A Distributed Max-Min Flow Control Algorithm for Multi-rate Multicast Flows[†]

Hyang-Won Lee*
mslhw@netsys.kaist.ac.kr

Jeong-Woo Cho*
ggumdol@netsys.kaist.ac.kr

Song Chong*
song@ee.kaist.ac.kr

Abstract— **We present a distributed algorithm to compute bandwidth max-min fair rates in a multi-rate multicast network. The significance of the algorithm, compared to previous algorithms [1], [2], [3], is that it is more *scalable* in that it does not require each link to maintain the saturation status of all sessions and virtual sessions travelling through it, more *stable* in that it converges asymptotically to the desired equilibrium satisfying the minimum plus max-min fairness even in presence of heterogeneous round-trip delays, and has explicit *link buffer control* in that the buffer occupancy of every bottlenecked link in the network asymptotically converges to the pre-defined value. In addition, we propose an efficient feedback consolidation algorithm which is computationally simpler than its hard-synchronization based counterpart and eliminates unnecessary consolidation delay by preventing it from awaiting backward control packets(BCPs) that do not directly contribute to the session rate. Through simulations we verify the performance of the proposed multi-rate multicast flow control scheme based on these two algorithms.**

Index Terms— **Multi-rate multicast, max-min flow control, feedback consolidation.**

I. Introduction

With the rapid growth of the Internet, demand for point-to-multipoint multimedia communications such as broadcasting, multi-party teleconferencing and multi-party game etc. has been increasing, and consequently the multimedia traffic has been growing in the Internet. Those kinds of multimedia data transfers usually exhibit huge data volume, a long holding time and, more importantly, redundant transmission of the same data to simultaneously serve multiple receivers so that other data transfers are likely to be overwhelmed and to starve for network resources since network resources are always finite. Multicast, which prevents the redundant transmission of data on a link in transferring data to multiple receivers, is thus an efficient solution in minimizing the resource consumption of such point-to-multipoint multimedia communications.

Suppose that an efficient and scalable multicast routing mechanism exists. The problem we address in this paper is multi-rate multicast flow control problem given a multicast tree pre-determined by the multicast routing mechanism. In multi-rate multicast, the incoming flow rate of a session at every branching point in its tree is enforced to be the maximum of the rates that can be accommodated by its participating branches. By doing so, the sending rate at the source will eventually be the maximum of the rates that can be accommodated by the entire paths to individual receivers. Since the source sends data at the maximum path rate, it is necessary to convert down the incoming flow rate at every branching point to the values that can be accommodated by its participating branches. Provided such a rate adaptation functionality at every branching point, each virtual session(VS), defined as each source-receiver pair in a multicast session, will eventually receive data at an independently trimmed rate which is equal to the rate allowed by its entire path.

Multi-rate multicast flow control algorithms focusing on provable fairness have been proposed and analyzed [1], [2], [3]. These algorithms differ in inter-session fairness achievable, respectively adopting bandwidth max-min fairness, aggregate utility maximization and utility max-min fairness as the target fairness. The problems with these algorithms, which are major concerns of this paper, are as follows. First, they are lack of scalability since they require each router to keep maintaining the saturation status of every session and VS travelling through it, yielding a major computational bottleneck. Secondly, they have no explicit control over link buffer occupancy so that the allocated rates can wander considerably before converging and link flow can exceed the capacity temporarily yielding uncontrolled link buffer occupancy before converging. Thirdly, no explicit and usable stability condition has been given particularly in presence of heterogeneous round-trip delays. Last, no treatment on the feedback explosion problem has been provided.

Our work in this paper is in line with these network-assisted multi-rate multicast schemes but solves the aforementioned drawbacks of the previous algorithms. We suppose that fine-grained multimedia transcoding techniques are available and feasible to implement at branching points. Note that this is no longer merely a supposition today because of the advent of efficient fine-grained scalable coding methods such as MPEG-4 FGS video standard [4]. Following the MPEG-4 FGS video standard, it is feasible to freely adjust the video rate to an arbitrary value in real time without time-consuming decoding and re-encoding operations, as long as the target rate is greater than or equal to that of the base layer. Our aim is to find a distributed algorithm to compute bandwidth max-min fair rates in a multi-rate multicast network, which is *scalable* in that it does not require each router to keep maintaining the saturation status of every session and VS travelling through it, *stable* in that the algorithm converges asymptotically to the desired equilibrium even in presence of heterogeneous round-trip delays with an explicit and usable stability condition, and has *explicit link buffer control* in that

*The authors are with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Daejon 305-701, Korea.

†This research was supported in part by University IT Research Center Project and in part by Korea Science and Engineering Foundation under Grant R01-2001-000-00317-0.

