Distributed and Optimal Congestion Control for Application-layer Multicast: A Synchronous Algorithm

Jinyao Yan*†,Martin May*, Bernhard Plattner*

*Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, ETH Zurich, CH-8092, Switzerland

†Computer and Network Center, Communication University of China, Beijing 100024, China

***Abstract*— In this paper, we study distributed and optimal congestion control for scalable video streams in application-layer multicast(ALM) . We propose a TCP friendly and fully distributed synchronous algorithm based on the utility-price model which maximizes the global utilities for the streams in the application-layer multicast tree. With the help of numerical study, we show our proposed algorithm optimizes the overall video quality for fine-grained scalable streams, while minimizing the messaging overhead in the application-layer multicast channel.**

I. Introduction

IP multicast is a networking technology, addressing the problem of efficient delivery of data over the Internet to a large number of receivers. Unfortunately, IP multicast is still not yet widely deployed in the public Internet, mainly due to deployment issues (technical and commercial). To overcome the technical deployment problems, a new type of multicast solution was developed: application-layer multicast, first presented in [5]. In application-layer multicast, end hosts implement the multicast functionality without relying on the support from the IP routers. In addition, application-layer multicast enables the flexibility of adding higher layer functionality, such as rate scaling at intermediate nodes.

To deal with the diverse and changing network conditions for the multicast streaming channel, congestion control protocols are used to adapt the sending rate such that the current available network resources are neither overloaded nor underutilized. However, in existing congestion control approaches for application-layer multicast, intermediate nodes determine the downstream sending rate without considering the application quality and other information from neighbor and parent nodes. For example in End System Multicast [5],the transmission rate for each flow is calculated locally using the unicast TCP-friendly rate control algorithm (TFRC) [2], considering neither the structure of the multicast tree nor streaming quality.

Therefore, the challenge is to develop an optimal and distributed congestion control algorithm of application-layer multicast with the goal to optimize the global utility (video-quality) of all receiving nodes in the multicast tree instead of optimizing the utilization of network resources of each unicast flow locally.

In the paper, we design a distributed congestion control algorithm based on the pricing model [3] for application-layer multicast, in particular for tree-based application-layer multicast for example Overcast [14] and NICE [15]. The advantages of our algorithm are

1) It maximizes the overall video quality for scalable video streams in the tree;
2) It is TCP friendly to other coexisting traffic according to the definition in 2.A;
3) And most importantly, our algorithm is fully distributed with the minimized messaging overhead and, in terms of complexity, easy to be implemented on end systems.

The pricing model proposed in [3] has already been applied to IP multicast in [10] and overlay multicast in Cui's algorithm [4]. In [4] however, the authors design a congestion control algorithm for overlay multicast that creates significant message overhead, where the flow rates and all physical link prices are explicitly exchanged with some centralized nodes for the optimal rate computation. Moreover, the bandwidth of TCP connections sharing the same links are not considered; hence, their algorithm is not TCP friendly.

II. Distributed and Optimal Congestion Control Algorithm

A. Network model and Problem Formulation

Consider an overlay network of $n+1$ end hosts, denoted as $H = \{h0, h1, \ldots, hn\}$. End host $h0$ is the source of the multicast channel. Other end hosts are consumers of the multicast channel. The structure of the overlay tree is given by the application-layer multicast protocol used. Non-leaf nodes are forwarding streaming data to its children and able to scale-down the streams, fulfilling the constraint of flow preservation. For our model, we assume that streams are fine-grained scalable [7]. The multicast channel consists of n end-to-end unicast flows, denoted as $F = \{f1, \ldots, fn\}$. Flow fi is the flow that terminates at hi. Each flow $f \in F$ has a rate

x_f. We collect all the x_f into a rate vector $x = (x_f, f \in F)$. We denote $U_f(x_f)$ as the utility of flow f, when f transmits at rate x_f. Let $I_f = [m_f, M_f]$ denote the rate range of flows. We assume that U_f is strictly increasing and concave, and twice continuously differentiable within I_s. F'_h is the set of flows sent from h. If a host h is the destination of a flow f_h and the source of another flow $f'_h \in F'_h$ then f'_h is the child flow of f_h, denoted as $f_h \to f'_h$. We denote h' is the child of h and h^p is the parent node of h, *i.e.* $h^p \to h \to h'$.

