

A Tracker-based P2P System for Live Multimedia Streaming Services

Haesun Byun, Meejeong Lee

*Department of Computer Science & Engineering, Ewha Womans University, Seoul, Korea

ladybhs@ewhain.net, lmj@ewha.ac.kr

Abstract— In this paper, we propose a tracker-based Peer to Peer (P2P) system for live multimedia streaming services. Our system is based on the Proactive network Provider Participation for P2P (P4P) framework which induces the cooperation between P2P systems and network service providers to achieve the best possible P2P application performance and efficient and fair usage of network resources. In the proposed scheme, the P2P application server obtains the network status information from servers of network providers. Also, the P2P application server receives the upload/download link capacity and the stream availability information from each peer. With the above information, the P2P application server suggests a peering list for the newly joining peer. The peering considers the routing cost, the congestion level of the links, the end-to-end delay, the matching of the upload/download link capacity, and the provision of the stable streaming service. We also propose a playback synchronization mechanism that enables each peer to start the playback within the limited variation from the playback positions of source peer. Through the simulation results, it is shown that the proposed peering mechanism enhances the playback continuity and system stability, and reduces the control overhead. Also, the playback synchronization mechanism shows the better performance with respect to the variation in playback positions of peers in the entire system.

Keywords— Peer to Peer Overlay, P4P Framework, Live Multimedia Streaming, Playback Synchronization

I. INTRODUCTION

Many P2P systems are designed for the enhancement of application performance under the assumption that there is no restriction on the usage of network resources. Thus, the network service providers may have the Internet traffic control problem is, it is hard to utilize the network resources efficiently and fairly. On the other hand, if the network service providers introduce a traffic control mechanism to solve the above problem, the performance of the P2P applications may degrade severely in some cases. The P4P working group of Distributed Computing Industry Association (DCIA), hence, proposed an architecture called P4P to allow for more effective cooperative traffic control between P2P systems and network service providers [1]. In the P4P, servers, which are operated by network providers, explicitly provide the network information such as network status, policies and network providers' capabilities to the P2P application servers(tracker-based) or peers(tracker-less). The P2P application servers and the peers consider both application requirements and the

network information to control the P2P traffic. The objectives of the P4P are the optimization of network resource efficiency as well as the improvement of application performance.

We propose a tracker-based P2P system for live multimedia streaming services. In the proposed scheme, the P2P application server obtains the network status information such as delay, link utilization status, and routing cost from servers of network providers. Also, the P2P application server receives the upload/download link capacity and the messages related to stream availability from each peer. With the above information from the servers and the peers, the P2P application server suggests a peering list for the newly joining peer. The peering considers the routing cost, the congestion level of links, the end-to-end delay to satisfy the live multimedia streaming services, the matching of upload/download link capacity of the joined peers, and the provision of the stable streaming services. Through the proposed peering, the continuity of the P2P application and the robustness of the P2P system can be enhanced and the efficiency of network resources can be increased.

The playback synchronization among peers is also one of the important factors that affect the quality of services in the live multimedia streaming in addition to the playback continuity and the system stability. In the data-driven approaches, buffer map, that means the availability of stream data, is used to synchronize between the peers [2][3][4]. That is, by examining partners' buffer map, the peer finds the largest sequence number of the received segments among its partners and set the segment number shifted by a system parameter from the largest sequence number as initial request position. Although the peer can reduce the initial buffering time when the data-driven approach is deployed, it can increase the variation in the playback positions between peers since the peer determines the initial request position with the buffer maps of the partners regardless of that of source peer. Therefore, we introduce a playback synchronization mechanism that enables each peer to start the playback within the limited variation from the playback positions of source peer.

Through the simulation results, it is shown that the peering mechanism of the proposed scheme enhances the playback continuity of multimedia streaming services and the stability of the P2P system, and reduces the control overhead. In addition, the peering mechanism reduces the traffic volumes

on the backbone links and evenly distributes the traffic load on the links. Also, the playback synchronization mechanism shows the better performance with respect to the variation in playback positions of peers in the entire system.

The rest of this paper is organized as follows: In Section 2, we give a detailed explanation on the architecture of the proposed scheme. We evaluate the performance of the proposed scheme in Section 3. Finally, Section 4 concludes the paper.

