A NOVEL PEER MANAGEMENT MODEL FOR LIVE PEER-TO-PEER STREAMING

Rongtao Liao, Shengsheng Yu, Lijun Dong

School of Computer Science and Technology
Huazhong University of Science and Technology, Wuhan, 430074, China
rt_liao@yahoo.com.cn, ssyu@mail.hust.edu.cn, ljdong@wtwh.com.cn

ABSTRACT

Due to the peer heterogeneity and dynamic, the peer management to construct high quality streaming overlay poses significant challenges in peer-to-peer streaming. In this paper, we propose a fully distributed scheme to achieve optimal peer selection and dynamic peer management, which gains global optimal performance in a highly diverse and dynamic peer-to-peer network. We designed this distributed management model based on an efficient peer selection algorithm and a set of dynamic maintenance scheme. It leverages the performance information and synchronization estimation between peers to select the best peers as senders. With the dynamic peer management protocol, our model is resilient to network dynamic that is characteristic in peer-to-peer networks. The validity and effectiveness of our approach are demonstrated in simulations.

Index Terms—Peer-to-Peer, Live Streaming, Peer Selection, Membership Management

1. INTRODUCTION

Despite so many important advantages, Peer-to-Peer (or P2P) streaming poses significant technical challenges. For peer heterogeneity, the available bandwidth of each peer may differ by at least an order of magnitude. Therefore, it is typical for a peer node to download media data from multiple upstream peers. In this case, the quality of a P2P streaming session depends on the efficient peer management which includes judicious selection of senders, constant monitoring of sender network status, and timely switching of senders when the sender or network fails or seriously degrades.

Previous works in P2P media streaming do not provide a systematic solution to the peer management. The approaches either require global membership information [1, 2, 3], or embed membership information in the overlay structure [4, 5], both of which would introduce scalability problems. The gossip style membership management adopted by recent systems [6, 7] use a random peer selection scheme which can not be easily modified to gain perfect global performance. In [8], it proposes a linear programming model that aim at minimizing general cost functions in P2P streaming. PROMISE [9], which infers and exploits properties of the underlying network, judiciously chooses the sending peers and orchestrates them to yield the best quality for the receiver. But the inference scheme is complex and peer set frequently changing interrupts the playback of the streaming.

In this paper, we present our solution to address such a problem in a highly diverse and dynamic P2P network. As PROMISE [9], each peer maintains two peer set, members of which may change dynamically during the session. The major properties of our membership management scheme include the following: (1) it leverages the performance information for the selection of sender peers. A weight factor is introduced to balance the influence of bandwidth and delay; (2) it estimates the synchronization between peers, those have more similar packets all the time will be selected as sender peers. This is based on a novel approach to calculate working set resemblance based on sketches; (3) it monitors the status of peers and connections and reacts to peer and connection failure with low overhead.

The remainder of this paper is organized as follows. In Sec. 2, we present the streaming application model based on our peer management model. In Sec. 3, we address the optimal peer selection and dynamic peer management. Simulation results are presented in Sec. 4.

2. LIVE P2P STREAMING APPLICATION MODEL

To achieve the goals of good performance streaming, we present a new P2P streaming framework based on our peer management model. For scalability, the localized overlay construction and maintenance approach is used, which means both the initial joining of a peer and the recovery from a sender failure are done by individual peers selecting their senders. The streaming framework is deployed on top of a P2P substrate, which responds for maintaining connectivity among peers, managing peer membership, and performing object lookup. Peer management and streaming operations are independent of the underlying P2P substrate.

1-4244-1447-4/07/$25.00 ©2007 IEEE

Figure 1 depicts the system diagram of a streaming node. The key modules are: (1) working set estimation, which estimates the relativity of streaming data between peers; (2) peer selection, which selects the best peers for standby set based on the candidate peers' characteristics and synchronization estimation;(3) overlay optimization, which potentially help the system to converge to a short average path, low latency, and good performance.

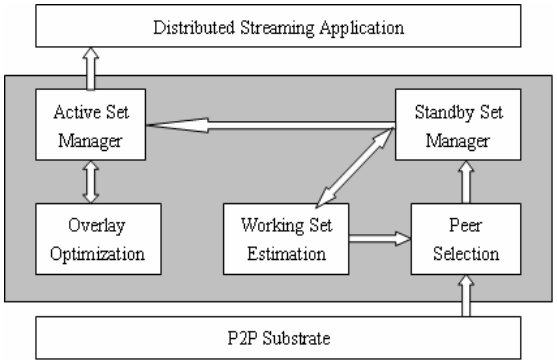

Fig.1. The system diagram of a streaming node

As mentioned above, each peer in the streaming session can only receive data from its sender peers and send data to its child peers, which is unlike DoNet [6].

