A Scalable Flow-Control Algorithm for Point-to-Multipoint Communications in High-Speed Integrated Networks

Xi Zhang and Kang G. Shin

Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109–2122
E-mail: {xizhang,kgshin}@eecs.umich.edu

ABSTRACT

We propose and study a scalable flow-control algorithm for multicast ABR (Available Bit Rate) service. A key idea behind the algorithm is the *Soft-Synchronization Protocol* (SSP), which derives a single "consolidated" RM (Resource Management) cell at each multicast branch-point from feedback RM-cells of different downstream branches that are not necessarily responses to the same forward RM-cell in each synchronization cycle. Using balanced and unbalanced binary-tree models, we analyze the scalability of SSP in terms of height and structure of the multicast tree. In contrast with the other existing schemes, SSP is shown to be able to effectively support synchronization of feedback RM-cells and make the effective RM-cell roundtrip delay virtually independent of the multicast-tree's height and structure, thus making it scalable.

Another key problem in multicast flow-control is how to deal with the variation of RM-cell roundtrip delay due to the randomly drifting bottleneck in a multicast tree. This problem is handled by adapting the second-order rate-control parameter to roundtrip-delay variations. Using a fluid model, we derive an analytical relationship between the second-order rate parameter and RM-cell roundtrip delay subject to both lossless transmission and finite buffer capacity constraints. This relationship ensures the feasibility of the second-order rate control in dealing with RM-cell roundtrip-delay variations and provides an insight on how the required buffer space can be controlled by adjusting the rate parameter. We develop an optimal control condition, under which the second-order rate control guarantees monotonic convergence of the system state to the optimal regime from an arbitrary initial value. The proposed second-order rate-control algorithm is also shown to be feasible and optimal in buffer-allocation efficiency and fairness at the bottleneck.

Index Terms — Multicast flow control, scalable algorithms, ABR service, ATM networks, feedback-based flow control, rate-based flow control, lossless transmission.

The work reported in this paper was supported by the U.S. Office of Naval Research under Grant No. N00014-94-0229. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agencies.

1 Introduction

Multicast ABR service is an increasingly important class of service in ATM (Asynchronous Transfer Mode) networks, and has a wide range of applications in distributed computation, group data/news-broadcasting, and LAN emulation. Conceptually, a multicast connection forms a multicast tree, in which a *single* copy of data is sent by the sender at the root, and the data copies are made at all branch-switches while the data being forwarded towards a group of receivers at the leaf nodes. The data generated by the sender moves through the multicast tree, traversing each tree edge only once. How to distribute the data to each member of the multicast group is transparent to the sender. When receivers join or leave a multicast group, the multicast tree is dynamically reconfigured.

Congestion control plays a critically-important role in providing high-quality services for diverse traffic through high-speed ATM networks. Supporting multicast ABR service poses a number of new challenges unseen in the unicast ABR context. One of the major problems, especially in large multicast trees, is commonly known as the *feedback implosion* problem [1]. Since our goal is to adjust the source transmission rate to match the bottleneck link bandwidth, the source needs to collect congestion information all branches in the multicast tree. Simultaneous congestion feedbacks from all branches can cause an implosion at the source. Hence, it is important to consolidate the congestion feedback at each branch-point and only the consolidated feedback is forwarded upward. Consolidation requires synchronization of feedbacks from all downstream branches at each branch-point. Since different branches may have different roundtrip delays, receiver-returned feedbacks may arrive at the branch-point at significantly different times. If a branch-point switch waits for feedbacks from all of its multicast-tree specified downstream branches, it may have to wait a long time, thus resulting in a long feedback delay. On the other hand, if the branch-point switch forwards an early feedback upstream without waiting for feedbacks from all of its downstream branches, the source may receive incomplete/incorrect information.

Another important but subtle problem associated with multicast flow control is that the bottleneck may drift from one path to another within a multicast tree. As a result, the roundtrip delay in the bottleneck path may change significantly. Since the roundtrip delay plays a critical role in determining the effectiveness of any feedback flow-control scheme, it is important to handle such dynamic drifting of the bottleneck. The flow-control scheme should also be able to detect and remove non-responsive or over-congested branches in order to prevent them from stalling the entire multicast connection.

Although the literature on unicast ABR is extremely rich, the results on multicast ABR flow control is quite limited. Roberts [2,3] proposed a multicast flow-control scheme which is based on EPRCA (Explicit Proportional Rate Control Algorithm) and extends unicast to multicast operations. Using a similar technique, the authors of [4–6] established a framework for extending an existing unicast congestion-control protocol to a multicast environment. Their schemes are simple and easy to migrate from a unicast environment to a multicast environment. They employ simple "hop-by-hop feedback" to deal with the feedback implosion at a branch-point switch, and retain the same source/destination control algorithms as in the unicast. The pseudocode of this scheme [2–6] is given in Figure 1, focusing on RM-cell processing. In this scheme, each switch maintains a register MER (Minimum Explicit Rate) and a flag MCI (Multicast Congestion Indication) for each multicast connection. This scheme ensures exactly one feedback RM-cell to be returned to the source for each forward RM-cell, and contains the minimum ER (Explicit Rate) and CI (Congestion Indication) information of all participating branches during the last RM-cell update interval. At each branch-point, a consolidated feedback RM-cell is generated and sent upstream a *single hop* upon receiving a forward RM-cell, thus marching "hop-by-hop" towards the root. All of the schemes in [2–6] are of this type.

```
If switch receives an RM(ACR, DIR = forward, ER, CI) cell:
    Multicast this RM cell to all participating branches;
    MXR:=ER, MXI:=CI;        ! MXR and MXI are temorary variables used to update MER and MCI
    Send RM (ACR, DIR:=feedback, ER:=MER, CI:=MCI) to the root node;
    MCI := MXI, and MER := MXR;
If switch receives an RM(ACR, DIR = feedback, ER, CI) cell:
    MCI:=OR(MCI, CI);        MER:=min(MER, ER);        Discard this RM cell;
```

Figure 1: Pseudocode for the "hop-by-hop feedback" algorithm.

While the hop-by-hop feedback scheme is effective in eliminating the feedback implosion problem and easy to implement by extending the existing unicast, it suffers from serious scalability and feedback inefficiency problems. First, the RM-cell roundtrip delay is large and proportional to path length in the multicast tree. Suppose the source sends an RM-cell once every N_{rm} data-cells in the multicast-tree of height m, then each feedback RM-cell will take mN_{rm} data-cell time units to reach the source from the leaf along the longest path since the feedback RM-cell is sent upstream only one hop upon arrival of a forward RM-cell. Thus, the hop-by-hop scheme does not scale well with the multicast-tree height. Second, this feedback is inefficient for the following reasons. For a multicast tree of m levels, the source needs to send m forward RM-cells for a feedback RM-cell to return to the source. Moreover, delaying any of these m forward RM-cells will directly affect the delay of the feedback RM-cell. Each branch-switch does not synchronize feedback RM-cells from different downstream branches. This could lead to incomplete/incorrect feedback information (ER and CI) to be consolidated in a feedback RM-cell. Third, the algorithms [2–6] based on hop-by-hop feedback cannot handle the variations in RM-cell roundtrip delay (resulting from the dynamic drifting of the bottleneck) because they apply the same set of rate-control parameters irrespective of the bottleneck location. However the roundtrip delay has a significant impact on maximum buffer requirement, average throughput, and buffer/bandwidth utilization. Fourth, the identification and removal of non-responsive[1] branches are not handled explicitly. If a branch does not respond for a long time or does not respond at all, then the hop-by-hop switch algorithm will keep using out-of-date feedback information, which could result in loss of a significant number of data-cells.

To overcome the above-mentioned problems, we propose a new multicast flow-control scheme, which can avoid the feedback implosion at the source while achieving excellent feedback efficiency through the "soft-synchronization" protocol (SSP). Specifically, the SSP derives a consolidated RM-cell at each branch-point from feedback RM-cells of different downstream branches that are not necessarily responses to the same forward RM-cell in each synchronization cycle. Using balanced/unbalanced binary-tree models, we analytically evaluate the SSP's scalability with respect to multicast-tree's height and structure. The analytical result shows that SSP can not only effectively provide feedback synchronization, but also reduce the RM-cell roundtrip delay to such low a level that it is virtually independent of the multicast-tree's height and structure, a sharp contrast with the hop-by-hop feedback scheme.

In order to cope with variations in RM-cell roundtrip delay, we have developed a second-order rate-control algorithm. More specifically, besides adapting the transmission rate based on congestion-information feedback, the source also adjusts the second-order parameters that determine the rate at which the transmission rate itself is adjusted. As a result of employing the second-order rate control, the maximum buffer requirement scales well with the dynamically-changing bottleneck-path length when the bottleneck location drifts from one path to another. Using a fluid model, we analyze the proposed scheme and study the system dynamics under a heavy load condition. We have developed an optimal control condition, under which the second-order rate control guarantees the monotonic convergence of system state to the optimal regime from

[1]A non-responsive branch may be either physically broken or over-congested, thus not returning any feedback for a very long time.

an arbitrary initial state. We also prove that the proposed second-order rate-control algorithm is feasible and optimal in convergence to buffer-allocation efficiency and fairness at the multicast-tree bottleneck.

The paper is organized as follows. Section 2 describes the proposed multicast flow-control scheme, and Section 3 establishes the system and control models for the proposed scheme, identifies the system controlling factors, and presents an analytical solution to the multicast-tree bottleneck dynamics. Section 4 analyzes the RM-cell roundtrip delays for both the proposed and other schemes using the balanced/unbalanced binary-tree models. The scalability of both schemes are compared numerically. Section 5 analyzes the relationship among the maximum buffer requirement, rate-control parameters, and RM-cell roundtrip delays. The second-order rate control is developed and its properties investigated in terms of efficiency and fairness. The paper concludes with Section 6.

2 The Proposed Scheme

As in the rate-based scheme, we also use the EFCI (Explicit Forward Congestion Indication) bit in data and RM cells to convey network-congestion information. A forward RM-cell is sent by the root (source) periodically or once every N_{rm} data-cells, and each leaf node (receiver) replies by returning to the source a feedback RM-cell with EFCI and ER (Explicit Rate) information. We redefine the RM-cell format [7] such that it contains both the cell-rate (first-order) and the rate-parameter (second-order) control information. More specifically, two new one-bit fields, BCI (Buffer Congestion Indication) and NMQ (New Maximum Queue), are defined. Our scheme classifies congestion into two types:

***bandwidth congestion*:** when the queue length $Q(t)$ at a switch exceeds a predetermined threshold Q_h. Under this condition the switch sets the local CI (Congestion Indication) bit to 1.

***buffer congestion*:** when the maximum queue length Q_{max} at a switch exceeds the target buffer occupancy Q_{goal}, ($Q_h < Q_{goal} < C_{max}$), where C_{max} is the buffer capacity. Under this condition, the switch sets the local BCI to 1.

When a branch-switch receives a feedback RM-cell from a downstream node, it consolidates the information on ER, CI, and BCI for the corresponding connection. The ER is set to the minimum of the ER computed by the branch-switch and the ERs received from the downstream nodes. The CI field associated with the connection state is set to 1 if the local CI is 1, or CI field in the RM-cell received from any one of the downstream nodes is 1. The BCI field associated with the connection state is computed exactly in the same way. Upon accumulating feedback information from all downstream branches, a *single* feedback RM-cell is generated with the consolidated congestion information and sent upward with CI and BCI fields set to their respective values in the connection state. Note that our algorithm allows branch-switches to consolidate feedback RM-cells that are *not* necessarily responses to the *same* forward RM-cell. This distinguishes our algorithm from both (i) "strict synchronization," where only the feedback RM-cells in response to the same forward RM-cell are consolidated, thus making the feedback delay determined by the longest path, and (ii) "no synchronization" at all [2–6], which may convey incomplete feedback information to the source, thus degrading feedback efficiency. In the proposed scheme, the feedback RM-cells are "softly synchronized" at each branch-point, providing fast, complete, and efficient feedback, and making the feedback delay scalable with the height and structure of multicast tree.

The proposed scheme can dynamically identify non-responsive downstream branches and remove them from the multicast connection. This is critical for multicast flow-control since non-responsive branches can stall, and even shut down, the entire multicast connection. In our scheme, the non-responsive branches are defined as those which are either over-congested or physically broken. Specifically, at each switch a responsive-branch state vector and a non-responsive timer are associated with each multicast connection. Each bit of the state vector corresponds to one of downstream branches of this multicast connection and the non-responsive timer contains a predetermined threshold. Whenever the switch receives a feedback

```
On receipt of an RM cell:
    if (LCI=1 ∧ CI=0) {                                  ! Buffer congestion control triggering condition
        if (BCI=1) {AIR := q × AIR};                      ! AIR (Additive Increase Rate) reduced multiplicatively
        elseif (BCI=0 ∧ LBCI=0) {AIR := p + AIR};         ! Increase AIR additively
        elseif (BCI=0 ∧ LBCI=1) {AIR := AIR/q};           ! BCI toggles from 1 to 0; stay around target AIR
        MDF := $e^{-AIR/BW\_EST}$;                        ! MDF (Multiplicative Decrease Factor) updating
        LNMQ := 1 };                                      ! Start a new measurement cycle
    if (CI=0) {ACR := ACR + AIR};                         ! Increase cell rate additively
    else {ACR := ACR × MDF};                              ! Decrease cell rate multiplicatively
    LCI := CI;        LBCI := BCI;                        ! Save CI value and BCI value for second-order control
```

Figure 2: The pseudocode for Source End System (SES).

RM-cell from a downstream branch, the bit of state vector is set to 1. Each time a forward RM-cell is received by the switch, the non-responsive timer is decreased by one. If feedback RM-cells are received from all downstream branches before the timer goes off, then a *fully-consolidated* RM-cell is generated and sent upward, and the timer is reset to the threshold. If the timer goes off before the switch receives feedback RM-cells from all downstream branches, then the non-responsive branches are removed from the multicast connection, and a *partially-consolidated* RM-cell is sent upward.

There are two rate-control modes at the source corresponding to the two types of congestion: bandwidth and buffer congestion. If the bandwidth congestion information with CI = 1 (or 0) is detected from a feedback RM-cell, then the cell rate is reduced multiplicatively (or increased additively) from its current value. The buffer congestion control is triggered when the source detects a transition from CI= 1 to CI= 0 (i.e., from a rate-decrease cycle to a rate-increase cycle). Depending on the BCI field, three different variations of this control is exercised by the source. If BCI in both the current and the last RM-cells received are 0, the rate-increase parameter is increased additively. When BCI toggles from 1 to 0, the rate-increase parameter is increased multiplicatively. If the current RM-cell has BCI set to 1, the rate-increase parameter is decreased multiplicatively. Each time when buffer congestion control is triggered, the source sets NMQ to 1 in the next forward RM-cell to "request" the switches to re-calculate Q_{max} for the next measurement-cycle.