II. TRACKER-BASED P2P OVERLAY SYSTEM

We first describe the basic components and the overview of the proposed scheme in Section 2.A. In Section 2.B and 2.C, we present the peering suggestion mechanism and the playback synchronization mechanism respectively.

A. Basic Components and Overview

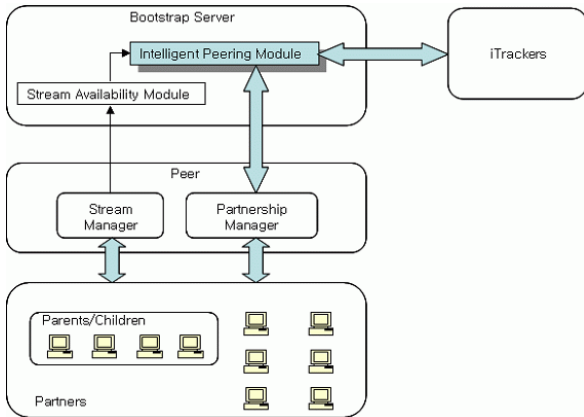


Figure 1. The proposed P2P overlay system design

Fig. 1 depicts the system design for the tracker-based P2P overlay. There are the peers, the bootstrap server, and the iTrackers in the system. Each peer is considered as a member in the overlay network and has a partial list of the currently active peers in the system as partner peers. The partial list is managed by the partnership manager. The stream manager pulls a stream from the parent peers and pushes it to child peers. Also, the stream manager periodically examines its' buffer and requests the missing segments to partner peers if some segment is missing.

The bootstrap server is a kind of application servers and is an entry point to the P2P system. The bootstrap server in the proposed scheme has the stream availability module and the intelligent peering module. The stream availability module receives the stream availability related messages from the peers. It periodically computes the stream availability weight with the stream availability information obtained from the messages received from the peers and the elapsed time since the messages reception. The stream availability weight indicates the degree of stability a peer can serve the stream with. The intelligent peering module selects the appropriate partner peers for the peering of the newly joining peer. The peering is determined by the network status from the iTrackers, the upload/download link capacity from the peers, and the

stream availability weight computed by the stream availability module. The iTracker is a node for monitoring the network status and delivers the network status information to the bootstrap server.

The proposed P2P system uses the concept of multiple substreams similar to the data-driven approaches. In this concept, a peer can be the parent for one or more substreams to a child peer. Also, a peer can be a parent peer for some substreams while it is a partner peer for the other substreams for a child peer. The child peer pulls the corresponding substream from the parent peer and the parent peer pushes all segments of the substream required by the child peer. Unlike the data-driven approaches, in the proposed scheme, if a peer does not receive a segment from the corresponding parent peer, the peer pulls the segment from a partner peer. As a result, the proposed P2P system consists of multiple tree overlays and auxiliary mesh overlay.

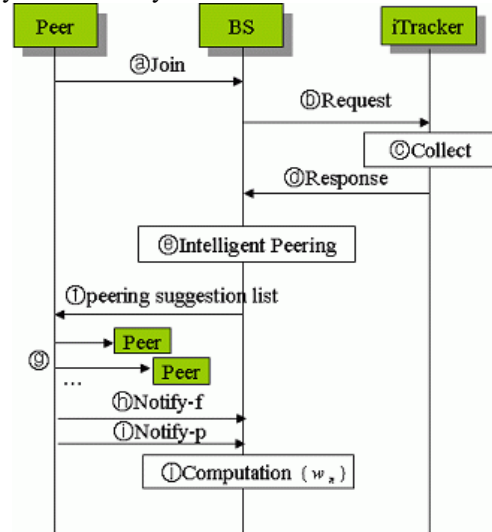


Figure 2. The procedure to construct the P2P overlay system

Fig. 2 presents the procedure to construct the proposed P2P overlay system. The newly joining peer sends the IP address, the ID of PoP (PID), the Autonomous System (AS) number, and the upload/download link capacity to the bootstrap server^①. The bootstrap server adds the newly joining peer to the member peer list of the system and periodically requests the network status information to the iTracker of the AS which the peer belongs to^②. The iTracker collects the network status information such as the routing cost, the delay, and the link utilization status for all links e on the path from PID- i to PID- j within its AS^③ and sends the results to the bootstrap server^④.

