An Analysis of User Dynamics in P2P Live Streaming Services

Kunwoo Park, Dukhyun Chang, Junghoon Kim, Wonjun Yoon, and Ted "Taekyoung" Kwon
School of Computer Science and Engineering,
Seoul National University, Korea
Email: {kwpark, dhchang, kjh, wjyoon}@mmlab.snu.ac.kr, tkkwon@snu.ac.kr

***Abstract*—Peer-to-peer (P2P) live streaming services are getting more popular as the average link capacity in the Internet becomes greater for end hosts. In order to provide a high quality P2P live streaming service, it is essential to handle user dynamics effectively to mitigate streaming disruptions caused by user churn. We collect massive user traces from the largest P2P live streaming site in Korea, namely, Afreeca. Based on the trace results obtained, we analyze user dynamics in the service. In particular, we analyze the user behaviors in terms of three criteria: long-stay users, short-stay users, and user churn. (1) The results show that a certain amount of long-stay users exist in a live streaming session, and it is challenging to recognize the long-stay users online, which will help to provide more resilient live streaming service. (2) Around 20% of the users stay in a session shorter than 1% of the session length. Short-stay users incur high control overhead and we discuss how to efficiently handle the short-stay users. (3) By intentionally disrupting a video streaming, we closely observe the users' leaving behavior and suggest the acceptable recovery time to keep the users remaining the session. We believe our findings can be useful not only to P2P live streaming systems, but also to every live streaming systems.**

I. Introduction

As the average link capacity in the Internet becomes greater for end hosts, streaming services for individual users are gaining much attention in the literature [15] [16] and industry [5] [6] as well. CacheLogic [1] reports peer-to-peer (P2P) traffic increased from 5% in 1999 to 50% in 2004 while other traffic had significantly decreased. A recent report by ipoque [2] shows P2P and streaming packets occupy 50% and 10% of the Internet traffic in 2008, respectively. Server-based solutions for streaming services are costly as the number of the users increases. Thus P2P technologies are often used in streaming applications leveraging ever increasing last mile link speeds, which is the focus of this paper.

Streaming services can be classified into video-on-demand (VoD) services and live streaming services. In VoD services, since the whole video session exists at the service start, effects of network fluctuations or streaming disruptions (due to user churn) can be mitigated by buffering. However, it becomes a critical issue to deliver video packets timely in live streaming services. Video streaming packets are generated at a source and sent to users continuously. To play the video in real-time, it is hard to maintain sufficient buffering. Therefore, it is more challenging to design an efficient distribution structure for live streaming services. In particular, P2P technologies make this issue more complicated due to user churn.

In this paper, we collect massive user traces of live streaming services, and analyze user behaviors, how they join and leave live streaming services. Trace results are classified by the streaming contents' type and popularity, which affects user behaviors significantly. Especially, we analyze the user behaviors in terms of three criteria.

- **Long-stay users:** A long-stay user refers to a user who stays in a session longer than 50% of the session length. Do stable users exist, and how to recognize them online?
- **Short-stay users:** A short-stay user refers to a user who stays in a session shorter than 1% of the session length. As we will discuss later, a substantial portion of users are short-stay users; they tend to leave the session within a short time. Why should a system designer consider short-stay users, and how to handle them efficiently?
- **User churn:** A live streaming system suffers from a high rate of user churn. Especially user churn is a challenging issue in two cases: (1) when a session is opened (flash crowd), (2) streaming services to some users are disrupted by other users' failure. How much churning rate should be supported by a live streaming system and what is the acceptable recovery time to keep the users remaining the session?

The contributions of this paper are twofold. First, we analyze the user behavior of a P2P live streaming service, collecting user traces from the largest live streaming site in Korea. Second, based on the analysis, we suggest system design considerations for live streaming services. We believe our findings can be useful not only to P2P live streaming systems, but also to infrastructure-based systems such as content delivery networks.

The rest of this paper is organized as follows. In Section II, we first introduce the live streaming service (a commercial service) which we collected traces from. Sections III and IV show the traces and their analysis with system design considerations. Related work are given in Section V. Finally, Section VI concludes this paper.

II. Data Collection

We analyze user traces collected from *Afreeca*, a P2P live streaming service provider [3]. In this section, we introduce services provided by Afreeca, and describe the user trace collection methodology we used.