2.1 The Source Algorithm

A pseudocode for the source control algorithm is presented in Figure 2. Upon receiving a feedback RM-cell, the source must first check if it is time to exercise the buffer-congestion (second-order) control. This algorithm is triggered when the source detects a transition from a rate-decrease cycle to a rate-increase cycle, that is, when LCI (local congestion indicator) is equal to 1, and the CI field in the RM-cell received is set to 0. In this phase, the rate-increase parameter is adjusted depending on the current value of the local BCI (LBCI) and the BCI field in the RM-cell received. As mentioned before, we consider three cases: (i) if BCI is set to 1 in the RM-cell received, the rate-increase parameter AIR (Additive Increase Rate) is decreased multiplicatively by a factor of q ($0 < q < 1$); (ii) if both LBCI and BCI are set to 0, the rate-increase parameter AIR is increased additively by a step of size $p > 0$, (iii) if LBCI = 1 and BCI = 0, AIR is increased multiplicatively by a factor of q. For all these three cases, the rate-decrease parameter MDF (Multiplicative Decrease Factor) is adjusted according to the estimated bottleneck bandwidth BW_EST. The local NMQ bit is marked and the BCI field is saved in LBCI. The source always exercises the cell-rate (first-order) control whenever an RM-cell is received. Using the same, or updated, rate-parameter, the source additively increases, or multiplicatively decreases, its ACR (Allowed Cell Rate) according to the CI field in the RM-cell received.

```
On receipt of a DATA cell:
  multicast DATA cell based on $conn\_patt\_vec$;      ! multicast data cell to all connected branches
  if ($data\_qu > Q_h$)  {$CI$ :=1;}                   ! 1) Bandwidth congestion ($EFCI$) control
  if ($data\_qu > Q_{max}$)  {$Q_{max}$ :=$data\_qu$;}  ! 2) update $Q_{max}$
  if ($Q_{max} > Q_{goal}$)  {$BCI$ :=1;}               ! 3) buffer congestion control
  else {$BCI$ :=0;}                                     ! 1), 2), 3) are applied to all connected branches
On receipt of a feedback RM cell  from $i-$th downstream branch:
  if ($conn\_patt\_vec(i) \neq 1$) {                    ! only process connected branch
    $resp\_branch\_vec(i)$ := 1;                        ! mark connected/responsive branch
    $MCI$:= $MCI \vee CI$;                              ! bandwidth-congestion indicator processing
    $MBCI$:= $MBCI \vee BCI$;                           ! buffer-congestion indicator processing
    $MER$:= min{$MER, ER$};                             ! ER information processing
    if ($conn\_patt\_vec \oplus resp\_branch\_vec = \underline{1}$) {   ! soft-synchronization
      send RM cell ($dir$ := $backward$, $ER$ := $\min_{resp-branches} MER$, $CI := \bigcup_{resp-branches} MCI$,
          $BCI := \bigcup_{resp-branches} MBCI$);      ! send fully consolidated RM cell upwards
      $no\_resp\_count$ := 0;   $MCI$ := 0;            ! reset no-responsive count and congestion indicator
      $resp\_branch\_vec$:= $\underline{0}$;}}          ! reset responsive branch vector
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 a new measurement cycle
  $no\_resp\_count$ := $no\_resp\_count$ + 1;          ! no-responsive branch checking
  if ($no\_resp\_count \geq N_{check}$){               ! there is a no-responsive branch
    $conn\_patt\_vec$ := $resp\_branch\_vec \oplus \underline{1}$;   ! update connection pattern vector
    if ($resp\_branch\_vec \neq \underline{0}$){        ! there is at least one responsive branch
      send RM cell ($dir$ := $backward$,$ER$ := $\min_{resp-branches} MER$, $CI := \bigcup_{resp-branches} MCI$,
          $BCI := \bigcup_{resp-branches} MBCI$);      ! send partially consolidated RM cell upwards
      $resp\_branch\_vec$ := $\underline{0}$;           ! reset responsive branch vector
      $MCI$ := 0;  $MER$ := $ER$;                       ! reset congestion control variables for RM cell
      $no\_resp\_count$:=0;}}                           ! reset no-responsive counter variable
On receipt of a connection request from $j-$th downstream branch:
  $conn\_patt\_vec(j)$ := 0;                            ! add/re-activate a branch in the established multicast connection
```

Figure 3: The pseudocode for Intermediate Switch System (ISS).

2.2 The Switch Algorithm

At the center of switch control algorithm is a pair of state-control vectors: (i) *conn_patt_vec*, the connection pattern vector where $conn\_patt\_vec(i)$=0 (1) indicates the i-th output port of the switch is (not) a downstream branch of the multicast connection. Thus, $conn\_patt\_vec(i)$=0 (1) implies that a data copy should (not) be sent to the i-th downstream branch and a feedback RM-cell is (not) expected from the i-th downstream branch; (ii) *resp_branch_vec*, the responsive branch vector which is reset to $\underline{0}$ initially and when a consolidated RM-cell is sent upward from the switch. $resp\_branch\_vec(i)$ is set to 1 if an feedback RM-cell is received from the i-th downstream branch. The lengths of these two state-control vectors are equal to the number of output links physically connected to this branch-switch. The connection pattern specified in *conn_patt_vec* is updated by *resp_branch_vec* each time when the non-responsive branch is detected or a new connection request is received from a downstream branch.

A simplified pseudocode of the switch control algorithm is given in Figure 3. This algorithm deals with both congestion detection and RM-cell processing. Upon receiving a data cell, the switch multicasts the data cell to its output ports specified by *conn_patt_vec*, if the corresponding output links are available, else enqueues the data cell in its branch's queue. Mark the branch's BCI ($EFCI$) if $Q(t) > Q_h$. Update Q_{max} for the second-order rate control (to be discussed later) if the branch's new $Q(t)$ exceeds the old Q_{max}. $BCI := 1$ if its updated $Q_{max} \geq Q_{goal}$, the target buffer occupancy.

When a feedback RM-cell is received by a switch, do $conn\_patt\_vec \oplus resp\_branch\_vec$. If the resulting modulo-2 sum equals $\underline{1}$, an all 1's vector, representing the synchronization of feedback RM-cells, then a completely-consolidated feedback RM-cell is generated and sent upward. But, if the modulo-2 addition is

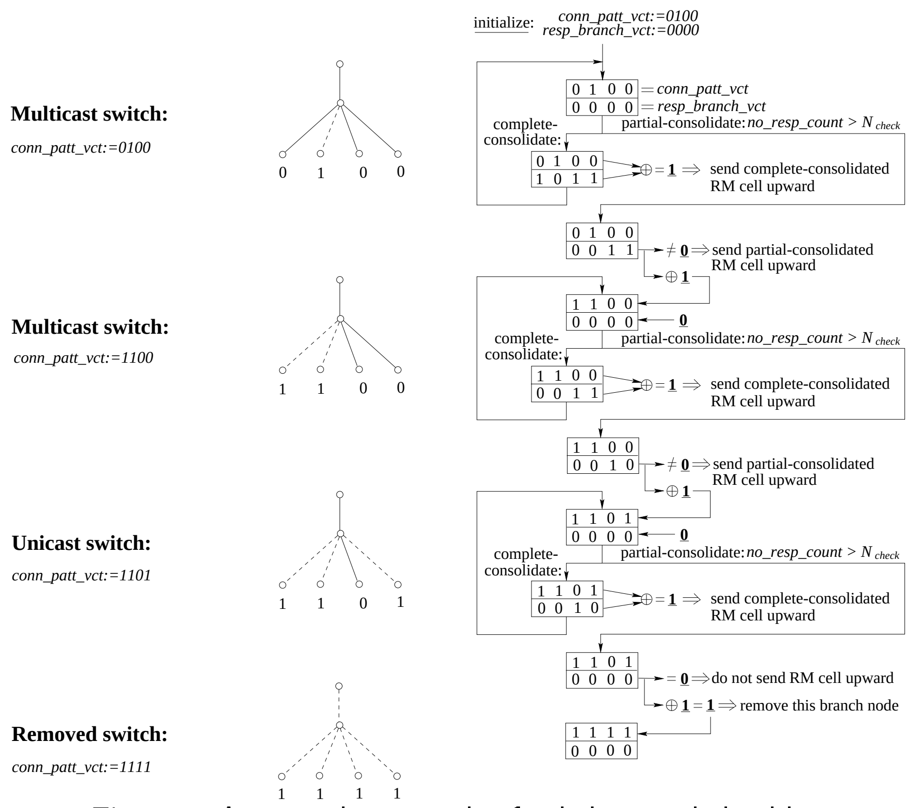


Figure 4: An execution example of switch control algorithm.

not equal to $\underline{1}$, the switch needs to await other feedback RM-cells for synchronization. Since the switch control algorithm does not require that a consolidated RM-cell be derived from the feedback RM-cells corresponding to the same forward RM-cell, the feedback RM-cell consolidation is "softly-synchronized."

When a forward (root-to-leaf) RM-cell is received, the switch first multicasts it to all connected branches specified by *conn_patt_vec*. Then, clear Q_{max} and the buffer congestion indicator MBCI if an NMQ request is received. To detect and remove the non-responsive or over-congested branches, *no_resp_count* is used. *no_resp_count* is initialized to 0 and reset to 0 whenever a consolidated RM-cell is sent upward. We use the forward RM-cell arrival time as a natural timeline for detecting/removing non-responsive/over-congested branches (such that it will work in the presence of faults in the downstream branches). Each time when a switch receives a forward RM-cell, the multicast connection's *no_resp_count* is increased by one. The updated *no_resp_count* is compared against its threshold N_{check}, which is determined by such factors as the difference between the maximum and minimum RM-cell roundtrip delays. If *no_resp_count* $\geq N_{check}$ and *resp_branch_vec* $\neq \underline{0}$ (i.e., there is at least one downstream branch is responsive), then the switch will stop awaiting arrival of feedback RM-cells and immediately generate a partially-consolidated RM-cell, and send it upward. Whenever *no_resp_count* $\geq N_{check}$, at least one non-responsive downstream switch is detected and will be removed by updating *conn_patt_vec* with the most recently obtained *resp_branch_vec*. Therefore, a downstream branch which has not sent any feedback RM-cell for N_{check} forward RM-cell time units will be removed from the multicast tree. On the other hand, a downstream node which is physically-linked to the branching-switch, but not part of the multicast tree, can join the multicast connection, at run-time, by submitting a connection request to its immediate upstream branching-switch. So, our algorithm supports the dynamic reconfiguration of the multicast tree.

Figure 4 shows an execution example, demonstrating how to implement RM-cell soft-synchronization, non-responsive branch detection, and reduction to a unicast connection. In this example, we assume the branching-switch has four physically-connected downstream branches, but only the first, the third, and the fourth belong to the multicast connection, thus initially, $conn\_patt\_vec = 0100$. If all connected branches send their feedback RM-cells before $no\_resp\_count \geq N_{check}$ (i.e., soft-synchronization is achieved), then a completely-consolidated RM-cell is generated and sent upward because $conn\_patt\_vec \oplus resp\_branch\_vec = 1111$. The branch connection pattern will remain the same as long as downstream branch-1, branch-3, and branch-4 keep sending feedback RM-cells upward before the condition $no\_resp\_count \geq N_{check}$ occurs. However, when only branch-3 and branch-4 respond in time, a partially-consolidated RM-cell is generated and sent upward. At the same time, $conn\_patt\_vec$ is updated as $conn\_patt\_vec = resp\_branch\_vec \oplus 1111 = 1100$, which defines a new multicast connection pattern at the branching-switch. A similar control procedure applies to the case of $conn\_patt\_vec = 1100$ which changes to $conn\_patt\_vec = 1101$ because branch-4 became non-responsive. Since there is only one responsive branch, the branching-switch supports unicasts. So, our algorithm includes unicast as a special case of multicast connection. If the only remaining branch-3 failed to respond in time resulting in $conn\_patt\_vec = 1111$, then the switch ceases generating and sending RM-cells upward, thus making the switch itself non-responsive to its upstream node (which will then be removed from the multicast tree).

3 The System Model

The proposed scheme can support both (i) binary rate-control, called the *CI-based scheme* since it only uses the CI bit for rate control, and (ii) explicit rate-control, called the *ER-based scheme* since the source rate is controlled by the ER fed back. However, we will focus only on the CI-based scheme (and will report the results on the ER-based scheme in a separate paper). The CI-based scheme is more attractive for LANs because of its least bandwidth cost for multicast signaling and simplicity in switch implementation. On the other hand, the ER-based scheme is more responsive to congestion and thus more suitable for WANs where bandwidth and roundtrip-delay product is large. However, the ER-based scheme's superiority to the CI-based scheme comes at the expense of complexity.

We model the proposed CI-based scheme by using the first-order fluid approximation, which characterizes the flow-control system with coupled time-delayed differential equations [8–14]. We assume the existence of only a single bottleneck[2] at a time with queue length $Q(t)$ and a "persistent" source with ACR $= R(t)$ for each multicast connection. Such a data-source model enables us to study the proposed scheme under the most stressful condition. Figure 5 depicts the system model for a multicast connection flow-controlled by the proposed scheme.

3.1 System Description

As shown in Figure 5, a multicast connection consists of n paths with different RM-cell roundtrip delays $\tau_1, \tau_2, \cdots, \tau_n$, and bottleneck bandwidths $\mu_1, \mu_2, \cdots, \mu_n$. There is only a single bottleneck on each path and its location may change with time. Thus, we use $T_f^{(i)}$ to represent the "forward" delay from the source to the bottleneck, and $T_b^{(i)}$ the "backward" delay from the bottleneck to the source via the leaf node of the i-th path. Clearly, $T_b^{(i)} = \tau_i - T_f^{(i)}$. Each path's bottleneck has its own $Q_i(t), i = 1, 2, \cdots, n$. According to the proposed control algorithm, all paths of a flow-controlled multicast connection share the same source rate $R(t)$ which dictates every path's dynamic behavior. As a result, all the paths in a multicast connection "interact" with each other via their source rate $R(t)$. Thus, the system model consists of n coupled subsystems, each of which corresponds to an individual path of the multicast connection. The i-th subsystem (path) is characterized by the following parameters in addition to $\tau_i, \mu_i, T_b^{(i)}$, and $T_f^{(i)}$:

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

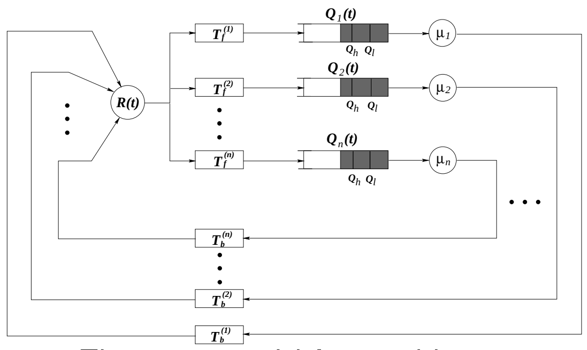


Figure 5: The system model for a multicast connection.