The bootstrap server performs the intelligent peering mechanism for the newly joining peer^⑤. In the intelligent peering mechanism, the bootstrap server makes the peering suggestion list using the network status information from the iTrackers, the stream availability weight, and the upload/download link capacity. After then, it delivers the peering suggestion list to the newly joining peer^⑥. The newly joining peer establishes connections with peers in the peering

suggestion list and determines the initial request position by performing the playback synchronization mechanism. Then, the newly joined peer requests the substreams starting from the initial request position to the parent peers^⑨. Once the newly joined peer receives the first segment from its parent peer, it sends a Notify-f message to the bootstrap server^⑩. Also, once the newly joined peer completes buffering up to the initial synchronization position determined by the playback synchronization mechanism, it sends a Notify-p message to the bootstrap server^⑪. These messages are used for the computation of the stream availability weight. Once the bootstrap server receives the messages, it periodically computes the stream availability weight for the peer with the elapsed time from the peer's arrival till it receives the messages, as well as the time elapsed since the messages reception^⑫.

In section 2.B, we describe the method that the iTracker computes the network status information, the methods that the bootstrap server computes the stream availability weight and makes the peering suggestion list. In Section 2.C, we present the playback synchronization mechanism.

B. P4P-framework based peering suggestion mechanism

Live multimedia streaming service is more sensitive than data services in terms of delay and loss rate. For the live multimedia streaming guaranteed the quality of services, the bootstrap server needs the peering cost reflecting direct restriction on delay rather than the peering cost presented in [3]. In the proposed scheme, hence, the iTracker presents the link utilization status and the delay between PoPs to be reflecting in the peering of P2P system to bootstrap server explicitly. Through the explicit information, the usage of network resources and the throughput of live multimedia streaming services can be supported efficiently. When the bootstrap server makes the suggestions for peering, we also reflect the upload/download link capacity and the stream availability of peers to select the most suitable peer in terms of bandwidth and provision of continuous streaming.

1) The collection of network status information at the iTracker

In the proposed scheme, the iTracker derives the routing cost(r_e) from OSPF weights and BGP preferences and periodically collects the delay(d_e) and the link utilization status(u_e) for the link e on the path from PID- i to PID- j in the AS. After then, it computes $r_{i,j}$, $d_{i,j}$, and $u_{i,j}$ as follows:

$$r_{i,j} = \sum_{e \in \text{path}(i,j)} r_e \quad (1)$$

$$d_{i,j} = \sum_{e \in \text{path}(i,j)} d_e \quad (2)$$

$$u_{i,j} = \prod_{e \in \text{path}(i,j)} u_e \quad (3)$$

, where u_e is computed by the equation (4) and (5) as follows:

$$l_e = \frac{b_e}{c_e} \quad (4)$$

, where c_e is the capacity of link e , and b_e is the amount of background traffic on link e (including the P2P traffic).

$$\begin{cases} \text{if the } l_e \text{ is small than } LT(\text{link threshold}), u_e = 1 \\ \text{otherwise, } u_e = 0 \end{cases} \quad (5)$$

2) The computation at the bootstrap server

The bootstrap server periodically computes the stream availability weight for the joined peers. First, the stream availability of a peer n is defined as follows:

$$p_n = \frac{1}{t_{init}^n} \cdot \alpha + \frac{1}{t_c^n} \cdot \beta + t_s^n \cdot \gamma \quad (6)$$

, where t_{init}^n denotes the elapsed time from the peer's arrival till it receive the first segment till the completion of. t_c^n denotes the time elapsed from the first segment reception the buffering up to the initial synchronization position. Refer to the 2.C for the definition of initial synchronization position. t_s^n denotes the time elapsed between the two messages, Notify-f and Notify-p. α , β , and γ denote the weights for the three kinds of time values respectively. That is, the bootstrap server gives the different weight since the range of the value for t_{init}^n , t_c^n , and t_s^n could be very different. Through the equation (6), as the peer obtains the stream from the parent peers more quickly, the stream availability of the peer increase. Also, as the peer resides in the system for a longer time, the higher the stream availability of the peer. Thus, the stream availability is considered for the stability of the parent peers and the continuity of the peer.

