Measurement and Modeling of a Large-scale Overlay for Multimedia Streaming

Long Vu, Indranil Gupta, Jin Liang, Klara Nahrstedt
Department of Computer Science, University of Illinois at Urbana-Champaign, IL61801
{longvu2,indy,jinliang,klara}@cs.uiuc.edu

ABSTRACT

This paper presents results from our measurement and modeling efforts on the large-scale peer-to-peer (p2p) overlay graphs spanned by the PPLive system which is arguably the most popular and largest multimedia streaming p2p system today. We believe that our findings can be used to understand large-scale p2p streaming systems for future planning of resource usage, and to provide useful and practical hints for future design of large-scale p2p streaming systems. Unlike other previous studies on PPLive, which focused on either network-centric or user-centric measurements of the system, our study is unique in (a) focusing on PPLive overlay-specific characteristics, and (b) being the first to derive mathematical models for its distributions of channel population size and session length.

Our studies also reveal characteristics of multimedia streaming p2p overlays that are markedly different from existing file-sharing p2p overlays. Specifically, we find that: (1) Small PPLive overlays (as many as 500 nodes) are similar to random graphs in structure, (2) Average degree of a peer in the overlay (i.e., its out-degree) is independent of channel population size, (3) The availability correlation between PPLive peer pairs is bimodal, i.e., some pairs have highly correlated availability, while others have no correlation, (4) Unlike p2p file-sharing users, PPLive peers are impatient, (5) Session lengths (discretized, per channel) are typically geometrically distributed, (6) Channel Population Size variations are larger than in p2p file-sharing networks, yet they can be fitted with polynomial mathematical models. We conclude with a series of suggestions on how our findings can improve IPTV future design.

1. INTRODUCTION

The proliferation of large-scale peer-to-peer (p2p) overlays such as Kazaa, Gnutella, Skype, PPLive [2], RONs [4], etc., has created the need to characterize, and to understand the emergent properties of these overlays. A large fraction of existing characteristic studies focus on *file-sharing* p2p applications, such as Kazaa, Gnutella, and Napster. Some of the more prominent studies among these are by Ripeanu et. al. [14] on Gnutella, by Saroui et. al. on Naspter and Gnutella [15], and by Bhagwan et. al. on Overnet [6]. Although these studies have created a better understanding of the characteristics of p2p overlays, there is a risk that many system designers may believe that some of these conclusions are shared by many other p2p overlays.

This paper shows that many of the well-held beliefs about the characteristics of p2p file-sharing overlays may be false when one changes the application atop the p2p overlay. Specifically, we undertake a crawler-based study of a deployed application overlay network called PPLive. We believe that results obtained from our studies can be used to understand large-scale p2p streaming systems for future planning of resource usage, and to provide useful and practical hints for future design of large-scale p2p streaming systems.

PPLive is the most well-known instance of an IPTV (Internet Protocol Television) applications which have seen a dramatic rise in popularity and have received significant attention from both industry and academia. The number of subscribers is predicted to increase from 3.7 million in 2005 to 36.9 million by 2009. Revenues could reach US$10 billion at the end of this period [1]. This promising market has encouraged the rapid development of IPTV technologies including tree-based multicasts [5, 10, 16], receiver-driven p2p streaming [8, 11, 13, 21] and chunk-driven p2p streaming [2, 20], in which chunk-driven approach has emerged as the most successful technology in terms of the number of simultaneous viewers [9].

Among deployed chunk-driven p2p streaming systems, PPLive stands out due to its heterogeneous channels and increasing popularity. As of May 2006, PPLive had over 200 distinct online channels, a daily average of 400,000 aggregated users, and most of its channels had several thousands of users at their peaks [2]. During the Chinese New Year 2006 event, a particular PPLive channel had over 200,000 simultaneous viewers [9]. In our experiments, we observed that there are about 400 daily online channels.