A. Afreeca

Afreeca [3] is the most popular live streaming service provider in Korea; for instance, there have more than 20,000,000 accumulated streaming sessions from 2006 until now. Also in 2009, Afreeca rated 300,000 simultaneous views at peak time. Actually, Afreeca does not provide any video contents of its own, but offers a live streaming service framework for users to broadcast their contents. The service framework of Afreeca is not well-known. According to the company's white papers [4], they use Dynamic Relay Distribution, which is a hybrid of server Farm and P2P technologies.

A user can either broadcast or subscribe to a particular session in Afreeca. A broadcasting user is referred to as a Broadcast Jockey (BJ), who plays a video content from his own computer, and shares his monitor screen with other users joining his session. Users join the session initiated by a BJ, watch the video and chat with each other. Popular video contents are television programs, gaming screens of the BJ, and the performance of the BJ over his web camera. Rebroadcasting television programs may incur copyright issues. Afreeca blocks rebroadcasting copyrighted contents only manually. When we collect user traces, only copyright-free sessions are taken into account.

B. Trace Collection

We set a TV tuner card in a PC and rebroadcasted various television programs over Afreeca. As a BJ, we initiated a session in Afreeca and collected user traces about who and when join or leave the session. We rebroadcasted the same TV program in four different sessions simultaneously on four PCs each. Since users joining and leaving four sessions behave independently, we believe traces of four sessions are statistically more meaningful than that of a single session. The data collection is performed over 9 months, from August 2008 to April 2009. The total rebroadcasting time for all the sessions is about 600 hours and the accumulated number of users in the trace set is around 70,000.

We classified TV programs based on their type and popularity. We expect user behaviors will differ depending on the type of the program they watch. Popularity of the program will also affect the user behavior. In this paper, we investigate three types of programs: TV dramas, news, and talk shows. Each type will be denoted by Type-D, Type-N, and Type-S, respectively. Due to the policy of Afreeca services, a session has a limit of the maximum simultaneous users (in most cases 200 users are the upper bound). Depending on the popularity, sessions are classified into Rank-A and Rank-B. A Rank-A session refers to a session whose number of users is equal to the maximum limit (users are backlogged). All the others are Rank-B sessions. In a Rank-A session, no more users can join the session until an existing user leaves the session, and the room will be filled with a new user in a short time. In a Rank-B session, the number of users does not reach the maximum capacity of a session, and hence a user can join or leave the Rank-B session at any time.

TABLE I
AVERAGE SI VALUES.

	Type-D	Type-S	Type-N
Rank-A	0.54	0.61	0.43
Rank-B	0.35	0.48	0.65

III. USER TRACE ANALYSIS

A. Long-stay Users

When you build a distribution structure for live streaming using P2P technologies, it is more reliable and efficient if the core part of the structure is built with long-stay users (also called *stable users*). To utilize these long-stay users, the first step is to verify the existence of long-stay users. After that, we discuss how to recognize these users online.

Figure 1 is a cumulative distribution function (CDF) of the stay duration of the users according to the program types and popularity levels. For instance, (D,A) refers to TV drama programs with high popularity. In (D,A), 63% of the users stay in the session for less than 50% of the session length, while 78% of the users in (D,B) stay in the session for less than 50% of the session length. We collected user traces in 2 to 5 sessions for each program type and popularity. From Figure 1, Rank-A sessions of Type-D and Type-S have more long-stay users than Rank-B sessions. However, Type-N programs exhibit different phenomena. (N,B) has 18% of users who watch the news longer than 75%, while only 8% of users in (N,A) watch the news for more than 75%, which means (N,B) has more long-stay users than (N,A).

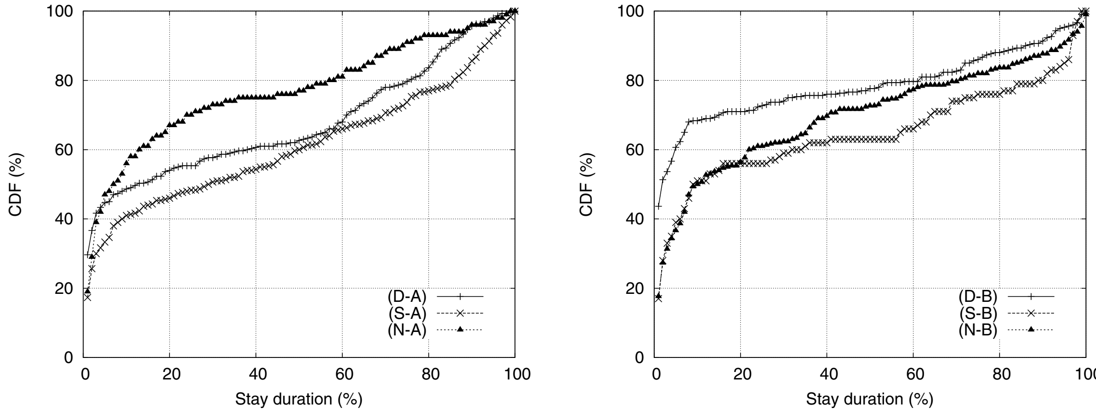
[11] quantifies the existence and importance of stable users by the Stability Index (SI) for each user, which is defined as

