha, \tau)$ by $\widetilde{\alpha_{\rm goal}}, \widetilde{\tau}$, and $Q_{\rm goal}$ in (79), respectively, yields (25). Hence the proof follows. ■

ACKNOWLEDGMENT

The authors would like to thank Prof. S. M. Meerkov at the University of Michigan for pointing out an error in Theorem 1 in an earlier version of this paper, Dr. W. T. Ng for his assistance in simulations, and the anonymous reviewers of this TRANSACTIONS and the IEEE INFOCOM'99 for their constructive comments.

REFERENCES

[1] L. Roberts, "Rate based algorithm for point to multipoint ABR service," ATM Forum, Contribution 94-0772, Sept. 1994.

[2] K.-Y. Siu and H.-Y. Tzeng, "On max-min fair congestion control for multicast ABR services in ATM," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 545–556, Apr. 1997.

[3] H. Saito, K. Kawashima, H. Kitazume, A. Koike, M. Ishizuka, and A. Abe, "Performance issues in public ABR service," *IEEE Commun. Mag.*, pp. 40–48, Nov. 1996.

[4] X. Zhang and K. G. Shin, "Statistical analysis of feedback synchronization signaling delay for multicast flow control," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 1152–1161.

[5] Y.-Z. Cho, S.-M. Lee, and M.-Y. Lee, "An efficient rate-based algorithm for point-to-multipoint ABR service," in *Proc. IEEE GLOBECOM*, Nov. 1997, pp. 790–795.

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

[7] S. Fahmy, R. Jain, R. Goyal, B. Vandalor, and S. Kalyanaraman, "Feedbackback consolidation algorithms for ABR point-to-multipoint connections in ATM networks," in *Proc. IEEE INFOCOM*, Apr. 1998, pp. 1004–1013.

[8] X. Zhang and K. G. Shin, "Performance analysis of feedback synchronization for multicast ABR flow control," in *Proc. IEEE GLOBECOM*, Dec. 1999, pp. 1269–1274.

[9] D. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks ISDN Syst.*, pp. 1–14, 1989.

[10] S. Sathaye, "ATM Forum Traffic Management Specifications Version 4.0," ATM Forum, Contribution 95-0013R7.1, Aug. 1995.

[11] X. Zhang and K. G. Shin, "Second-order rate-control based transport protocols," in *Proc. IEEE Int. Conf. Network Protocols*, Nov. 2001, pp. 342–350.

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

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

[14] F. Bonomi, D. Mitra, and J. Seery, "Adaptive algorithms for feedback-based flow control in high-speed, wide-area ATM networks," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1267–1283, Sept. 1995.

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

[16] J. Bolot and A. Shankar, "Dynamical behavior of rate-based flow control mechanism," in *ACM SIGCOMM Computer Communication Review*, vol. 20, Apr. 1990, pp. 35–49.

[17] X. Zhang, K. G. Shin, D. Saha, and D. Kandlur, "Scalable flow control for multicast ABR services," in *Proc. IEEE INFOCOM*, Mar. 1999, pp. 837–846.

[18] M. Ritter, "Network buffer requirements of the rate-based control mechanism for ABR services," in *Proc. IEEE INFOCOM*, Mar. 1996, pp. 1190–1197.

[19] A. Heybey, "The Network Simulator," Laboratory for Computer Science, Massachusetts Inst. of Technology, Cambridge, MA, Sept. 1990.

Xi Zhang (S'89–SM'98) received the B.S. and M.S. degrees from Xidian University, Xi'an, China, and the M.S. degree from Lehigh University, Bethlehem, PA, all in electrical engineering and computer science. He received the Ph.D. degree in electrical engineering and computer science (EE-Systems) from the University of Michigan, Ann Arbor.

He is currently an Assistant Professor in the Department of Electrical Engineering, Texas A&M University, College Station. He was an Assistant Professor and Founding Director of the Division of Computer Systems Engineering, Department of Electrical Engineering and Computer Science, Beijing Information Technology Engineering Institute, Beijing, China, from 1984 to 1989. He was a Research Fellow with the School of Electrical Engineering, University of Technology, Sydney, Australia, and the Department of Electrical and Computer Engineering, James Cook University, Queensland, Australia, under a fellowship from the Chinese National Commission of Education. He worked as a Summer Intern in the Networks and Distributed Systems Research Department, Bell Labs, Murray Hills, NJ, and at AT&T Labs Research, Florham Park, NJ, in 1997. His current research interests focus on the design, modeling, and performance analysis of protocols and systems for multicast and unicast computer communications over wired and wireless networks supporting QoS guarantees, statistical communication theory and random signal processing, and distributed computer-control systems. He has authored and coauthored more than 30 technical papers.