There are several measurement studies about PPLive characteristics [3, 9, 19]. These existing studies tend to predominantly look at either network-centric metrics (e.g., video traffic, TCP connections, etc.), or at user-centric metrics (e.g., geographic distribution, user arrival and departure, user-perceived quality, channel population, etc.). Our crawler-based measurement studies are unique in focusing primarily on *overlay-based characteristics*, which lie somewhere in between the user-centric view and the network centric view. Of course, overlay characteristics are influenced both by an amalgamation of both user behavior and by the design of the underlying protocol and the network, yet they stand apart themselves. Our studies also expose new avenues for improving performance, reliability, and quality of IPTV systems in the future.

Results obtained from our extensive experiments (stretching from April 2006 until the end of Dec 2006) indicate that PPLive overlay characteristics differ from those of p2p file-sharing. Our major findings are: (1) Small PPLive overlays (as many as 500 nodes) are similar to random graphs in structure, (2) Average degree of a peer in the overlay is independent of channel population size, (3) The availability correlation between PPLive peer pairs is bimodal, i.e., some pairs have highly correlated availability, while others have no correlation, (4) Unlike p2p file-sharing users, PPLive peers are

This research was supported in part by NSF CAREER grant CNS-0448246, NSF ITR grant CMS-0427089, NSF ANI 03-23434, and NSF CNS05-09314.
ACM-Qshine 2007, August 14 - 17, 2007, Vancouver, British Columbia.

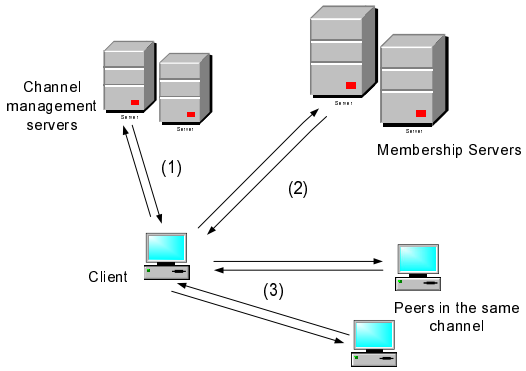

Figure 1: PPLive membership and partnership protocols.

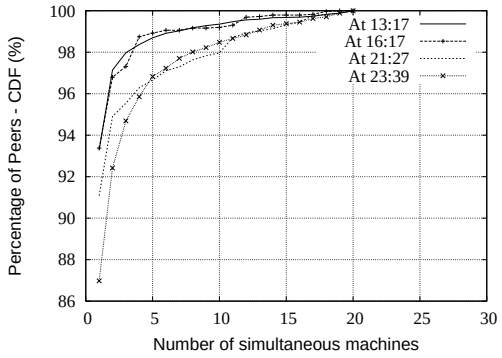

Figure 2: 10 machines running snapshot operation crawl 98% peers in one channel, compared to when 20 machines are used. We use 10 Planetlab nodes in our experiments.

impatient, and (5) Session lengths are typically geometrically distributed, (6) Channel Population Size variations are larger than in p2p file-sharing networks, yet they can be fitted with polynomial mathematical models. All the above conclusions, except (4), are markedly different from the well-known characteristics of p2p file-sharing systems [6, 14, 15].

The rest of this paper is organized as follows. We describe PPLive basics and preliminary definitions in Section 2. Section 3 presents and justifies our crawler methodology. We study the structure of the PPLive overlay in Section 4 and evaluate host availability interdependence in Section 5. Section 6 shows why PPLive peers are impatient. Section 7 discusses the impacts of episode-based PPLive channels on the overlay and its size. We conclude and discuss in Section 8.

2. PPLIVE BASICS

Before embarking on a characteristic study of PPLive, we briefly summarize its basic architecture as well as the structure of its channels, in each case giving some basic definitions that will be reused later in the paper. For more details, see our technical report [17].