0-7803-8794-5/04/$20.00 © 2004 IEEE

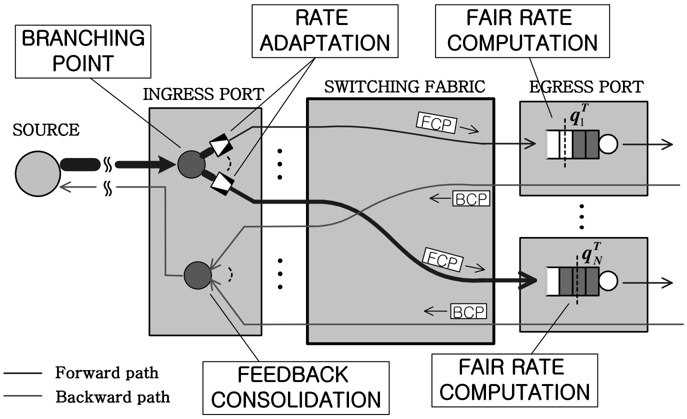

Fig. 1. The functional block diagram of the proposed scheme at a branching node

TABLE I

DATA STRUCTURES USED AT A BRANCHING NODE

	data structures
egress port j:	FairRate, ErrorSum
ingress port:	Token[M], MaxBranch[M], MaxBranchRate[M], BranchRate[M][N]
source:	SDR, ADR, MDR, PDR
FCP/BCP:	ADR, MDR
	description
FairRate	Fair rate at outgoing link j
ErrorSum	Cumulative $q_j[k] - q_j^T$
M	Maximum # of multicast sessions
N	Maximum # of branches per session
$Token[i]$	Token for upstream transmission of BCP
$MaxBranch[i]$	Session i's branch having maximum rate
$MaxBranchRate[i]$	Maximum rate of session i's branches
$BranchRate[i][j]$	Rate of session i's branch j
SDR	Current transmission rate at source
ADR	Allowed data rate
MDR	Minimum data rate to be guaranteed
PDR	Peak rate constraint

the buffer occupancy of every bottleneck link in the network asymptotically converges to the desired value. In addition, we propose an efficient soft-synchronization feedback consolidation algorithm which is computationally simpler than the hard-synchronization counterpart [5], eliminates unnecessary consolidation delay by preventing the algorithm from awaiting backward control packets(BCPs) that do not directly contribute to the session rate, and limits the number of BCPs travelling through a link in the backward direction to that of forward control packets(FCPs) travelling through it in the forward direction, thereby solving the feedback explosion problem [6].

II. THE ALGORITHM

Fig. 1 depicts the functional block diagram of the proposed multi-rate multicast flow control scheme at a branching node in a multicast network. In the forward direction, a multicast flow branches at the ingress port card of the node and is forwarded onto all of its outgoing links via the switching fabric. Rate adaptors associated with each outgoing link are also located at the ingress port card. At each egress port card, there is a single FIFO queue to multiplex all flows travelling through the outgoing link. The fair rate computation algorithm runs independently at each egress port card using the occupancy information of the FIFO queue. The source of a multicast session issues and transmits an FCP in the forward direction repeatedly upon every transmission of F data packets, in order to communicate flow-control related information with the routers in the tree. FCPs are also multicasted as data packets are. The receivers of the multicast session send these control packets back to the source as soon as they receive them. These control packets in the backward direction are BCPs. The feedback consolidation algorithm runs at the ingress port card in the backward direction. It merges the BCPs received from different branches into one BCP. We assume in the paper that the forward path and the backward path of each VS are identical and the result of the fair rate computation is written onto BCPs instead of FCPs. If the forward path and the backward path are not identical as in the current IP routing protocol, the result must be written onto FCPs.

Before we state the algorithm in details, we summarize data structures to be maintained at a branching node in Table I and provide the pseudocode of the proposed router and source algorithms in Fig. 2.

A. Router Algorithm

1) Fair Rate Computation: The proposed fair rate computation is based on PI control in the feedback control theory [7] [8] and has the following form. For each outgoing link j, its fair rate, $f_j[k]$, is calculated periodically upon every T epoch by

$$f_j[k] = -C_p(q_j[k] - q_j^T) - C_I \sum_{n=0}^{n=k} (q_j[n] - q_j^T) \qquad (1)$$

where $C_p > 0$ and $C_I > 0$ are the proportional and the integral control gains respectively, $q_j[k]$ is the queue length at the link buffer j, and q_j^T is its target queue length.

In contrast to the previous fair rate allocation algorithms in [1], [2], [3], the proposed algorithm is completely independent of the number of sessions and VSs travelling through the link and thus highly scalable. Moreover, it jointly controls rate allocation and link buffer control, meaning that as the iteration proceeds, it makes the link buffer occupancy converge to the target value, i.e., $\lim_{k\to\infty} q_j[k] = q_j^T$, while finding the max-min fair rate, which will be proved later in Section III. Such an explicit control of the link buffer occupancy is desirable in practice since without this, the allocated rates can wander considerably before converging and the link flow can exceed the capacity temporarily yielding uncontrolled link buffer occupancy before converging, and the link buffer occupancy even in the steady state can be arbitrary and thus it is unpredictable. None of the previous algorithms have such a feature, either.

