Weight-Based Fair Intelligent Bandwidth Allocation for Rate Adaptive Video Traffic

TECHNICAL REPORT NUMBER 526

June 2000

Xiaomei Yu, Doan. B. Hoang, David D. Feng
Weight-Based Fair Intelligent Bandwidth Allocation for Rate Adaptive Video Traffic

Xiaomei Yu, Doan B. Hoang, and David D. Feng

Basser Department of Computer Science
The University of Sydney, NSW 2006, Australia

Abstract -- In this paper, we present a weight-based network bandwidth sharing algorithm, Fair Intelligent Bandwidth Allocation (FIBA) for transporting rate-adaptive video traffic using feedback, and report on its performance under a general weight-based and PCR-based share policies. Through simulations, we obtained following results. (1) the FIBA algorithm is capable of allocating bandwidth fairly for the MCR plus weighted allocation fairness criteria and MCR plus allocation proportional to PCR fairness criteria, among competitive, rate-adaptive video sources, (2) the FIBA algorithm is able to reallocate smoothly when there are renegotiations of the minimum guaranteed cell rate, weight, or peak cell rate by some connections, (3) the FIBA algorithm is able to reallocate smoothly when a new connection is admitted, (4) the FIBA algorithm is able to reallocate smoothly when a connection is throttled somewhere earlier along the connection path, (5) the algorithm prevents congestion, especially during the initial periods when buffer queues can build up significantly.

Keywords -- rate adaptive video traffic, fair bandwidth allocation, network bandwidth sharing algorithm, rate-based congestion control.

I. INTRODUCTION

Compressed video traffic is likely to become the most significant component workload of future networks. Compressed video is inherently bursty and will become more so with the advance of compression technology and VLSI chip design. Compressed video sources can adapt their rate dynamically by adjusting compression parameters. Naturally, it suggests that ABR type of feedback control can be employed for more efficient bandwidth sharing.

Attempts have been made to employ ABR-like feedback control as a distributed feedback control for sharing network bandwidth among competing video connections [1-4]. For such an application to be fruitful, two main difficulties have to be overcome. The first difficulty is caused by the source behavior. A rate-adaptive video source does not behave like an ABR-source in that its rate does not change continuously either linearly or exponentially. The source rate jumps from one Explicit Rate to the another as indicated by the feedback mechanism. This makes it difficult for a switch algorithm to converge properly to target fair share allocation. Secondly, to guarantee the quality of service demanded by a rate-adaptive video application, a rate allocation algorithm must be definitive rather than suggestive in allocating bandwidth. This implies that a switch needs to keep more information concerning the availability of the network bandwidth resources and concerning the demand of its connections. Under these circumstances an ABR adaptive mechanism will not adapt well to video traffic with large jumps in their rates, without appropriate considerations concerning convergence to target rate.

In [1, 2], an MCR-proportional max-min policy was proposed to support rate-adaptive video. However, it was unclear of what distributed feedback control algorithm to be used to achieve such network bandwidth sharing policy. A MCR plus weighted allocation bandwidth sharing algorithm is proposed in [4]. The algorithm decouples the source's actual transmission rate from the Allowed Cell Rate variable used for protocol convergence. However, the algorithm can be improved in several aspects.

We proposed a novel distributed feedback control algorithm -- Fair Intelligent Bandwidth Allocation (FIBA) which is computationally less costly and also includes a mechanism to prevent congestion, especially in the transient periods [5].

In this paper, we present a weight-based FIBA algorithm. Through simulations, we show that the FIBA algorithm is capable of allocating bandwidth fairly for the MCR plus weighted allocation fairness criteria, and it is flexible for the renegotiation of MCR and weight, and new connection adding. We also show that it is able to reallocate smoothly when a connection is throttled somewhere earlier along the connection path. Furthermore, the results also show that FIBA can prevent congestion, especially during the initial periods when buffer queues can build up significantly.

Another fairness criteria -- MCR plus allocation proportional to PCR is also presented and studied in this paper. The simulation result shows that the FIBA algorithm is capable of allocating bandwidth fairly for the MCR plus allocation proportional to PCR fairness
criteria. It is able to reallocate smoothly when there is renegotiation of the minimum guaranteed cell rate or peak cell rate by some connections, or there is a new connection admitted.