β:	Multiplicative decrease factor for rate reduction
α:	Additive rate-increase slope
p, q:	Increase step-size and decrease factor of rate increase parameter
$Q^{(i)}_{goal}$:	Target buffer occupancy
Q_h, Q_l:	High/low queue-length thresholds for detecting traffic overload/underload

We use a synchronous model for rate control in which the fixed rate-update interval Δ (RM-cell update interval) is usually a fraction of the roundtrip delay. According to the proposed source algorithm, the additive increase and the multiplicative decrease of source rate during the n-th rate-update interval can be modeled by the linear-increase and exponential-decrease in a continuous-time domain as follows [10]:

$$R_n = \begin{cases} R_{n-1} + a; & \text{additively increase} \\ bR_{n-1}; & \text{multiplicative decrease} \end{cases} \Longrightarrow \quad R(t) = \begin{cases} R(t_0) + \alpha(t - t_0); & \text{linear increase} \\ R(t_0)e^{-(1-\beta)\frac{(t-t_0)}{\Delta}}; & \text{exponential decrease} \end{cases}$$

where $a = AIR$ and $b = MDF$; t is the current time and t_0 is the time of the last rate-update; $\alpha = a/\Delta$ and $\beta = 1 + log\ b$ within a rate-adjustment interval (RM-cell update interval) Δ.

3.2 System Control Factors

At any given time, the most congested path governs the dynamic behavior of a flow-controlled multicast connection. To explicitly model this feature, we introduce the following definition.

Definition 3.1 *The* ***multicast-tree bottleneck (path)*** *is the path whose feedback dominates the source rate-control actions. The* ***multicast-tree RM-cell roundtrip delay*** *is the RM-cell roundtrip delay experienced in the multicast-tree bottleneck path.*

According to the proposed algorithm, the rate-control actions are based on the following feedback signals: (1) $ER = \min_{i\in\{1,2,\cdots,n\}}\{ER(i)\}$; (2) $CI = \bigcup_{i\in\{1,2,\cdots,n\}}\{CI(i)\}$; (3) $BCI = \bigcup_{i\in\{1,2,\cdots,n\}}\{BCI(i)\}$ where $ER(i)$, $CI(i)$, and $BCI(i)$ are the requested minimum rate, bandwidth congestion indication, and buffer congestion indication for path i, respectively. Obviously, ER is determined by the minimum bottleneck bandwidth, and CI or BCI is marked first by the path with minimum available bandwidth. Thus, the multicast-tree bottleneck is located along the path which has the minimum bottleneck bandwidth.

Soft-synchronization of Feedback RM-cells and Multicast RM-Cell Roundtrip Delay: Let τ be the multicast RM-cell roundtrip delay and τ_{max} (τ_{min}) be the delay of the longest (shortest) path which consists of maximum (minimum) number of hops if we assume that each intermediate link/switch has the same processing delay. Since, in the steady state, the arrival rates of feedback RM-cells from different paths are the same and equal to that of the forward RM-cell stream, the roundtrip delay of the RM-cells flowing through each path, like a pipeline, depends on the length (number of hops) of that path.

Based on our proposed algorithm, initially τ is equal to τ_{max} for the very first feedback RM-cell since the soft-synchronization algorithm requires that the feedback RM-cells be synchronized at each branch-switch. However, after the first cycle of RM cell, τ is determined by the multicast-tree bottleneck path's roundtrip delay which can be any value between τ_{min} and τ_{max} depending on the location of the bottleneck within the tree.

State Equations for the Multicast-Tree Bottleneck Path: Since the multicast-tree bottleneck dominates the source rate-control actions, it suffices to analyze the multicast flow-control system by focusing on the multicast-tree bottleneck's state equations. Let $Q(t)$ (Q_{goal}) be the queue-length function (the target buffer occupancy) at the multicast-tree bottleneck path, and let $\tau = T_f + T_b$ be the multicast-tree RM-cell roundtrip delay at the multicast-tree bottleneck path. Then, the multicast-tree bottleneck state is specified by two state variables, $R(t)$ and $Q(t)$. The multicast-tree bottleneck path's state equations are given by:
Source-rate function:

$$R(t) = \begin{cases} R(t_0) + \alpha(t - t_0); & \text{if } Q(t - T_b) < Q_h \\ R(t_0)e^{-(1-\beta)\frac{(t-t_0)}{\Delta}}; & \text{if } Q(t - T_b) \geq Q_h \end{cases} \tag{3.1}$$

Multicast-tree bottleneck path's queue-length functions:

$$Q(t) = \begin{cases} 0; & \text{if } Q(t) = 0 \text{ and } R(t) < \mu \\ \int_{t_0}^{t}[R(v - T_f) - \mu]dv + Q(t_0); & \text{if (1) } R(t) > \mu; \text{ or (2) } R(t) < \mu \text{ and } Q(t) > 0 \end{cases} \tag{3.2}$$

where $R(t)$ represents the fluid approximation to cell-transmission throughput. The average throughput is then given by $\lim_{t\to\infty} \frac{1}{t}\int_0^t R(v)dv$.

3.3 Analytic Solutions of Multicast-tree Bottleneck Path Dynamics

Here we only present the analytical expressions of the key performance measures for the dynamics of multicast-tree bottleneck path, and omit their derivations, which can be found in [14].

(1) Maximum Queue Length of the multicast-tree bottleneck path:

$$Q_{max} = \int_0^{T_{max}} \alpha t\, dt + \int_0^{T_d} (R_{max}e^{-(1-\beta)\frac{t}{\Delta}} - \mu)dt = \frac{\alpha}{2}T_{max}^2 + \alpha\frac{\Delta}{1-\beta}T_{max} + \mu\frac{\Delta}{1-\beta}\, log\, \frac{\mu}{R_{max}}, \tag{3.3}$$

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

(2) Oscillation Period of the multicast-tree bottleneck path:

$$T = T_{max} - \frac{\Delta}{1-\beta}\, log\, \frac{\mu}{R_{max}} + \tau + T_l + T_r, \tag{3.4}$$

where T_l is non-negative real root of non-linear equation:

$$e^{-(1-\beta)\frac{T_l}{\Delta}} + \frac{1-\beta}{\Delta}T_l - \left[\left(\frac{Q_{max} - Q_l}{\mu}\right)\left(\frac{1-\beta}{\Delta}\right) + 1\right] = 0 \tag{3.5}$$

and $T_r = \frac{\mu}{q\alpha}\left(1 - e^{-(1-\beta)\frac{T_l+\tau}{\Delta}}\right)$ assuming $Q_{max} > Q_{goal}$.

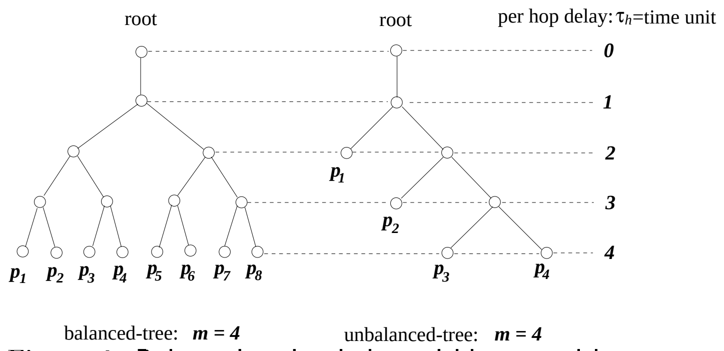

root
root
per hop delay: τ_h=time unit
0
1
2
3
4
p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8
p_1
p_2
p_3
p_4
balanced-tree: $m = 4$
unbalanced-tree: $m = 4$

Figure 6: Balanced and unbalanced binary multicast trees.

(3) Average Throughput of the multicast-tree bottleneck path:

$$\overline{R} = \frac{1}{T}\left[\mu T_{max} + \frac{\alpha}{2}T_{max}^2 + R_{max}\frac{\Delta}{(1-\beta)}\left(1 - e^{-(1-\beta)\frac{T_e}{\Delta}}\right) + T_r R_{min} + \frac{\alpha}{2}T_r^2\right], \tag{3.6}$$

where $T_e = \tau + T_l - \frac{\Delta}{1-\beta}\, log\, \frac{\mu}{R_{max}}$ and $R_{min} = \mu e^{-(1-\beta)\frac{\tau+T_l}{\Delta}}$.

All performance measures are shown to be functions of multicast-tree RM-cell roundtrip delay τ which varies with time. Thus, we explicitly include the variation of τ into our model, and study its relationship with the multicast-tree topology and flow-control performance.

4 The RM-cell Roundtrip Delay Analysis

4.1 Model

To simplify the analysis of RM-cell roundtrip delays, we assume that each switch (link) has the same processing (queueing and propagation) delay. This assumption can be relaxed easily because the difference in switch processing delay and link propagation delay can be translated into the difference in number of hops. We use the hop delay, τ_h, which is the sum of the switch processing delay and link propagation delay in taking one hop, as the *time unit.* We also assume the lossless transmission of RM-cells, as loss of RM-cells is taken care of by the non-responsive branch detection/removal mechanism. For the worst-case study and comparison, we only consider two types of multicast trees: *balanced* and *unbalanced* binary trees. Since we are only concerned with a path's RM-cell roundtrip delay which is determined by its length, it suffices to consider binary trees. Notice that in a balanced binary tree, the number n of paths from the root to leaves is equal to the height m of the tree while in a balanced binary tree $n = 2^{(m-1)}$. Figure 6 illustrates these two types of trees with height $m = 4$. The RM-cell roundtrip delay for a given path may vary at the beginning of the flow-control operation (in an initial state) when feedback RM-cells are not yet 'regularly' synchronized. The RM-cell roundtrip delay becomes stable after feedback RM-cells are regularly synchronized (in a steady state). We present the properties of RM-cell roundtrip delays with different flow-control schemes, beginning with a hop-by-hop feedback scheme.

4.2 Properties of Hop-by-Hop Feedback

Theorem 4.1 *If an* ***unbalanced-tree*** *multicast connection of height $m \geq 2$ is flow-controlled by a hop-by-hop feedback scheme with an RM-cell interval $\Delta \geq 1$, then the RM-cell roundtrip delay, $\tau_u(j, \Delta)$, of the j-th (counting from left to right) path, P_j, remains the same in both steady and initial states, and is* *determined by:*

$$\tau_u(j,\Delta) \;=\; \begin{cases} 2 + j\ \Delta; & \text{if } 2 \leq \Delta \leq \tau_{max} \\ 2(j+1); & \text{if } \Delta = 1 \end{cases} \tag{4.1}$$

where $1 \leq \Delta \leq \tau_{max}$,[3] $\tau_{max} = 2m$, *and* $1 \leq j \leq (m-1)$.

Proof. P_j's length is $j+1$ (in number of hops) and its leaf is located at the $(j+1)$-th level of the multicast tree (see Figure 6 for the case of $m = 4$). The proof consists of two parts, corresponding to the two parts of Eq. (4.1).

<u>Part 1</u>: Assume $2 \leq \Delta \leq \tau_{max} = 2m$. We can rewrite the first part of Eq. (4.1) as $\tau_u(j,\Delta) = 2 + j\Delta = (j+2) + j(\Delta - 2) + j \overset{\Delta}{=} C_1 + C_2 + C_3$. Then, according to the hop-by-hop feedback scheme, the three components of $\tau_u(j,\Delta)$ can be found as follows.

$C_1 = j + 2$ is the time for a forward RM-cell to traverse from the root to P_j's leaf node, then to return to the first branch-switch from the leaf toward the root (see Figure 6 for the case of $m = 4$). It takes $(j+1)$ time units for a forward RM-cell to reach P_j's leaf from the root and 1 time unit to immediately (by hop-by-hop feedback scheme) move one hop back to the first consolidating branch-switch, so $C_1 = (j+1) + 1 = j + 2$.

$C_2 = j(\Delta - 2)$ is contributed by feedback RM-cells waiting at branch-switches for the subsequent forward RM-cells, each moving one-hop upward the feedback RM-cell at each branch-switch. Let's start with $\Delta = 2$, implying that feedback RM-cells do not have to wait for forward RM-cells in order to move upward as they arrive at each branch-switch at the same time as a forward RM-cell arrives at the branch-switch. Thus, $C_2 = j(\Delta - 2) = 0$ holds. Now, suppose $\Delta = 2 + \ell$ with $\ell \geq 1$. Then, feedback RM-cells always arrive at branch-switches ℓ time units earlier than forward RM-cells, and thus, have to wait $\ell = (\Delta - 2)$ time units before making one-hop move. So, $C_2 = j(\Delta - 2)$, since there are j branch-switches along P_j.

$C_3 = j$ is the time for P_j's feedback RM-cells to traverse from its first branch-switch from the leaf without waiting for forward RM-cells to arrive, which is j, since there are j hops between the first branch-switch and the root.

<u>Part 2</u>: Assume $\Delta = 1$, then feedback RM-cells will never wait for forward RM-cells at any branch-switch, implying that $C_2 = 0$, and $C_1 = j+2$ and $C_3 = j$ remain the same as in Part 1. Thus, $\tau_u(j,\Delta) = C_1 + C_3 = 2(j+1)$ for $\Delta = 1$, completing the Proof. □

Corollary 4.1, which gives the equations for calculating the path delay in a balanced tree for hop-by-hop feedback, follows directly from Theorem 4.1 by letting $j = m - 1$ in Eq. (4.1).

Corollary 4.1 *If a balanced-tree multicast connection of height $m \geq 2$ is flow-controlled by the hop-by-hop feedback scheme with the RM-cell interval $\Delta \geq 1$, then RM-cell roundtrip delays of all paths, $\tau_b(j,\Delta)$, are the same in both steady and initial states, and are determined by:*

$$\tau_b(j,\Delta) = \max_{j \in \{1,2,\cdots,(m-1)\}} \{\tau_u(j,\Delta)\} \;=\; \begin{cases} \tau_{max} + (m-1)(\Delta - 2); & \text{if } 2 \leq \Delta \leq \tau_{max} \\ \tau_{max}; & \text{if } \Delta = 1 \end{cases} \tag{4.2}$$

where $\tau_{max} = 2m$, $1 \leq j \leq 2^{(m-1)}$, *and* $\tau_u(j,\Delta)$ *is* P_j*'s RM-cell roundtrip delay for an* ***unbalanced*** *multicast tree of the same height.*

[3]Theorem 4.1 still holds even when $\Delta \geq \tau_{max} = 2m$. But the RM-cell update interval Δ is usually a fraction of the RM-cell roundtrip delay. So, we do not consider the case of $\Delta \geq \tau_{max} = 2m$ even if it is analytically correct.

