Improving Quality of Experience in P2P IPTV

Yaw-Chung Chen
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
ycchen@cs.nctu.edu.tw

Chih-Yuan Liao
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
zyliao@cht.com.tw

***Abstract*—Nowadays many IPTV services are deployed based on P2P structure for the sake of saving the cost of server equipment and bandwidth requirement. Since peers may join and leave the P2P system arbitrarily, it may cause the service quality unstable. In this paper, we compare IP addresses to find partner peers in proximity, and use specific packet pairs to probe partner peers that have larger bandwidth, so as to reduce the switching delay thru pushing the video chunks proactively. We also use two buffers that store data of both current and newly selected channel to improve the stability of media playback. We use OMNET++ simulator to evaluate our proposed schemes, the numerical results show that our schemes significantly reduce the channel switching delay and improve the smoothness of video playback.**

***Keywords*—P2P IPTV; media streaming; quality of experience; switching delay; playback.**

I. Introduction and Background

IPTV (Internet Protocol TeleVision) provides TV services thru the Internet. In traditional centralized client-server IPTV services, when the number of viewers grows large, it becomes costly to provide satisfactory quality of service. For example, MOD (Multimedia on Demand) [1] is an IPTV platform that provides VoD (Video On demand) and TV services. These video services are deployed within the operator's network by using Gigabit Switch Routers and High Performance Edge Routers, which are quite expensive.

A. P2P Live Streaming

With the successful P2P file sharing applications such as BitTorrent (BT) [2], various P2P streaming services including PPLive[3], PPStream[4], SopCast[5] etc. have been widely deployed due to their high scalability and low deployment cost. The P2P streaming structure can be implemented in two types, tree-based and mesh-based. In the latter, participating peers form a randomly connected overlay, or a mesh [7]. A peer selects some peers from the peer list as its partners, then exchanges its buffer map for checking the chunk availability, and uses chunk scheduling strategy to get media content from its partners. Its advantages are robustness and ease of implementation, but it suffers long startup delay, thus a good peer selection strategy and a chunk scheduling strategy are very important.

In both IPTV and P2P IPTV, only the currently watched channel is delivered to the viewer because of the bandwidth limitations in the access network [6]. In P2P IPTV, video playback of the newly selected channel may be delayed due to the latency of searching those peers watching the newly selected channel, loading and buffering of video content. Such kind of delay may annoy the P2P IPTV viewers.

B. Peer Selection Strategies

The peer selection strategy focuses on robustness that regards the peer churn, and network efficiency that concerns the packet download delay. Early research mainly adopted Random Peer Selection Strategy and Round-Robin Peer Selection Strategy [8], which are simple and efficient but these schemes may select parents that could not provide sufficient upload bandwidth, and result in the bandwidth bottleneck of the system [9], hence RTT (Round- Trip Time) based and ABW (Available Bandwidth) based strategies were proposed. The former only considers the distance of the peer to its candidate peer, it may probably select those parent peers with low bandwidth. ABW-based strategy measures the remaining capacity of the path between two nodes in the network [10]. It can select the partner peer with larger available bandwidth, but the ABW estimation is time consuming, an example is TOPP (Trains of Packet Pairs) [11] which may spend too much time in estimating the available path bandwidth.

C. Chunk Scheduling Strategies

In mesh based P2P streaming architecture, a peer must maintain a set of parent peers in order to exchanging chunk availability information. There are two popular strategies: Rarest First (RF) method and Greedy method. The former selects a chunk that has the least number of copies in the system. The latter selects the chunk that is closest to its playback deadline [12]. Rarest First strategy is widely adopted in BitTorrent and CoolStreaming.

II. Proposed Approaches

In this work, we proposed a peer selection scheme, an available bandwidth estimation scheme and a chunk scheduling algorithm to reduce the switching delay and improve the smoothness of playback. There is a parent-child relationship between connected peers, and content is always delivered from the parent to child [13]. Since the peer with low capabilities can be a bottleneck of service in heterogeneous P2P network [14], we focus on how to quickly select appropriate partners and improve the chunk scheduling strategy.

A. Peer selection scheme

Our peer selection scheme consists of two steps, comparing IP address and estimating available path bandwidth. After a new peer joined the P2P network and selected a channel, it firstly connected to the tracker for requesting an IP list of candidate partners. The tracker will compare all the candidate peers' IP addresses with the one in newly joined peer, and those candidate peers with longest IP address bits matched will be selected. We will select up to *M* peers which have IP address most close to the IP of the newly joined peer. Since IP addresses feature characteristic of locality, the distance between two close IP addresses is unlikely too far. This can be verified by sending IP addresses to IP Location Finder [15] and check their locations. After selecting *M* partners, the tracker sends the partner's IP+PORT list to the new peer, which will then select its parent peers from the list.