Now,let us suppose that the overlay network consists of L bottleneck links, denoted as $\Gamma = \{1, 2, \ldots, L\}$. The TCP friendly available bandwidth for the multicast channel at each bottleneck link $l \in \Gamma$ in the direction is c_l. Note that the bottleneck link is directional. We store all the c_l in vector $C = (c_l, l \in \Gamma)$. We assume each flow f has a single bottleneck at particular point of time [8] [11] [12], denoted as $l(f) \in \Gamma$. For each bottleneck link l, $F(l) = \{f \in F \mid l(f) = l\}$ is the set of flows in the channel that pass through it, and we assume that they are sibling flows when the overlay tree is well formed.

We define further a $\Gamma \times F$ matrix A. $A_{lf} = 1$, if flow f goes through bottleneck link l in the direction, *i.e.*,$l = l(f), f \in F(l)$. Otherwise, $A_{lf} = 0$. It follows that the rate summation of all flows **in the direction** in the channel that go through the bottleneck link l should not exceed c_l . Formally, such TCP friendly available bandwidth constraint at bottleneck link is expressed as follows:

$$A \cdot x \leq C \tag{1}$$

Definition 1: A congestion control algorithm for application-layer multicast is TCP friendly, if and only if the coexisting TCP traffic achieves no fewer throughputs than what they would achieve if all flows of the application-layer multicast channel were using TCP as congestion control algorithm.

Let $T^h_{f'_h}$ be the TCP friendly available bandwidth for unicast for f'_h at the bottleneck link $l(f'_h)$ measured by TFRC algorithm. We collect all $T^h_{f'_h}$ into vector $T^h = (T^h_{f'_h}, f'_h \in F'_h)$. C^h is the vector of TCP friendly available bandwidth for multicast channel for F'_h, $C^h = A^h \cdot T^h$. A^h is a $\Gamma_h \times F'_h$ Matrix, where $\Gamma_h = \{l(f'_h) \mid f'_h \in F'_h\}$. If f'_h flow goes through the bottleneck link $l_h \in \Gamma_h$ in the direction, *i.e.*,$l_h = l(f'_h)$, then $A^h_{l_h f'_h} = 1$, otherwise $A^h_{l_h f'_h} = 0$. The location of the bottleneck of f'_h can be inferred as in [6] [9]. Since we assume only sibling flows share bottlenecks, namely non-sibling flows are independent, inequality (1) can be decomposed into:

$$A \cdot x \leq C \Leftrightarrow A^h \cdot x^{F'_h} \leq C^h \Leftrightarrow A^h \cdot x^{F'_h} \leq A^h \cdot T^h \tag{2}$$

, where vector $x^{F'_h} = (x_{f'_h}, f'_h \in F'_h), \forall h \in H$.

Moreover, the rate of downstream is constrained by the rate of upstream, namely, if $f \to f'$, then $x_{f'} \leq x_f$. We define the data constraint or flow preservation $F \times F$ matrix B. $B_{f_1 f_2} = -1$,if $f_2 \to f_1$,*i.e.*$f_1 = f'_2$; $B_{f_1 f_2} = 1$, if $f_1 = f_2$, and f_1 has a parent flow; Otherwise $B_{f_1 f_2} = 0$. Hence, given the application-layer multicast tree, the data constraint can be formalized as follows:

$$B \cdot x \leq 0 \tag{3}$$

As shown in figure 1(a) and 1(b), we use the same example in Cui's algorithm [4] to illustrate our model compared with theirs. In [4], authors assume links and flows are undirected. In reality, flows are directed. Therefore, in particular for link 3 and link 5, the link capacity constrains the flows that pass through it in each direction **independently** as indicated in inequality (4)instead of inequality (5). The analysis in [13] showed the directed link model leads to better accuracy than the undirected link model.