Dr. Zhang is a member of the Association for Computing Machinery.

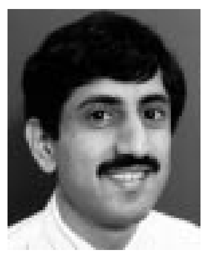
Kang G. Shin (S'75–M'78–SM'83–F'92) received the B.S. degree in electronics engineering from Seoul National University, Seoul, Korea, in 1970, and the M.S. and Ph.D degrees in electrical engineering from Cornell University, Ithaca, NY, in 1976 and 1978, respectively.

He is the Kevin and Nancy O'Connor Chair Professor of Computer Science and Founding Director of the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor. From 1978 to 1982, he was on the faculty of Rensselaer Polytechnic Institute, Troy, NY. He has held visiting positions at the U.S. Airforce Flight Dynamics Laboratory, AT&T Bell Laboratories, the Computer Science Division, Department of Electrical Engineering and Computer Science, University of California at Berkeley, the International Computer Science Institute, Berkeley, CA, the IBM T. J. Watson Research Center, and the Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA. He also chaired the Computer Science and Engineering Division, Electrical Engineering and Computer Science Department, University of Michigan, for three years beginning in January 1991. His current research focuses on QoS-sensitive networking and computing as well as on embedded real-time OS, middleware and applications, all with emphasis on timeliness and dependability. He has supervised the completion of 42 Ph.D. theses, and authored or coauthored over 600 technical papers and numerous book chapters in the areas of distributed real-time computing and control, computer networking, fault-tolerant computing, and intelligent manufacturing. He has co-authored (with C. M. Krishna) the textbook *Real-Time Systems* (New York: McGraw-Hill, 1997).

Dr. Shin is a Fellow of the IEEE and the Association for Computing Machinery, and a member of the Korean Academy of Engineering. He was the General Chair of the 2000 IEEE Real-Time Technology and Applications Symposium, the Program Chair of the 1986 IEEE Real-Time Systems Symposium (RTSS), the General Chair of the 1987 RTSS, the Guest Editor of the 1987 August special issue of IEEE TRANSACTIONS ON COMPUTERS on Real-Time Systems, a Program Co-Chair for the 1992 *International Conference on Parallel Processing*, and has served on numerous technical program committees. He also chaired the IEEE Technical Committee on Real-Time Systems during 1991–1993, was a Distinguished Visitor of the Computer Society of the IEEE, an Editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED COMPUTING, and is currently an Editor of ACM TRANSACTIONS ON EMBEDDED COMPUTING SYSTEMS, and an Area Editor of INTERNATIONAL JOURNAL OF TIME-CRITICAL COMPUTING SYSTEMS, and COMPUTER NETWORKS. He received the Outstanding IEEE TRANSACTIONS ON AUTOMATIC CONTROL Paper Award in 1987, the Research Excellence Award in 1989, the Outstanding Achievement Award in 1999, the Service Excellence Award in 2000, and the Distinguished Faculty Achievement Award in 2001 from the University of Michigan. He also coauthored papers with his students which received the Best Student Paper Awards from the 1996 IEEE Real-Time Technology and Application Symposium and the 2000 UNSENIX Technical Conference.

Debanjan Saha received the B.Tech. degree from I.I.T., India, and the M.S. and Ph.D. degrees from the University of Maryland, College Park, all in computer sciences.

He is the Manager of Internetworking and Advanced Applications at Tellium, Inc., Oceanport, NJ. Previous to his tenure at Tellium, he spent several years at IBM Research and Lucent Bell Labs, where he led design and development of protocols for IP routers and multi-service switches. He is actively involved with various standards bodies, most notably IETF and OIF. He also serves as Editor of international journals and magazines, and Technical Committee Member of workshops and conferences. He is a noted author of numerous technical articles on various topics of networking and is a frequent speaker at academic and industrial events.



Dilip D. Kandlur (M'91) received the M.S.E. and Ph.D. degrees in computer science and engineering from the University of Michigan, Ann Arbor.

He heads the Networking Software and Services Department at the IBM T.J. Watson Research Center, Hawthorne, NY. Since joining the IBM T.J. Watson Research Center, his research work has covered various aspects of providing quality of service in hosts and networks and their application to multimedia systems, network and server performance, web caching, etc. In particular, he has worked on protocols and architectures for providing quality of service in IP and ATM networks, their impact on transport protocols, and their realization in protocol stacks for large servers and switch/routers. He holds ten U.S. patents and has been recognized as an IBM Master Inventor. He is a member of the IEEE Computer Society and currently is Vice-Chair of the IEEE Technical Committee on Computer Communications.

Dr. Kandlur has been awarded an Outstanding Technical Achievement Award for his work in creating the QoS architecture for IBM server platforms.