To reduce the playback delay, we select those peers with the shortest round trip time T_p and maximum available bandwidth (ABW). We used a pair of packets, both with smallest size (UDP header + IP header + Payload = 28bytes) to estimate the round trip time T_p, and used another two packets with different sizes (smallest packet size and MTU size) to estimate each partner peer's ABW. Due to the high bandwidth of nowadays backbone network, we assume that the bandwidth bottleneck always occurs in the peer's access network. The requesting peer firstly starts a timer and sends a smallest-sized packet to partner peers one by one and waits for their ACKs in smallest-sized packet. Upon receiving an ACK, it stops the associated timer and records the elapsed time T_p; and starts another timer and sends a smallest-sized packet to the same peer and waits for the ACK in MTU-sized packet. Upon receiving the MTU-sized ACK, the sending peer records the elapsed time T_t. The process is repeated until the peer received all partner peers' MTU-sized ACKs. The time difference T_t - T_p represents the time required by a partner to transmit extra data bytes, so we can calculate each partner peer's available bandwidth as follows:

$$ABW = [(MTU - 28) / (T_t - T_p)] \text{ bytes/s} \qquad (1)$$

Our ABW estimation is performed three times on each path to improve the accuracy. *N* peers with the maximum ABW will be selected as its parents. If multiple partner peers have the same ABW, peers with shorter T_p will be selected. The total parents' ABW must exceed the playback rate, and every peer at most shares a half of its available bandwidth; otherwise the video playback may not be smooth.

B. Chunk scheduling algorithm

Assuming that every media chunk has same size with a chunkID. After peer selection, each peer will have *N* parents, the peer will request chunkID_1/MOD *N* from its first parent, chunkID_2/MOD *N* from its second parent, and so-forth. Every parent will create a schedule table to store each child's ID, parent number *i* and number of parents *N*, as shown in Fig. 1. Once a peer received a chunk from its parent, it can quickly decides whether this chunk must be pushed to its child peer or not. This algorithm reduces the frequency of exchanging buffer map and chunk request overhead.

In order to improving playback continuity, we use two buffer maps. The first buffer map stores the currently watched channel data. Once the channel switching occurs, the new channel's media chunk will be stored into the second buffer map. The peer will continue to push the remaining chunks in the first buffer map to its child peers.

When a peer is switching the channel, it restarts the peer selection scheme, notifies all its parents to stop pushing chunks, and sends the number of remaining chunks in its own buffer to all its children peers and inform them regarding its switching channel. To keep the partner list updated, we set a threshold time $T_{threshold.}$ If the peer watches the channel for $T_{watch} < T_{threshold}$, it will select a new parent with the maximum ABW in the partner list. Otherwise, the peer will reload its partner list from the tracker and restart peer selection and available bandwidth estimation. Two buffer maps help the children peers get media chunk much steadily.

To evaluate the traffic overhead of our proposed chunk scheduling scheme, we set $T_{threshold}$ to 10 minutes. If we compare the packet overhead caused by switching channel with the amount of data packets of streaming video, the packet overhead is much smaller and could be ignored. With 1 Mbps streaming video, it generates approximately 88 MTU-sized packets per second. When $T_{watch} < T_{threshold}$, the packet overhead can be ignored because our approach only needs one packet to notify the new parent peer with maximum available bandwidth in the partner list. If $T_{watch} > T_{threshold}$, the packet overhead is about twelve times the number of partner peers because our approach needs four packets to estimate the available bandwidth for three times. The packet overhead and time overhead can be calculated as follows:

$$\textit{Packet overhead} = 12 * \textit{Number of partners} \qquad (2)$$

$$\textit{Time overhead} \approx 3 * (\textit{Transmission delay} + 2 * \textit{RTT}) \qquad (3)$$

Assume that the number of partners is 10, the distance between the peer and the farthest partner is 500 kilometers and the available bandwidth between a peer and its partner peer is the minimum bandwidth among all of its partners. The bandwidth is 512Kbps, the signal speed is 200,000 km/s and the peer has watched the channel for 10 minutes. We can calculate the packet overhead which is 120 packets, and the