3. DESIGN OF PEER MANAGEMENT MODEL

In this section, we present the details of the peer management scheme. First, we describe the peer characteristics and resemblance estimation method. Then, an optimal peer selection strategy is proposed to guarantee global performance. The basic dynamic management protocol is also presented.

3.1. Peer Characteristics and Similarity Estimation

In order to cope with the heterogeneity of peers, we associate each peer p with two parameters: offered rate R_p and delay D_p. The offered rate (R_p) is the maximum sending rate that a peer can contribute to the system. The delay (D_p) denotes the network delay between source to peer p. The offered rate and delay information is collected by a daemon running on each participating peer.

Another significant consequence of the heterogeneity of the P2P network is that content is likely to be disseminated non-uniformly across peers. Significant discrepancies between working sets are likely to arise due to uncorrelated losses, bandwidth differences, asynchronous joins and topology reconfigurations. So the peer selection scheme, which just based on network performance such as offer rate and delay, is not enough. Obviously, those peers who have more similar data is likely to have more data current peer need because of the limited buffer windows. The magnitude of the working set resemblance is useful for the peer selection strategy we proposed. Also, it is essential that the data be conveyed between the peers to compute the resemblance as efficiently as possible.

Let peers A and B have working sets A_s and B_s containing packets of a streaming source S. In [10], it presents a novel approach to calculate working set resemblance based on sketches. Given a random mapping function Ω_i, for a set $S = (s_1, s_2, \ldots, s_n)$, let $\Omega_i(S) = \{\Omega_i(s_1), \Omega i(s_2), \ldots, \Omega_i(s_n)\}$, and let $min\Omega_i(S) = min\Omega_i(s_k)$, k∈ [1,n]. It can be proved that $min\Omega_i(A_s) = min\Omega_i(B_s)$ with probability $\gamma_{AB}=|A_s \cap B_s|/|A_s \cup B_s|$. The quantity γ_{AB} represents the resemblance between the two sets. To estimate the resemblance, the peer A computes $min\Omega_i(A_s)$ for some fixed number of mapping functions, and similarly for B and B_s. By estimating minima vector $v(A)$ and $v(B)$, the resemblance γ_{AB} can be obtained.

3.2. Selecting Best Peers

Time synchronization. The time synchronization S_{AB} is a property denotes the data relativity between two peers A and B during the streaming session. If the buffer of A always has data B need during streaming, we consider A and B in good synchronization. The synchronization is a dynamic parameter, we should estimate it periodically. In our model, we represent the time synchronization as a function of the working set resemblance and the buffer size of pair of peers. The resemblance estimation method is proposed for file sharing. But in the case of streaming, many peers will hold buffers in different size to keep the data. When a peer with large buffer compares the minima vector with a peer with small buffer, the result is tend to small, which can't represent the accurate synchronization between the two peers. To address this issue, we introduce buffer size ratio to eliminate the influence introduced by different buffer size. The time synchronization of peer A and B is defined as $S_{AB} = \gamma_{AB} \times \chi_{AB}$, where γ_{AB} is the working set resemblance between peer A and B, and χ_{AB} is the ratio of the peers' buffer size depends on the buffer size of the two peers as follows:

$$\chi_{AB} = \begin{cases} |B|/|A|, & if\ |B| \geq |A| \\ |A|/|B|, & others \end{cases} \quad (1)$$

where $|A|$ is the buffer size of peer A , the same as $|B|$.

Peer performance. The peer performance A_p is a property indicate the network performance of peer p, it can also denote the ability the node can contribute to the streaming. In our model , we think the delay and offer rate is the most important property which influent the streaming performance a lot, so we represent the peer performance as a function of the offered rate R_p and delay D_p. As mention above, the two parameters can be obtained easily. The peer performance A_p can be represented as the following form:

$$A_p = \frac{(1-\alpha)R_p}{\alpha D_p}, \quad (2)$$

where α is a constant value indicate the weight of D_p in the estimation of peer performance, which can be modified as needed. Peers with high expected performance values indicate that these peers are likely to provide good and sustained sending rate and relative low delay to the source.

Peer goodness. We define the goodness of peer p, G_p, as a function of its performance property A_p and the time synchronization between two peers. If peer p is a candidate peer of peer A, then to peer A's perspective, the G_p has the following form:

$$G_p = S_{Ap} \times A_p, \quad (3)$$

where S_{Ap} is the time synchronization between peer p and A.