4.3 Properties for the Proposed Scheme

Lemma 4.1 *Consider an unbalanced-tree multicast connection T of height of $m > 2$. Let P_i be a relatively shorter path than another path $P_{\tilde{i}}$ such that $1 \leq i < \tilde{i} \leq m-1$. If T is flow-controlled by the proposed scheme with the RM-cell update interval $\Delta \geq 1$, then $P_{\tilde{i}}$'s feedback RM-cell does not have to wait for P_i's feedback RM-cell to synchronize feedback RM-cells at any branch-node.*

Proof. When the switch algorithm checks for feedback RM-cell synchronization ($conn\_patt\_vec \oplus resp\_branch\_vec = \underline{1}$?) at a branch-switch, the feedback RM-cell on a shorter path P_i always arrives at the branch-switch earlier than that (in response to the same forward RM-cell) from a longer path $P_{\tilde{i}}$. Thus, $P_{\tilde{i}}$'s feedback RM-cell can be synchronized, without waiting, *at least* with the feedback RM-cell in response to the same forward RM-cell, from the shorter path P_i at each branch-switch. So, the feedback RM-cell on a longer path never waits for feedback RM-cells on a shorter path to synchronize feedback RM-cells. □

Lemma 4.2 *Let P_j be the j-th path in an unbalanced multicast tree T as defined in Lemma 4.1 with $1 \leq j \leq (m-1)$. Then, the following four statements are equivalent for the steady-state RM-cell roundtrip delay:*

S1. *P_j's feedback RM-cell doesn't wait for a longer path $P_{\tilde{j}}$'s ($\tilde{j} > j$) feedback RM-cell to achieve feedback synchronization at the first branch-switch from P_j's leaf;*

S2. *P_j's feedback RM-cell doesn't wait for feedback RM-cells for synchronization at* any *branch-switch on P_j;*

S3. *$\exists k \in \{0, 1, 2, \cdots\}$ such that $2(m-j-1) - k\Delta = 0$, where $1 \leq j \leq (m-1)$ and $1 \leq \Delta \leq \tau_{max} = 2m$;*

S4. *P_j's steady-state RM-cell roundtrip delay $\tau_u(j, \Delta)$ attains its minimum and is given by:*

$$\tau_u(j, \Delta) = \min_{\Delta}\{\tau_u(j, \Delta)\} = 2(j+1), \quad \text{where } 1 \leq j \leq (m-1) \text{ and } 1 \leq \Delta \leq \tau_{max} = 2m.$$

Proof. In contrast to the case described in Lemma 4.1, the feedback RM-cell from a shorter path *may or may not* have to wait for the feedback RM-cell from a longer path for feedback synchronization.

S1 $\Longrightarrow$ S2: If P_j's feedback RM-cell does not wait at the first branch-switch from P_j's leaf, then it becomes part of the feedback RM-cell from a longer path after its RM-cell is consolidated at the first branch-switch. By Lemma 4.1, it does not wait for feedback RM-cells from any path at all subsequent branch-switches on P_j to achieve feedback synchronization.

S2 $\Longrightarrow$ S3: If P_j's feedback RM-cell does not wait for a longer path's feedback at any branch-switch, then at least it doesn't wait for synchronization at the first branch-switch from P_j's leaf, i.e., P_j's feedback RM-cell arrives at its first branch-switch at the exact same time when the feedback RM-cell from the longest path arrives at this branch-switch. But it takes $(2m - j)$ time units for an RM-cell to traverse from the root to the leaf of the longest path then return to P_j's first branch-switch. On the other hand, it takes $(j+2)$ time units for an RM-cell to traverse from the root to P_j's leaf then return to its first branch-switch. Thus, the arrival time of P_j's feedback RM-cell at its first branch-switch is $(j+2) + k\Delta$ where $k = 0, 1, \cdots$, corresponding to the $(k+1)$-th RM-cell, respectively. Then, $\exists k \in \{0, 1, \cdots\}$ such that the feedback RM-cell from the longest path is synchronized with P_j's feedback RM-cell whose arrival time is $(j+2) + k\Delta$ at its first branch-switch, and satisfies the following constraint (on k):

$$(j+2) + k\Delta \leq 2m - j < (j+2) + (k+1)\Delta \text{ for } 1 \leq j \leq (m-1) \text{ and } 1 \leq \Delta \leq \tau_{max} = 2m. \qquad (4.3)$$

But P_j's feedback RM-cell does not wait for the feedback RM-cell from a longer path at any branch-switch on P_j. Thus, $\exists k \in \{0, 1, \cdots\}$ such that $(j+2) + k\Delta = 2m - j$.

S3 $\Longrightarrow$ S4: From $2(m-j-1) - k\Delta = 0$, we know that P_j's feedback RM-cell does not wait for a longer path's feedback at the first branch-switch from the leaf. According to the proof of **S1 $\Longrightarrow$ S2**, P_j's feedback RM-cell does not wait for a longer path's feedback at all branch-switches on P_j. Therefore, P_j's steady-state RM-cell roundtrip delay only consists of the pure transmission (propagation plus processing, but no waiting) delay, meaning that $\tau_u(j,\Delta) = 2(j+1)$. Since P_j's feedback RM-cell may or may not have to wait for the feedback from a longer path for synchronization, depending on the value of Δ for given m and j, $\tau_u(j,\Delta)$ is lower-bounded by $2(j+1)$, and thus **S4** follows.

S4 $\Longrightarrow$ S1: If $\tau_u(j,\Delta)$ attains its minimum, then P_j's feedback RM-cell does not wait for feedback RM-cells from a longer path for synchronization at all branch-switches on P_j, and hence not at the first branch-switch from the leaf of P_j. This completes the proof. $\square$

Theorem 4.2 *Let P_j be the j-th path of an unbalanced multicast tree T as defined in Lemma 4.1 ($1 \le j \le (m-1)$). **If** T is flow-controlled by the proposed scheme with an RM-cell update interval Δ ($1 \le \Delta \le \tau_{max} = 2m$)* [4]*, **then** the following claims hold:*

Claim 1. *The number of P_j's feedback RM-cells going through **initial state** is determined by:*

$$k_j^* \stackrel{\Delta}{=} \max_{k\in\{0,1,2,\cdots\}} \{k \mid 2(m-j-1) - k\Delta \ge 0\},\ j = 1,2,\cdots,(m-1); \tau_{max} = 2m;\ \textit{and}\ 1 \le \Delta \le \tau_{max}; \quad (4.4)$$

Claim 2. *P_j's RM-cell roundtrip delay in **steady state** is determined by:*

$$\tau_u(j,\Delta) = \tau_{max} - k_j^*\Delta, \quad j = 1,2,\cdots,(m-1), \tau_{max} = 2m,\ \textit{and}\ 1 \le \Delta \le \tau_{max}; \qquad (4.5)$$

Claim 3. *The i-th RM-cell roundtrip delay in P_j's **initial state** is determined by:*

$$\tau_u(j,\Delta,i) = \begin{cases} \tau_{max} - (i-1)\Delta; & \textit{if } k_j^* \ge 1 \textit{ and } 1 \le i \le k_j^* \\ \tau_u(j,\Delta); & \textit{if } k_j^* \ge 1 \textit{ and } i > k_j^* \textit{ (become steady state)} \\ \tau_{max}; & \textit{if } k_j^* = 0 \textit{ (become steady state)} \end{cases} \qquad (4.6)$$

where $j = 1,2,\cdots,(m-1)$, $\tau_{max} = 2m$, and $1 \le \Delta \le \tau_{max}$.

Proof. For convenience of presentation, we begin with the proof of **Claim 2**.

Claim 2: By the statements **S3** and **S4** of Lemma 4.2 and the proofs of **S2 $\Longrightarrow$ S3** and **S3 $\Longrightarrow$ S4**, P_j's steady-state RM-cell roundtrip delay $\tau_u(j,\Delta)$ can be expressed as the sum of transmission delay $2(j+1)$ and synchronization delay W_j

$$\tau_u(j,\Delta) = 2(j+1) + W_j \qquad (4.7)$$

where W_j is the net waiting time for P_j's feedback RM-cell to synchronize with the feedback on a longer path at the first branch-switch from P_j's leaf. Based on Lemma 4.2's proof of **S2 $\Rightarrow$ S3** and Eq. (4.3), $\exists k \in \{0,1,2,\cdots\}$ such that W_j can be expressed as

$$W_j = (2m-j) - [(j+2) + k\Delta] = 2(m-j-1) - k\Delta. \qquad (4.8)$$

Since the feedback RM-cell on a longer path is always synchronized with the most recently arrived feedback on a shorter path at P_j's first branch-switch as a constraint Eq. (4.3), the minimum possible

[4]Theorem 4.2 still holds for $\Delta > \tau_{max} = 2m$, but Δ is typically a fraction of the maximum RM-cell roundtrip delay $\tau_{max} = 2m$.

synchronization-waiting time (≥ 0) determines W_j. By Lemma 4.1, $W_j \geq 0$. Thus, k in Eq. (4.8) is determined by

$$k_j^* \overset{\triangle}{=} \max_{k\in\{0,1,2,\cdots\}} \{k \mid 2(m-j-1) - k\Delta \geq 0\}, \qquad 1 \leq j \leq (m-1). \tag{4.9}$$

where k_j^* is obtained by minimizing $W_j = 2(m-j-1) - k\Delta \geq 0$ over k. Combining Eqs. (4.7), (4.8), and (4.9), we get Eq. (4.5).

Claim 1: Let n_i be the number of P_j's feedback RM-cells going through the initial state. Since the first feedback RM-cell received by the root always experiences the longest path's roundtrip delay, in initial state the RM-cell roundtrip delay decreases from $\tau_{max} = 2m$ to its steady-state value $\tau_u(j,\Delta)$ ($\leq \tau_{max} = 2m$). Thus, the number of RM-cells which go through the initial state is given by

$$n_i = \frac{\tau_{max} - \tau_u(j,\Delta)}{\Delta} = \frac{\tau_{max} - (\tau_{max} - k_j^*\Delta)}{\Delta} = k_j^* = \max_{k\in\{0,1,2,\cdots\}} \{k \mid 2(m-j-1) - k\Delta \geq 0\} \tag{4.10}$$

which results in Eq. (4.4).

Claim 3: Based on the proposed switch algorithm, all forward RM-cells in the initial state are consolidated in the first feedback RM-cell and sent back to the root at time $t = \tau_{max} = 2m$ to start feedback synchronization. In addition, the very first RM-cell's roundtrip delay is always equal to $\tau_{max} = 2m$ for all paths. Thus, if $k_j^* \geq 1$ for P_j, the i-th ($1 \leq i \leq k_j^*$) initial-state RM-cell will experience a roundtrip delay of $\tau_u(j,\Delta,i) = \tau_{max} - (i-1)\Delta$, since it enters the system $(i-1)\Delta$ time units later than the very first RM-cell. After k^* RM-cells pass through the flow-controlled system (i.e., $i > k_j^*$ for P_j), the system reaches steady state and P_j's RM-cell roundtrip delay becomes a constant (independent of i) specified by $\tau_u(j,\Delta,i) = \tau_u(j,\Delta)$. If $k_j^* = 0$ for P_j, i.e., P_j's feedback must be synchronized with those feedback RM-cells corresponding to the same forward RM-cell. Thus, the system enters steady state from the very first RM-cell since P_j's RM-cell roundtrip delay does not have initial-state (i.e., $k_j^* = 0$). Therefore, $\tau_u(j,\Delta,i) = 2m = \tau_{max}$, if $k_j^* = 0$ for P_j. This completes the proof. □

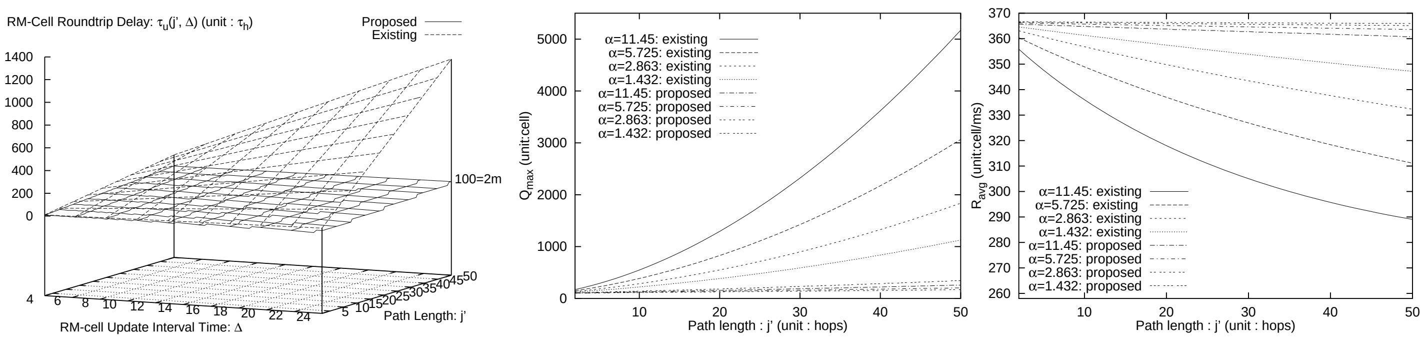
The corollary below, giving the equations for calculating the path delay under the proposed scheme for a balanced-tree structure, follows directly from Theorem 4.2 by letting $j = m-1$ in Eq. (4.4) which leads to $k_{(m-1)}^* = 0$ and thus $\tau_b(j,\Delta) = \tau_u(m-1,\Delta) = \tau_{max}$ by Eq. (4.5).

Corollary 4.2 *If a balanced-tree multicast connection of height of $m \geq 2$ is flow-controlled by the proposed scheme with the RM-cell interval $\Delta \geq 1$, then all paths' RM-cell roundtrip delays, $\tau_b(j,\Delta)$, are the same in both steady and initial states and are determined by:*

$$\tau_b(j,\Delta) = \max_{j\in\{1,2,\cdots,(m-1)\}} \{\tau_u(j,\Delta)\} = \tau_{max} \tag{4.11}$$

where $\tau_{max} = 2m$, $1 \leq j \leq 2^{(m-1)}$, and $\tau_u(j,\Delta)$ is P_j's RM-cell roundtrip delay for an unbalanced multicast tree of the same height.

Remarks. Comparing Theorem 4.2 and Theorem 4.1, we can make the followings observations. First, for hop-by-hop feedback, the initial-state and steady-state RM-cell roundtrip delays are the same. In contrast, with the proposed scheme, the initial-state RM-cell roundtrip delay, if any, is larger than, and lower bounded by, the steady-state roundtrip delay. For the proposed scheme, the initial-state acts/behaves like a "warm-up" for feedback RM-cells to be synchronized at each branch-switch, during which the initial-state RM-cell roundtrip delays converge to their corresponding steady-state values. The "warm-up" periods for P_j-th



(a) $\tau_u(j', \Delta)$ vs. (j', Δ) with $m = 50$. (b) Q_{max} vs. j' with $m = 50$. (c) $\overline{R}$ vs. j' with $m = 50$.