$$SI = min(\frac{2s}{L-t}, 1),$$

where L is the whole session length, s is the user stay duration, and t is the arrival time of the user. Table I shows the average SI (ASI) values of users for each program. The ASIs of (D,A) and (S,A) are greater than (D,B) and (S,B), respectively. The unusual characteristics of Type-N are also observed in the table. The ASI of (N,B) is 0.65, which is the highest value in the table.

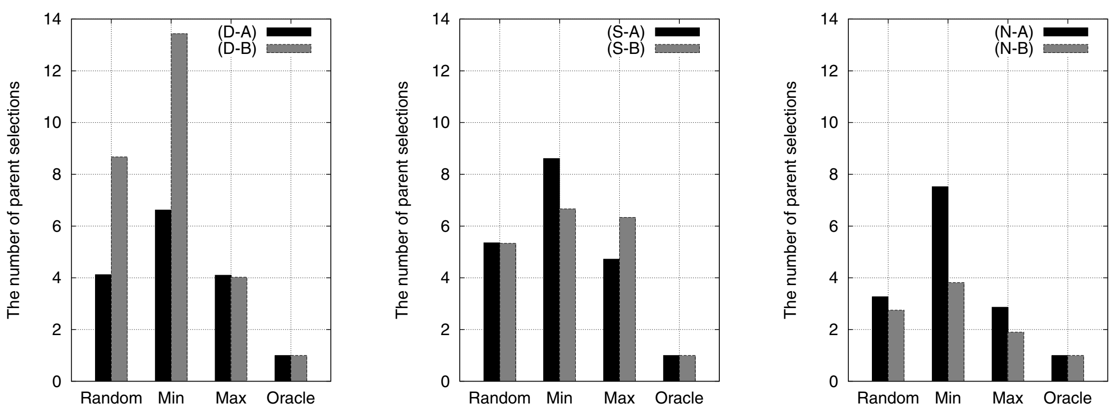
Recognizing long-stay users can help to build a stable P2P streaming system. To this end, we compare a few measures to recognize long-stay users online. **Oracle**, which has the omniscient information about when each user will join and leave the session. Thus, it will place the most stable users at the core of the distribution structure and achieve the best performance. However, this is not feasible and some online heuristic algorithms should be designed to estimate how long a user will stay in the session. The second measure, **Max**, is to place older users at the relatively core part of the distribution structure.

To validate the effectiveness of **Oracle** and **Max**, we carry out a simple experiment. A source forms a tree to distribute some streaming packets. Each node is a user, and each user can join and leave the session following the log we collected.



(a) Rank-A sessions (b) Rank-B sessions

Fig. 1. CDF of the number of long-stay users.



(a) Type-D programs (b) Type-S programs (c) Type-N programs

Fig. 2. The number of parent selections.

A node has a parent, and can have up to 5 child nodes at maximum. Whenever a node's parent leaves the session, the node should select another parent to participate in the distribution structure. In addition to **Oracle** and **Max**, we also consider **Min** and **Random**:

- **Oracle:** Each node has the omniscient information about other nodes' behavior, hence selects a parent node who will leave the session last among the current nodes who can accept child nodes.
- **Max (Min):** Each node is assumed to know which node has participated in the session for the longest (shortest) time. Thus, the node will select the longest (shortest) staying node as its parent.
- **Random:** Each node randomly selects its parent node.

