A SCHEME FOR PEER-TO-PEER LIVE STREAMING WITH MULTI-SOURCE MULTICAST AND FORWARD ERROR CORRECTION

[1]Victor Gau, [2]Peng-Jung Wu, [2]Chung-Nan Lee, [1]Jenq-Neng Hwang

[1]University of Washington, Box 352500
Department of Electrical Engineering
Seattle, WA 98195, USA
[2]Department of Computer Science and Engineering,
National Sun Yat-Sen University, Kaohsiung, Taiwan

ABSTRACT

In this paper, we propose a scheme for peer-to-peer (P2P) live streaming with multi-source multicast and forward error correction. In our scheme, there is a control topology for membership management, and a multi-source multicast tree for data delivery. The control topology facilitates peers to locate multiple sources for media content, and the multi-source multicast tree make the system adaptive to node churn and packet loss. Simulation results show that the performance of our proposed method is significantly better than that of BitTorrent-Like (BT-Like) systems.

Index Terms— peer-to-peer, live streaming

1. INTRODUCTION

With the progress in the internet technology, traditional TV broadcasting has an alternative way to deliver to audience, i.e., the TV contents are compressed, packetized, and delivered to the audience through the Internet connections. The server has options to stream data by using multiple unicast connection, IP multicasting, applicaton-layer multicasting (ALM), or a mixture of these methods. Multiple unicasting is not scalable. IP Multicasting is infrastructure dependable and not widely deployed yet. Therefore, ALM is thus becoming promising because of its scalability and flexibility. ALM can be realized by either content delivery networks (CDN) or peer-to-peer (P2P) technologies, and the latter one requires much less capital investment.

Recent progress in P2P live streaming is prone to mesh-pull (BitTorrent (BT)-Like) approaches for streaming. In these approaches, each peer keeps a buffer consisting of video chunks. Peers exchanges video chunks based on gossip-like protocols. The beauty of the mesh-pull approach is to utilize the upload bandwidth of all peers to achieve better video quality. The data transmission rate could be up to 300~400 Kbps or even higher. However, the startup and playout delays are considerably long, ranging from 20 seconds to several minutes. Typical mesh-pull approaches include CoolStreaming, PPLive, PPStream, SopCast and so on. Most of these systems provide little information about their proprietary technologies [6].

NICE was intended for low bandwidth real-time applications, e.g., real-time stock quote, internet radio and so on [5]. It would not adapt well in a high rate of node churn, because when nodes join and leave, the subsequent cluster mergence and split would propagate to several layers. In addition, the nodes at higher layer would need to deliver data to $O(\log_k N)$ members, which make it not scalable in bandwidth-intensive application as the scale increases.

The mesh-pull approaches, e.g., PPStream etc., could make use of the upload capacity of all peers to support high bandwidth playback. And due to its multi-source nature, node failures would not significantly affect the playback. However, a very large buffering space is required for mesh-pull approaches and a very long buffering and playback delay would be experienced. In NICE protocol, the video content would be efficiently delivered; however, there would be potential bottlenecks for high bandwidth playback, and the system would not be adapted well when nodes leave or fail.

In this paper, we proposed a scheme for P2P live streaming with multi-source multicast structure. The proposed scheme takes advantage of the strengths from both the BT-like mesh-pull and NICE-like tree-push approaches so as to reduce the long startup/playout delays of mesh-pull technologies and the bandwidth limitation of multicast-push technologies. The rest of the paper is organized as follows: Section 2 introduces different schemes for p2p streaming. Section 3 describes the proposed method. Section 4 shows our simulation results. Section 5 concludes this paper.

2. SINGLE-/MULTI-SOURCE MULTICAST TREES WITH/WITHOUT FEC

In this section, we present different schemes for P2P live streaming, and briefly describe our proposed method at the end. The benefit of employing FEC is that when there are

packets lost during the transmission, the whole data packets would possibly be recovered based on FEC technologies. For example, FEC(*N=8,K=5*) denotes a block erasure coding scheme, where *N* is the number of total transmitted packets, *K* is the number of actual data packets, i.e., *N-K* redundant packets are created for error correction purpose. If the number of lost packets is less than or equal to *N-K*, the missing packets can be recovered by FEC on receiver.

Due to the bursty nature of the Internet, packet loss pattern would not be sparsely along the data stream. Instead, the packet loss would not happen frequently but when it happens, the patterns would be densely distributed in a time frame. In such bursty nature, the single source with FEC scheme would not effectively recover the lost packets, because the number of lost packets would be greater than *N - K*.