The paper is organised as follows. The weight-based Fair Intelligent Bandwidth Allocation algorithm is presented in Section II. Section III describes the simulation environment and its parameters. Section IV presents simulation results and analysis. Section V concludes the paper by summarizing the contributions and direction for further research.

II. THE WEIGHTED FAIR INTELLIGENT BANDWIDTH ALLOCATION ALGORITHM

In this section, we first present our target fair share calculation regarding with MCR plus weighted allocation fairness criteria and MCR plus allocation proportional to PCR fairness criteria. We will describe the main ideas employed in the algorithm and then we will state the algorithm. [5] describes various design considerations and the algorithm in detail.

A. Target fair share allocation

To utilise the total link bandwidth we have to know at all time the exact amount of bandwidth available to be allocated. Given the fairness criteria for allocation the target\_fair\_share for a connection can be calculated.

In a MCR plus weighted allocation policy, the target\_fair\_share for a connection with weight \( w \) is given by

\[
\text{target\_fair\_share} = \text{mcr} + \frac{C - \text{sum\_mcr}}{\text{sum\_w}} \times w,
\]

where \( C \) is the link capacity, \( \text{sum\_mcr} \) and \( \text{sum\_w} \) is the sum of all the MCR, and the sum of all the weight for all connections passing through the switch, respectively.

Consider the case of target fair share limited by its PCR or by bottleneck from other switches along the path of the connection, target fair share is actually calculated as

\[
\text{target\_fair\_share} = \text{mcr} + \frac{C - \text{sum\_mcr}}{\text{sum\_w}} \times w + \frac{\text{spare\_per\_w\_sum}}{\text{sum\_per\_per\_sum}},
\]

where \( \text{spare\_per\_w\_sum} \) is unused bandwidth from constrained connections per the unit weight of the rest, unconstrained connections.

In a MCR plus allocation proportional to PCR policy, the target\_fair\_share for a connection is given by

\[
\text{target\_fair\_share} = \text{mcr} + \frac{C - \text{sum\_mcr}}{\text{sum\_per}} \times \frac{\text{per}}{\text{sum\_per} + \text{spare\_per\_per\_sum}},
\]

where \( \text{sum\_pcr} \) is the sum of all the PCR for all connections passing through the switch, and \( \text{spare\_per\_pcr\_sum} \) is unused bandwidth from constrained connections per the unit PCR of the rest, unconstrained connections.

B. The basic ideas of the FIBA

The aim is to employ an ABR-like feedback control mechanism, but taking into account various constraints related to peak cell rate, congestion bottleneck, and available bandwidth, to allocate bandwidth fairly and efficiently to all connections. The essential ideas of the Fair Intelligent Bandwidth Allocation are as follows:

- The algorithm has to guarantee a minimum rate to all connections that are admitted to the network by the network admission control procedure.
- It has to keep track of the amount of bandwidth available for allocation.
- It has to keep a running average account of current share allocation for each connection.
- It has to observe the current transmission rate of each connection and make appropriate adjustment to bring the current share allocation of each connection to the target fair share for that connection.
- If some connections are restricted by their peak rate and cannot take up their allocated fair share, the unused bandwidth is allocated fairly among the rest of the connections.
- In case where a connection is throttled by a switch along its path to the destination, the algorithm has monitor the situation and reallocate bandwidth fairly.
- In case where congestion is a problem, the algorithm exercises its congestion control mechanism to stabilize the network around a target operating point.

C. Algorithms

The weight-based Fair Intelligent Bandwidth Allocation algorithm is shown below.