After the simulation, we counted the number of parent selections experienced by each user, which gives a clue for the quality of live streaming over P2P overlays. Figure 2 shows the result. **Oracle** performs best obviously. Since the numbers of joining and leaving users are different among the experiments, the number of parent selections is normalized with respect to that of **Oracle**. **Min** performs worst among the four criteria for all the experiments. **Random** is better than **Min**, but it performs slightly worse than **Max**. This result is consistent with the ASI values of Table I. The ASI of (N,B) is the highest (0.65), and (N,B) in Figure 2 shows the least number of parent selections. It is surprising that the performance of **Max** and **Random** is not quite different. In (S,B), **Max** even performs slightly worse than **Random**. From these results, selecting old users as the parent is not such an effective measure to provide high quality streaming services.

B. Short-stay Users

Users join a session and leave the session at will. A user leaves a session because the content is not what she expected, or the streaming quality is not good enough, or she has other things to do in the meantime. To provide a streaming service to a user (no matter she is a short-stay or a long-stay user), a system needs to allocate resources (CPU, storage, network bandwidth) and put the user in the right position in the P2P distribution structure, and run the cleanup process when the user leaves the session. Therefore, short-stay users consume more resources than long-stay users in terms of the control

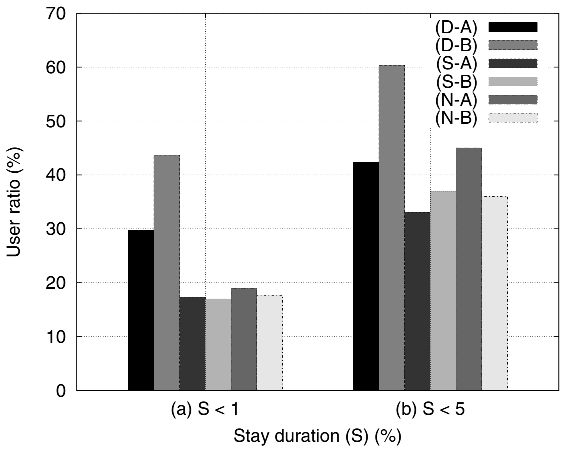


Fig. 3. Proportion of short-stay users.

(setup and cleanup) overhead of the system.

Figure 3 shows the proportion of short-stay users. Figure 3 (a) is the proportion of users whose stay duration is less than 1% of the session length, and (b) is the proportion of users whose stay duration is less than 5% of the session length. In Type-D, more than 40% of the users stay less than 5% in the session. Except for the Type-N, short-stay users increases when the popularity of the program becomes low. Especially Type-D, compared to Type-S and Type-N, has a high proportion of short-stay users.

As Figure 3 shows, roughly 20% of users stay less than 1% of the session length. If a distribution structure is complex (maybe to provide resilient service such as multiple trees or a mesh topology), the system will waste lots of resources to perform setup and cleanup for the short-stay users. Things get worse when disruption occurs. If the streaming video is disrupted, users will leave the session (which will be detailed in section IV-B) and the session will now have a room for new users. A new user, who notices the video is disrupted, will again leave the session. As a result, a system will even suffer from high user churning due to the heavy distribution structure, which is to prevent. On the other hand, light-weight structures (such as simple single-tree structures) efficiently handle the short-stay users, but has a limitation of service resiliency. Therefore, it would be ideal to simply locate a new user in the structure without incurring heavy control overheads (not wasting the existing long-stay users' resources). After a user has stayed in a session for some time, it is affordable to allocate more resources to the user (e.g., control overhead).

We propose an overhead indicator function, OI, as

$$OI = \frac{1}{N}\sum_{i=1}^{N}\frac{C_i}{S_i}.$$

N is the total number of users, S_i is the stay duration of user i, and C_i is the control overhead invested to user i. The control overhead can be defined as the number of messages exchanged or the number of other users involved to setup and cleanup for the user. OI is the average overhead incurred by each user.

TABLE II
THE NUMBER OF PARENT SELECTIONS AND OI.