For video streaming in NICE, a source specific tree with single source without FEC structure is used. The peers in the NICE tree relay the data packets along the multicast tree to all other peers. The advantage of this protocol is that the buffering and playback delays are a lot shorter than BT-Like systems.

NICE uses a control topology to manage and control membership, and its multicast tree is implicitly defined by the control topology, as shown in Fig. 1.

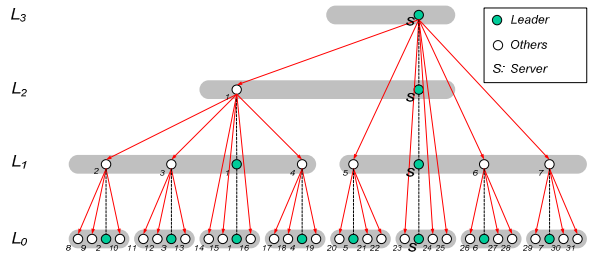


Fig.1. The control topology (black dashed lines) and multicast tree (red solid lines) of NICE.

Fig. 2 shows the structure of multi-source with FEC, where *S1~S8* denote the sources and R is the receiver. The benefit of multi-source is that when one or few parents leave, other parents can still provide most remaining part of streaming video packets. In case that the number of leaving parents plus the number of dropped packets is smaller than the number of redundant FEC packets, the missing packets can still be fully recovered by the FEC scheme.

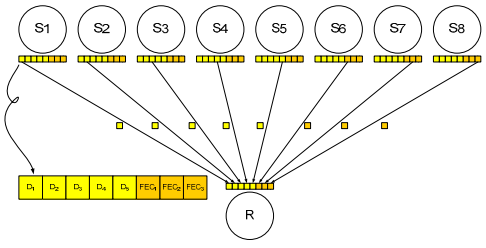


Fig.2. An example of 8-source structure with FEC(N=8,K=5).

In our proposed method, the system contains two entities: control topology and multi-source multicast tree. The control topology is a hierarchical structure similar to the NICE's control topology. The multi-source multicast tree is built atop the control topology. The control topology is to control and manage membership and to cluster the peers based on network distances and to make easy the selection of multi-source data paths. The multi-source multicast tree is to reduce the impact of node churn and to solve the problems like potential bandwidth bottlenecks that would occur in NICE.

3. PROPOSED METHOD

In this section, we describe how we organize the peers into a multi-source multicast tree. We first describe how we organize peers into a hierarchical control topology. After that, we describe the formation of the multi-source delivery paths.

3.1. Formation of the Control Topology

As mentioned in previous section that the formation of our proposed control topology is similar to NICE's control topology [5]. In the join process, the new peer p loops from the highest layer to find closest parent along the control topology using round trip time (RTT), until it identifies a closest parent at layer 1 and joins a cluster at layer 0. During the join process, the peer will record all the peers it visits along the control topology.

In the control topology, we define *k* as the lower bound of the cluster size. If more and more peers join the multicast group and the cluster size become greater than 3*k*-1, then the oversize cluster will be split into two clusters to make sure the cluster size is within the range of [*k, 3k-1*]. Similarly, if more and more peers leave the multicast group, occasionally the cluster size will be smaller than *k*. Then, we will perform cluster mergence to guarantee the cluster size is within [*k, 3k-1*].

3.2. Formation of the Multi-Source Multicast Tree

In mesh-pull (BT-Like) systems, a tracker would be used to keep track of all peers. When a new comer joins the protocol, it requests a peer list from the tracker. The tracker randomly picks a list of peers from its database and responds the new comer with this randomly picked peer list. The newly joined peer then starts to exchange data with peers in the peer list and try to pull the data from peers in the list. BT-Like systems are multi-source in nature; therefore, it is adaptive to node churn.

We notice that when the scale is small, it is not efficient to use the multi-source multicast structure. Therefore, we let the peer request for direct unicast from the server and keep a list of peers as backup paths. Because data is efficiently pushed to the peers and the peers would server as seeds later,

we call this transient process the seeding process. After the seeding process, the newly joined peer starts to request for multi-source delivery for robustness.

In our proposed method, after joining the multicast group, the newly joined peer can easily locate N parents to request for the required part of the data. The newly joined peer actually selects parents from its cluster mates and its leader's cluster mates. And the newly joined peer needs to ask the selected parent (*S1*~*SN*) to provide packets with ((sequence number mod *N*) = = Source Number), e.g., the new peer p will request packets with (sequence number mod *N* = = 1) from *S1*.