In order to guarantee minimum data rate(MDR) for a multicast session during its entire holding time, it is necessary to have admission control to check the availability of bandwidth resource at all links in the tree and decide whether it can be accepted or not. Suppose that there exists an admission control such that at every link in the network the sum of MDRs of all sessions sharing the link is always less than its capacity. Then, the fair rate computation in (1) can easily be extended to support MDR by allocating $f_j[k] + MDR$ to

⟨ Router algorithm ⟩

⟨ Egress port j ⟩

```
Upon every T epoch (say, time k)
    $ErrorSum \mathrel{+}= q_j[k] - q_j^T$
    $FairRate = -C_P(q_j[k] - q_j^T) - C_I \cdot ErrorSum$
Upon receipt of a BCP from its downstream
    If $BCP.ADR > FairRate + BCP.MDR$
        $BCP.ADR = FairRate + BCP.MDR$
    Send the BCP to its intended ingress port
```

⟨ Ingress port ⟩

```
Upon receipt of a forward packet of session $i$
    If the packet is an FCP
        $Token[i] = 1$
    Make as many copies as intended branches
    Execute rate adaptation on each branch $j$ to enforce $BranchRate[i][j]$
    Forward each output to its intended egress port
Upon receipt of a BCP of session $i$ from egress port $j$
    $BranchRate[i][j] = BCP.ADR$
    If $j == MaxBranch[i]$
        $MaxBranchRate[i] = BCP.ADR$
        If $Token[i] == 1$
            Send the BCP to its upstream node
            $Token[i] = 0$
        Else
            Terminate the BCP
    Else if $BCP.ADR > MaxBranchRate[i]$
        $MaxBranchRate[i] = BCP.ADR$
        $MaxBranch[i] = j$
        If $Token[i] == 1$
            Send the BCP to its upstream node
            $Token[i] = 0$
        Else
            Terminate the BCP
    Else
        If $Token[i] == 1$
            $BCP.ADR = MaxBranchRate[i]$
            Send the BCP to its upstream node
            $Token[i] = 0$
        Else
            Terminate the BCP
```

⟨ Source algorithm ⟩

```
Transmit data including FCPs at SDR $\leq ADR$
Upon every transmission of D data packets
    Transmit an FCP with
        $FCP.ADR = \infty$
        $FCP.MDR = MDR$
Upon receipt of a BCP
    $ADR = \min\{BCP.ADR, PDR\}$
```

Fig. 2. Pseudocode of router/source algorithms

the sessions who require MDR guarantee. This achieves so-called *minimum plus max-min fairness*, implying that MDRs of all sessions in the network can be guaranteed and whatever the bandwidth remains after the guarantee will be shared by competing sessions in the max-min fair sense.

The major role of BCPs travelling in the backward direction is to inform upstream nodes of the fair rates computed locally by each link in the tree. Consider an egress port, say j. Upon receipt of a BCP from its downstream node, the fair rate computed locally by this port, $f_j[k] + BCP.MDR$, is compared with the fair rate of its downstream, being carried by the $BCP.ADR$ field of the BCP, and the smaller value is written onto the field and delivered to the upstream. The pseudocode of this fair rate computation and the BCP operation at an egress port is given in the first box of Fig. 2.

2) Feedback Consolidation: A multicast session branches at the ingress port of its branching node to its intended outgoing links. Consider branch j of multicast session i. The rate allocated by branch j to session i is stored in $BranchRate[i][j]$, which is updated upon receipt of a session i's BCP from the egress port j as $BranchRate[i][j] = BCP.ADR$ since the $BCP.ADR$ carries the information on the rate allowed by branch j. Rate adaptation is executed on branch j before forwarding data to its intended egress port j to enforce $BranchRate[i][j]$. On the other hand, if an FCP of session i arrives at the ingress port, we set $Token[i] = 1$. An arriving BCP of that session in the backward direction sees this token. If $Token[i] == 1$, the BCP knows that it is eligible to continue to travel through the ingress link in the backward direction. If $Token[i] == 0$, the BCP must stop travelling. By doing so, the number of session i's BCPs travelling through the ingress link in the backward direction is always restricted to that of session i's FCPs travelling through the link in the forward direction, and consequently, at every link in the network the number of BCPs in the backward direction is restricted to that of FCPs in the forward direction. Therefore, this single-bit token operation solves the feedback explosion problem.

The question remains is how to consolidate BCPs arriving from multiple branches into the limited number of BCPs who are eligible to continue to travel toward the upstream node. One way to do is the hard-synchronization based consolidation scheme [5]. In this scheme, each branching node in the tree waits for as many BCPs as the number of its branches or at least one BCP from all of its branches, and upon receipt of the last one, it merges these BCPs into the last BCP by overwriting the $BCP.ADR$ field of the last BCP as the maximum of ADR values delivered by these BCPs. The major drawback of this scheme is that such a hard synchronization can cause unnecessarily large consolidation delay since the branching node waits for BCPs from all of its branches including the ones that do not directly contribute to the maximum value. Moreover, BCPs are subject to loss and hence a timeout mechanism is necessary to avoid waiting for lost BCPs forever.

Our approach to this problem is to relax this hard synchronization constraint using the *locality* information. The key idea in our locality-based consolidation scheme is to cache both ID and rate of a branch who is likely to have the maximum rate among all branches based on history, and to send this cached rate to the upstream node by BCPs seeing $Token == 1$. The BCP arrives when $Token == 0$ is terminated to avoid the feedback explosion problem as explained. We call this branch as max-branch and store its ID and rate in $MaxBranch$ and $MaxBranchRate$ respectively. $MaxBranch$ and $MaxBranchRate$ are maintained for each multicast session and updated as follows. Consider an ingress port where session i branches. Suppose that a session i's BCP arrived from egress port j. If the port j is the previous max-branch of session i, i.e., $j == MaxBranch[i]$, then $MaxBranchRate[i]$ is updated by the $BCP.ADR$ value of this new BCP and $MaxBranch[i]$ is kept unchanged, expecting that the port j is still the max-branch. In this case, if $Token[i] == 1$, the BCP is sent to its upstream node with its $BCP.ADR$ being unchanged because it is believed to be the one from the max-branch. On the other hand, if the port j is