Algorithm 1. End System Behaviour

Source Behaviour:

- The source starts with \( \text{ACR}=\text{ICR}, \text{ICR}=\text{MCR} \);
- For every Nrm transmitted data cells, the source sends a RM cell (\( \text{CCR}=\text{ACR}, \text{MCR}=\text{MCR}, \text{ER}=\text{PCR}, \text{TACR}=\text{PCR}, \text{W}=\text{W} \));
- Upon the receipt of a backward RM cell, set \( \text{ACR}=\text{ER} \)

Destination Behaviour:

- The destination end system of a connection
Algorithm 2. Switch algorithm

If cell is RM (DIR = forward....)
if RM (acr, mcr, er, tacr, w,..) cell signals connection initiation
{ /* acr, mcr, er, pcr, tacr = per, w values here apply to individual VCI */
    mcsa = mcr; sum_mcr = mcr + sum_mcr;
    sum_w = partial_sum_w = w + sum_w;
    spare_per_w_sum = 0; x = 1;
} else
{
    if (mcr or weight has been changed) {
        recalculate sum_mcr or sum_w, and set
        partial_sum_w = sum_w;
        spare_per_w_sum = 0; x = 1;
    }
    if (last_tacr != 0) and (last_tacr != tacr ) {
        target_fair_share = u0 * w + mcr;
        /* recalculate fair share for the connection */
        partial_sum_w = sum_w;
        spare_per_w_sum = 0;
    }
    calculate ( );
    if (target_fair_share > tacr) {
    /* the connection cannot take up its fair share because of bottleneck restriction elsewhere */
        partial_sum_w = partial_sum_w - w;
        spare_per_w_sum = spare_per_w_sum + (target_fair_share-tacr) / partial_sum_w;
        target_fair_share = last_tacr = tacr;
    }
    tacr = target_fair_share;
    /* pulling the mean current share allocation towards the target_fair_share */
    if (abs(mcsa - target_fair_share)) < e
    /* e is a small real value in the order of 10^-3 */
        mcsa = target_fair_share
    else if (mcsa > target_fair_share)
        mcsa = mcsa - ((mcsa + acr) / 2 - target_fair_share) *AV;
    else
        mcsa = mcsa + (target_fair_share - (mcsa + acr) / 2) *AV;
}

If cell is RM (DIR = backward....)
Q0 = BUR * BufferSize;
if (QueueLength > Q0 )
DPF = (BufferSize - QueueLength) / (BufferSize -Q0);
else
    DPF = (a - 1)(Q0 - QueueLength) / Q0 + 1;
ER= Min (ER, DPF * mcsa)

Calculate ( )
{
    spare = temp = 0;
    /* spare, temp are temporary parameters */
    for VCs passing swi and traversing link i
    {
        if (this VC can take up its share of the spare bandwidth) or (x == 1) {
            u0 = (C - sum_mcr)/sum_w +
                spare_per_w_sum;
            target_fair_share = u0*w + mcr;
            spare_per_w_sum = spare_per_w_sum +
                target_fair_share = pcr;
            temp = temp + w;
        }
        if (target_fair_share > pcr) {
            /* target_fair_share is restricted by peak cell rate */
            spare = spare + target_fair_share - pcr;
            target_fair_share = pcr;
            temp = temp + w;
        }
    }
    if (spare > 0) {
        /* spare bandwidth due to pcr constraint */
        partial_sum_w = partial_sum_w - temp;
        spare_per_w_sum = spare_per_w_sum +
                        spare/partial_sum_w;
    }
    x =0;
}

Parameters used in the algorithm
- x: x is set to 1 if there is a new connection, or mcr/weight has been changed. It will be reset to 0 if the new connection calculation is completed.
- mcsa: mean current share allocation.
- last_tacr: is used to identify the changing of tacr.
- u0 = the fair share per unit weight = (Capacity - (Sum of MCR))/(Sum of Weight) + (spare bandwidth)/(Sum of Weight for connections which share the spare bandwidth).
- sum_mcr: sum of the MCR of all connections.
- partial_sum_w: sum of weight of connections that are not limited by PCR or by bottleneck elsewhere.
- spare_per_w_sum: unused bandwidth from connections that are constrained by PCR, and by bottleneck from other switches along the path of the connection per the unit weight of the rest, unconstrained connections.
- target_fair_share: is target rate. target_fair_share = u0*w + mcr.
III. SIMULATION ENVIRONMENT

Our simulation run on the ATM HFC Network Simulator [8] developed by the National Institute of Standards and Technology (NIST). There are two network configurations employed in the simulation: a three-node configuration (Fig. 1), and a parking lot configuration (Fig. 2). For the networks in the simulation, all ATM switches are assumed to have output port buffering. Each output port employs the simple first-in-first-out queueing discipline for all cells destined to that port. The simulation parameters are shown in Table 1.