Figure 7: Effects of path length, tree height, and RM-cell interval on $\tau(j', \Delta)$, Q_{max}, and $\overline{R}$.

path $(1 \leq j \leq (m-1))$ are determined by k_j^* values given in Eq. (4.4). Second, for the proposed scheme in both initial and steady states, the RM-cell roundtrip delay $\tau_u(j, \Delta)$ is upper-bounded by $\tau_{max} = 2m$ (see Eqs. (4.5) and (4.6)). The increase rate of $\tau_u(j, \Delta)$ as a function of m, is $O(m)$ in the worst case. Third, with the hop-by-hop feedback scheme, the RM-cell roundtrip delay $\tau_u(j, \Delta)$ cannot be upper-bounded by $\tau_{max} = 2m$ (see Eq. (4.1)). Instead, $\tau_u(j, \Delta)$ is very sensitive to path length j and RM-cell update interval Δ, and can increase at a rate up to $O(m^2)$ in the worst case.

4.4 Comparison between the Proposed and the Existing Schemes

To quantitatively compare the RM-cell roundtrip delays under the hop-by-hop feedback and the proposed schemes, we present the numerical results drawn from Theorems 4.1 and 4.2 and the fluid modeling in Section 3. We focus here on the unbalanced multicast tree, which allows us to study the worst case of RM-cell roundtrip delay variations. We first consider how the multicast-tree path length j' $(= j + 1$ for $P_j)$, height m (the special case for $j' = m$ or $j = m - 1$), and RM-cell update interval Δ affect the RM-cell roundtrip delay $\tau_u(j', \Delta)$ under the two different schemes. In Figure 7(a), $\tau_u(j', \Delta)$ is plotted against a bivariate (j', Δ) with $m = 50$. It is found that $\tau_u(j', \Delta)$'s for the both schemes increase with j' and Δ. However, $\tau_u(j', \Delta)$ for the existing (hop-by-hop feedback) scheme increases much faster, and is always larger, than that for the proposed scheme, and tends to blow up (as high as 1200 τ_{hop}) as j' and Δ get larger. On the other hand, for the proposed scheme, $\tau_u(j', \Delta)$ increases very slowly as a function of Δ and j'. In fact, $\tau_u(j', \Delta)$ for the proposed scheme is upper-bounded by $2m = 100$ as shown in Theorem 4.2. We also observed that $\tau_u(j', \Delta)$ for the existing scheme is very sensitive to m (the maximum for j'). By contrast, m's impact on $\tau_u(j', \Delta)$ for the proposed scheme is very small as compared to the existing scheme. Thus, as shown in Figure 7(a), $\tau_u(j', \Delta)$ for the proposed scheme is virtually independent of j', m (and so the multicast-tree structure), and Δ, compared to the existing scheme. This is because (1) the synchronization-waiting time is much longer for the existing scheme than the proposed one; (2) the number of forward RM-cells required for a feedback RM-cell to return from a leaf node to the root in the existing scheme is proportional to m, while in the proposed scheme, any single RM-cell can return from the leaf node back to the root by itself.

Now, let's examine how the multicast-tree path length j' and height m affect the multicast-tree bottleneck path's maximum queue length Q_{max} and average throughput $\overline{R}$ under the two different schemes. We assume the multicast-tree bottleneck bandwidth $\mu = 155$ Mbps ≈ 367 cells/ms; each link-hop delay $\tau_h = 0.1$ ms; the RM-cell update interval Δ=4 $(\tau_h) = 0.4$ ms. In Figure 7(b) and Figure 7(c), using Eqs. (3.3), (3.4), and (3.6), Q_{max} and $\overline{R}$ under the two different schemes are plotted against j' with $m = 50$ while varying α. For the existing scheme, Q_{max} is observed to increase dramatically (see Figure 7(b)) while the average throughput $\overline{R}$ drops significantly (see Figure 7(c)) as the multicast-tree path length j' and tree height

m (the maximum for j') increase. This undesirable trend worsens as α gets larger. In contrast, with the proposed scheme under the same parameters settings, both Q_{max}-increase and $\overline{R}$-drop are very small when j' and m (even as α varies) increase. Again, Q_{max} and $\overline{R}$ under the proposed scheme are found to be virtually independent of the multicast-tree structure. Hence, the proposed scheme is more scalable than the existing scheme in terms of maximum buffer requirement and average throughput when the multicast-tree height and structure vary.

5 Adaptation to Variation of Multicast-Tree Roundtrip Delay

In a real network environment, there is always cross-traffic at each link, which causes the multicast-tree bottleneck path to drift from one path to another. So, the RM-cell roundtrip delay varies dynamically within $[\tau_{min}, \tau_{max}]$. The main and direct impact of RM-cell roundtrip-delay variations is on the maximum buffer requirement for the bottleneck path.

5.1 Maximum Buffer Requirement and Loss Control

Although SSP makes the RM-cell roundtrip delay τ for the proposed scheme much smaller than that for the existing scheme, as shown in Section 4, τ's swing between τ_{min} and τ_{min} is still large enough to make a notable impact on Q_{max}, particularly when m is large. Q_{max} also depends on the source rate-control parameter α (see Section 3). Thus, Q_{max} is a function of α and τ, i.e., $Q_{max}(\alpha, \tau)$. In reality, the buffer capacity, C_{max}, on the bottleneck path is finite, and hence, to ensure lossless transmission, the condition $Q_{max} \leq C_{max}$ must hold. This constraint divides the (α, τ)-space into two regions as follows.

Definition 5.1 *If $C_{max} < \infty$, then the **feasible** (α, τ)**-space:** $\Omega \triangleq \{(\alpha, \tau) \mid \alpha > 0, \tau > 0\}$ is partitioned into two parts: **lossless transmission-region:** $\mathcal{F} \triangleq \{(\alpha, \tau) \mid (\alpha, \tau) \in \Omega,\ Q_{max}(\alpha, \tau) \leq C_{max}\}$ and **lossy transmission-region:** $\mathcal{L} \triangleq \Omega \setminus \mathcal{F}$.*

The lemma presented below gives an upper bound for $Q_{max}(\alpha, \tau)$ as a function of α, τ, and Q_h in Ω.

Lemma 5.1 *If $(\alpha, \tau) \in \Omega$, then $Q_{max}(\alpha, \tau) \leq (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2$.*

Proof. Since $(\alpha, \tau) \in \Omega$, we have $T_{max} > 0$ and $T_d > 0$ (see Eq. (3.3) for definition of T_{max} and T_d). Thus, by the analytical solution given in Eq. (3.3), we get

$$\begin{aligned}
Q_{max}(\alpha, \tau) &= \int_0^{T_{max}} \alpha t\, dt + \int_0^{T_d} (R_{max} e^{-(1-\beta)\frac{t}{\Delta}} - \mu) dt && (5.1)\\
&\leq \frac{1}{2}(\alpha T_{max})(T_{max} + T_d) && (5.2)\\
&= \frac{1}{2}\left[\alpha\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\left(\frac{\alpha\Delta}{1-\beta}\, log\left[1 + \frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\right] && \\
&= \frac{1}{2}\left[\alpha\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\left(\mu\, log\left[1 + \frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\right] && (5.3)\\
&\leq \frac{1}{2}\left[\alpha\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)^2 + \left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\left(\mu\left[\frac{\alpha}{\mu}\left(\tau + \sqrt{\frac{2Q_h}{\alpha}}\right)\right]\right)\right] && (5.4)
\end{aligned}$$

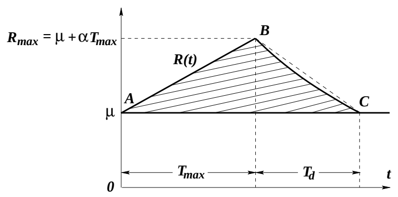


Figure 8: Q_{max} (shaded area) is upper-bounded by the area of $\triangle ABC$.

$$= \quad (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2 \tag{5.5}$$

The calculation of $Q_{max}(\alpha, \tau)$ in Eq. (5.1) is detailed in [14]. $Q_{max}(\alpha, \tau)$ is also equal to the area (shaded area in Figure 8) between the source rate $R(t)$ and the available bandwidth μ at the multicast-tree bottleneck over the time interval of $T_{max} + T_d$, which is upper-bounded by the area of its circumscribed triangle $\triangle ABC$ as shown in Figure 8. The righthand-side of Eq. (5.2) is the area of $\triangle ABC$. Since $\alpha > 0$ due to $(\alpha, \tau) \in \Omega$, in Eq. (5.3) we can use the equality of $\alpha\left(\frac{\Delta}{1-\beta}\right) = \mu$ which is set to balance the increasing and decreasing speeds of $R(t)$ [9]. Eq. (5.4) is due to the fact that $log\ x \leq\ x - 1$ and $log\ x \approx\ x - 1$ for x close to 1. This completes the proof. □

As analyzed in [9,10,13,14], Q_{max} is a monotonic increasing function of both α and τ (which is also verified in Figure 7(b), and also reflected in Lemma 5.1), and thus can be controlled by adjusting α for given τ. The theorem given below establishes an explicit relationship among α, τ, and Q_h subject to constraints of lossless transmission and $C_{max} < \infty$.

Theorem 5.1 *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* $(\alpha, \tau) \in \mathcal{F}\ \ \forall\ (\alpha, \tau) \in \{(\alpha, \tau) \mid \tau\sqrt{\alpha} \leq K, (\alpha, \tau) \in \Omega\}$;

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

Proof. **Claim 1:** Let $K \overset{\Delta}{=} \tau\sqrt{\alpha}$ which is a positive real number for $(\alpha, \tau) \in \Omega$. We further define $\zeta(K) = \zeta(\tau\sqrt{\alpha}) \overset{\Delta}{=} (K + \sqrt{2Q_h})^2$, the *upper bound function* of $Q_{max}(\alpha, \tau)$ obtained from Eq. (5.5). Thus, by Lemma 5.1 $\zeta(K) \geq Q_{max}(\alpha, \tau)$ for $(\alpha, \tau) \in \Omega$ and we have

$$Q_{max}(\alpha, \tau) \leq \zeta(K) = [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{max})] + C_{max}. \tag{5.6}$$

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

$$K^2 + 2\sqrt{2Q_h}K < (C_{max} - 2Q_h) \qquad \text{i.e.,} \qquad [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{max})] < 0. \tag{5.7}$$

and $\{(\alpha, \tau) \mid \tau\sqrt{\alpha} \leq K,\ (\alpha, \tau) \in \Omega\} \neq \emptyset$. Thus, $\forall\ (\alpha, \tau) \in \{(\alpha, \tau) \mid \tau\sqrt{\alpha} \leq K,\ (\alpha, \tau) \in \Omega\}$ where K is specified by Eq. (5.7), by Eqs. (5.6) and (5.7), we get $Q_{max}(\alpha, \tau) \leq [K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{max})] + C_{max} < C_{max}$, which implies $(\alpha, \tau) \in \mathcal{F}$, thus $\mathcal{F} \neq \emptyset$.

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

$$Q_{max}(\alpha, \tau) \leq \zeta(K) = K^2 + 2\sqrt{2Q_h}K + 2Q_h = C_{max}, \tag{5.8}$$

which reduces to a quadratic equation: $K^2 + 2\sqrt{2Q_h}K + (2Q_h - C_{max}) = 0$. Solving this for K and taking the positive root: $K_l = \sqrt{C_{max}} - \sqrt{2Q_h} > 0$ since $C_{max} > 2Q_h$. By Eq. (5.8), $(\alpha, \tau) \in \mathcal{F}$, $\forall$ $(\alpha, \tau) \in \{(\alpha, \tau) \mid \tau\sqrt{\alpha} \leq K_l,\ (\alpha, \tau) \in \Omega\}$, implying that all points located below or on the curve of function $K_l = \tau\sqrt{\alpha} \notin \mathcal{L}$. Thus, $\mathcal{L}$ is lower bounded by this function, completing the proof. □

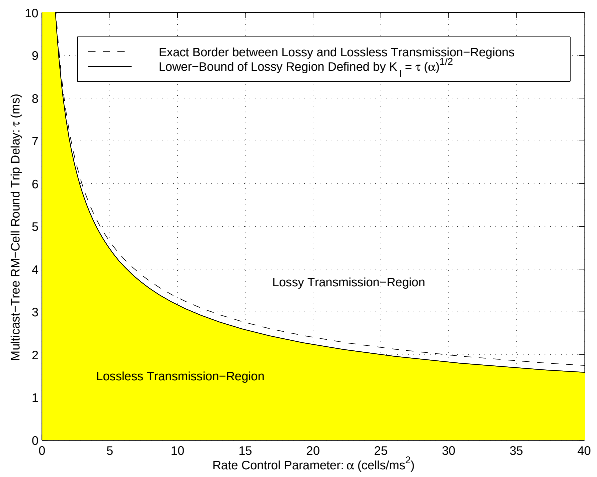


Figure 9: Lossy and lossless transmission regions divided by the lower bound of lossy transmission region.

Remarks on Theorem 5.1. **(1)** Claim 1 shows that Q_{max} is controllable, and identifies a sufficient condition for lossless. Moreover, Claim 1 describes 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 a close-form expression for α as function of τ and C_{max}, which is impossible to obtain (due to the non-linearity of Eq. (3.3)). Furthermore, since the lower bound function $\tau\sqrt{\alpha} = \sqrt{C_{max}} - \sqrt{2Q_h}$, which divides $\mathcal{F}$ and $\mathcal{L}$, is obtained by setting $Q_{max} \leq \zeta(K) = C_{max}$, by letting $Q_{max} = C_{max}$, we get $Q_{max} = (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2$, which can be used to estimate Q_{max} when the bound is tight. **(3)** Another interesting fact revealed by Theorem 5.1 is that Q_{max} is virtually independent of the multicast-tree bottleneck 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 relative difference 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 of $\mathcal{L}$, and the configurations of $\mathcal{F}$ and $\mathcal{L}$ are plotted in Figure 9, with $C_{max} = 400$ cells, Q_h=50 cells, and $\mu = 367$ cell/ms (about 155 Mbps). The exact border between $\mathcal{F}$ and $\mathcal{L}$ is obtained by solving Eq. (3.3) numerically. The lower bound of $\mathcal{L}$ plotted in Figure 9 is found to be very close to its exact value (the exact border between $\mathcal{L}$ and $\mathcal{F}$). In addition, the smaller α is, the tighter the bound is, which is consistent with the approximation *log* $x \approx x - 1$ when x is close to 1 (see Eq. (5.4)).

5.2 Second-Order Rate Control

Based on Theorem 5.1, 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 $\alpha = \frac{R(t)}{dt}$ — which we call *α-control* — is the second-order control over $R(t)$, providing one more dimension to control the dynamics of the proposed flow-control.

5.2.1 α-Control

The α-control is a discrete-time process as it is only exercised when the source rate control is in a "decrease-to-increase" transition based on the buffer congestion feedback signal in the n-th rate-control cycle,[5] $BCI(n) = 0$ (1) if the maximum queue length in the n-th rate-control cycle: $Q^{(n)}_{max} \leq Q_{goal}$ ($Q^{(n)}_{max} > Q_{goal}$), where Q_{goal} ($Q_h < Q_{goal} < C_{max}$) is the target buffer occupancy (also called *setpoint*) in the equilibrium state. If the multicast-tree bottleneck changes 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 will overflow, implying that the current α is too large for the increased τ. The source must reduce α to prevent cell loss. On the other hand, if τ decreases from its current value due to the shift of the multicast-tree bottleneck from a longer path to a shorter one, then Q_{max} will decrease. When $Q_{max} < Q_{goal}$, only a small portion of buffer space will be utilized, implying that the current α is too small for the decreased τ. The source should increase α to avoid buffer under-utilization and to improve system responsiveness in grabbing available bandwidth. Keeping $Q_h < Q_{goal} < C_{max}$ has two benefits: (1) the source can quickly grab available bandwidth; (2) it can achieve high throughput and high utilization.