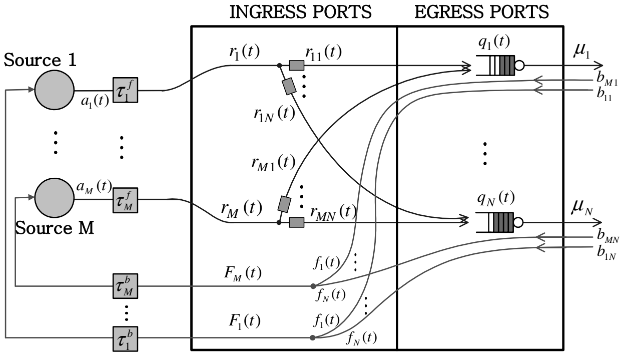

Fig. 3. The system model

not the previous max-branch of session i and the $BCP.ADR$ value of this BCP is greater than the rate of previous max-branch of session i, i.e., $BCP.ADR > MaxBranchRate[i]$, then both $MaxBranch[i]$ and $MaxBranchRate[i]$ are updated by j and $BCP.ADR$ since it is obvious that the max-branch was changed. In this case, if $Token[i] == 1$, the BCP is sent to its upstream node with its $BCP.ADR$ being unchanged because it is from the max-branch. If neither of two conditions in the above is satisfied, i.e., the port j is not the previous max-branch of session i and the $BCP.ADR$ value of the BCP is not greater than the rate of previous max-branch of session i, there is no reason to update $MaxBranch[i]$ and $MaxBranchRate[i]$. In this case, if $Token[i] == 1$, the BCP is sent to its upstream node with its $BCP.ADR$ being replaced as current $MaxBranchRate[i]$. The details of the proposed feedback consolidation scheme is given in the second box of Fig. 2.

Note that in our scheme, no BCP is waiting for other BCPs for consolidation. Every BCP arriving is processed on the fly and either sent to the upstream or terminated immediately. Therefore, it completely eliminates the unnecessary consolidation delay to await slow BCPs and thus can improve the transient response when the network condition changes. Moreover, it requires neither the sequence number nor the timeout mechanism to be implemented.

B. Source Algorithm

The source transmits data packets including FCPs at the rate of $SDR(\leq ADR)$ where ADR is updated upon receipt of a BCP as $ADR = \min\{BCP.ADR, PDR\}$. It also generates an FCP with $FCP.ADR = \infty$ and $FCP.MDR = MDR$ upon every transmission of D data packets

III. Analysis

A. System Model

The system model we consider is depicted in Fig. 3 where we model a single branching node explicitly and the other nodes in the network implicitly for the sake of analytical tractability. The branching node has M sessions passing through it and N outgoing links with individual FIFO queues. Thus, each session can have at most N branches. We model the system as a continuous-time fluid flow system. For the reader's convenience, the variables used throughout the analysis are summarized in Table II. By neglecting the buffer floor, the dynamics of the link buffer at each outgoing link j is modelled by

TABLE II
Variables used in the analysis

$a_i(t)$	The source transmission rate(SDR) of session i
$r_i(t)$	The ingress rate of session i at the branching node
$r_{ij}(t)$	The transmission rate of session i to link j
$q_j(t)$	The queue length of link buffer j
q_j^T	The target queue length of link buffer j
μ_j	The capacity of link j
$f_j(t)$	The fair rate computed at link j
$F_i(t)$	The fair rate of session i after consolidation
b_{ij}	The fair rate allocated by the downstream nodes of branch j of session i
m_i	The minimum rate constraint(MDR) of session i
p_i	The peak rate constraint(PDR) of session i ($p_i > m_i$)
S	The set of all sessions passing through the branching node
M	The cardinality of the set S
L	The set of all outgoing links at the branching node
N	The cardinality of the set L
L_i	The set of links to which session i branches
S_j	The set of sessions passing through link j
Q_j	The set of locally bottlenecked sessions at link j
$\lvert Q_j \rvert$	The cardinality of the set Q_j
τ_i^f	The forward-path delay of session i
τ_i^b	The backward-path delay of session i
τ_i	The round-trip delay of session i, i.e., $\tau_i^f + \tau_i^b$

$q_i^s = \lim_{t\to\infty} q_j(t)$, $f_j^s = \lim_{t\to\infty} f_j(t)$, $F_i^s = \lim_{t\to\infty} F_i(t)$,
$r_i^s = \lim_{t\to\infty} r_i(t)$, $r_{ij}^s = \lim_{t\to\infty} r_{ij}(t)$

$$\dot{q}_j(t) = \sum_{i\in S_j} r_{ij}(t) - \mu_j, \quad j \in L. \tag{2}$$

The fair rate computation at link j in (1) can be rewritten as

$$f_j(t) = -C_p\{q_j(t) - q_j^T\} - C_I \int_0^t \{q_j(t) - q_j^T\}\, dt, \quad j \in L. \tag{3}$$