2.1 PPLive Protocols

Figure 1 shows the actions of a PPLive node to join the network: (1) retrieve a list of channels from channel management servers via $HTTP$; (2) for the interested overlay, retrieve a small set of member nodes from the membership servers via UDP; (3) use this seed partner list to harvest (learn about) other candidate partners by periodically probing existing partners (and sometimes membership servers) via UDP. If a PPLive node is inside NAT or firewalls, UDP in the above steps may be replaced by TCP - further details are in [3, 9, 17].

In this paper, we refer to a given $< IP, port >$ tuple as a "node" or a "peer" - the combination of both a client machine and a user. The term "client" refers only to the machine (e.g., workstation) that the PPLive node is running on, while "user" refers to the human user, and these should not be confused with node or peer.

2.2 PPLive Overlay and Node Degree

In this paper, we focus on episode channels. Each of these channels forms an overlay of nodes. We formally define the PPLive overlay as a graph $G = (V, E)$. Recall that each PPLive overlay corresponds to an individual PPLive channel. Each node (or peer) is defined as a given $< IP, port >$ tuple and belongs to V. Each partner (or neighbor) of this node, appearing in its partner list, then corresponds to an edge (or link) in E.

k response degree:. We are interested only in the out-degree of a node in the overlay, and henceforth call this simply as "degree". The *k response degree of a node* is defined as a aggregated set of partners in the first k responding packets from a node. In other words, when node gets queried repeatedly (once a second) for its partners, it returns its partners and they are aggregated. Because the overlay is dynamic and how a node returns its partners is unknown, we spend 15 seconds (corresponding to 15 responses) to estimate the "partner list" of a queried node. Our experiment verifies that this hypothesis is sufficient to estimate node degree [17]. We use a default setting of $k = 15$ in our experiments, especially for our Partner Discovery Operator in section 3. Henceforth in this paper, the term *node degree* stands for *k response degree*.

Active Peer:. Given an overlay G and a peer v, v is considered to be an *active peer* in G if either v appears in the membership list for G at one of the membership servers, or v is present in the partner list of some other peer u that is also an active peer. Notice that the definition is recursive. Formally, we define the predicate:

ACTIVE(v, G) = $\{v \in$ *Membership Server List for* $G\}$ *OR* $\{\exists u$: *ACTIVE*(u, G) *AND* $v \in u.PartnerList(G)\}$

This definition is more inclusive than that in [9] because our definition also includes "silent" peers that may behind NATs. These peers are about 50% of total peers in PPLive overlays[17].

Channel Population Size:. Channel Population Size is the number of active peers attending the channel in a certain period of time, usually one execution of the crawler. We use channel population size interchangeably with "channel size", and "overlay size."

3. STUDY METHODOLOGY

	Chan. size	**PS Len**	**#Pro**	**Pro Len**	**Type**
A	32K-45K	6h15m	6	36m-2h	Movie
B	10K-15K	4d4h	300	20m	Cartoon
C	8K-12K	1d2h16m	40	40m	Movie

Table 1: Three channels in our experiments.

Settings:. Our crawler works in the following manner: a machine (either a Linux machine in our cluster at UIUC, or a PlanetLab node) joins a given PPLive channel, and then crawls peers attending that channel. The crawler is used to execute what we call the *Snapshot Operation* which will be described shortly. Ethereal is used to trace traffic between PPLive nodes and servers.

Most of our experiments were focused on three specific repre-

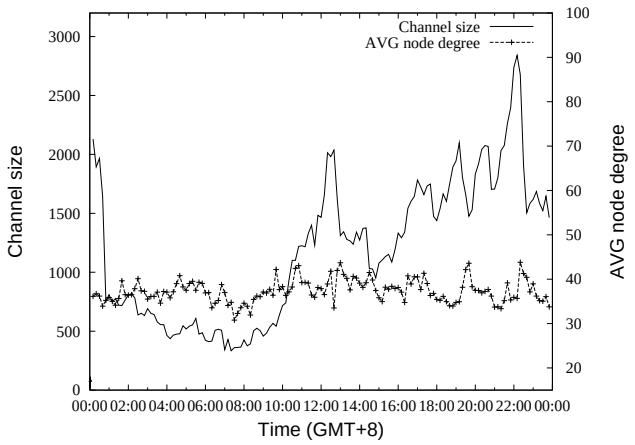

Figure 3: Average node degree is independent of channel population size (or channel size)