The bootstrap server, then, computes the stream availability weight for the each peer in the same PoP. The stream availability weight of the peer n is defined as follows:

$$w_n = \frac{p_n}{\sum_{a \in A} p_a} \quad (7)$$

, where A is the set of active peers in the same PoP.

3) The peer selection at the bootstrap server

The bootstrap server selects the optimal peers using the stream availability weight, the upload/download link capacity of the peers, and the network status information. The selection is completed by three stages similar to[1]. Assume that a peer needs m partner peers. Also, assume that the number of substreams is s and the number of peers selected as a parent peer is k .

First, it selects among those that are located at PID- i to which the newly joining peer belongs. The bootstrap server selects up to s peers, one parent peer per each substream, in ascending order of the stream availability weight among peers with the available upload bandwidth in the PID- i . If it cannot found enough peers, it performs the second stage.

In the second stage, the bootstrap server expands the selection to other PIDs but in the same AS. In this stage, the bootstrap server finds the PIDs with the conditions that the $d_{i,j}$ is small than the delay threshold and the $u_{i,j}$ is 1. Then, it sorts

PoPs in descending order with respect to r_{ij} of the PIDs. After then, it selects up to s peers. If it cannot found enough peers, it performs the third stage.

In the last stage, the bootstrap server selects peers from the other AS up to a total of s peers. The selection depends on the preference of the ISPs which the peer belongs to.

If the bootstrap server selects k parent peers for s substreams, it additionally selects $(m-k)$ partner peers. The procedure of partner peer selection is the same to that of the parent peer except that the available upload bandwidth is not considered. This is because that the partner sends the single urgent segment rather than sending a substream continuously, that is, there is no continuous consumption of bandwidth.

C. The playback synchronization mechanism

In the data-driven approaches, the peer determines the position shifted by the system parameter T_p from the largest sequence number in the buffer map of the partner peers for the initial request position. The T_p is the maximum allowable latency for a partner behind to others. The initial request position is used to synchronize the playback between the peer and its partner peers. However, the time-sensitive efficiency can not be guaranteed based on the initial request position. First, we will discuss the problem. The average bit rate required for one single substream transmission is R/s . Suppose that the parent peer p is capable of providing upload bandwidth $r_{\uparrow} > R/s$ to the child peer q . In this case the child peer q can quickly catch up with the parent peer p for the corresponding substream. If the peer q does not serve as a partner peer until it finishes the catch up process, the playback synchronization problem does not arise. However, if a newly joining peer n establishes partnership with the peer q before it catch up with the peer p , the newly joining peer n will receive the buffer map that falls behind significantly. In the worst case, if the peer n receives the buffer map that falls behind the source significantly from all partner peers, the playback synchronization between the peer n and the source peer can not be guaranteed within the T_p .

In the P2P live multimedia streaming services, the playback synchronization among the peers is one of important factors for the interaction among the users. We introduce a playback synchronization mechanism that enables each peer to start the playback within the limited variation from the playback positions of source peer. Fig. 3 describe the relation of the initial request position, the initial synchronization position, and the estimated playback position. In our playback synchronization mechanism, each peer estimates the playback position of the source peer. For the estimation, the source peer notifies its playback position to its partner peers. The peers receiving the playback position from the source peer periodically estimates the next playback position of the source peer based on the number of frames per second. The peer delivers the estimated playback position to its partners and so on.

With the reception of the estimated playback position from one or more peers, the peer needs to determine the initial request position. The peer finds the largest playback position

of the received playback positions. The largest playback position becomes the initial synchronization position. The peer sets the segment number shifted by a system parameter T_p from the initial synchronization position as the initial request position.