$f_j(t) + m_i$ is allocated to the session i travelling through the link j and this rate is compared with b_{ij}, to choose the minimum, yielding $\min\{f_j(t) + m_i, b_{ij}\}$. b_{ij} is assumed to be constant to avoid analyzing the coupled dynamics which is too complex to deal with. Assuming that the feedback consolidation contains no consolidation delay and errors, we have

$$F_i(t) = \max_{j\in L_i}[\min\{f_j(t) + m_i, b_{ij}\}], \quad i \in S. \tag{4}$$

The consolidated fair rate of session i, $F_i(t)$, is fed back to the source after the backward-path delay τ_i^b. Then, the source i transmits data at the following rate.

$$a_i(t) = \min[F_i(t - \tau_i^b), p_i], \quad i \in S \tag{5}$$

After some computations, we can obtain

$$r_{ij}(t) = \begin{cases} f_j(t - d_i) + m_i & i \in Q_j \\ \min[b_{ij}, p_i] & i \in S_j - Q_j. \end{cases} \tag{6}$$

where $0 \leq d_i \leq \tau_i$. See [9] for the derivation.

B. Steady State and Fairness

Suppose that the closed-loop system has an equilibrium point at which the derivatives of the system variables are zero, i.e., $\lim_{t\to\infty}\dot{q}_j(t) = 0$ and $\lim_{t\to\infty}\dot{f}_j(t) = 0$ for all $j \in L$. At the equilibrium point, (2), (3), (4), (5) and (6) give us that

$$\sum_{i\in S_j} r_{ij}^s = \mu_j, \quad q_j^s = q_j^T, \quad \forall\ j \in L, \tag{7}$$

$$F_i^s = \max_{j\in L_i}[f_j^s + m_i, b_{ij}], \quad a_i^s = \min[F_i^s, p_i], \quad \forall\ i \in S, \tag{8}$$

$$r_{ij}^s = \begin{cases} f_j^s + m_i & i \in Q_j \\ \min[b_{ij}, p_i] & i \in S_j - Q_j \end{cases} \quad \forall\ i \in S,\ \forall\ j \in L. \tag{9}$$

By combining the first equation in (7) and (9), we obtain

$$f_j^s = \frac{\mu_j - \sum\limits_{i\in Q_j} m_i - \sum\limits_{i\in S_j - Q_j} \min[b_{ij}, p_i]}{|Q_j|}, \quad \forall\ j \in L. \tag{10}$$

By substituting (10) for f_j^s in (9), we obtain that for $\forall\ i \in S$ and $\forall\ j \in L$,

$$r_{ij}^s = \begin{cases} \dfrac{\mu_j - \sum\limits_{i\in Q_j} m_i - \sum\limits_{i\in S_j - Q_j} \min[b_{ij}, p_i]}{|Q_j|} + m_i & i \in Q_j \\ \min[b_i^s, p_i] & i \in S_j - Q_j. \end{cases} \tag{11}$$

The following theorem summarizes the result.

Theorem 3.1: Provided that $\sum_{i\in S_j} m_i < \mu_j, \forall j \in L$, and $\min[b_{ij}, p_i] > m_i, \forall i \in S_j - Q_j$, there exists an unique equilibrium point at which 1) the occupancy of each link buffer is equal to its target value ($q_j^s = q_j^T, \forall j \in L$), 2) the capacity of each link is fully utilized ($\sum_{i\in S_j} r_{ij}^s = \mu_j, \forall j \in L$), 3) for every multicast session, its MDR is guaranteed at all the branches in the tree ($r_{ij}^s > m_i$, $\forall i \in S$, $\forall j \in L$) and for every link, its unreserved portion of capacity, $\mu_j - \sum_{i\in S_j} m_i$, is shared by all the sessions travelling through it in the max-min fair sense.

C. Asymptotic Stability

In this subsection we study the local stability of the closed-loop system in the neighborhood of the equilibrium point where the system is governed by (2), (3) and (6). Note that we omit all the proofs and some subsidiary propositions due to the limited space. See [9] for the details.

Consider an outgoing link j which has at least 1 locally-bottlenecked session. By substituting (6) for $r_{ij}(t)$ in (2), we get

$$\dot{q}_j(t) = \sum_{i\in Q_j} f_j(t - d_i) + \underbrace{\sum_{i\in Q_j} m_i + \sum_{i\in S_j - Q_j} \min[b_{ij}, p_i] - \mu_j}_{constant} \tag{12}$$

where $d_i \le \tau_i$. The constant part in the equation can be viewed as an external disturbance. By denoting the disturbance by D and substituting (3) for $f_j(t - d_i)$ in (12), we obtain the following closed-loop equation of the system.

$$\dot{q}_j(t) = D - \sum_{i\in Q_j}\left[C_P\{q_j(t-d_i) - q_j^T\} + C_I\int_0^{t-d_i}\{q_j(t) - q_j^T\}\,dt\right]. \tag{13}$$

The closed-loop system given by (13) is a special case of the one in [10], and we note that all the results associated with the asymptotic stability here are a special case of those in [10].