sentative channels from PPLive - Table 1 shows their basic features. We anonymize these channels by naming them as A, B, and C. A is one of the most popular channels, B has shorter programs and the largest program set, while C is somewhat in between A and B. Since a large fraction of PPLive users are in China, we use Chinese Time Zone (GMT+8) in our plots.

Studied Metrics:. The primary metrics of interest in this paper are: (1) node degree in the overlay, (2) overlay structure (or overlay clustering coefficient), (3) availability correlation among nodes in the overlay, (4) channel population size (or overlay population size), and (5) node session lengths per overlay. In order to support this, we use our crawler to develop a *Snapshot Operation* and a *Partner Discovery Operation*. Our "Snapshot" is different from the notion of Chandy and Lamport's "consistent snapshots" [7].

Snapshot Operation:. This operation works by repeatedly fetching partner lists and querying returned entries as follows. First, the initiator requests the initial peer list from one of the PPLive membership servers, and uses this to initialize a local list denoted as L. Second, the initiator then continuously scans L in a round-robin fashion, by sending a request for partner list to each entry, and appending to L new peers (i.e., ones that it has not heard about before) received in the partner list replies. Third, when the initiator has received fewer than n new peers among the last Δ peers received as partner lists, Snapshot Operation terminates. We use $n = 8, \Delta = 1000$ in all of our experiments; with this setting, the snapshot operation typically takes between 3 to 8 minutes depending channel population size. To avoid flooding network with our messages, new snapshot operations are initiated only once every 10 minutes.

The above snapshot operation is different from the crawler of [9] in two ways – their crawler runs once each minute and for about 15 seconds; thus to crawl a large part of the network, it imposes a high load on PPLive. Second, their crawler counts only responding peers, thus undercounting the channel size, i.e., they use a more restrictive definition for $ACTIVE(u)$ than our definition in Section 3. Third, their crawler stops after a fix amount of time, regardless the channel size while our crawler stops depending overlay size.

To increase the coverage of our snapshot, we run it in parallel on multiple machines. Figure 2 shows the captured number of peers with m machines as a fraction of the captured number of peers with 20 machines (at four different times). We observe that 10 machines cover about 98% of peers covered by 20 machines. Hence, in this paper, we decided to use 10 geographically distributed PlanetLab nodes to run simultaneous snapshot operations.

Partner Discovery:. This operation obtains the k response degree of a node defined in section 2.2. We repeatedly request a peer (once every second) to send its partner list. The first k received responses are aggregated to create the k response degree.

4. PPLIVE OVERLAY STRUCTURE MAY BE SIMILAR TO RANDOM GRAPHS

It is well-known that the degree distribution in file-sharing p2p networks is scale-free and hence likely small world [14, 15]. This section shows that like file-sharing p2p overlays, the average node degree in the PPLive overlay is also independent of the channel population size. However, in direct contradiction to file-sharing p2p overlays, the structure of the PPLive overlay is closer to that of random graphs, under certain situations.

4.1 Average Node Degree is Independent of Channel Population Size

We collect the k response partner lists of nodes using our Snapshot Operation and Partner Discovery engine (Section 3). During a given snapshot operation (over 10 minutes), we simultaneously run the Partner Discovery to obtain the k response degree of 300 randomly selected peers that are both active and responsive. Figure 3 shows the variation of the average k response node degree and Channel Population Size (henceforth degree) of a channel during a 24 hour period.