IV. SIMULATION RESULTS

In this section we present the simulation results of the weight-based FIBA. We mainly studied the Allowed Cell Rate for each connection, and the maximum switch buffer queue length for several cases: 1) parking lot network, 2) three-node network, which used to test the case when a connection is throttled somewhere earlier along the connection path, 3) renegotiation of MCR, 4) renegotiation of weight, 5) adding new connections, 6) setting ICR=PCR to see the buffer queue control ability of FIBA. For case 2 and 4, we also study the effect of connection rate constraints on the final target fair share.

The simulation results of FIBA with MCR plus allocation proportional to PCR policy is also presented for 3 cases: 1) parking lot network, 2) renegotiation of MCR, 3) renegotiation of PCR.

<table>
<thead>
<tr>
<th>Component</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>Speed</td>
<td>10 Mbps</td>
</tr>
<tr>
<td></td>
<td>Distance</td>
<td>1000 Km</td>
</tr>
<tr>
<td></td>
<td>Delay</td>
<td>5 ms/Km</td>
</tr>
<tr>
<td>Interswitch Link</td>
<td>Speed</td>
<td>100 Mbps</td>
</tr>
<tr>
<td></td>
<td>Distance</td>
<td>1 Km</td>
</tr>
<tr>
<td></td>
<td>Delay</td>
<td>5 ms/Km</td>
</tr>
<tr>
<td>Host-Switch Link</td>
<td>Speed</td>
<td>MCR</td>
</tr>
<tr>
<td></td>
<td>Nrm</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>Input Queue Size</td>
<td>3000 cells</td>
</tr>
<tr>
<td>Host</td>
<td>ICR (Initial Cell Rate)</td>
<td>4 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5000 cells</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.05</td>
</tr>
</tbody>
</table>

A. FIBA with MCR plus weighted allocation policy

1) Parking Lot Configuration

For this configuration as shown in Fig.2, there are 4 connections started from several switches, but they terminate at the same switch. The output port links of the last two switches are potential bottleneck links. The target fair share for each connection in the parking lot configuration is limited by the Switch 3. Fig. 3 shows the ACR for each connection. Fig. 4 shows the maximum queue length at Switch 1 and Switch 2 and Switch 3. Table 2 shows the minimum rate requirement, the peak cell rate, weight, the target fair share allocation and the Allowed Cell Rate at the source. The target fair share figure is used to judge how well the algorithm allocates cell rate to connections.

Figure 1. A three-node network.

Figure 2. A parking lot network.

Figure 3. The ACR of all connections for the parking lot configuration.
2) Three-node Configuration

In three-node configuration, only VC1 passes through all of switches. This configuration can be used to test the case of a connection is throttled somewhere earlier along the connection path. The smallest target fair share of VC1 among the target fair shares of VC1 calculated at Switch1 and Switch 2 will be taken as final target fair share of VC1. We use RM cell to transmit this value in our algorithm. Fig. 5 shows the ACR for each connection. Fig. 6 shows the target fair share for all connections. At initial, the target fair share of VC3 is 5, which is bigger than its PCR (4). The algorithm sets the target fair share of VC3 as its PCR and calculated the spare bandwidth and allocated it to other connections. Fig. 7 shows the maximum queue length at Switch 1 and Switch 2. Table 3 shows the MCR, PCR, weight, the target fair share and the Allowed Cell Rate at the source.

3) MCR Renegotiation

Each time when a connection changes its minimum cell rate, the target_fair_share allocation for all connections in the network will change. However, it has to be checked that the sum of the new set of MCRs does not exceed the link’s capacity on any link it traverses. If this condition is not satisfied, then the newly negotiated minimum rate is rejected. Our algorithm is able to reiterate and converge to a new set of fair share for all connections. For the three-node configuration, Fig. 8 shows the ACR for each connection before and after the MCR renegotiation. The change of MCR for VC1 from 0.5 Mbps to 2.5 Mbps occurs at time $t = 300$ ms. Table 4 shows the MCR, PCR, weight, the target fair share and ACR at source for all connections before and after the MCR renegotiation.
TABLE 4
The MCR, Renegotiated MCR, The Target Fair Share and ACR at Source for All Connections Before and After MCR Renegotiation