Now, we define the controller gains, C_P and C_I, to be

$$C_P = \frac{A}{|Q_j|}, \quad C_I = \frac{B}{|Q_j|} \tag{14}$$

where A and B are some positive constants. The open-loop transfer function of the closed-loop system (13) is then given by

$$F(s) = \left(\frac{A}{|Q_j|}\frac{1}{s} + \frac{B}{|Q_j|}\frac{1}{s^2}\right)\sum_{i\in Q_j} e^{-d_i s}, \tag{15}$$

which is obviously a special case of

$$F(s) = \left(\frac{A}{s} + \frac{B}{s^2}\right)\sum_{i\in Q_j} \rho_i e^{-d_i s} \tag{16}$$

where $\rho_i \ge 0, \forall i \in Q_j$ and $\sum_{i\in Q_j} \rho_i \le 1$. From now on, we use this generalized form of open-loop transfer function to find the stability condition.

First, we consider a single source case, i.e., $|Q_j| = 1$ with round-trip delay d and $\rho_1 = 1$ and $\rho_i = 0, \forall i > 1$. Then, the open-loop transfer function becomes

$$F(s) = \underbrace{\left(\frac{A}{s} + \frac{B}{s^2}\right)}_{\triangleq G(s)} e^{-ds} \tag{17}$$

and letting $s = j\omega$ yields

$$F(j\omega) = \left(-\frac{B}{\omega^2} - j\frac{A}{\omega}\right) e^{-j\omega d}. \tag{18}$$

The following theorem states the stability condition for the closed-loop system with a single delay.

Theorem 3.2: The closed-loop system with a single delay $d \ge 0$ is asymptotically stable if and only if the delay is bounded by

$$0 \le d < \frac{\arccos\left(\frac{B}{\bar{\omega}^2}\right)}{\bar{\omega}} \tag{19}$$

where $\bar{\omega}$ is a unique $\omega > 0$ such that $|F(j\omega)| = 1$.

We have found the upper bound of the round-trip delay for the single source system to be asymptotically stable. It is, however, difficult to apply the stability condition (19) as it is to the design of a controller. We modify the condition into an usable form in the following corollary.

Corollary 3.1: Let $U = Ad$ and $V = Bd^2$. Then the closed-loop system is asymptotically stable if and only if

$$0 < U < \frac{\pi}{2} \text{ and } 0 < V < \omega_1^2 \cos\omega_1 \tag{20}$$

where ω_1 is the unique solution of $U = \omega\sin\omega$ for $0 < \omega < \pi/2$.

The stability condition for the case of heterogeneous round-trip delays can be given by the theorem below.

Theorem 3.3: The closed-loop system with heterogeneous delays is asymptotically stable for all $0 \le d_i \le \bar{d}$ and for all ρ_i satisfying $\sum_{i\in Q_j} \rho_i \le 1$ if and only if the closed-loop system of the single-delay case with delay $\bar{d}$ is asymptotically stable.

Consequently, once the upper bound of all the round-trip delays is known, the stable gain for the heterogeneous-delay case can be obtained from $A = U/\bar{d}$ and $B = V/\bar{d}^2$ where U and V satisfies (20).

TABLE III

RECOMMENDED VALUES FOR THE DESIGN PARAMETERS. $\bar{d} = \max_{i \in S} d_i$ AND Δ IS ONE PACKET TRANSMISSION TIME.

Rate Allocation Algorithm			FCP Transmission
A	B	T	D
$\frac{0.5}{\bar{d}}$	$\frac{0.1}{\bar{d}^2}$	32Δ	32

TABLE IV

THE TRAFFIC MODEL, THEORETICAL FAIR RATES(MBPS) AND $|Q_j|$ OVER TIME(SEC).

session	MDR	PDR	Arrival	Departure
	(Mbps)		(sec)	
1	15	150	1	3
2	20	150	2	4
3	10	20	0	∞
4	10	150	0	∞
5	25	150	0	∞
6	30	150	0	∞

session	0∼1	1∼2	2∼3	3∼4	4∼
1	-	38.75	30	-	-
2	-	-	38.3	40	-
3	20	20	20	20	20
4	47.5	38.75	30	40	47.5
5	60	53.75	43.3	53.3	60
6	67.5	53.75	45	53.3	67.5
link			$\lvert Q_j \rvert$		
L3	0	2	3	2	0
L5	3	2	2	2	3

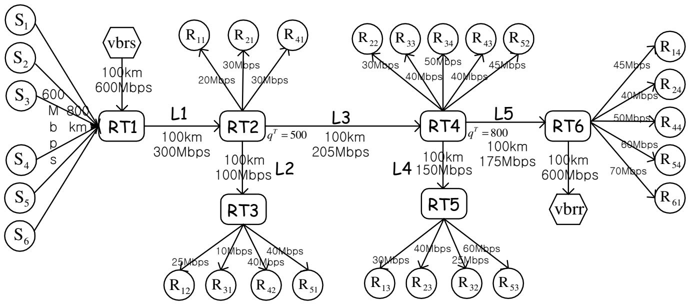

Fig. 4. Multiple-link configuration

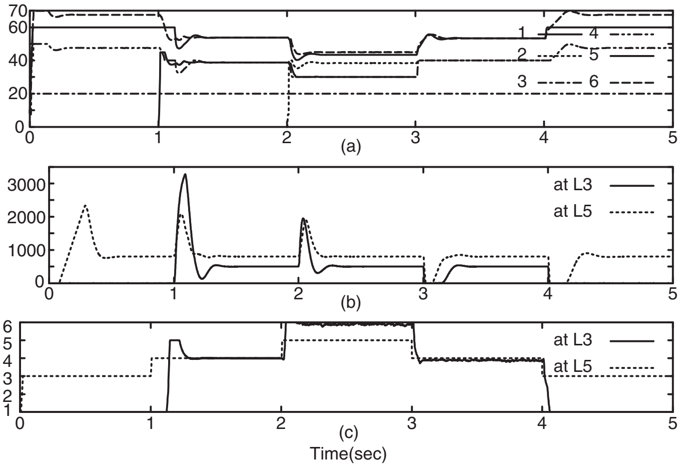

Fig. 5. Results without VBR background traffic: (a) Sender transmission rates(Mbps), (b) Queue length(packets), (c) Estimated number of locally bottlenecked sessions.