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} \leq \begin{pmatrix} 6 \\ 3 \\ 8 \\ 8 \\ 15 \\ 10 \\ 10 \\ 2 \\ 2 \end{pmatrix} \tag{4}$$

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} \leq \begin{pmatrix} 6 \\ 3 \\ 8 \\ 15 \\ 10 \\ 2 \\ 2 \end{pmatrix} \tag{5}$$

In Fig.1, there are five end-to-end unicast flows (F =5). The network consists of 4 directional bottleneck links (L=4). Hence, TCP friendly available bandwidth constraint at bottleneck link in our model, *i.e.* inequality (1), becomes:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} \leq \begin{pmatrix} 6 \\ 8 \\ 2 \\ 2 \end{pmatrix} \tag{6}$$

Since only sibling flows may share bottlenecks, namely only x_1 and x_2, x_4 and x_5 may share bottlenecks. As shown in (2), inequality (6) can be decomposed into:

$$\begin{cases} \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 3 \end{pmatrix} \\ \begin{pmatrix} 1 \end{pmatrix} \begin{pmatrix} x_3 \end{pmatrix} \leq \begin{pmatrix} 1 \end{pmatrix} \begin{pmatrix} 8 \end{pmatrix} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_4 \\ x_5 \end{pmatrix} \leq \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 2 \end{pmatrix} \end{cases}$$

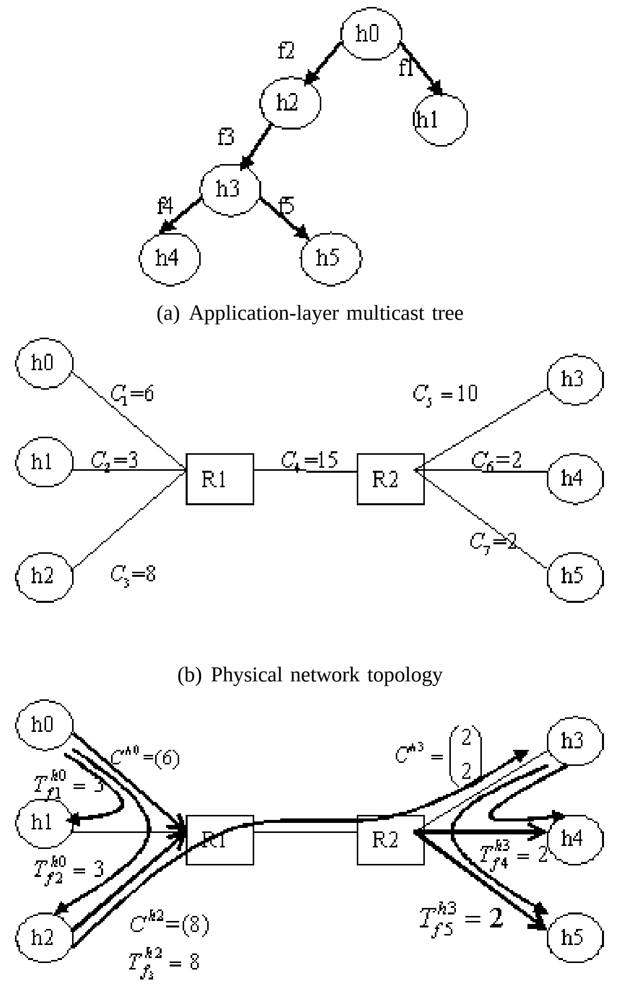
$$\Leftrightarrow \begin{cases} x_1 + x_2 \leq 6 \\ x_3 \leq 8 \\ x_4 \leq 2 \\ x_5 \leq 2 \end{cases} \tag{7}$$

Now, we find out the inequality (7) from our model is much simpler than the full link capacity constraint inequality (4).

The decomposed inequality (7) makes our proposed algorithm fully distributed with the lowest message complexity.

And inequality (3) in this example becomes:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} \leq 0 \tag{8}$$



(a) Application-layer multicast tree

(b) Physical network topology

(c) TCP friendly available bandwidth constraints (Bold links are bottlenecks in the arrow direction,no TCP cross traffic introduced in this example)

Fig. 1. Illustrating sample of the proposed network model (The unit used for bandwidth is Mbps)

We collect the notations in the model into table 1.