The main purpose of α-control is to handle the buffer congestion resulting from the variation of τ. We set four goals for α-control: (1) ensure that $Q^{(n)}_{max}$ quickly converges to, and locks within, the neighborhood of Q_{goal}, which is upper-bounded by C_{max}, from an arbitrary initial value by driving $Q^{(n)}_{max}$'s corresponding α_n to the neighborhood of $\alpha_{goal} = Q^{-1}_{max}(Q_{goal})$ for a given τ; (2) minimize the oscillation amplitude of $Q^{(n)}_{max}$ for given rate-control parameters; (3) maintain statistical fairness on the buffer occupancy among multiple multicast connections sharing a common multicast-tree bottleneck; (4) minimize the extra cost incurred by the α-control. To achieve these goals, we propose a "converge-and-lock" α-control law (see Figure 2) in which the new value α_{n+1} is determined by α_n, and the feedback information BCI on Q_{max}'s current and one-step-old values, $Q^{(n)}_{max}$ and $Q^{(n-1)}_{max}$. The α-control law can be expressed by:

$$\alpha_{n+1} = \begin{cases} \alpha_n + p; & \text{if } BCI(n-1,n) = (0,0), \quad (Q^{(n-1)}_{max} \leq Q_{goal} \wedge Q^{(n)}_{max} \leq Q_{goal}) \\ q\alpha_n; & \text{if } BCI(n) = 1, \quad (Q^{(n)}_{max} > Q_{goal}) \\ \alpha_n / q; & \text{if } BCI(n-1,n) = (1,0), \quad (Q^{(n-1)}_{max} > Q_{goal} \wedge Q^{(n)}_{max} \leq Q_{goal}) \end{cases} \tag{5.9}$$

where q is the α-decrease factor such that $0 < q < 1$ and p is the α-increase step-size whose value will be discussed next.

5.2.2 Properties of α-Control

Definition 5.2 *The α-control is said to be in an* ***equilibrium*** *state if $Q^{(n)}_{max}$ has already converged to a certain regime and oscillates around Q_{goal} with constant amplitude and frequency; α-control is said to be in a* ***transient*** *state if it is not in an equilibrium state.*

Definition 5.3 *The* ***neighborhood*** *of target buffer occupancy Q_{goal} in an* ***equilibrium*** *state is specified by $\{Q^l_{goal}, Q^h_{goal}\}$ with $Q^l_{goal} \triangleq \max_{n\in\{0,1,2,\cdots\}} \{Q^{(n)}_{max} \mid Q^{(n)}_{max} \leq Q_{goal}\}$ and $Q^h_{goal} \triangleq \min_{n\in\{0,1,2,\cdots\}} \{Q^{(n)}_{max} \mid Q^{(n)}_{max} > Q_{goal}\}$, where $Q^{(n)}_{max}$ is governed by the proposed α-control law.*

Definition 5.4 *$\{Q^{(n)}_{max}\} \triangleq \{Q_{max}(\alpha_n)\}$ is said to* ***monotonically*** *converge to Q_{goal}'s neighborhood at time $n = n^*$ from its initial value $Q^{(0)}_{max} = Q_{max}(\alpha_0)$, if $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (0,0,0,0,\cdots,0,1,0,1,0,\cdots)$ for $\alpha_0 \leq \alpha_{goal}$; and $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (1,1,1,1,\cdots,1,0,1,0,1,\cdots)$ for $\alpha_0 > \alpha_{goal}$.*

[5] A rate-control cycle of $R(t)$ consists of a linear-increase phase and an exponential-decrease phase.

The α-control can be applied either in a *transient* state before $Q_{max}^{(n)}$ converging to Q_{goal}'s neighborhood, $\{Q_{goal}^l, Q_{goal}^h\}$, or in an *equilibrium* state after $Q_{max}^{(n)}$ has been bounded by $\{Q_{goal}^l, Q_{goal}^h\}$. Note that Q_{goal}^l and Q_{goal}^h are the closest attainable points to 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 of α_0. The α-control aims at making $Q_{max}^{(n)}$ converge rapidly in its transient state and lock steadily within Q_{goal}'s neighborhood in the equilibrium state. The monotonic-convergence of $Q_{max}^{(n)}$ to Q_{goal}'s neighborhood ensures that $Q_{max}^{(n)}$ is confined to the target operation regime at the first time it reaches the target regime, and is locked within that target regime. Thus, monotonic-convergence is desired in terms of shortness of transient state and the stability of equilibrium state. The theorem given below characterizes the properties of α-control in transient state, which provides a sufficient condition on α-control parameters selections for the monotonic-convergence of $Q_{max}^{(n)}$ and calculates the neighborhood of Q_{goal} and gives a formula to compute the number of the transient state cycles.

Theorem 5.2 *Suppose the proposed α-control law given by Eq. (5.9) is applied to a multicast connection with its bottleneck characterized by Q_{goal}, Q_h, and τ.* ***If*** *(1) $\alpha = \alpha_0$, an arbitrary initial value at time $n = 0$, (2) $0 < q < 1$, and (3) $p \le \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2$,* ***then*** *during the* ***transient*** *state,*

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

$$Q_{goal}^l = \begin{cases} Q_{max}\left(q^{n^*}\alpha_0\right); & \text{if } \alpha_0 > \alpha_{goal} \\ Q_{max}\left(q(n^*p+\alpha_0)\right); & \text{if } \alpha_0 \le \alpha_{goal} \end{cases} ; \quad Q_{goal}^h = \begin{cases} Q_{max}\left(q^{(n^*-1)}\alpha_0\right); & \text{if } \alpha_0 > \alpha_{goal} \\ Q_{max}\left(n^*p+\alpha_0\right); & \text{if } \alpha_0 \le \alpha_{goal} \end{cases} \tag{5.10}$$

(2) *n^*, the number of transient-state cycles, is determined by*

$$n^* = \begin{cases} \left\lceil \frac{\log\left[\frac{\alpha_{goal}}{\alpha_0}\right]}{\log q} \right\rceil; & \text{if } \alpha_0 > \alpha_{goal} \\ \left\lceil \frac{\alpha_{goal}-\alpha_0}{p} \right\rceil; & \text{if } \alpha_0 \le \alpha_{goal} \end{cases} \tag{5.11}$$

where α_{goal} corresponds to Q_{goal}, and is the positive real root of equation

$$\frac{\alpha_{goal}}{2}\left(\tau + \sqrt{\frac{2Q_h}{\alpha_{goal}}}\right)^2 + \mu\left(\tau + \sqrt{\frac{2Q_h}{\alpha_{goal}}}\right) + \frac{\mu^2}{\alpha_{goal}} \log \frac{\mu}{\mu + \alpha_{goal}\left(\tau + \sqrt{\frac{2Q_h}{\alpha_{goal}}}\right)} - Q_{goal} = 0. \tag{5.12}$$

Proof. **Claim (1)**: We prove this claim by considering the following two cases depending upon the range of the initial value of rate-control parameter α_0.

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

$$Q_{max}(\alpha_{n^*+1}) = Q_{max}(q\alpha_{n^*}) \le Q_{goal} \tag{5.13}$$

Since $(\tau\sqrt{\alpha_{goal}} + \sqrt{2Q_h})^2 \geq Q_{max}(\alpha_{goal}) = Q_{goal}$ by Lemma 5.1, we have $\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2 \leq \alpha_{goal}$. But, since $p \leq \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2$, we get $p \leq \left(\frac{1-q}{q}\right)\alpha_{goal}$, which reduces to $q(\alpha_{goal}+p) \leq \alpha_{goal}$. On the other hand, since $\alpha_{n^*-1} \leq \alpha_{goal}$, we have $q(\alpha_{n^*-1}+p) \leq q(\alpha_{goal}+p) \leq \alpha_{goal}$, implying

$$\alpha_{n^*+1} = q\alpha_{n^*} \leq \alpha_{goal} \tag{5.14}$$

because $\alpha_{n^*-1} + p = \alpha_{n^*}$. Thus, $Q_{max}(q\alpha_{n^*}) \leq Q_{max}(\alpha_{goal}) = Q_{goal}$, which is Eq. (5.13). Due to Eq. (5.13), $BCI(n^*, n^*+1) = (1,0)$. Applying α-control law, we get $\alpha_{n^*+2} = \alpha_{n^*+1}/q$. But $\alpha_{n^*+1} = q\alpha_{n^*}$, giving $\alpha_{n^*+2} = q\alpha_{n^*}/q = \alpha_{n^*} > \alpha_{goal}$; thus, $BCI(n^*+1, n^*+2) = BCI(0,1)$. Applying the α-control law again, $\alpha_{n^*+3} = q\alpha_{n^*+2} = q\alpha_{n^*} = \alpha_{n^*+1}$. But by Eq. (5.14), $\alpha_{n^*+3} = q\alpha_{n^*} \leq \alpha_{goal}$, and thus $BCI(n^*+2, n^*+3) = (1,0)$. Repeating the above procedure, we have $\forall k \in \{0,1,2,\cdots,\}$

$$\begin{cases} \alpha_{n^*+(2k+1)} = \alpha_{n^*+1} = q\alpha_{n^*} \leq \alpha_{goal} \\ \alpha_{n^*+2k} = \alpha_{n^*} > \alpha_{goal}; \end{cases} \tag{5.15}$$

implying that $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (0,0,0,0,\cdots,0,1,0,1,0,\cdots)$. By Definition 5.4, $Q_{max}^{(n)}$ monotonically converges to Q_{goal}'s neighborhood $\{Q_{goal}^l, Q_{goal}^h\}$. In addition, in the equilibrium state,

$$\begin{cases} Q_{max}(q\alpha_{n^*}) = Q_{max}(q(n^*p+\alpha_0)) = \max_{n\in\{0,1,2,\cdots\}} \{Q_{max}^{(n)} \mid Q_{max}^{(n)} \leq Q_{goal}\}; \\ Q_{max}(\alpha_{n^*}) = Q_{max}(n^*p+\alpha_0) = \min_{n\in\{0,1,2,\cdots\}} \{Q_{max}^{(n)} \mid Q_{max}^{(n)} > Q_{goal}\}. \end{cases} \tag{5.16}$$

Thus, by Definition 5.3, $Q_{goal}^l = Q_{max}(q(n^*p+\alpha_0))$ and $Q_{goal}^h = Q_{max}(n^*p+\alpha_0)$.

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

$$\begin{cases} \alpha_{n^*+(2k+1)} = \alpha_{n^*+1} = \alpha_{n^*}/q > \alpha_{goal} \\ \alpha_{n^*+2k} = \alpha_{n^*} \leq \alpha_{goal}; \end{cases} \tag{5.17}$$

which implies that $BCI(0,1,2,3,\cdots,n^*-1,n^*,n^*+1,n^*+2,n^*+3,\cdots) = (1,1,1,1,\cdots,1,0,1,0,1,\cdots)$. Therefore, by Definition 5.4, $Q_{max}^{(n)}$ monotonically converges to Q_{goal}'s neighborhood $\{Q_{goal}^l, Q_{goal}^h\}$. In addition, in the equilibrium state,

$$\begin{cases} Q_{max}(\alpha_{n^*}) = Q_{max}(q^{n^*}\alpha_0) = \max_{n\in\{0,1,2,\cdots\}} \{Q_{max}^{(n)} \mid Q_{max}^{(n)} \leq Q_{goal}\}; \\ Q_{max}(\alpha_{n^*}/q) = Q_{max}(q^{(n^*-1)}\alpha_0) = \min_{n\in\{0,1,2,\cdots\}} \{Q_{max}^{(n)} \mid Q_{max}^{(n)} > Q_{goal}\}. \end{cases} \tag{5.18}$$

Thus, by Definition 5.3, $Q_{goal}^l = Q_{max}(q^{n^*}\alpha_0)$ and $Q_{goal}^h = Q_{max}(q^{(n^*-1)}\alpha_0)$.

Claim (2): We also consider two cases.
Case 1. $\underline{\alpha_0 > \alpha_{goal}}$: Let α^l_{goal} correspond to $Q^l_{goal} = Q_{max}(\alpha^l_{goal})$. From Eq. (5.10), we have $\alpha^l_{goal} = q^{n^*}\alpha_0$, which reduces to $n^* = \frac{log\ \frac{\alpha_0}{\alpha^l_{goal}}}{log\ \frac{1}{q}} \geq \frac{log\ \frac{\alpha_0}{\alpha_{goal}}}{log\ \frac{1}{q}} = \frac{log\ \frac{\alpha_{goal}}{\alpha_0}}{log\ q}$ where the second equation is due to $\alpha_{goal} \geq \alpha^l_{goal}$. But since $q\alpha_{goal} < \alpha^l_{goal}$, i.e., $\frac{log\ \frac{\alpha_0}{\alpha^l_{goal}}}{log\ \frac{1}{q}} - \frac{log\ \frac{\alpha_0}{\alpha_{goal}}}{log\ \frac{1}{q}} < 1$, we get $n^* = \lceil \frac{log\ [\frac{\alpha_{goal}}{\alpha_0}]}{log\ q} \rceil$.
Case 2. $\underline{\alpha_0 \leq \alpha_{goal}}$: Let α^h_{goal} correspond to $Q^h_{goal} = Q_{max}(\alpha^h_{goal})$. From Eq. (5.10), we have $\alpha^h_{goal} = n^*p + \alpha_0$, which reduces to $n^* = \frac{\alpha^h_{goal}-\alpha_0}{p} > \frac{\alpha_{goal}-\alpha_0}{p}$ where the inequality is due to $\alpha_{goal} < \alpha^h_{goal}$. Since $\alpha^h_{goal} - \alpha_{goal} < p$, i.e., $\frac{\alpha^h_{goal}-\alpha_0}{p} - \frac{\alpha_{goal}-\alpha_0}{p} < 1$, we get $n^* = \lceil \frac{\alpha_{goal}-\alpha_0}{p} \rceil$.

Since α_{goal} corresponds to $Q_{goal} = Q_{max}(\alpha_{goal})$, it can be obtained by solving the non-linear Eq. (3.3) with Q_{max} substituted by Q_{goal}, which leads to Eq. (5.12), completing the proof. □

The following theorem characterizes the properties of α-control in the equilibrium state by describing the bounds of the neighborhood of Q_{goal} and their relationship with the α-control parameters.