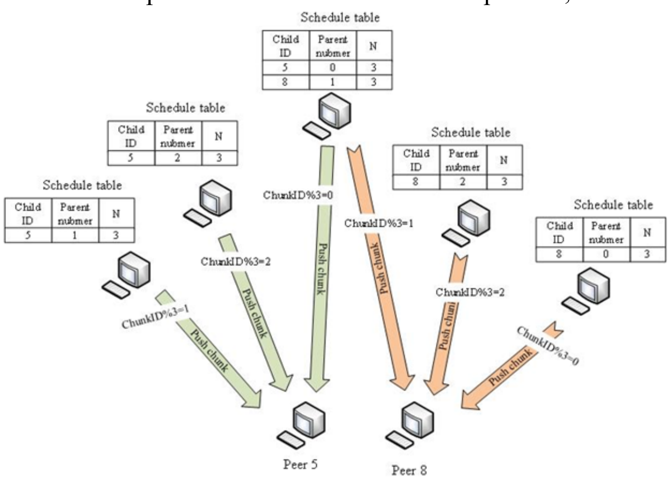

Fig. 1 Two peers and their parent peers.

number of MTU-sized packets for video streaming in ten minutes is (88*60*10) = 52800 packets, it means that our approach only has 0.227% packet overhead, the time overhead ≈ 3*(MTU/(512*1024/8)) + (2*(500/ 200000)))≈73.66 ms, this is far less than the duration of watching a channel, so it can be ignored.

III. Simulation Experiment and Numerical Results

We evaluate our proposed scheme in OMNet++16]. We construct a centralized directory model that consists of a tracker module, a server module and certain number of peer modules. When a peer joined the network, it will connect to the tracker for requesting a partner list. Upon receiving the partner list, the peer will select some peers to download chunks, all the peers will form a mesh-based overlay.

We assume that P2P IPTV peers are all in Taiwan, hence the distance between any two peers should be no longer than 500 kilometers, and all the peers using HiNet Internet Services. According to the broadband access service in Chunghwa Telecom [17], the upload bandwidth distribution of peers is shown in Table I. We tested some popular P2P IPTV services such as PPTV, PPStream, and SopCast regarding their channel switching delay and playback rate, the average channel switching delay ranges from 7 to 20 seconds, and the data rate of 1Mbps is sufficient for smooth playback.

To measure the real upload time of one chunk in different bandwidth, we use a pair of server and client to test. When the client connects to the server, the server sends a request packet to the client, which responds by sending 88 MTU-sized packets to the server. After the server received all these packets, it records the total upload time. The result of this measurement is shown in Table II. W can calculate the Ideal/Real time ratio and the real upload bandwidth as follows:

Ideal / Real time ratio = Ideal upload time/Real upload time

Real upload bandwidth = Ideal upload bandwidth (Ideal / Real) time ratio*

Our experiment shows that the real upload bandwidth is about 40%~60% of ideal upload bandwidth. This is due to TCP congestion control algorithm, with which the real throughput is about half of ideal throughput. Therefore, in our simulation, each peer was assigned a random number between 0.4 and 0.6 to calculate its real upload bandwidth. Thus, if the playback rate is 1Mbps, a peer must have download bandwidth no less than 2Mbps. In our simulation, we only generate peers with download bandwidth larger than 2Mbps. Table III shows the distribution of peers' bandwidth and Table IV shows parameters used for the simulation environment.

Table I Distribution of peer's upload bandwidth

	Upload bandwidth(bps)	Percentage
ADSL	2M/64K	10.00%
	5M/384K	13.00%
	8M/640K	11.00%
FTTB	6M/2M	12.00%
	20M/5M	25.00%
	60M/15M	20.00%
	100M/20M	9.00%

A. Results and Performance Analysis

We compared our proposed scheme with both the random peer selection scheme and TOPP scheme. We use our proposed structures, but use TOPP ABW estimation scheme, and record the average switching delay with different number of peers. We also measured the channel switching delay with buffering time 2 and 4 seconds, as shown in Table V and Table VI, respectively. It shows that our proposed scheme spends less time than both random peer selection scheme and TOPP scheme in switching channels, because our scheme selects parent peers with the largest available bandwidth, and every parent peer refuses those connections from children peers with shared upload bandwidth lower than the playback rate. In contrast, random peer selection scheme may select neighbors with small upload bandwidth or connect to too many children and lead to wasted bandwidth. The TOPP scheme may waste too much time to estimate the end-to-end ABW. Hence, our scheme achieved better performance in switching delay than both random peer selection scheme and TOPP scheme. Also when the number of peers grows, the average switching time of our proposed scheme remains unchanged. While the time in random peer selection scheme is longer and the time of TOPP is unstable.