Select	D,A	D,B	S,A	S,B	N,A	N,B
Oracle	1.00	1.00	1.00	1.00	1.00	1.00
Random	4.61	7.42	5.19	6.00	3.57	2.09
Mesh	0.21	0.13	0.22	0.20	0.11	0.16
OI	D,A	D,B	S,A	S,B	N,A	N,B
Oracle	0.05	0.07	0.04	0.03	0.05	0.03
Random	0.08	0.12	0.06	0.05	0.08	0.05
Mesh	0.42	0.45	0.31	0.21	0.27	0.22

We performed a simple experiment to compare OI values for three different distribution structures: A tree with **Oracle**, a tree with **Random**, and a mesh with up to 5 bidirectional links for each node (a new node sets up three links with the existing nodes). First two structures are the same as the ones in the previous section. In our experiment, we counted the number of connected nodes to setup or cleanup for a node in the structure as a control overhead C. Table II shows the number of parent selections and OI of each structure. Since a node in mesh has up to 5 bidirectional links to other nodes, the number of parent selection is less than that of **Oracle**. On the other hand, the mesh suffers from high control overhead which leads to high OI value. Since Type-D programs have more than 30% of short-stay users, their OIs are higher than those of Type-S and Type-N programs. If a system has a large number of short-stay users, a mesh structure becomes more inefficient. Therefore, a streaming system which can adapt to the dynamic churning rate will be most desirable.

IV. EFFECT OF DISRUPTIONS

A. *An Example Session with Two Disruptions*

Figure 4 shows the trace result of a popular session (Rank-A) of a TV drama program (Type-D) with two intentional disruptions. The x-axis indicates the time progress of the session in percentage. Depending on the time progress, there are 6 zones. The y-axis is the ratio of the number of current users to the maximum user capacity of the session for the green line, while the blue and red lines indicate the ratio of the accumulated number of joining and leaving users to the maximum user capacity of the session, respectively. Note that difference between the accumulated number of joining users and the number of leaving users equals to the number of current users.

When a BJ initiates a Rank-A session (Zone-A), the flash crowd effect is observed. After the session is fully occupied with users (when Zone-B starts), the number of joining users for a given interval is roughly the same as the number of leaving users for the same interval. Users join the session whenever there is a room due to high popularity. To observe the effect of service disruption, the BJ intentionally pauses (when Zone-C starts) broadcasting for 200 seconds and then resumes broadcasting. After sometime, the number of current users reaches the maximum capacity again (when Zone-C

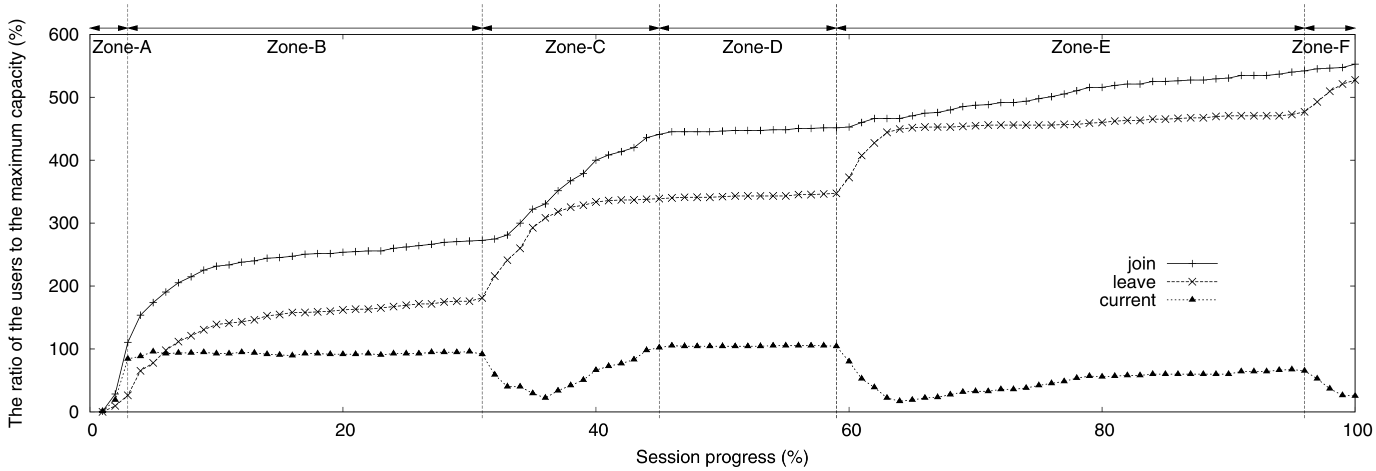


Fig. 4. Number of remaining users in a Type-D, Rank-A session.

ends), Zone-D exhibits the similar pattern to Zone-B. When the time progress is 59%, the BJ pauses the broadcasting again. When there is the second disruption after more than half of the drama has progressed (Zone-E), the joining rate of the users is far less than that of Zone-C. At the moment of the time progress being 97%, the main performance of the program ends and the ending credit starts (Zone-F starts). In Zone-F, users are leaving the session slowly and 20% of the users remain in the session until the last moment.