Theorem 5.3 *For the same multicast connection under the proposed α-control as in Theorem 5.2,* ***if*** *$0 < q < 1$ and $p \leq \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2$,* ***then*** *during the* ***equilibrium*** *state, the fluctuation amplitudes of $Q^{(n)}_{max}$ around Q_{goal} are bounded as*

$$\begin{cases} Q^h_{goal} - Q_{goal} \leq \tau^2\alpha_{goal}\left(\frac{1}{q} - 1\right) + \tau\sqrt{8\alpha_{goal}Q_h}\left(\frac{1}{\sqrt{q}} - 1\right), \\ Q_{gaol} - Q^l_{goal} \leq \tau^2\alpha_{goal}(1-q) + \tau\sqrt{8\alpha_{goal}Q_h}(1-\sqrt{q}), \end{cases} \tag{5.19}$$

and the diameter of Q_{goal}'s neighborhood is bounded by

$$Q^h_{gaol} - Q^l_{goal} \leq \tau^2\alpha_{goal}\left(\frac{1}{q} - q\right) + \tau\sqrt{8\alpha_{goal}Q_h}\left(\frac{1}{\sqrt{q}} - \sqrt{q}\right), \tag{5.20}$$

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

Proof. Since $0 < q < 1$ and $p \leq \left(\frac{1-q}{q}\right)\left(\frac{\sqrt{Q_{goal}}-\sqrt{2Q_h}}{\tau}\right)^2$, by Theorem 5.2 $Q^{(n)}_{max}$ is guaranteed to converge to Q_{goal}'s neighborhood in the equilibrium state. Define an error function for given τ as $\gamma(\alpha) \overset{\triangle}{=} \zeta(\tau\sqrt{\alpha}) - Q_{max}(\alpha) = (\tau\sqrt{\alpha} + \sqrt{2Q_h})^2 - Q_{max}(\alpha)$, which is non-negative since $Q_{max}(\alpha) \leq \zeta(\tau\sqrt{\alpha})$.

Note that $\gamma(\alpha)$ is a monotonically increasing function of α, the proof of which is quite lengthy, thus omitted (this was already verified in Figure 9). Since $\alpha^h_{goal} \geq \alpha_{goal}$, $\gamma(\alpha^h_{goal}) - \gamma(\alpha_{goal}) \geq 0$, from which we get

$$\begin{aligned} Q^h_{goal} - Q_{goal} &\leq \left[Q^h_{goal} - Q_{goal}\right] + \left[\gamma(\alpha^h_{goal}) - \gamma(\alpha_{goal})\right] \\ &= Q^h_{goal} - Q_{goal} + \left[\left(\tau\sqrt{\alpha^h_{goal}} + \sqrt{2Q_h}\right)^2 - Q^h_{goal}\right] - \left[\left(\tau\sqrt{\alpha_{goal}} + \sqrt{2Q_h}\right)^2 - Q_{goal}\right] \\ &\leq \tau^2\left(\frac{1}{q}\alpha_{goal} - \alpha_{goal}\right) + \tau\sqrt{8Q_h}\left(\sqrt{\frac{1}{q}\alpha_{goal}} - \sqrt{\alpha_{goal}}\right) \qquad (5.21) \\ &= \tau^2\alpha_{goal}\left(\frac{1}{q} - 1\right) + \tau\sqrt{8\alpha_{goal}Q_h}\left(\frac{1}{\sqrt{q}} - 1\right) \qquad (5.22) \end{aligned}$$

where Eq. (5.21) is due to the fact that $\alpha^h_{goal} \leq \frac{1}{q}\alpha_{goal}$ resulted from the α-control law, proving the first part of Eq. (5.19). The second part of Eq. (5.19) can be proved similarly by using the fact that $\alpha^l_{goal} \geq q\alpha_{goal}$ due to the α-control law. Combining the Eq. (5.22) and the second part of Eq. (5.19) Eq. (5.20) follows, completing the proof. □

Remarks: The α-control law is similar to, but different from, AIMD (additive-increase and multiplicative-decrease) in the following sense. During the transient state, the α-control law behaves like AIMD, which offers convergence to the target operating regime in terms of efficiency and fairness (to be discussed below) in buffer allocation. On the other hand, in the equilibrium state, the α-control law guarantees the maximum buffer occupancy to lock within its setpoint regime as long as $Q_{max}(n)$ enters a constant neighborhood of Q_{goal}, regardless of the initial value α_0. In contrast, the AIMD algorithm does not guarantee this monotonic convergence because the α-control is a time-discrete process and its convergence is dependent on α_0. As a result, under AIMD, the equilibrium state with a small oscillation may never be reachable, and the fluctuation amplitude of $Q^{(n)}_{max}$ can be so large that the stability of the flow-control system and buffer control may not be achieved. Moreover, monotonic convergence enables $Q^{(n)}_{max}$ to converge to the neighborhood of Q_{goal} in the least amount of time.

5.2.3 Efficiency and Fairness of α-Control for Multiple Multicast Connections

Since $Q_{max}(\alpha)$ is a one-to-one function between Q_{max} and α, buffer-allocation control can be equivalently treated by α-allocation. We introduce below the criteria to evaluate the α-control law for buffer management in terms of α-allocation.

Definition 5.5 *Let vector $\boldsymbol{\alpha}(k) = (\alpha_1(k), \alpha_2(k), \cdots, \alpha_n(k))$ represent the rate-control parameters at time k for n multicast connections sharing a common bottleneck characterized by $\alpha_{goal} = Q^{-1}_{max}(Q_{goal})$. The **efficiency** of α allocation is defined by the **closeness** between the total α-allocation, $\alpha_t(k) \stackrel{\triangle}{=} \sum_{i=1}^n \alpha_i(k)$, and its target level α_{goal}.*

Neither over-allocation, $\alpha_t(k) > \alpha_{goal}$, nor under-allocation $\alpha_t(k) < \alpha_{goal}$ is desirable and efficient, as over-allocation may result in cell loss and under-allocation yields poor transient response, buffer utilization, and transmission throughput. The goal of α-control is to drive $\boldsymbol{\alpha}(k)$ to α_{goal} as close as possible and as fast as possible from any initial state.

Definition 5.6 *The **fairness** of α-allocation $\boldsymbol{\alpha}(k) = (\alpha_1(k), \alpha_2(k), \cdots, \alpha_n(k))$ for n connections of the same priority sharing the common multicast bottleneck at time k is measured by the **fairness index** defined as $\phi(\boldsymbol{\alpha}(k)) \stackrel{\triangle}{=} \frac{[\sum_{i=1}^n \alpha_i(k)]^2}{n\ [\sum_{i=1}^n \alpha_i^2(k)]}$.*

The fairness index is defined according to the *maxmin fairness* criterion [15] which ensures all the connections in an equivalent class to have an equal share of bottleneck resources. Notice that $\frac{1}{n} \leq \phi(\boldsymbol{\alpha}(k)) \leq 1$. $\phi(\boldsymbol{\alpha}(k)) = 1$ if $\alpha_i(k) = \alpha_j(k), \forall i \neq j$. This corresponds to "best" fairness. $\phi(\boldsymbol{\alpha}(k)) = \frac{1}{n}$ if the entire α is allocated to only one of n active connections. This corresponds to "worst" fairness and $\phi(\boldsymbol{\alpha}(k)) \to 0$ as $n \to \infty$. So, $\phi(\boldsymbol{\alpha}(k))$ should be as close to 1 as possible.

The α-control is a negative feedback control over the rate-control parameter, and computes $\alpha(k+1)$ based upon the current value $\alpha(k)$ and the feedback $BCI(k-1,k)$. Thus, $\alpha(k+1)$ can be expressed by the control function as $\alpha(k+1) = g(\alpha(k), BCI(k-1,k))$. For implementation simplicity, we only focus on a *linear* control function $g(\cdot,\cdot)$ such that $\alpha(k+1) = g(\alpha(k), BCI(k-1,k)) \stackrel{\triangle}{=} p + q\alpha(k)$, where coefficients p and q are determined by feedback information $BCI(k-1,k)$. The theorem given below (its proof is omitted due to lack of space) states that the proposed α-control law is feasible and optimal among all linear control functions.

Theorem 5.4 *Suppose n multicast connections sharing a common bottleneck are synchronously flow-controlled by the proposed α-control. Then, **(1)** in **transient** state, the α-control law is feasible and optimal linear control in terms of convergence to the efficiency and fairness (to be discussed below) of buffer allocation; **(2)** in **equilibrium** state, the α-control law is feasible and optimal linear control in terms of maintaining the efficiency and fairness of buffer allocation.*

Proof. We prove this theorem by considering the transient state and equilibrium state, respectively.
(I) In Transient State. The α-control function can be expressed by

$$\alpha_i(k+1) \quad = \quad \begin{cases} p_I + q_I\alpha_i(k); & \text{if } BCI(k-1,k) = (0,0), (\alpha_t(k) < \alpha_{goal}) \\ p_D + q_D\alpha_i(k); & \text{if } BCI(k) = 1, (\alpha_t(k) > \alpha_{goal}) \end{cases} \tag{5.23}$$

We now derive the constraints to determine the control-function coefficients p_I, q_I, p_D, and q_D to guarantee convergence to both efficiency and fairness.
(1) Convergence to Efficiency. To ensure $\alpha_t(k)$ to converge to its target α_{goal}, the α-control must be a negative feedback at each α-control cycle, i.e.,

$$\begin{cases} \alpha_t(k+1) = \sum_{i=1}^n \alpha_i(k+1) > \sum_{i=1}^n \alpha_i(k) = \alpha_t(k); & \text{if } BCI(k-1,k) = (0,0), (\alpha_t(k) < \alpha_{goal}) \\ \alpha_t(k+1) = \sum_{i=1}^n \alpha_i(k+1) < \sum_{i=1}^n \alpha_i(k) = \alpha_t(k); & \text{if } BCI(k) = 1, (\alpha_t(k) > \alpha_{goal}) \end{cases} \tag{5.24}$$

which, by using Eq. (5.23), reduces to

$$\begin{cases} q_I > 1 - \frac{np_I}{\sum_{i=1}^n \alpha_i(k)}, & \forall n \text{ and } \forall \sum_{i=1}^n \alpha_i(k); \text{ if } BCI(k-1,k) = (0,0), \\ q_D < 1 - \frac{np_D}{\sum_{i=1}^n \alpha_i(k)}, & \forall n \text{ and } \forall \sum_{i=1}^n \alpha_i(k); \text{ if } BCI(k) = 1, \end{cases} \tag{5.25}$$

(2) Convergence to Fairness. Convergence of $\boldsymbol{\alpha}(k)$ to fairness can be expressed by

$$\lim_{k\to\infty} \phi(\boldsymbol{\alpha}(k)) = \lim_{k\to\infty} \frac{[\sum_{i=1}^n \alpha_i(k)]^2}{n\ [\sum_{i=1}^n \alpha_i^2(k)]} = 1. \tag{5.26}$$

Plugging the linear-control function $g(\cdot,\cdot)$ into the fairness index and defining $\theta \triangleq \frac{p}{q}$, we get

$$\phi(\boldsymbol{\alpha}(k+1)) = \frac{(\sum_{i=1}^n [p + q\alpha_i(k)])^2}{n\sum_{i=1}^n [p+q\alpha_i(k)]^2} = \frac{(\sum_{i=1}^n [\theta + \alpha_i(k)])^2}{n\sum_{i=1}^n [\theta+\alpha_i(k)]^2} = \phi(\boldsymbol{\alpha}(k)) + [1-\phi(\boldsymbol{\alpha}(k))]\left[1 - \frac{\sum_{i=1}^n \alpha_i^2(k)}{\sum_{i=1}^n [\theta+\alpha_i(k)]^2}\right].$$

Note that $\phi(\boldsymbol{\alpha}(k+1)) - \phi(\boldsymbol{\alpha}(k)) = [1-\phi(\boldsymbol{\alpha}(k))]\left[1 - \frac{\sum_{i=1}^n \alpha_i^2(k)}{\sum_{i=1}^n [\theta+\alpha_i(k)]^2}\right]$ is a monotonic-increasing function of $\theta \triangleq \frac{p}{q}$ and $\phi(\boldsymbol{\alpha}(k+1)) \geq \phi(\boldsymbol{\alpha}(k))$ *iff* $\theta \geq 0$. Thus, if $\theta > 0$, fairness increases: $\phi(\boldsymbol{\alpha}(k+1)) > \phi(\boldsymbol{\alpha}(k))$; if $\theta = 0$, the fairness maintains: $\phi(\boldsymbol{\alpha}(k+1)) = \phi(\boldsymbol{\alpha}(k))$. Since $\theta \triangleq \frac{p_I}{q_I}$ in α-increase phase and $\theta \triangleq \frac{p_D}{q_D}$ in α-decrease phase, we get four possible cases as follows:

$$\begin{cases} \textbf{1. } \text{if } \frac{p_D}{q_D} > 0 \wedge \frac{p_I}{q_I} > 0, \text{then } \phi(\boldsymbol{\alpha}(k+1)) > \phi(\boldsymbol{\alpha}(k)) \text{ in both } \alpha\text{-dec and } \alpha\text{-inc;} \\ \textbf{2. } \text{if } \frac{p_D}{q_D} > 0 \wedge \frac{p_I}{q_I} = 0, \text{then } \phi(\boldsymbol{\alpha}(k+1)) > \phi(\boldsymbol{\alpha}(k)) \text{ in } \alpha\text{-dec and } \phi(\boldsymbol{\alpha}(k+1)) = \phi(\boldsymbol{\alpha}(k)) \text{ in } \alpha\text{-inc;} \\ \textbf{3. } \text{if } \frac{p_D}{q_D} = 0 \wedge \frac{p_I}{q_I} > 0, \text{then } \phi(\boldsymbol{\alpha}(k+1)) = \phi(\boldsymbol{\alpha}(k)) \text{ in } \alpha\text{-dec and } \phi(\boldsymbol{\alpha}(k+1)) > \phi(\boldsymbol{\alpha}(k)) \text{ in } \alpha\text{-inc;} \\ \textbf{4. } \text{if } \frac{p_D}{q_D} = 0 \wedge \frac{p_I}{q_I} = 0, \text{then } \phi(\boldsymbol{\alpha}(k+1)) = \phi(\boldsymbol{\alpha}(k)) \text{ in both } \alpha\text{-dec and } \alpha\text{-inc;} \end{cases} \tag{5.27}$$

Eq. (5.27) implies that control-function coefficients p_D, p_I, q_D, and q_I must all have the same signs if not zeroes. Combining Eq. (5.27) with Eq. (5.23), we conclude that these four control-function coefficients must be all positive if not zero, and q_I and q_D must be positive since $\alpha(k)$ is always positive. The convergence condition given in Eq. (5.25) adds further constraints on q_D such that $0 < q_D < 1$. Thus, the constraints on the control-function coefficients, in terms of convergence to fairness and efficiency, can be summarized as

$$\textbf{constraint}\{0 < q_D < 1, 0 < q_I, p_D \geq 0 \wedge p_I > 0, p_D > 0 \wedge p_I \geq 0\} \tag{5.28}$$

which include cases **1.**, **2.**, and **3.** as described in Eq. (5.27).