<table>
<thead>
<tr>
<th>MCR (Mbps)</th>
<th>PCR (Mbps)</th>
<th>Weight</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>before</td>
<td>after</td>
<td>before</td>
<td>after</td>
<td>before after</td>
</tr>
<tr>
<td>VC1</td>
<td>0.5</td>
<td>2.5</td>
<td>7.5</td>
<td>1 1.5 3 1.51 3.01</td>
</tr>
<tr>
<td>VC2</td>
<td>1.5</td>
<td>1.5</td>
<td>9</td>
<td>3 4.5 3 4.51 3.01</td>
</tr>
<tr>
<td>VC3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4 4 4 4</td>
</tr>
<tr>
<td>VC4</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>2 8.5 7 8.5 7.02</td>
</tr>
</tbody>
</table>

4) Weight Renegotiation

The target fair share allocation for all connections in the network will be recalculated for each time a connection change its weight. Our algorithm is able to reallocate the new optimal vector for connections. For the three-node configuration, the change of weight for VC1 from 1 to 8 occurs at time \( t = 300 \) ms. Fig. 9 shows the target fair share for all connections. We can see that the algorithm recalculates the target fair share very quickly after the weight is changed. Fig. 10 shows the ACR for each connection before and after the weight renegotiation. Table 5 shows the MCR, PCR, weight, target fair share and ACR at source for all connections before and after weight renegotiation.

TABLE 5
The Weight, MCR, PCR, The Target Fair Share, and ACR at Source for All Connections Before and After Weight Renegotiation

<table>
<thead>
<tr>
<th>Weight</th>
<th>MCR (Mbps)</th>
<th>PCR (Mbps)</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>before</td>
<td>after</td>
<td>before</td>
<td>after</td>
<td>before after</td>
</tr>
<tr>
<td>VC1</td>
<td>1</td>
<td>8</td>
<td>0.5</td>
<td>1.5 3.7 1.51 3.71</td>
</tr>
<tr>
<td>VC2</td>
<td>3</td>
<td>3</td>
<td>1.5</td>
<td>9 4.5 2.7 4.51 2.71</td>
</tr>
<tr>
<td>VC3</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2 4 3.6 4 3.62</td>
</tr>
<tr>
<td>VC4</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1 10 8.5 6.3 8.52 6.51</td>
</tr>
</tbody>
</table>

5) Introduction of new connection

When a new connection is accepted into the network, it brings with it an additional MCR demand and weight demand. The algorithm has to take this requirement into account and has to readjust the fair share for all connections. For the three-node configuration, Fig. 11 shows that algorithm can cope with this situation smoothly. Table 6 shows the MCR, PCR, weight, target fair share, and ACR for all connections after the introduction of VC1 at time \( t = 300 \) msec.
TABLE 6
MCR, PCR, WEIGHT, THE TARGET FAIR SHARE AND ACR AT SOURCE FOR ALL CONNECTIONS IN THE THREE-NODE NETWORK. VC1 START AT t = 300 ms

<table>
<thead>
<tr>
<th></th>
<th>MCR (Mbps)</th>
<th>PCR (Mbps)</th>
<th>Weight</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC1</td>
<td>0.5</td>
<td>7.5</td>
<td>1</td>
<td>before 1.5</td>
<td>after 1.5</td>
</tr>
<tr>
<td>VC2</td>
<td>1.5</td>
<td>9</td>
<td>3</td>
<td>4.5</td>
<td>6.01</td>
</tr>
<tr>
<td>VC3</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>VC4</td>
<td>1</td>
<td>10</td>
<td>2</td>
<td>10</td>
<td>8.5</td>
</tr>
</tbody>
</table>

6) Prevent queues from building up

For three-node network, we set ICR=PCR at source to test the performance of switch buffer control provided by the FIBA. There are 7.5 + 9 + 4 = 20.5 Mbps passing through the Link1 (C = 10 Mbps). And for Link2 (C = 10Mbps), there are 7.5 + 10 = 17.5 Mbps passing through. Congestions occur at Switch1 and Switch2. Fig. 12 shows the queue lengths of Switch1 and Switch2 with ICR=PCR in the three-node network. The maximum queue length of Switch 1 is 283 cells, and Switch 2 is 138 cells. Fig. 13 shows the ACR of each connection with ICR=PCR. The result shows that the FIBA can prevent queues from building up in the transient periods.