B. User churn

To mitigate the disruption in P2P streaming systems, many techniques such as multi-path or heavy signaling are adopted. However, there is a tradeoff between control overhead and the disruption level. To be more specific, the disruption level can be interpreted as the disruption time or recovery duration. A system designer invests the system resources to provide as seamless services as possible. The disruption time and required network resource have an inverse proportional relationship; sending HEARTBEAT messages to neighboring nodes at short intervals will allow a node to detect the absence of a departing neighbor shortly, but it will occur significant overhead.

Figure 5 shows the ratio of the number of remaining users to the number of users at the beginning of the disruption. Only users who were in the session at the onset of the disruption are counted. This time, x-axis is the time in seconds. Each disruption continued for 200 seconds. Most of the users in (D,A) session wait until 80 seconds (maybe anticipating the disruption will end soon) and then rapidly move out from the session after 80 seconds. For the provisioning, for the sessions like (D,A), the users are more tolerable to the longer disruption time. So, we can invest less resources to mitigate disruptions for (D,A) sessions. However, users in (D,B), (S,A), (N,A), and (N,B) quickly leave the session after 30 seconds. For these types of programs, we should provision a quick session recovery for a node failure although a system may waste some resources.

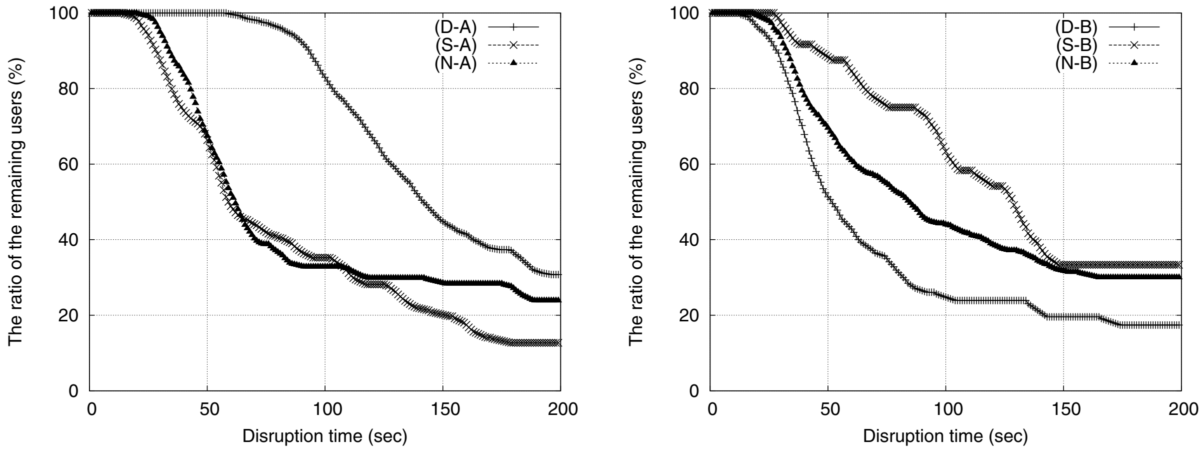
When we initiate a Rank-A session, we observe the flash crowd, which we call surge-in. A contrasting situation is observed when the stream is disrupted or the program is finished, which we call surge-out. In either case, an efficient mechanism to handle user churn is important because a system receives massive requests within a short period. If a system suffers from surge-in or surge-out, even the existing users will leave the session due to degraded quality, which makes the situation worse. Therefore handling surge-in and surge-out phenomena should be carefully designed. Table III shows the peak numbers of surge-in, surge-out, and their sum per second. Overall, churning rate soars when a disruption occurs only in (D,A) sessions.