Problem Formulation: Our objective is to devise a distributed congestion control algorithm that maximizes the total utility, *i.e.*, the overall video quality of all streams in the application-layer multicast tree:

$$\max_{m_f \leq x_f \leq M_f} \sum_f U_f(x_f) \tag{9}$$

And that fulfills the following constraints:

$$y = \begin{cases} A \cdot x \leq C \\ B \cdot x \leq 0 \end{cases}$$

B. Algorithm

Solving the problem (9) directly requires coordination among sources. To get a distributed solution, we solve its dual problem. Then, by the Kuhn-Tucker theorem, we obtain the maximizer [3] [4]:

$$x_f(\mu^\alpha, \mu^\beta) = [U_f'^{-1}(\lambda_f^\alpha + \lambda_f^\beta)]_{m_f}^{M_f} \tag{10}$$

Where $\mu^\alpha = (\mu_l^\alpha, l \in L)$ and $\mu^\beta = (\mu_f^\beta, f \in F)$ are vectors of Lagrangian multipliers.

Vectors $\lambda^\alpha = (\lambda_f^\alpha, f \in F)$ and $\lambda^\beta = (\lambda_f^\beta, f \in F)$ are defined as follows:

$$\lambda_f^\alpha = \sum_{l=l(f)} \mu_l^\alpha = \mu_{l(f)}^\alpha \tag{11}$$

$$\lambda_f^\beta = \mu_f^\beta - \sum_{f \to f'} \mu_{f'}^\beta \tag{12}$$

For μ^α, μ_l^α can be understood as the link price of bottleneck link l. Consequently, for λ^α, λ_f^α is the bottleneck link price that f has to pay for its single bottleneck, namely $\mu_{l(f)}^\alpha$. For μ^β, μ_f^β is the relay price that f must pay its parent flow f^p for relaying data to f. If f has no parent flow, then $\mu_f^\beta = 0$. Meanwhile, for f^p, μ_f^β can be understood as its relay benefit from f. For λ^β, we can interpret λ_f^β as data price of f, which is the relay price μ_f^β subtracts the relay benefit from all its children $\sum_{f \to f'} \mu_{f'}^\beta$.

We solve its dual problem using the gradient projection method [3]. γ is the step size. We can get:

$$\mu_{l(f)}^\alpha(t+1) = [\mu_{l(f)}^\alpha(t) + \gamma(\sum_{f \in F(l)} x_f(\mu^\alpha(t), \mu^\beta(t)) - c_{l(f)})]^+ \tag{13}$$

$$\mu_f^\beta(t+1) = [\mu_f^\beta(t) + \gamma(x_f(\mu^\alpha(t), \mu^\beta(t)) - x_{f^p}(t))]^+ \tag{14}$$

Equation (13) is consistent with the law of supply and demand: if the demand $\sum_{f \in F(l)} x_f$ for bandwidth at bottleneck link $l(f)$ exceeds its TCP friendly supply $c_{l(f)}$ for the channel, the TCP friendly bandwidth constraint is violated. Thus, the link price $\mu_{l(f)}^\alpha$ is raised. Otherwise, $\mu_{l(f)}^\alpha$ is reduced. In equation (14), if f demands a flow rate higher than its parent flow f^p, the relay price μ_f^β is raised. Otherwise, μ_f^β is reduced.

We present our algorithm in Table 2. We assume the network is synchronous such that updates at sources and links are synchronized to occur at times $t = 1, 2 \ldots$. Each end host h is assumed to be capable of communicating with neighbors and be capable of measuring A^h, T^h and computing for each flow f_h'.

We choose the TCP friendly available rate of unicast flow as the initial rate in the algorithm, *i.e.*,$x_{f_h'}(0) = T_{f_h'}^h$. The closer to optimal rate the initial rate is, the faster the algorithm converges to the optimal rate. The algorithm is extendable to the asynchronous environment where prices are updated at different times [3] [4].