B. FIBA with MCR plus allocation proportional to PCR policy

1) Parking Lot Configuration

Fig. 14 shows the ACR of each connection for PCR-based sharing policy. Fig. 15 shows the maximum switch buffer queue length at Switch 1 and Switch 2 and Switch 3. Table 7 shows the MCR, PCR, the target fair share, and ACR at the source.

2) MCR Renegotiation

Fig. 16 shows the ACR for each connection before and after the MCR renegotiation. The change of MCR for VC1 from 1.5 Mbps to 0.5 Mbps occurs at time t = 300 ms. Table 8 shows the MCR, PCR, the target fair share and ACR at source before and after MCR renegotiation.

---

![Figure 12](image12.png)

Figure 12. Queue Length of Switch1 and Switch2 with ICR=PCR in three-node network.

![Figure 13](image13.png)

Figure 13. ACR for all connections with ICR=PCR in three-node network.

![Figure 14](image14.png)

Figure 14. The ACR of all connections for PCR-based policy in the parking lot configuration.

![Figure 15](image15.png)

Figure 15. The maximum switch buffer queue length of Switch 1, Switch 2 and Switch 3.

---

<table>
<thead>
<tr>
<th></th>
<th>MCR (Mbps)</th>
<th>PCR (Mbps)</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC1</td>
<td>1.5</td>
<td>3.5</td>
<td>2.86</td>
<td>2.87</td>
</tr>
<tr>
<td>VC2</td>
<td>1</td>
<td>2</td>
<td>1.77</td>
<td>1.78</td>
</tr>
<tr>
<td>VC3</td>
<td>1</td>
<td>5</td>
<td>2.94</td>
<td>2.95</td>
</tr>
<tr>
<td>VC4</td>
<td>0.5</td>
<td>5</td>
<td>2.44</td>
<td>2.45</td>
</tr>
</tbody>
</table>

---

7
TABLE 8
THE MCR, PCR, THE TARGET FAIR SHARE, AND ACR BEFORE AND AFTER MCR RENEGOTIATION

<table>
<thead>
<tr>
<th></th>
<th>MCR (Mbps)</th>
<th>PCR (Mbps)</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>before</td>
<td>after</td>
<td>before</td>
<td>after</td>
</tr>
<tr>
<td>VC1</td>
<td>1.5</td>
<td>0.5</td>
<td>3.5</td>
<td>2.86</td>
</tr>
<tr>
<td>VC2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1.77</td>
</tr>
<tr>
<td>VC3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>2.94</td>
</tr>
<tr>
<td>VC4</td>
<td>0.5</td>
<td>0.5</td>
<td>5</td>
<td>2.44</td>
</tr>
</tbody>
</table>

3) PCR Renegotiation

Fig. 17 shows the ACR for each connection before and after PCR renegotiation in the parking lot network. The change of PCR for VC1 from 3.5 Mbps to 9 Mbps occurs at time $t = 300$ ms. Table 9 shows the MCR, PCR, the target fair share and ACR at source for all connections before and after PCR renegotiation.

![Figure 17. The ACR of all connections for the parking lot configuration. VC1 changed its PCR from 3.5 Mbps to 9 Mbps at $t = 300$ ms.](image)

C. Discussion on the Results

Consider again the simulation results for the three-node configuration in IV-A-2. VC1 passes through Switch 1 and Switch 2. Initially, the target fair share of VC1 at switch 2 is 3.333 Mbps. When the target fair share of VC1 at switch 1, which is 1.5 Mbps, arrives at Switch 2, the algorithm compares two values and takes the smallest one as the final target fair share, and allocates the unused bandwidth to other connections.

Also in the simulation, the initial target fair share of VC3 is bigger than its PCR, so VC3 cannot take up is allocated share, the algorithm is able to recalculate the target faire share and allocate the unused bandwidth as shown in Fig. 6. We have demonstrated that our feedback algorithm is able to reallocate smoothly when a connection is throttled somewhere earlier along the path, or when a connection is limited by its peak cell rate.