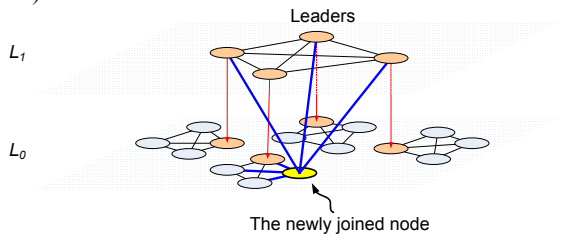


Fig.3. A conceptual diagram about how a newly joined node locates its parents for multi-source multicast structure (L_2 an L_3 are not shown).

Our proposed method is a blend of the advantages of NICE and the advantages of BT-Like systems.

4. SIMULATION

In our simulation, we assume that there are twenty domains: domains 1-10 indicate the campus network, i.e., 100Mbps upload rate, and domains 11-20 indicate residential network, i.e., 400-500Kbps upload rate. The connection of two nodes in the same domain in residential network will go through the unique link between residential network and campus network twice to simulate the relatively high End-to-End Delay nature of the broadband internet access through ADSL or cable modem. The one way delay within the same domain in campus network is in (*0ms*, *6ms*). The one way delays among domains in campus network of different countries are in [*100ms*, *300ms*]. The one way delays between campus network and residential network are in [*50ms*, *120ms*].

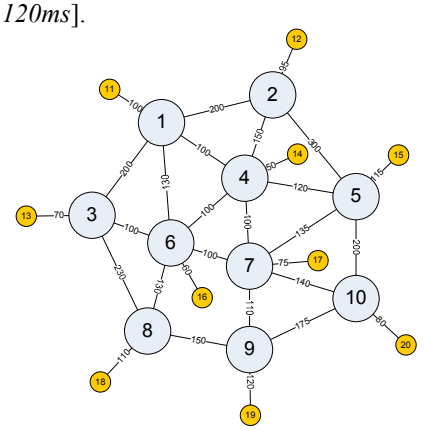


Fig.4. The simulation scenario.

4.1. Performance Metrics

In our simulation, we measure End-to-End Delay, Relative Delay Penalty, Packet Arrival Time, and Time Difference of Packets, so as to estimate the efficiency of our protocol.

End-to-End Delay (EED) $d(x_i, x_j)$ is the measure of the direct unicast distance between two nodes x_i and x_j. EED is often used to measure the proximity of nodes in the same cluster.

Relative Delay Penalty (RDP) is the ratio of the delay along the overlay from the source to the member to the delay of direct unicast path, mathematically

$$RDP = \frac{D(x_s, x_r)}{d(x_s, x_r)}$$

where $D(x_s, x_r)$ denotes the total delay from the server x_s to the receiver node x_r along the data delivery path.

The average RDP (ARDP) is then defined as

$$ARDP = \frac{1}{n}\sum_{r=1}^{n}\frac{D(x_s, x_r)}{d(x_s, x_r)}$$

where n is the total number of nodes joining the multicast group.

The Packet Arrival Time (T) indicates the time when a peer successfully receives all consequent data packets, i.e. all eight packets in FEC(N=8, K=5).

$$T(x_p) = \max(D(x_s, x_r) + d(x_p, x_r)),$$

where $x_r \in x_p's$ active parents.

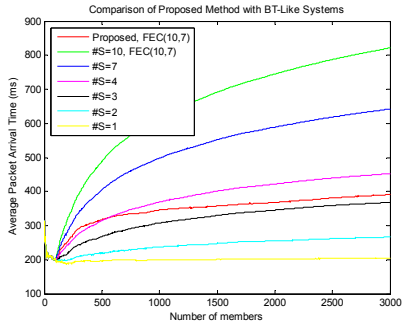
The time difference of packets indicates the time period that the peer needs to wait for the last packet after it receives the first packet. This is to measure the dispersion of the eight packets in FEC(N=8, K=5).

$$B(x_p) = \max\left(\left|D(x_s, x_{r1}) + d(x_p, x_{r1}) - D(x_s, x_{r2}) - d(x_p, x_{r2})\right|\right),$$

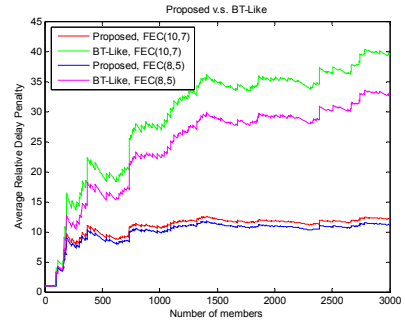
where x_{r1}, $x_{r2} \in x_p's$ active parents.