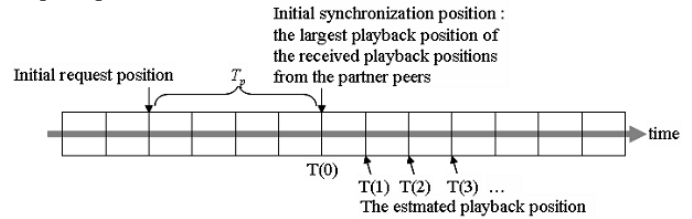


Figure 3. Relation of the initial request position, the initial synchronization position, and the estimated playback position

III. PERFORMANCE EVALUATION

We have constructed a simulation model with NS-2[5] to evaluate the performance of the proposed scheme(LiveP4P). For comparison, we have also implemented CoolStreaming[2] which is one of the representative data-driven mesh system. The topology of the simulation network consists of three AS and there is one iTracker in each AS. The number of routers in a single AS is 36; we consider each router as PoP. Peers are connected to each PoP. The propagation delay between the peers and the PoPs were set to 0.1~0.2ms and the propagation delay between the PoPs has been set to 0.5~1ms. 200Mbytes of link capacity were used for the routers. The link capacity of the peers and the distribution rate of the peers were set as follows: 10% of 100Mbps upload/download capacity peers; 20% of 20Mbps peers; 50% of 10Mbps peers; 20% of 2Mbps peers. The peers join the system in accordance with a Poisson process. The live multimedia stream length was 500 seconds and the data rate of the source peer is considered to be 400Kbps with 30frames per second. Buffer length is set to be 60seconds and the parameter T_p for synchronization is set to be 20seconds. Also, once 20seconds of contiguous data are available, playback begins, and once 15seconds of contiguous data are available, buffer detection for parent substitution begins. For comparison, the setting guidelines in CoolStreaming can be found in [2]. In our experiment, the number of peers is varied from 100 to 1000.

A. Application Performances

Fig. 4 and 5 show the average continuity index and the ratio of peers with segment loss rate less than 1%. The average continuity index is the number of segments that arrive before playback deadlines over the total number of segments. As shown Fig. 4, both schemes maintain this value at a high level (98%).

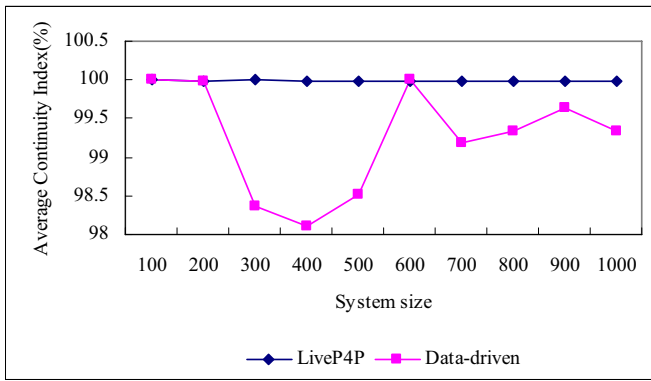


Figure 4. Average continuity index

In the proposed scheme, almost all of peers satisfy the segments loss rate less than 1% as shown Fig. 5. In the data-driven approach, however, only 60~99% of peers satisfies the segments loss rate less than 1%.

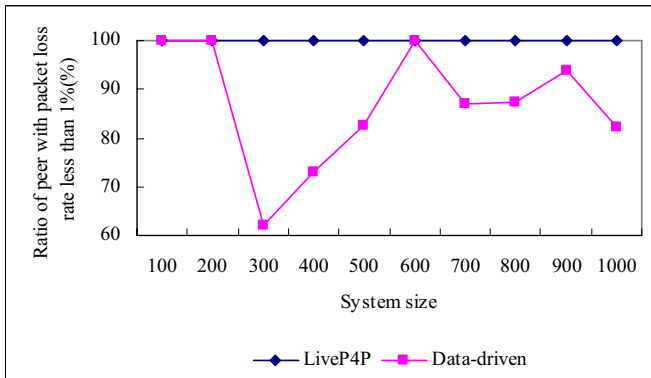


Figure 5. Ratio of peer with packet loss rate less than 1%

Fig. 6 depict the start-up latency, which is the time from to requesting to join the session and till receiving enough segments (20seconds continuous segments in our experiment setting) to start playback. For the start-up latency, the data-driven approach has smaller minimum value than the proposed scheme. This is because that the initial request position in the data-driven approach is earlier position than that in the proposed scheme. In the data-driven approach, the possibility that the parent peers might have the 20seconds continuous segments in its buffer is higher than the proposed scheme. However, the proposed scheme shows the better performance with respect to the maximum and average of start-up latency values. That is, the possibility that the parent peers do not have the 20seconds continuous segments in its buffer is also higher than the proposed scheme. Once a peer obtains the part the substreams, the possibility that can receive the substreams continuously in our scheme is higher since the bootstrap server does the peering based on the stream availability and the upload/download bandwidth matching. Similar reasons also explain the results of continuity index shown in Fig. 4 and 5.