IV. OPTIMAL PARAMETERS AND IMPLEMENTATION CONSIDERATIONS

A. Optimal Gain

Of the pairs (U, V) satisfying (20), we find (U, V) at which the asymptotic decay rate of the closed-loop system is maximized. The asymptotic decay rate is dominated by the principal eigenvalue which has the largest real part of the system poles. Thus, the maximum asymptotic decay rate is achieved when the real part of the principal eigenvalue is minimized. We numerically found that the asymptotic decay rate is maximized approximately at $(U, V) = (0.5, 0.1)$. Therefore, we use $(A, B) = (0.5/\bar{d}, 0.1/\bar{d}^2)$ to find a stable and optimal controller gain. See [9] for the detail of the numerical approach.

B. Estimation of $|Q_j|$

Based on the pair (A, B) found in the above subsection, we get the controller gain as $(C_P, C_I) = (A/|\hat{Q}_j|, B/|\hat{Q}_j|)$ where $|\hat{Q}_j|$ is obtained through the estimation of the number of locally bottlenecked sessions $|Q_j|$. Note that the overestimation of $|Q_j|$ is necessary because by Theorem 3.3, if $|Q_j|$ is underestimated, i.e., $\sum_{i \in Q_j} \rho_i > 1$, the system could be unstable. See [9] for the detail of the estimation.

V. SIMULATION RESULTS

In this section, we verify through discrete-event simulations that the proposed algorithm works as designed and the proposed LB(Locality-Based) feedback consolidation algorithm results in the better transient performance than the hard-synchronization based consolidation algorithm [5][1], we call it WFA(Wait For All). The parameter values used throughout the simulation are summarized in Table III.

We examine the proposed algorithm in the multiple-link configuration shown in Fig. 4. There are 6 sessions and 21 VSs, and the VBR background traffic whose sender and receiver are respectively vbrs and vbrr. Multicast session $i(1\sim5)$ has the sender S_i and its receivers $R_{ij}(j{=}1\sim4)$, and the unicast session 6 has one receiver R_{61}. The length of each link connecting a sender and the router RT1 is different from each other, and the maximum length is set to be 800 kilometers. The capacities of the links between senders and RT1 are equally set to 600Mbps to ensure that no sessions are throttled there. To see the effectiveness of our feedback consolidation algorithm when some BCP arrivals are delayed significantly longer than the other BCP arrivals, the length of a receiver access link in each session is set to be 10,000 kilometers while the other receiver access links are equally 50 kilometers long. The top half of Table IV shows the traffic model. We vary MDR, PDR, and arrival and departure time to see their impact on the network performance. From the network topology and traffic model, we can compute the theoretical fair rates over time as summarized in the bottom half of Table IV.

[1] In the WFA consolidation algorithm, each branching node must receive at least one BCP from all participating branches for the feedback consolidation operation.

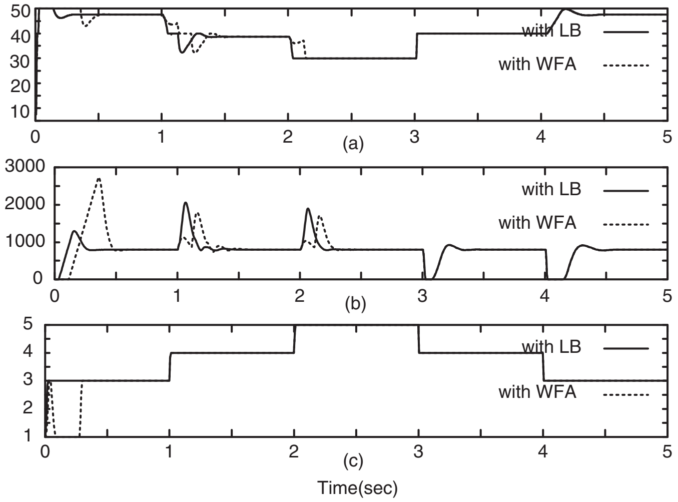

Fig. 6. Results (LB vs WFA): (a) Transmission rate of S_4(Mbps), (b) Queue length at L5(packets),(c) Estimated number of locally-bottlenecked sessions at L5.