Table II. Upload one chunk (128KB) in different distance.

County	Distance (KM)	Ideal Upload (Mbps)	Ideal upload time (128KB) (ms)	Real upload time (128KB) (ms)	Real upload one packet(MTU) (ms)	Ideal/Real time ratio (%)
Taoyuan (Close to Server)	<10	1.48	676	1229	13.97	54.98
Taoyuan (Neighbor)	<10	5.34	187	386	4.39	48.51
Hsinchu (Hukou)	25	1.6	625	1367	15.53	45.72
Hsinchu (Home)	30	5	200	370	4.2	54.05
Miaoli (Tai-an)	89	1.22	820	1835	20.85	44.67
Taichung (Cingshuei)	118	2	500	1062	12.07	47.08
Yunlin (Siluo)	172	1.73	578	1399	15.90	41.32
Tainan (Sinying)	232	1.98	505	1181	13.42	42.76
Pingtung (Wanluan)	332	1.73	578	1293	14.69	44.70

Table III. Distribution of peers' bandwidth in the simulations.

Upload bandwidth(bps)	Percentage
5M/384K	13% → 14.45%
8M/640K	11% → 12.22%
6M/2M	12% → 13.33%
20M/5M	25% → 27.78%
60M/15M	20% → 22.22%
100M/20M	9% → 10%
Total percentage	90% → 100%

Table IV. Parameters for the simulation environment.

parameter	Value
Play Rate	1 Mbps
Number of Probed Partners	10~20
Number of Parents	3~10
Number of Peers	100~10000
Buffer Time	10 s
Chunk Size	128 KB
Duration of Simulation	30 Minutes

Table V. Channel switching delay with buffer time=2 seconds.

Number of peers	100	500	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Random peer selection	8.762	9.1	10.11	10.23	10.5	10.89	11.43	12.33	12.6	12.6	12.69	12.9
TOPP	4.48	4.86	4.6	4.61	4.73	5.29	4.66	4.93	5.2	6.46	5.89	6
Our proposed scheme	3.5	3.47	3.49	3.49	3.5	3.45	3.41	3.47	3.48	3.48	3.47	3.49

Table VI. Channel switching delay with buffer time=4 seconds.

Number of peers	100	500	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Random peer selection	10.38	10.53	11	11.58	11.84	12.14	12.22	12.52	12.69	12.87	13.29	13.38
TOPP	6.9	6.9	7.61	7.78	7.82	7.6	8.11	7.85	8	8.6	8.45	8.69
Our proposed scheme	5.48	5.3	5.5	5.44	5.41	5.47	5.47	5.49	5.45	5.48	5.4	5.33

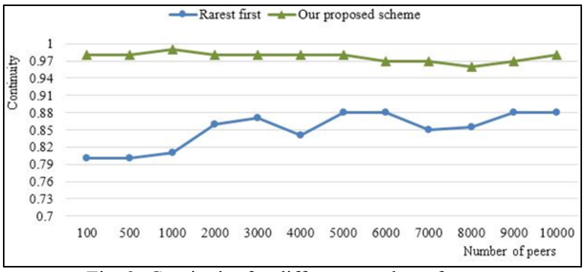

Fig. 2. Continuity for different number of peers.

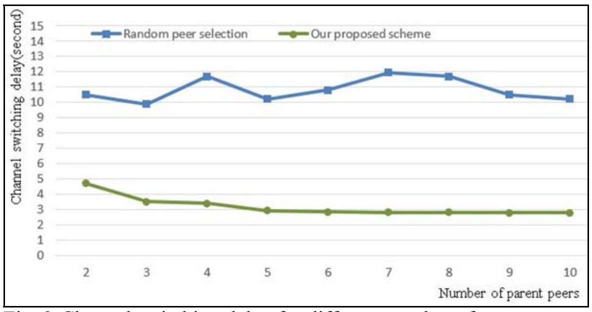

Fig. 3 Channel switching delay for different number of parent peers.

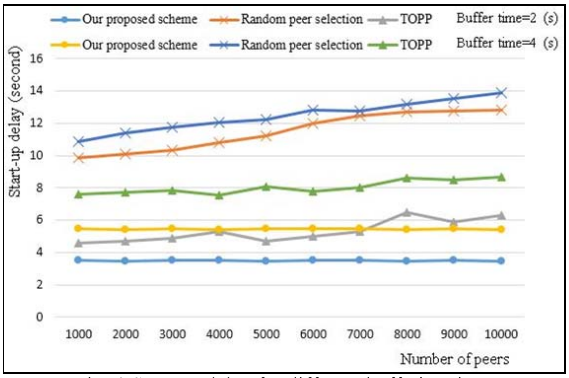

Fig. 4 Start-up delay for different buffering time.