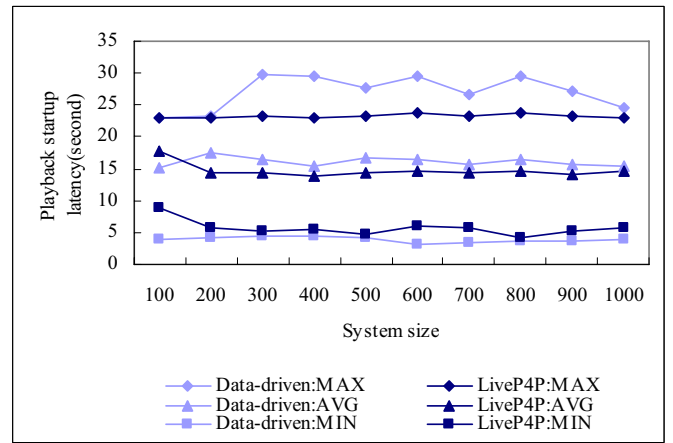


Figure 6. Playback start-up latency

B. P2P System Performances

Next we present the results of the control overhead and the synchronization variation.

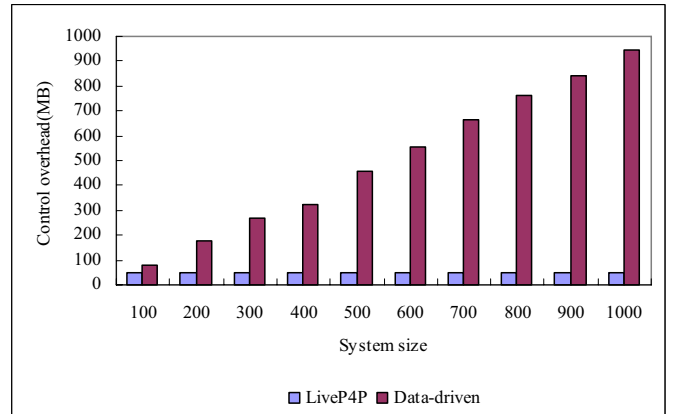


Figure 7. Control messages overhead

Fig. 7 shows the total control overhead of all peers. The data-driven approach introduces high control overhead because of the continuous exchange of buffer maps among the partners. The proposed scheme keeps the constantly control overhead regardless the system size.

The time difference of playback synchronization is affected by the initial request position and the start-up latency. Fig. 8 depict the time difference between the source peer and the peers in playback. As shown Fig.8, the proposed scheme shows the same or the better performance in the maximum, average, minimum and standard deviation values of playback point variation. The time difference of the playback position is small in the proposed scheme since the peers determine the initial request position based on the initial synchronization position that is estimated for the playback position of the source peer, and besides the average start-up latency in our scheme is also small as shown in Fig. 6.

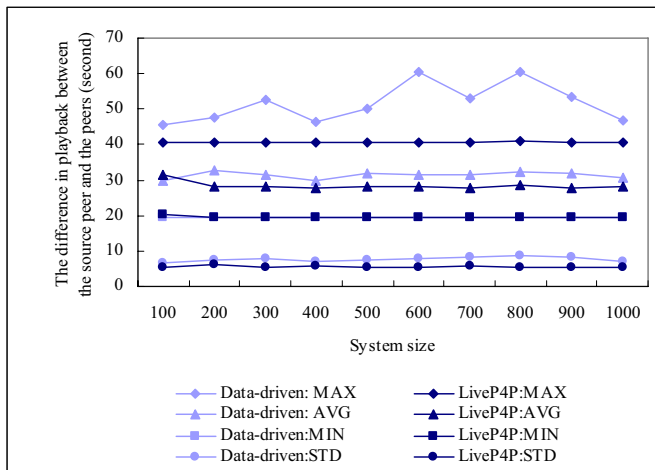


Figure 8. Difference between the source peer and the peers in playback point

C. Network Performances

Finally, we present the results of the network traffic volumes and the variation of traffic volumes on the backbone links.