TABLE III
PEAK NUMBERS OF SURGE-IN, SURGE-OUT AND THEIR SUMS WITH AND WITHOUT AN INTENSIVE DISRUPTION.

w/o disruption	D,A	D,B	S,A	S,B	N,A	N,B
total	4	3	4	3	4	3
surge-in	3	3	3	3	3	3
surge-out	3	3	3	3	3	3
w disruption	D,A	D,B	S,A	S,B	N,A	N,B
total	20	4	4	2	3	4
surge-in	19	4	4	2	3	2
surge-out	3	4	4	2	3	4

V. RELATED WORK

A few measurement studies on P2P based live streaming services have been published in the literature. There are two approaches in obtaining data about user dynamics: (1) streaming service providers (or their collaboration partners) can obtain user trace directly, and (2) the subset of user traces can be obtained by installing some monitoring programs into some participants (or a crawler is used if possible). Chu *et al.* [12] show the measurement results of End System Multicast (ESM). ESM is an internet broadcast system administrated by the authors themselves. They report comprehensive results such as mean session bandwidth of a receiver and packet loss ratio. They also address the connectivity constraints posed by NATs and firewalls. Ali *et al.* [13] analyze PPLive [8]



(a) Rank-A sessions (b) Rank-B sessions

Fig. 5. The ratio of the number of remaining users to the number of users at the beginning of the disruption.

and SOPCast [9], which are commercial P2P live video streaming services. The authors collected packet traces using Ethereal [10] and revealed the protocols of the two systems. Using the packet traces, the distributions of packet sizes, packet sending and receiving rates, and the relationships between peers are analyzed. Hei *et al.* [14] collected traces of PPLive by using a crawler and sniffing packets. The authors analyze participating users' characteristics such as the daytime effect, joining/leaving pattern, and geographic distribution of the users.

VI. Conclusions and Future Work

This study focuses on the user dynamics in live streaming services. We analyze user behaviors, how they join and leave live streaming services. Trace results are classified by the streaming programs' type (Type-D for TV dramas, Type-N for news, Type-S for talk shows) and popularity (Rank-A for high popularity, Rank-B for log popularity). By the experiment we performed to recognize long-stay users online, selecting old users as the parent (**Max**) is not such an effective measure to provide high quality streaming services. Around 20% of the users are classified as short-stay users, and around 30% of the users stayed in the session less than 5% of the session length. Type-D programs have higher portions of short-stay users than others; 30% in (D-A) session and 40% in (D-B) session. Since short-stay users incur high control overhead, efficient handling mechanism is needed. Surge-in (flash crowd) or surge-out phenomena is often observed among the users in live streaming services. Therefore, a system should provision high rate of user churn. We intentionally disrupted a streaming session for 200 seconds to see number of remaining users in the session during the disruption. Most of the users in (D,A) session wait until 80 seconds, while users in (D,B), (S,A), (N,A), and (N,B) sessions quickly leave the session after 30 seconds.

We conclude that the type and popularity of streaming programs significantly affects user behaviors in the session. Consequently, a live streaming system can provide better streaming quality by reflecting the type and popularity of the streaming program. Moreover, if a distribution structure can adapt the current rate of user churn, it can save the system resources. We are currently working on modeling the user behaviors in live streaming depending on the session type and popularity. Our next step is to develop a user pattern generator program in live streaming services based on our model, which can reproduce user behaviors similar to real trace sets.

Acknowledgments

This work was partly supported by NAP of Korea Research Council of Fundamental Science & Technology and the IT R&D program of MKE/IITA [2008-F-034-03].

References

[1] CacheLogic. http://www.cachelogic.com/.
[2] ipoque Internet Study 2008/2009. http://www.ipoque.com/resources/internet-studies/.
[3] Afreeca. http://www.afreeca.com/.
[4] Nowcom http://www.nowcom.co.kr/page/27.
[5] Zattoo. http://zattoo.com/.
[6] YouTube. http://www.youtube.com/.
[7] Akamai. http://www.akamai.com/.
[8] PPLive. http://www.pplive.com/.
[9] SOPCast. http://www.sopcast.org/.
[10] Ethereal. http://www.ethereal.org/.
[11] F. Wang, J. Liu, and Y. Xiong, "Stable peers: Existence, importance, and application in peer-to-peer live video streaming," in *INFOCOM,* 2008.
[12] Y.-H. Chu, A. Ganjam, E. T. S. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early experience with an internet broadcast system based on overlay multicast," in *USENIX Annual Technical Conference*, 2004.
[13] S. Ali, A. Mathur, and H. Zhang, "Measurement of commercial peer-to-peer live video streaming," in *ICST Workshop on Recent Advances in Peer-to-Peer Streaming*, 2006.
[14] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale p2p iptv system," in *Multimedia, IEEE Transactions on*, 2007.
[15] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An Analysis of Live Streaming Workloads on the Internet," in *Internet Measurement Conference*, 2004.
[16] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live p2p streaming approaches," in *INFOCOM,* 2007.