TABLE I

SUMMARY OF NOTATIONS IN THE MODEL

Notation	Definition
$h \in H = \{h0, h1, \ldots, hn\}$	End Host
$h^p \to h \to h' \in H'_h$	h^p is the parent node of h, h' is a child of h
H'_h	Set of child of h
$f \in F = \{f1, f2, \ldots, fn\}$	Unicast flow in ALM channel
$fi \to hi$	Flow fi terminated at hi
fh	Flow terminated at h
$x = (x_f, f \in F)$	Flow rate set of $f \in F$
$l \in \Gamma = 1, 2, \ldots, L$	Bottleneck Link l
$c_l \in C$, $l \in \Gamma$	TCP friendly available bandwidth for the channel of bottleneck link l
$f_h \to f'_h \in F'_h$	f_h is a child flow of f_h
F'_h	Set of flow sent from h in the channel
$\Gamma_h = \{l_h \| l(f'_h), f'_h \in F'_h\}$	Set of bottlenecks of flow f'_h
$l(f) \in \Gamma$	The bottleneck link that f goes through
$F(l)$	Set of siblings flows that go through bottleneck link l
$A = (A_{lf})_{L \times F}$	Bottleneck constraint matrix
$B = (B_{f'f})_{F \times F}$	Data constraint matrix
$A^h = (A_{lf})_{\Gamma_h \times F'_h}$	Bottleneck constraint matrix of F'_h
$T^h_{f'_h}$	TCP friendly available bandwidth for unicast for f'_h at $l(f'_h)$
T^h	Collection of $T^h_{f'_h}$ for $f'_h \in F'_h$
$C^h = A^h \cdot T^h$	Vector of TCP friendly available bandwidth for ALM channel for F'_h
$I_f = [m_f, M_f]$	Feasible Range of $U_f(x_f)$
$U_f(x_f)$	Utility Function of streams at rate x_f
$S(l) = \{s \in S \| l \in L(s)\}$	Set of sources that use link l
$\Gamma(s) \in \Gamma$	Set of links that source s uses
$x^{F'_h} = (x_{f'_h}, f'_h \in F'_h)$	Flow rate set of F'_h

TABLE 2: Synchronous Algorithm of End Host h

Link Price Update (by bottleneck link: $l = l(f'_h) \in \Gamma_h$):
At $t = 1, 2 \ldots$
Update price of the bottleneck link:
$\mu^{\alpha}_{l(f'_h)}(t+1)$
$= [\mu^{\alpha}_{l(f'_h)}(t) + \gamma(\sum_{f'_h \in F(l)} x_{f'_h}(t) - c_{l(f'_h)})]^+$
Relay Price Update (by flow $f'_h \in F'_h$): At $t = 1, 2 \ldots$
Update relay price of f'_h:
$\mu^{\beta}_{f'_h}(t+1) = [\mu^{\beta}_{f'_h}(t) + \gamma(x_{f'_h}(t) - x_{f_h}(t))]^+$
Stream rate Adaptation (by flow $f'_h \in F'_h$), At $t = 1, 2 \ldots$
1 Receive relay prices $\mu^{\beta}_{f'_{h'}}(t)$
from all children flow $\{f'_{h'} \mid f'_h \to f'_{h'}\}$
2 Calculate:
$\lambda^{\alpha}_{f'_h}(t) = \mu^{\alpha}_{l(f'_h)}(t)$
$\lambda^{\beta}_{f'_h}(t) = \mu^{\beta}_{l(f'_h)}(t) - \sum_{f'_h \to f'_h} \mu^{\beta}_{f'_{h'}}(t)$
3 Adjust rate:
$x_{f'_h}(t+1) = [U'^{-1}_{f'_h}(\lambda^{\alpha}_{f'_h}(t) + \lambda^{\beta}_{f'_h}(t))]^{M_f}_{m_f}$
communicates with the rate $x_{f'_h}(t+1)$ for flow f'_h
5 Send $\mu^{\beta}_{f'_h}(t+1)$ to h^p

III. NUMERICAL STUDY OF RATE CONTROL FOR FINE-GRAINED SCALABLE STREAMING

A. The utility of streams

The utility function used in [3] was $U_f(x_f) = \ln(x_f)$, which did not reflect the application quality of video streams.To tailor the utility function to the application quality as in [1], we use the rate–distortion function as the utility of our algorithm for each flow $f \in F$.