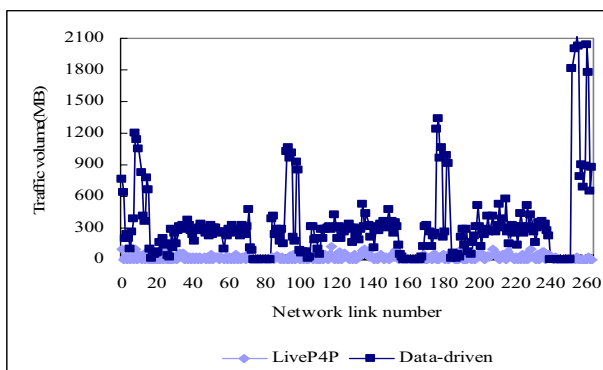


Figure 9. Traffic volumes

Fig. 9 shows the traffic volumes on the each backbone network link when the number of peers is 1000. The traffic volume of the data-driven approach is approximately 320MB on the backbone links on average, while the proposed scheme reduces the backbone traffic volume to approximately 20MB. For the newly joining peer, in the proposed scheme, the bootstrap server preferably let the peers in the same PoP serve the newly joining peer. Thus, the network traffic on the backbone links can be reduced.

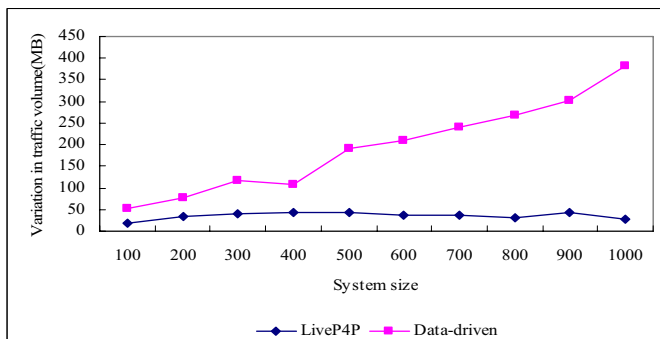


Figure 10. Variation of traffic volumes

To study the load balancing of traffics on the backbone links, we measured the variation in traffic volumes on average for the each backbone network link. As shown Fig. 10, the variation in traffic volumes increases in the data-driven approach as we increase the number of peers in the system. On the other hand, the variation is kept constant in the proposed scheme. The proposed scheme can evenly distribute the traffic loads since the bootstrap server controls the traffic using the network status information such as delay and link utilization status from the network servers for the peering.

IV. CONCLUSIONS

We presented the tracker-based P2P system for live multimedia streaming services. We evaluated the performance of our scheme and compared it with the data-driven approach.

Our evaluations demonstrate that it improves the performance with respect to the average continuity index, the ratio of peer with packet loss rate less than 1%, and the start-up latency. We also show that our playback synchronization mechanism can reduce the time difference in playback between the peers. Finally, our scheme shows the better performance with respect to the network efficiency.

ACKNOWLEDGMENT

This work was supported by the Korea Science and Engineering Foundation(KOSEF) grant funded by the Korea government(MEST) (No. 2010-0000269)

REFERENCES

- [1] H. Xie, R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: Provider portal for (P2P) applications," in Proc. of ACM SIGCOMM, pp.351-362, 2008.
- [2] S. Xie, B. Li, G. Y. Keung, and X. Zhang, "Coolstreaming: Design, Theory, and Practice," IEEE Transactions on multimedia, Vol. 9, No. 8, pp.1661-1671, 2007.
- [3] X. Zhang, J. Liu, B. Li, T.S.P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," IEEE Infocom, pp.2102-2111, 2005.
- [4] www.pplive.com
- [5] www.isi.edu/nsnam/ns