In Fig. 3, we found that changing number of parents affects the channel switching delay, because increasing number of parent peers provides peers more source peers for requesting chunks. The start-up delay is also evaluated. We set the playback rate to 1Mbps, the buffering time to 2 and 4 seconds, and the chunk size to 128KB. Assuming that every peer has sufficient download bandwidth, we compare our proposed scheme on start-up delay with the random peer selection scheme and TOPP scheme. Fig. 4 shows that our scheme can reduce start-up delay approximately 6~9 seconds than the random peer selection scheme and approximately 1~3 seconds than TOPP scheme when buffering time is 2 seconds; and approximately 5~8 seconds than the random peer selection scheme and approximately 2~3 seconds than TOPP scheme when buffering time is 4 seconds, because our scheme can quickly connect to the peers which have larger upload bandwidth, and pushing method can avoid a lot of request time.

IV. Conclusion

We proposed an efficient peer selection scheme and a chunk scheduling strategy to improve the quality of experience of P2P IPTV. We combine the advantages of mesh-pull based and tree-push based approaches. We established a simulation environment in OMNET++ to evaluate peer selection scheme and chunk scheduling strategy, and compare the proposed schemes with the random peer selection and TOPP schemes. The results show that our proposed schemes reduce about 20-40% channel switching delay than TOPP scheme and about one-third channel switching delay of random peer selection scheme, as well as keeps high continuity. Hence, our proposed schemes can really improve the quality of experience of P2P IPTV system.

References

[1] MOD: http://mod.cht.com.tw/, Accessed May 2016.

[2] BitTorrent: http://www.bittorrent.com/, Accessed May 2016.

[3] PPLive: http://www.pplive.com/, Accessed May 2016.

[4] PPStream: http://www.pps.tv/, Accessed May 2016.

[5] SopCast: http://www.sopcast.org/, Accessed May 2016.

[6] Fernando M. V. Ramos, “Mitigating IPTV Zapping Delay,” IEEE Communications Magazine Vol. 51, No. 8, pp. 128-133, August 2013.

[7] N. Magharei, R. Rejaie, and Y. Guo, “Mesh or Multiple-Tree: A Comparative Study of Live P2P Streaming Approaches,” in Proceedings of INFOCOM 2007, 26th IEEE International Conference on Computer Communications, pp. 1424-1432, May 2007.

[8] C. Lu, H. Zhang, and L. Sheng, “Research and Design on Peer Selection Strategy of P2P Streaming,” in Proceedings of 5th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1-4, Sept. 2009.

[9] J. Ghoshal, M. Wang, L. Xu, and B. Ramamurthy, “Variable neighbor selection in live peer-to-peer multimedia streaming networks,” in Proceedings of 5th International Conference on Broadband Communications, Networks and Systems: pp. 344-346, Sept. 2008.

[10] T.-H. Hsiao, M.-H. Hsu, and Y.-B. Miao, “Adaptive and Efficient Peer Selection in Peer-to-Peer Streaming Networks,” in Proceedings of Parallel and Distributed Systems (ICPADS), IEEE 17th International Conference on, pp. 753-758, Dec. 2011.

[11] B. Melander, M. Bjorkman, and P. Gunningberg, “A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks,” in Proceedings of IEEE Global Telecommunications Conference, Vol. 1, pp. 415-420, Nov. 2000.

[12] Y. Zhou, D.-M. Chiu, and J. C. S. Lui, “A Simple Model for Chunk-Scheduling Strategies in P2P Streaming,” IEEE/ACM Transactions on Networking Vol. 19, No. 1, pp. 42-54, Feb. 2011.

[13] N. Magharei, and R. Rejaie, “PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming,” IEEE/ACM Transactions on Networking Vol. 17, No. 4, pp. 1052-1065, Aug. 2009.

[14] C. K. Lee, W. Hyun, and S. G. Kang, “Adaptive chunk scheduling method for P2P-based multimedia streaming service,” in Proceedings of 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT), pp. 91-94, Nov. 2011.

[15] IP Location Finder: http://www.iplocation.net/, Accessed Aug 2016

[16] OMNET++: http://www.omnetpp.org/, Accessed May 2016.

[17] Chunghwa Telecom: http://www.cht.com.tw/, Accessed May 2016.