We decided to use MPEG-4 fine-grained Scalable video steams [7] in our numerical study, due to its ability to be sent at any given rate either determined by a congestion control algorithm at server side or any intermediate node in overlay multicast. MPEG-4 fine-grained Scalable video streams can be dynamically adapted to the varying condition of the network by truncating the streams to any desired bit rate. We get the utility (video quality) function for *Forman*(CIF, 30fps, 300frames) as an example (appendix in [1]):

$$U_f(x_f) = -D_f(x_f) = -2^{-0.8625x_f + 6.657} \tag{15}$$

where D_f stands for the distortion of the stream and *Megabit per second* is used as unit for streaming rate x_f. The utility function (15) is strictly increasing and concave,and twice continuously differentiable. It follows that solving problem (9) is equivalent to maximizing the overall video quality or minimizing the overall video distortion in the channel.

The primary concept of incorporating the rate-distortion function of a videoencoding scheme into congestion control is directly applicable to other video-encoding schemes beyond FGS.

B. Results

We setup the same physical network topology and overlay multicast tree as in the example shown in figure 1. In our experiments, stepsize γ is 0.001. Now we compare results of our proposed algorithm with Cui's algorithm and unicast algorithm. Figure 2 shows the comparison of resulting total utility. The optimal rates(Mbps) allocated by Cui's algorithm, which uses utility function $U_f(x_f) = \ln(x_f)$, are $x_1^* = 2, x_2^* = 4, x_3^* = 4, x_4^* = 2$ and $x_5^* = 2$, and the total utility is $\sum_{f\in F} U_f(x_f^*) = -110.049$. However, the optimal rates (Mbps) allocated by our algorithm are $x_1^* = 2.420, x_2^* = 3.580, x_3^* = 3.580, x_4^* = 2$ and $x_5^* = 2$, which is the same as the optimal result allocated by Cui's algorithm when using the same utility function (15). Then the total utility is $\sum_{f\in F} U_f(x_f^*) = -108.542$. If first we allocate the rates independently as unicast flows using TFRC algorithm, and then apply the data constraint, we get a different set of rates as unicast flows: $x_1^* = 3, x_2^* = 3, x_3^* = 8, x_4^* = 2$ and $x_5^* = 2$. In the second step, the data constraint is applied to this set of rates: x_3^* is changed to 3. Thus, the total utility is $\sum_{f\in F} U_f(x_f^*) = -111.422$, which is worse than than the optimal result -108.542. Indeed, the rates in the second set can be the initial rates for our algorithm. For the sake of the overall media quality of all receivers, our proposed distributed algorithm serves a small subset of receivers with low quality stream and use the newly available bandwidth for the remaining large subset receivers. In this example, our algorithm allocates more bandwidth in the shared bottleneck $c_1 = 6Mbps$ to flow $f2$ with more children in its sub–tree than the bandwidth to flow $f1$.

We compare the messaging overhead of algorithms in figure 3. The results show our proposed algorithm produces about one fourth messaging overhead of Cui's algorithm in each round. In our algorithm, there is no link price update message to be exchanged between hosts, since each flow, thus its host, has only one bottleneck link producing link price. Indeed, we find out most link price messages exchanged in Cui's algorithm report the link prices are zero. We conclude our algorithm maximizes the total utility of overlay multicast, while minimizes the messaging overhead of the distributed algorithm.

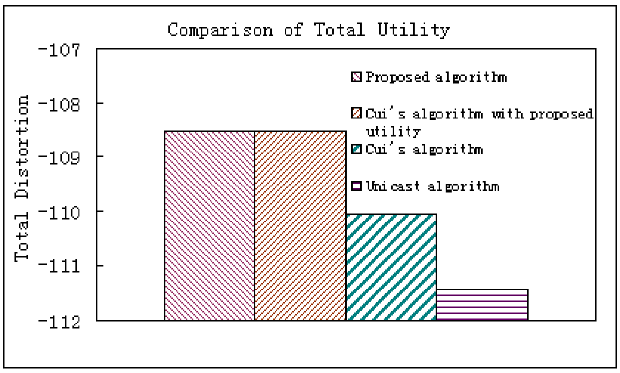


Fig. 2. Comparison of Total Utility

IV. CONCLUSIONS

In the paper, we designed a distributed and TCP friendly congestion control algorithm for application-layer multicast based on the pricing model which optimizes the overall