Best active peers set. This is the subset of peers that are likely to provide the best quality to the receiver. We are now ready to state the selection problem: given the obtained peer network property and the time synchronization, find the best set of active peers in candidate peer set Φ, which can be phrased as:

$$\max \sum_{p \in \Phi^{active}} G_p, \quad (4)$$

$$\text{subject to} \quad \Phi^{active} \subseteq \Phi. \quad (5)$$

Practically, in our model, each peer maintains one active peer set and one standby peer set. When the number of standby peer set is less than a threshold, the selection algorithm is called to find appropriate candidate peers in the global P2P network with the help of the P2P substrate. When one active peer is cut off due to its rejection or leaving, a new peer is selected from the standby set.

Given that we have U as the active peer set and V as the standby peer set, and let $\varphi(s)$ denote the rate at which the source s is streaming. We must have:

$$\sum_{x \in U} \partial(x, p) \geq \beta\varphi(s), \quad (6)$$

$$|V| \geq \theta. \quad (7)$$

The function $\partial(x, p)$ denotes the streaming rate peer p receives from peer x. We use the factor $\beta \geq 1$ to indicate that the system can tolerate packet loss. The constant θ is the smallest number of the standby peer set.

3.3. Basic Dynamic Protocol

3.3.1. Peer Join

When a peer p_i initially joins the system, it will first issue a lookup request to the underlying P2P substrate, which will return a set of candidate peers who have the needed stream. The candidate peer set typically contains more than 10 peers. The protocol then request the candidate peers to return their characteristics and working set estimation vector. Using the peers' characteristics and estimation vector, the estimation algorithm works out the peer goodness of all the candidate peers, then sort them descending, into a candidate peer set queue CQ_i. Candidate peers are selected into the standby peer set queue SQ_i of p_i from the local queue CQ_i as follows:

```
while(CQ_i ≠ Φ){
 cp=dequeue(CQ_i);
send_message(Try,cp, PeerInfo)
msg = receive_message(cp); //reply from peer cp
if (msg.type = = TryOK)
  enqueue(SQ_i,cp);
}
```

cp= dequeue(Q) and *enqueue(Q, cp)* show procedures where a peer *cp* is dequeued from a queue *Q* and enqueued into *Q*, respectively.

As shown above, peer p_i send **Try** message to each candidate peer *cp*. The candidate peer *cp* is expected to send a **TryOK** message back; unless it has been exhausted all of its uplink bandwidth and all its child peers have large peer performance than peer p_i. In this case, a **TryRej** message is sent back, and the peer p_i will try other candidate peers one bye one. After the standby peer set is initialized, peer p_i will start its active peer set manager module and standby peer set manager module. Because the active peer set is null initially, the active peer set manager module will select peers from standby peer set until it satisfies the constraint in (6).

Once the active set is determined, peer p_i establishes parallel connections with all peers in the active set. Peer p_i assigns a sending rate to each of the active senders based on the sender's offered rate and goodness. The streaming session continues as far as there is no need to switch to a different active sending set. A switch is needed if a peer fails or some other new peer is added into the active peer set.

3.3.2. Dynamic Peer Maintenance

Standby Peer Set. During the streaming, the standby peer set manager of p_i will periodically send a **Try** message to its standby peers, noted as $SQ_i = \{SP_{i1}, SP_{i2}, \ldots, SP_{in}\}$. A peer (e.g. SP_{i1}) receiving the **Try** message should reply with a **TryOK** or **TryRej** message to p_i. If no **TryOK** or **TryRej** is received from SP_{i1} for several periods, or just **TryRej** is received, the peer SP_{i1} is removed from the standby peer set SQ_i. The **TryOK** message contains some state information such as the current performance and the working set estimation vector of the peer SP_{i1}. If the parameters of the peer SP_{i1} have been changed, the standby peer set manager must re-compute their own goodness and sort them into the standby peer set queue SQ_i again. If no enough peers are in standby peer set, the standby peer set manager will send a lookup request to the underlying P2P substrate to find other candidate peers to fill the standby set until the constraints in (7) is satisfied.