Since α-control is exercised on a per-connection basis, and the i-th source does not have any information on $\alpha_j(k)$, $\forall j \neq i$ and value of n (α-control is a distributed algorithm), the convergence condition given in Eq. (5.25) cannot be explicitly used to further specify the control-function coefficients. In the absence of such information, each connection must satisfy the negative feedback condition as follows, which represents a stronger condition for convergence to the efficiency:

$$\begin{cases} \alpha_i(k+1) > \alpha_i(k) \Longrightarrow p_I + (q_I - 1)\alpha_i(k) > 0, \forall i, & \text{if } BCI(k-1,k) = (0,0); \\ \alpha_i(k+1) < \alpha_i(k) \Longrightarrow p_D + (q_D - 1)\alpha_i(k) < 0, \forall i, & \text{if } BCI(k) = 1; \end{cases} \tag{5.29}$$

Eq. (5.29) yields further constraints in determining control-function coefficients. Since $(q_D - 1)\alpha_i(k) < 0$ (due to Eq. (5.28)) may have an arbitrarily small absolute value, the second inequality in Eq. (5.29) requires $p_D = 0$, which implies $p_I > 0$ by Eq. (5.28) for convergence to fairness in α-increase. The first inequality in Eq. (5.29) requires $q_I \geq 1$ to ensure $p_I + (q_I - 1)\alpha_i(k) > 0 \ \forall \alpha_i(k) > 0$. Since $\theta = \frac{p_I}{q_I}$ (fairness increases only in α-increase phase) and $\phi(\boldsymbol{\alpha}(k+1)) - \phi(\boldsymbol{\alpha}(k))$ is an increasing function of θ, we let q_I take its minimum $q_I = 1$ which is the optimal value for the convergence to fairness. Thus, we obtain the feasible and optimal linear control function defined by the following constraints:

$$\textbf{constraint}\{0 < q_D < 1, q_I = 1, p_D = 0, p_I > 0\} \tag{5.30}$$

which is the exactly what we proposed for the α-control in the transient state, i.e.,

$$\alpha_i(k+1) \quad = \quad \begin{cases} p + \alpha_i(k); & \text{if } BCI(k-1,k) = (0,0) \\ q\alpha(k); & \text{if } BCI(k) = 1 \end{cases} \tag{5.31}$$

where $p_I = p > 0$, $q_I = 1$, $p_D = 0$, and $0 < q_D = q < 1$.

(II) In Equilibrium State. The α-control function is expressed by

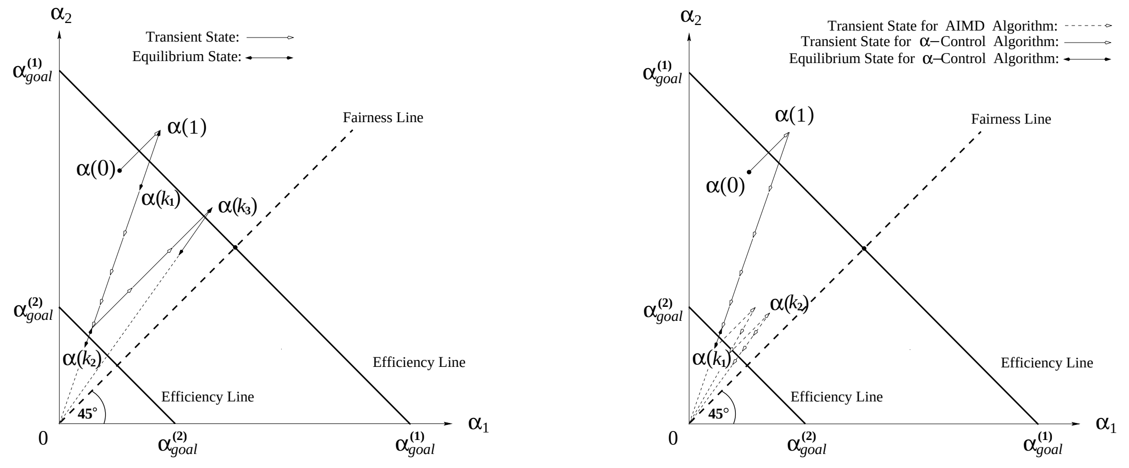
$$\alpha_i(k+1) \quad = \quad \begin{cases} \frac{1}{q}\alpha_i(k); & \text{if } BCI(k-1,k) = (1,0), (\alpha_t(k) = \alpha^l_{goal}) \\ q\alpha_i(k); & \text{if } BCI(k) = 1, (\alpha_t(k) = \alpha^h_{goal}) \end{cases} \tag{5.32}$$

Since $p_D = 0$, $p_I = 0$, $q_D = q$ $(0 < q < 1)$, and $q_I = \frac{1}{q} > 1$, this control function belongs to case **4.** in Eq. (5.27) where $\theta = 0$. Thus, the fairness is maintained as the α-control enters the equilibrium state. On the other hand, when $p_D = 0$ and $p_I = 0$, the constraints for convergence to efficiency become:

$$\textbf{constraint}\{0 < q_D < 1, q_I > 1\} \tag{5.33}$$

which is also consistent with Eq. (5.29) and Eq. (5.25). Thus, the convergence to efficiency is also maintained for that connection. This completes the proof. □

Remark. Theorem 5.4 is an extension from bandwidth control [16] to buffer control. The results presented here differ from [16] as follows. Unlike bandwidth control exerted on the RM-cell transmission rate, α-control is exercised once every rate-control cycle. As a result, α-control distinguishes transient state from



α2
Transient State:
Equilibrium State:
α(1)
α(0)
α(k1)
α(k3)
α(k2)
Fairness Line
Efficiency Line
Efficiency Line
45°
0
α1
Transient State for AIMD Algorithm:
Transient State for α–Control Algorithm:
Equilibrium State for α–Control Algorithm:

(a) $\boldsymbol{\alpha}(k)$ convergence to efficiency and fairness. (b) Comparison: α-control vs. AIMD algorithms.
Figure 10: Examples of α-allocation convergence to efficiency and fairness.

equilibrium state, and applies different control algorithms, ensuring convergence to, and lock within, a small neighborhood of the target operating regime. Since the total allocation $\alpha_t(k)$ keeps on going up and down due to the cross-traffic in real networks (or equivalently, the target α-allocation for each multicast connection is "moving" up and down), it is sufficient to ensure convergence to fairness/efficiency in transient state and maintain the achieved fairness/efficiency in equilibrium state. We do not need fairness to go up in both transient and equilibrium states.

Now, we describe two examples in a 2-dimensional space (for two connections) to graphically show the α-allocation convergence under the α-control in terms of efficiency and fairness. As shown in Figure 10, any α-allocation of two multicast connections at the k-th α-control is represented as a point $\boldsymbol{\alpha}(k) = (\alpha_1(k), \alpha_2(k))$ in a 2-dimensional space. All allocation points (α_1, α_2) for which $\alpha_1 + \alpha_2 = \alpha_{goal}$ form the *efficiency line*, and all points for which $\alpha_1 = \alpha_2$ form the *fairness line* which is a 45° line. It is easy to verify that additive increase, $(\alpha_1, \alpha_2) + p \overset{\triangle}{=} (\alpha_1 + p, \alpha_2 + p)$, corresponds to moving up ($p > 0$) along a 45° line, and multiplicative decrease or increase, $q(\alpha_1, \alpha_2) \overset{\triangle}{=} (q\alpha_1, q\alpha_2)$ ($0 < q < 1$ or $q > 1$), corresponds to moving along the line that connects the origin to (α_1, α_2).

Example 1. Two multicast connections sharing a bottleneck are flow-controlled by the α-control law. The multicast-tree bottleneck is characterized by: $\mu = 184$ cells/ms, $Q_{goal} = 200$ cells, $Q_h = 20$ cells, and $\tau = 2$ ms (so, $\alpha_{goal} = 18$ cells/ms^2). Consider a scenario where α_{goal} is equal to $\alpha_{goal}^{(1)} = 18$ initially, but reduces to $\alpha_{goal}^{(2)} = 6$ at the k_1-th α-control, and then returns to $\alpha_{goal}^{(1)}$ after the k_2-th α-control (see Figure 10(a)). The variation of α_{goal} is due to the variations in τ between $\tau^{(1)} = 2$ ms and $\tau^{(2)} = 3.34$ ms, or due to the variation in number of connections (between $n = 2$ and $n = 6$). We take $q = 0.8$ for both connections, specifying the total $p \leq 3.82$ for the two multicast connections by the condition (3) in Theorem 5.2 with $Q_{goal} = 200$ and $\tau = 2$ ms. Thus, $\frac{1}{2}p = 1.91$ for each of the two multicast connections. Suppose $\boldsymbol{\alpha}(0) = (3, 12.75)$ initially. Then, by α-control, $\boldsymbol{\alpha}(1) = \boldsymbol{\alpha}(0) + 1.91 = (4.9, 14.65)$ and $\boldsymbol{\alpha}(2) = 0.8\boldsymbol{\alpha}(1) = (3.92, 11.72)$ since $\alpha_1(0) + \alpha_2(0) = 15.75 < \alpha_{goal}^{(1)}$ and $\alpha_1(1) + \alpha_2(1) = 19.55 > \alpha_{goal}^{(1)}$. Thus, α-control enters equilibrium state around $\alpha_{goal}^{(1)}$ during which $\boldsymbol{\alpha}(k)$ fluctuates between $(3.92, 11.72)$ and $(4.9, 14.65)$. When α_{goal} reduces to $\alpha_{goal}^{(2)}$, equilibrium is broken and $\boldsymbol{\alpha}(k)$ converges to a new equilibrium state multiplicatively by 5 α-control cycles, and fluctuates between $(1.32, 3.87)$ and $(1.65, 4.83)$. Finally,

α_{goal} returns back to $\alpha_{goal}^{(1)}$, $\boldsymbol{\alpha}(k)$ converges to the new equilibrium state additively through 3 α-control cycles and fluctuates between $(6.13, 8.66)$ and $(7.67, 10.83)$. We observe that in transient state, α-control not only guarantees the monotonic convergence to the neighborhood of efficiency-line in the increase or decrease phase, but also improves the fairness index from $\phi(\boldsymbol{\alpha}(0)) = 0.72$ to $\phi(\boldsymbol{\alpha}(k_3)) = 0.94$ as shown in Figure 10(a), where $\boldsymbol{\alpha}(k_3) = (7.67, 10.83)$ is closer to the fairness line than $\boldsymbol{\alpha}(0) = (3, 12.75)$.
Example 2. In the second example, all the parameter settings and $\boldsymbol{\alpha}(0)$ are the same as in Example 1 except that α_{goal} reduces to, and stays with, $\alpha_{goal}^{(2)}$ after $\boldsymbol{\alpha}(k)$ reaches $\boldsymbol{\alpha}(1)$. Moreover, the trajectories under both α-control and AIMD algorithms are compared in Figure 10(b). Both schemes share the control trajectory from $\boldsymbol{\alpha}(0)$ up to $\boldsymbol{\alpha}(k_1)$. However, after $\boldsymbol{\alpha}(k)$ is driven to $\boldsymbol{\alpha}(k_1)$, the two trajectories split. Under α-control, $\boldsymbol{\alpha}(k)$ converges to an equilibrium state and locks itself within a small neighborhood of $\alpha_{goal}^{(2)}$: $\{(1.32, 3.87), (1.65, 4.83)\}$. In contrast, under the AIMD algorithm, $\boldsymbol{\alpha}(k)$ does not confine itself within a small neighborhood of $\alpha_{goal}^{(2)}$ and, in fact, $\boldsymbol{\alpha}(k)$ cannot even reach any equilibrium state. The resultant maximum buffer allocation "overshoot" for the AIMD algorithm at $\boldsymbol{\alpha}(k_2)$ is as high as $Q_{max}^{(k_2)} - Q_{goal} = 261 - 200 = 61$ cells, which is about 9 times as large as that for α-control (with the maximum "overshoot" equal to $Q_{max}^{(k_1+1)} - Q_{goal} = 207 - 200 = 7$). So, even though the AIMD algorithm is better than α-control in term of speed of convergence to fairness, the AIMD's maximum buffer requirement and potential loss ratio, which are more important to buffer control and design, are much higher than α-control, especially when the RM-cell roundtrip delay variation $\tau^{(2)} - \tau^{(1)}$ is large.

6 Conclusion

We proposed and evaluated a flow-control scheme for multicast ABR service, which scales well with the multicast-tree structure and size, and is efficient in dealing with variations in RM-cell roundtrip delay. The proposed scheme consists of two key components: soft-synchronization protocol (SSP) and second-order rate control. Using balanced and unbalanced binary-tree models, we analyzed the scalability of SSP in terms of the height and structure of the multicast tree. The analysis results show that as compared to the existing scheme, SSP can not only effectively achieve feedback synchronization, but also make the effective RM-cell roundtrip delay virtually independent of the multicast-tree's height and structure.

We proposed an algorithm which adapts the second-order rate-control parameter (α) to roundtrip-delay variations such that the maximum buffer requirement scales well with the bottleneck-path length. Using the fluid approximation, we modeled the proposed scheme for binary feedback and derived the system performance measures for multicast ABR services under the most stringent traffic condition. We also developed an optimal control condition, under which α-control guarantees the monotonic convergence of system state to the optimal regime from an arbitrary initial value. This α-control is also shown to be feasible and optimal in buffer-allocation efficiency and fairness at the multicast-tree bottleneck.

The oscillation of system state around the optimal operating point is the price to pay for the simplicity of binary feedback. This oscillation can be avoided by applying ER-feedback flow control. Moreover, it is necessary to add error-control facilities to the proposed scheme and study its performance in a more general environment. These are matters of our future inquiry.

References

[1] J. Crowcroft and K. Paliwoda, "A multicast transport protocol," in *Proc. of ACM SIGCOMM*, pp. 247–256, August 1988.

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

September 1994.

[3] L. Roberts, *Point-to-Multipoint ABR Operation*, ATM Forum contribution 95-0834, August 1995.

[4] K.-Y. Siu and H.-Y. Tzeng, "Congestion control for multicast service in ATM networks," in *Proc. of GLOBE-COM*, pp. 310–314, November 1995.

[5] H. Saito, K. Kawashima, H. Kitazume, A. Koike, M. Ishizuka, and A. Abe, "Performance issues in public ABR service," *IEEE Communications magazine*, vol. 11, pp. 40–48, November 1996.

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

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

[8] L. Kleinrock, *Queueing systems - Volume II, Computer Applications*, John Wiley and Sons, 1976.

[9] J. Bolot and A. Shankar, "Dynamical behavior of rate-based flow control mechanism," *ACM SIGCOMM Computer Communication Review*, vol. 20, no. 4, pp. 35–49, April 1990.

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

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

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

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

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

[15] J. M. Jaffe, "Bottleneck flow control," *IEEE Trans. on Communications*, vol. 29, no. 7, pp. 954–962, July 1981.

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