These Figures first indicates that although the average node degree varies, it stays within a small range - between 30 to 43 over the course of day. More importantly though, *there appears to be no correlation between the variation of average degree and the channel size.* Thus we conclude that the average degree of a PPLive node does not depend on the channel population size.

This behavior can be explained since a peer in an overlay only needs to communicate once in a while with a few of peers to exchange data, advertise availability, and discover new partners. Therefore, even when the channel becomes large, a peer can still preserve viewing quality without establishing too many new connections.

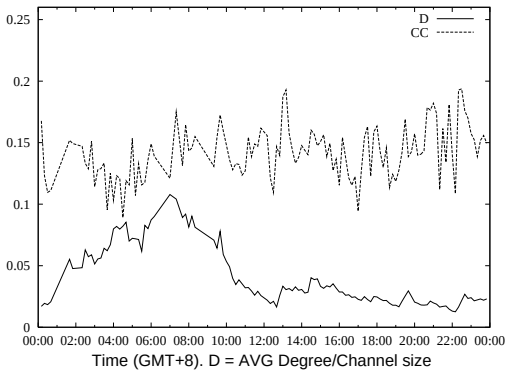

Figure 4: Overlay resembles a random graph when channel size is small (around 500 nodes) but becomes more clustered when channel size grows. Relationship between D and CC.

4.2 Randomness of Overlay Depends on Channel Population Size

The distinction between a random and a non-random graph can be measured by a metric called "Clustering Coefficient" (CC) [18].

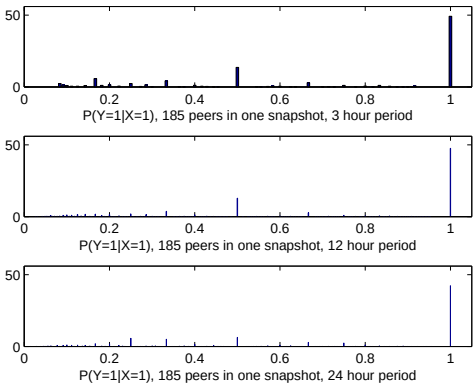

Figure 5: Peers occurring in the same snapshot may occur together again. Plot shows PDF of availability correlation. Y-axis is % host pairs.

Informally, the CC in a graph is defined as follows: for a random node u with two neighbors v and w in its partner list, the CC is the probability that either v is in w's partner list, or vice versa. The farther the CC is from the above value, the less random (i.e., more clustered) the graph is. For our experiment, we first calculate the average degree of the PPLive overlay (measured as in Section 4.1), and use it to calculate D as follows:

$$D = (Average\ node\ degree)/(Channel\ size) \quad (1)$$

We then compare the CC (measured as described below) to D (the unconditional probability that v links to w). The CC information is measured simultaneously to the degree measurement, and as follows: in each snapshot, we randomly select 300 active peers (i.e. root nodes). For each root node R, we first use partner discovery to obtain its partner list. Second, we pick randomly two active partners P_1 and P_2 in R's partner list and obtain their partner lists, again via the partner discovery operation. Third, we verify whether P_1 is in P_2's partner list or not, and vice versa. If P_1 is in P_2's partner list (or vice versa), we increase a variable called $Count$ by 1. $Count$, initialized to 1, represents the total number of edges existing in all such partner pairs. Then, CC is computed as follows:

$$CC = Count/(2 \times RootNodeNum) \quad (2)$$

In which, $RootNodeNum$ is the number of root nodes whose two active partners P_1 and P_2 are verified. Figure 4 plots, for two different values of $k = 5, 15$, the 24-hour variation of D and CC. This experiment was done at the same time and for the same channel as Figure 3. Thus, notice that from 04:00 am to 08:00 am, when the channel population size is small, the value of CC approaches the value of D, especially for higher values of k. This indicates that *when Channel Population Size is small, the structure of the PPLive overlay graph approaches a random graph*.