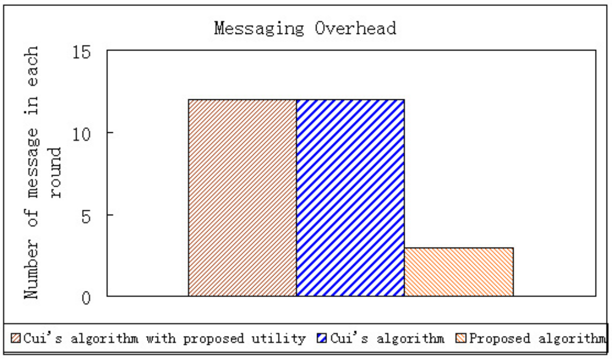


Fig. 3. Comparison of Messaging Overhead

video quality for scalable video streams in the multicast tree. Our algorithm is fully distributed with minimized message complexity and feasible to implement on end host without any centralized node. Currently, we are working on evaluating our algorithm in the asynchronous and real environment.

REFERENCES

[1] Jinyao Yan, Kostas Katrinis, Martin May, and Bernhard Plattner, "Media and TCP-Friendly Congestion Control for Scalable Video Streams (Extended Version)", TIK report 234, ETH Zurich, Nov. 2005, http://www.tik.ee.ethz.ch/ katrinis/pubs/TIK-Report234.pdf. An abbreviated version was in IEEE Transactions on Multimedia,8(2),196-206, 2006

[2] S. Floyd, M. Handley, and J. Padhye "Equation-Based Congestion Control for Unicast Application", ACM SIGCOMM'00, September 2000.

[3] S. Low, D. E. Lapsley, "Optimization Flow Control, I: Basic Algorithm and Convergence", IEEE/ACM Transactions on Networking, vol. 7, no. 6, December 1999.

[4] Y.Cui, Y.Xue and K.Nahrstedt, "Optimal Resource Allocation in Overlay Multicast", Parallel and Distributed Systems, IEEE Transactions on, 17(8), 808-823

[5] Y. Chu, S. G. Rao, and H. Zhang,"A case for End System Multicast" in Proc. ACM Sigmetrics, June 2000

[6] Venkata N. Padmanabhan, Helen J. Wang and Philip A. Chou "Supporting heterogeneity and congestion control in peer-to-peer multicast streaming" LNCS Vol.3279, Springer, 2005

[7] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 Video Standard", IEEE Trans.on CSVT, Vol.11, No.3, March 2001, pp301-317V.

[8] Van Jacobson,"Congestion Avoidance and Control", ACM SIGCOMM'88, pages 314-329, Aug. 1988.

[9] T. Bu, N G Duffield, F Lo Presti, D Towsley, "Network tomography on general topologies",ACM SIGMETRICS 2002, Marina Del Rey, USA,21-30

[10] Kar, S. Sarkar and L. Tassiulas, "Optimization basd rate control for multirate multicast sessions",in IEEE INFOCOM, 2001.

[11] S. Keshav, "Packet-Pair Flow Control", tech.rep. AT&T Bell Laboratories, Murray Hill, New Jersey.1994.

[12] M. Jain and C. Dovrolis, "End-to-End available bandwidth:Measurement methodology,dynamics,and relation with TCP throughput", IEEE/ACM Trans. on Networking (TON), 2003,11(4):537-549.

[13] Jinyao Yan, Martin May, and Bernhard Plattner, Comments on 'Optimal Resource Allocation in Overlay Multicast"', to appear in IEEE Transactions on Parallel and Distributed Systems, 2008.

[14] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, James W. O'Toole, Jr., "Overcast:Reliable Multicasting with an Overlay Network", Proceedings of the 4th Symposium on Operating Systems Design and Implementation, October 2000

[15] Suman Banerjee, Christopher Kommareddy, Bobby Bhattacharjee, "Scalable Application Layer Multicast", in ACM SIGCOMM 2002,Pittsburgh, PA, 2002.