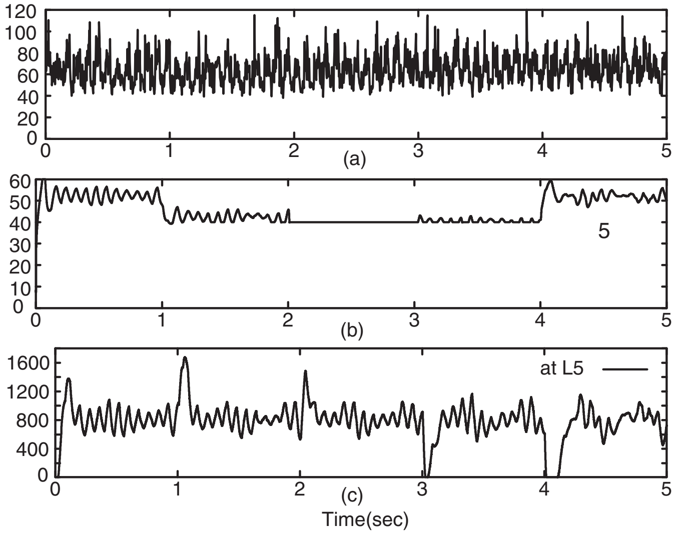

Fig. 7. Results with VBR background traffic: (a) Trace of H.26L VBR video traffic, (b) Transmission rate of S_5(Mbps), (c) Queue length at L5(packets).

The simulation results without VBR background traffic are shown in Fig. 5. Each source's transmission rate in Fig. 5(a) exactly follows the theoretical fair rates given in Table IV although there is a transient period whenever a session arrives or leaves. Fig. 5(b) shows that the queue lengths at L3 and L5 converge to their target values, 500 and 800 packets, in steady state. The reason for the empty queue at L3 in [0,1) and [4,5) is that no sessions are locally bottlenecked at L3 in those periods. This result can also be observed in [0,1) and [4,5) in Fig. 5(c) which shows that there are no locally bottlenecked sessions. Observe that the estimated number of locally bottlenecked sessions in Fig. 5(c) is not equal to the theoretical result in some periods, which implies that $|Q_j|$ is overestimated in those periods. In overall, these results provide an evidence that the local stability condition we found in Section III may serve as the global stability condition as well.

Fig. 6 compares the performance of the two consolidation algorithms. In overall, LB yields better and more rapid transient performance than WFA as we expected in Section II, which is because consolidation delay is smaller in LB.

We also examine how the performance of the proposed algorithm is affected by the VBR background traffic, which is generated by superimposing 21 different H.26L encoded video clips [11] and has the average rate of approximately 60Mbps as in Fig. 7(a). The representative results are shown in Fig. 7(b),(c). Compared to the rate trace of S_5 in Fig. 5(a), the one in Fig. 7(b) is shifted down approximately by 10 Mbps due to the addition of VBR traffic and includes high-frequency fluctuation. Lastly, the queue length shown in Fig. 7(c) fluctuates around its target value 800 packets. In brief, we can conclude that the unpredictable high-frequency traffic can lead to the high-frequency oscillation but never causes the system instability, which means that the performance is well bounded under our control.

VI. Conclusions

In this paper, we proposed a distributed max-min flow control framework for multi-rate multicast flows focusing on the fair rate allocation and the feedback consolidation. The proposed fair rate allocation algorithm is highly scalable because it does not need the individual flow information for the rate computation. In addition, our locality-based feedback consolidation algorithm efficiently reduces the consolidation delay and solves the feedback explosion problem. We mathematically showed that the proposed algorithm achieves the minimum plus max-min fairness, the target queue length and consequently the full link utilization in steady state. Moreover, we found the stability condition in an usable form taking into account the heterogeneous round-trip delays.

References

[1] S. Sakar and L. Tassiulas, "Distributed algorithms for computation of fair rates in multirate multicast trees," in *Proc. IEEE INFOCOM'00*, Tel-Aviv, Israel, Apr. 2000, pp. 52–61.

[2] K. Kar, S. Sakar, and L. Tassiulas, "Optimization based rate control for multirate multicast," in *Proc. IEEE INFOCOM'01*, Anchorage, Alaska, USA, Apr. 2001, pp. 123–132.

[3] S. Sakar and L. Tassiulas, "Fair allocation of utilities in multirate multicast networks," in *Proc. 37th Annual Allerton Conference on Communication, Control and Computing*.

[4] W. Li, "Overview of fine granularity scalability in mpeg-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301–317, Mar. 2001.

[5] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, and S. Kalyanaraman, "Feedback consolidation algorithms for abr point-to-multipoint connections," in *IEEE ATM Forum '97*, 1997.

[6] L. Roberts, "Rate based algorithm for point to multipoint abr services," in *ATM FORUM/94-0772R1*, Nov. 1994.

[7] K. J. Astrom and B. Wittenmark, *Computer Controlled Systems: Theory and Design*. NJ: Englewood Cliffs: Prentice-Hall, 1984.

[8] B. R. Barmish, *New Tools for Robustness of Linear Systems*. New York: MacMillan, 1994.

[9] H. W. Lee, J. w. Cho, and S. Chong, "A distributed max-min flow control algorithm for multirate multicast flows," *http://netsys.kaist.ac.kr/∼mslhw/globecom2004extd.pdf*.

[10] J. w. Cho and S. Chong, "Stabilized max-min flow control using pid and pii^2 controllers," *http://netsys.kaist.ac.kr/∼ggumdol/pidpii2.pdf*.

[11] [Online]. Available: http://trace.eas.asu.edu/h26l/longtraces.html