This is explained by the fact that for "small" channels (with as many as 500 nodes), peers indeed connect fairly randomly to each other. As the channel population size increases (10:00 am onwards in Figure 4), the CC is only about six times that of the value of D. This is still indicative of some randomness of the graph, although it is clear large channel population sizes lead to more clustering.

5. PEER AVAILABILITY INTERDEPENDENCE

File-sharing p2p systems are known to have host availabilities uncorrelated [6]. In comparison, we show that: (1) unlike in p2p file-sharing systems, PPLive peer pairs occurring together in some snapshot have highly *correlated* availabilities, while (2) like in p2p file-sharing systems, peer pairs that are randomly selected will have highly *uncorrelated* availabilities.

We measure the correlation between the availability of two peers X and Y by using a similar technique as in [6]. Specifically, let $X = 1$ (resp. $Y = 1$) be the event that peer X (resp. Y) occurs as an active peer in a given snapshot. Then, for the peer pair (X, Y), we calculate $P(Y = 1|X = 1)$, i.e., the conditional probability that given X is present in a given snapshot, Y will be too. We then compare this conditional probability to the unconditional probability that peer Y occurs in a given snapshot, i.e., $P(Y = 1)$. The closer the two values, the more uncorrelated are X's and Y's availability patterns.

5.1 Nodes in the Same Snapshot Have Correlated Availability

Given traces of a series of snapshots (for Channel A) taken over a contiguous time period (we use three settings: 3 hours, 12 hours, and 24 hours), we select a set of 185 peers from one particular snapshot. Note that we have 144 snapshots for 24 hours. Figure 5 shows the conditional probability $P(Y = 1|X = 1)$, for each node pair in this set. 50% of node pairs show a high correlation in availability, i.e., $P(Y = 1|X = 1) = 1$.

We believe there are two factors contributing to this behavior: first, user pairs that appeared in the same snapshots are likely to have similar interests in terms of channel viewing contents. Second, and perhaps more importantly, certain peer pairs that occur together in a snapshot are perhaps "well-matched" as streaming relays for each other. It is possible that PPLive's inter-overlay optimizations [12] cause one client's presence to draw in other well-matched clients for relaying.

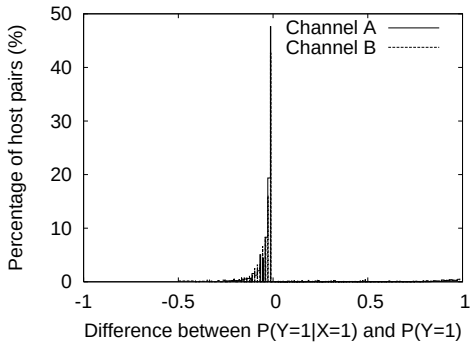

Figure 6: Randomly selected pairs of peers have uncorrelated availabilities. Plot shows PDF of availability correlation from 500 random peers taken from channels A and B.

5.2 Random Node Pairs Have Independent Availabilities

We ran a similar experiment as in Section 5.1, except that we selected 500 random peers from among 39412 crawled peers of channel A, as well as 500 random peers from 11527 peers of channel B, each collected over a 24 hour period. Then, we computed the difference between $P(Y = 1|X = 1)$ and $P(Y = 1)$ for each host pair (among the set of 500) over the 144 snapshots, corresponding to 24 hours. In contrast to results in Section 5.1, Figure 6 shows that random peer pairs have completely independent availability behavior. In particular, 87% peer pairs in channel B (92% in channel A) lie between +0.2 and -0.2, indicating independence in

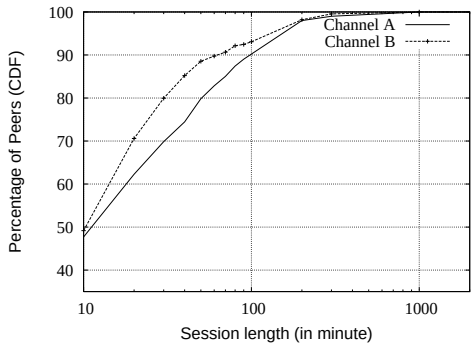

Figure 7: PPLive peers are impatient. CDF of session lengths is shown. The X-axis is on a log-scale.

availability among these peers. This is explainable because random peers are unlikely to have either correlation in user interest, or be well-matched in relaying feeds.