From the above simulation, we've shown that our feedback algorithm is able to reiterate and converge to a new set of fair share for all connections when MCR, weight or PCR is renegotiated, or when a new connection is admitted.

From Fig. 6, it is observed that the algorithm converges to the final target fair share for all connections after few round trip times. From Fig. 9, it is also observed that the algorithm converges to the final target fair share in a similar fashion, when there is a weight change at $t = 300$ ms. The rate and the smoothness of the convergence of the algorithm can be adjusted by the averaging factor, $A$ and the queue control parameter $\alpha$. Larger values of $A$ tend to speed up the convergence process, but generally increase the fluctuations. The factor $\alpha$ used in the algorithm could shorten the convergence time significantly. With larger the value of $\alpha$, the harder the algorithm pushes the current mean share allocation towards the final target fair share allocation.

![Figure 16. The ACR of all connections for the parking lot configuration. VC1 changed its MCR from 1.5 Mbps to 0.5 Mbps at $t = 300$ ms.](image)

TABLE 9
THE MCR, PCR, RENEGOTIATED PCR, THE TARGET FAIR SHARE AND ACR BEFORE AND AFTER PCR RENEGOTIATION

<table>
<thead>
<tr>
<th></th>
<th>PCR (Mbps)</th>
<th>MCR (Mbps)</th>
<th>Target Fair Share (Mbps)</th>
<th>ACR at Source (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>before</td>
<td>after</td>
<td>before</td>
<td>after</td>
</tr>
<tr>
<td>VC1</td>
<td>3.5</td>
<td>9</td>
<td>1.5</td>
<td>2.86</td>
</tr>
<tr>
<td>VC2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.77</td>
</tr>
<tr>
<td>VC3</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2.94</td>
</tr>
<tr>
<td>VC4</td>
<td>5</td>
<td>5</td>
<td>0.5</td>
<td>2.44</td>
</tr>
</tbody>
</table>
For congestion control, a switch buffer queue control scheme is employed to keep the queue length to some predetermined operating level \((Q_0 = BUR * Buffer\ size)\). When congestion is present, the network will be pushed to operate above \(Q_0\), the queue control factor DPF will then be reduced to pull the network back around \(Q_0\). From above simulations, we’ve observed that our algorithm is able to maintain the switch queue length around the \(Q_0\) level. And from the simulation results presented in IV-A-6, where there are bottleneck links, it is observed that our algorithm is able to prevent queues from building up in the transient periods.

From simulation results presented, we’ve also demonstrated that FIBA algorithm can be employed effectively with different fairness criteria: MCR plus weighted allocation policy and MCR plus allocation proportional to PCR policy.

V. CONCLUSION

Explicit rate ABR feedback control scheme provides several important features for transporting rate-adaptive video. It preserves the simplicity of CBR connection admission control. Rate renegotiation of guaranteed minimum cell rate can be done more frequently and effectively by way of RM cells. Guaranteed minimum cell rate can ensure acceptable quality even in periods of congestion. The information returned in the RM cells may be used to adapt the bit rate of the video decoder appropriately.

This paper presents a weight-based network bandwidth sharing algorithm, FIBA for transporting rate-adaptive video using feedback control. The fairness criteria presented in this paper are MCR plus weighted allocation policy and MCR plus allocation proportional to PCR policy. We have shown that FIBA is capable of allocating bandwidth fairly for these fairness criteria. We’ve demonstrated that FIBA is able to reallocate smoothly when a new connection is admitted or when there are renegotiations of the minimum cell rate, weight, or peak cell rate by some connections, or when a connection is throttled somewhere earlier along the connection path. We have also shown that the algorithm can prevent queue from build up in the initial periods.

Several issues may require further consideration. For real-time video transporting, a suitable source rate adaptation control algorithm relative with image quality deserves attention. Other fairness criteria may also be considered. It is demonstrated that ABR-like explicit rate feedback mechanism is feasible and desirable for transporting rate adaptive traffic, an investigation into the performance of the mechanism when VBR, ABR services are combined, would be of great interest.

REFERENCES