4.2. Simulation Results

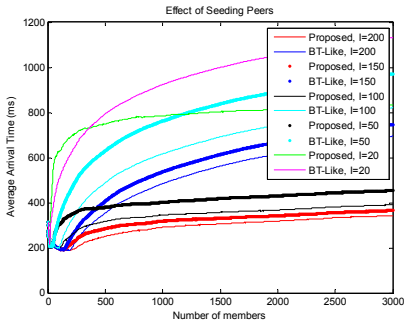
We assume that when new peer joins BT-Like systems, the tracker will respond the new peer with a peer list of 30 peers, whereas in our proposed method, the number peers in the peer list is within [2*k*, 6*k*-4] with *k* = 5 in our simulation. In literature, measurement studies show that BT-Like systems adds about 27% more traffic into the streaming [6]. Our proposed method with FEC(*N=10,K=7*) incurs about 30% more traffic. Fig. 5 (a) shows the comparison of our proposed method with BT-Like systems. As shown in the figure, our proposed method always use 10-source multicast. #S=1 means that the BT-Like system can get all packets from the fastest peer in its peer list. From the picture, if a peer in the BT-Like systems could not find all the packets from the first three fastest peers in the peer list, our proposed method will have better performance.



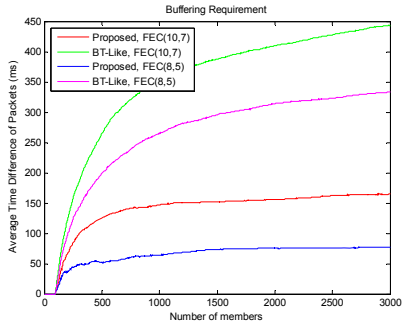
(a)



(b)



(c)



(d)

Fig.5. (a) Average packet arrival time. (b) ARDP of the proposed method and BT-Like Systems. (c) Effects of seeding peers. (d) Average time differences of packets.

Fig. 5(b) shows the average relative delay penalty of our proposed method and BT-Like approaches. This gives us a rough idea that if the unicast distance to the server is *c*, the time to get the streaming data is about ARDP**c*. The simulation results show that the ARDP of BT-Like systems are about 3 times of our proposed method.

Fig. 5(c) shows the effect of seeding peers. The number shows that the system does not start with multi-source structure at the very beginning. For example, I = 20 means that after the number of peers has exceeded 20, then we start to adopt multi-source multicast tree structure. Before that, the system runs with single source multicast, but with several backup peers in peer lists.

Fig. 5(d) shows the average time difference of the packets. The larger the difference, the bigger the buffer would be required. Fig. 5(d) shows that the average time difference of packets of our proposed method is about 3~5 times better than BT-Like systems.

5. CONCLUSION

In this paper, we proposed a multi-source multicast scheme with forward error correction. Our simulation results show that our proposed method has shorter playback delay, and shorter buffering time (Fig. 5 (a), (b), (d)). The results also show that the initial seeding process would help to reduce the playback delay and buffering time (Fig. 5(d)). More real world experiments will be conducted to verify the results.

6. REFERENCES

[1] V. Padmanabhan, H. Wang, and P. Chou. Resilient Peer-to-Peer Streaming. 11th IEEE International Conference on Network Protocols (ICNP'03), Atlanta, Georgia, November, 2003.

[2] D. Tran, K.Hua, and T. Do. A Peer-to-Peer Architecture for Media Streaming. To appear at IEEE JSAC Special Issue on Advances in Service Overlay Networks, 2003.

[3] H. Deshpande, M. Bawa, and H. Garcia-Molina. Streaming Live Media over a Peer-to-peer Network. Technical report, Stanford University, 2001.

[4] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava. On Peer-to-Peer Media Streaming. In Proc. of IEEE ICDCS'02, Vienna, Austria, July 2002.

[5] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. Technical report, UMIACS TR-2002-53 and CS-TR 4373, Department of Computer Science, University of Maryland, College Park, MD 20742, USA, May 2002.

[6] G. Marfia, A. Sentivelli, S. Tewari, M. Gerla and L. Kleinrock, "Will IPTV ride the peer-to-peer stream?", IEEE Communications Magazine, Special Issue on Peer-to-Peer Streaming, June 2007.

[7] X. Hei, C. Liang, J. Liang, Y. Liu, K. W. Ross, " A Measurement study of a large-scale p2p iptv system," Department of Computer and Information Science at Polytechnic University, Tech. Rep., 2006.

[8] X. Zhang, J. Liu, B. Li, and T. Yum, "Coolstreaming/donet: A data-driven overlay network for efficient live media streaming," in IEEE INFOCOM, 2005.