Active Peer Set. Unlike standby peer set, the active peer set will be stable to eliminate network jitter incurred by frequently reconnecting. A peer switching is needed if a peer is failed or receiving a **ConnectRej** message from a peer in active peer set, noted as $AQ_i = \{AP_{i1}, AP_{i2}, ..., AP_{im}\}$. The active peer set manager of peer p_i will periodically send a **Connect** message to its active peers (eg. AP_{i1}). If no **ConnectOK** or **ConnectRej** is received from peer AP_{i1} for several periods, or just **ConnectRej** is received, the peer AP_{i1} is removed from the active peer set AQ_i. If a peer in active set (e.g. AP_{ik}) has been exhausted of all its outbound bandwidth and all its child peers have large peer performance than current peer p_i, and a new join peer (e.g. *NP*) has large peer performance than peer p_i, then peer p_i will be preempted by the new peer *NP*. In this case, the peer AP_{ik} will reply with a **ConnectRej** to p_i when a **Connect** message is received from p_i. During the streaming, the active peer set manager will update the receive rate of current peer, when constraints in (6) is not satisfied, it will select some peers from standby peer set until the constraints is satisfied.

4. PERFORMANCE EVALUATION

We have carried out simulations to investigate the performance of our peer management scheme over random networks generated by BRITE topology generator. Simulations show distribution of clients according to the distance (in hops) to the source, i.e. according to the level in overlay topology. Here, we fixed the number of peers to 500 and use the hop to source to denote the delay to source. Each plotted value is an average of a hundred simulation runs.

Table 1. Distribution of peers in the overlay structure

Hops to Source	Optimal Peer Selection			Random
	α=0.5	α=0.8	α=0.9	
1	8.00%	8.40%	8.20%	0.00%
2	24.20%	26.00%	25.20%	0.60%
3	25.40%	22.60%	19.40%	1.20%
4	21.40%	22.80%	24.60%	3.20%
5	14.00%	13.40%	15.40%	6.20%
6	5.20%	4.40%	5.00%	7.40%
7	1.00%	1.80%	1.40%	8.60%
8	0.20%	0.40%	0.40%	14.80%
9	0.60%	0.00%	0.40%	12.20%
10	0.00%	0.00%	0.00%	11.80%
11	0.00%	0.20%	0.00%	11.40%
12	0.00%	0.00%	0.00%	8.60%
13	0.00%	0.00%	0.00%	6.40%
14	0.00%	0.00%	0.00%	2.40%
15	0.00%	0.00%	0.00%	2.40%
16	0.00%	0.00%	0.00%	1.60%
17	0.00%	0.00%	0.00%	0.40%
18	0.00%	0.00%	0.00%	0.60%
19	0.00%	0.00%	0.00%	0.20%

Table 1 shows, with the implementations of our model, α=0.5, α=0.8 and α=0.9, that there is a pike of peers at the 2nd to 5th level of the overlay structure. However, with the random peer selection scheme, most of the peers are situated between 8th to 11th level of overlay. It can be concluded that peers are closer to the source in hops, therefore delays and time to first packet are consequently shorter in our model than in all other random based implementations.

5. CONCLUSION AND FUTURE WORK

In this paper, we presented the design and evaluation of our peer management model for live P2P streaming. Our model achieves scalable and decentralized membership management by using an efficient peer selection algorithm and a set of dynamic maintenance scheme. All the algorithm depends on local resource measurement only and thus has good scalability.

Currently, we are working on deploying this system in Internet to test its applicability and performance, thus it enables us to test our model in a large scale P2P system.

6. REFERENCES

[1] V. N. Padmanabhan, H. J. Wang, P. A. Chow, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking", in Proc. of NOSSDAV 2002, May 2002.

[2] Y. H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast", in Proc. of ACM SIGMETRICS, June 2000.

[3]M. Kwon and S. Fahmy, "Topology-aware overlay networks for group communication", in Proc. of NOSSDAV 2002, May 2002.

[4] D. A. Tran, K. A. Hua, and T. Do, "ZIGZAG: An Efficient Peer- to- Peer Scheme for Media Streaming", in Proc. of INFOCOM, IEEE, 2003, March 2003.

[5] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast", in Proc. of ACM SIGCOMM'02, August 2002.

[6] X. Zhang, J. Liu, B. Li, and T. P. Yum, "DONet: A data-driven overlay network for efficient live media streaming", in Proc. of IEEE INFOCOM'05, Miami, FL, 2005.

[7] R. Rejaie and S. Stafford, "A framework for architecting peer-topeer receiver-driven overlays", in Proc. of NOSSDAV'04, 2004.

[8] M. Adler, R. Kumar, K. W. Ross, D. Rubenstein, T. Suel, and D. D. Yao, "Optimal Peer Selection for P2P Downloading and Streaming", in Proc. of IEEE INFOCOM 2005, March 2005.

[9] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast", in Proc. of the 11th ACM international conference on Multimedia, Berkeley, CA, USA, November, 2003.

[10] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery Across Adaptive Overlay Networks", in Proc. of ACM SIGCOMM 2002, August 2002.