In conclusion, unlike p2p file-sharing systems, media streaming p2p systems may exhibit a highly correlation availability among certain node pairs. Systems designers will have to account for this, regardless of whether it arises from user interests or from internal optimized PPLive design (in the latter case it is a good p2p system design principle).

6. PPLIVE PEERS ARE IMPATIENT

It has been widely reported, e.g., [15], that users of p2p file-sharing systems are "patient," i.e., they do not mind waiting hours, and sometimes even days, for a file as large as 1 GB, to download. In the PPLive environment, due to the streaming nature of the content, the opposite is true. In other words, PPLive users are very impatient in terms of staying in a particular channel. They usually switch channels during their watching time.

Figure 7 shows session lengths of 5000 random peers taken from 38675 peers in channel A, and 5000 random peers taken from 11625 peers in channel B. We observe that about 50% sessions are shorter than 10 minutes, 60% of A's sessions and 70% of B's sessions are shorter than 20 minutes, and over 90% sessions from both channels are 100 minutes or shorter. This implies that *PPLive nodes are impatient*, i.e., they rarely stick to a channel for too long.

This opposite behavior arises out of both a difference in application characteristics, as well as from user behavior. Since p2p file-sharing overlays like Kazaa are *batch-mode* delivery systems in which the human users can go away from the client machine while it continues to download content, session lengths are long. In comparison, PPLive application is a *streaming-mode* one, where user can obtain utility from the application only if she is actively present near the client machine. If the user is not sitting at her machine, she has no incentive to keep PPLive running, hence the session times are shorter. This is especially true to PPLive, most of streaming content in PPLive channel is for entertainment and users usually use PPLive in leisure times [9].

There are other reasons (both application and user-based) contributing to the short session lengths. First, PPLive users are likely to switch from one channel to another because of a loss of interest - home television viewing often suffers from the same malady! Secondly, PPLive nodes face a longer start-up delay [9] than nodes in p2p file-sharing systems or traditional TV. We have observed that newly joining nodes need tens of seconds to a minute to join a channel, with the latency being even higher if the channel is really small (due to the scarcity of potential neighbors). Furthermore, the long start-up delays increase the likelihood of the user switching to a different (and more popular) channel. The difference in session

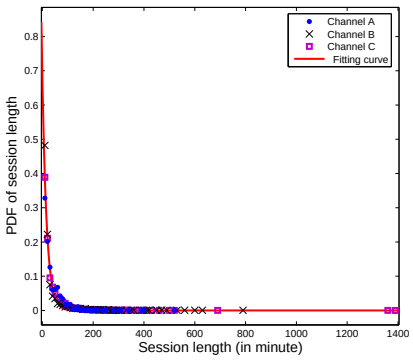

Figure 8: PDF of session length fits a geometric series.

length distributions for channels A and B in Figure 7 also brings out an important observation – *short-term sessions (session is shorter than 60 minutes) depend on channel characteristics, while long-term sessions (session is longer than 120 minutes) depend on user preference*. For short term behavior – notice that number of short-term sessions in B is *always* about 10% more than that in A. This behavior arises because channel A is more popular, as well as because channel B's programs are shorter (around 20 minutes) compared to channel A. In contrast, long-term sessions of the two channels converge, indicating that long-term sessions depend on user preference – for any channel, about the same fraction of users stays "too long." In other words, if the channel content is favorite to user, she may be willing to watch for a very long time without switching to another channel.

Channel	a	b
A	0.6378	-0.05944
B	1.183	-0.09878
C	1.079	-0.09594

Table 2: Coefficients of geometric series, fitted by Matlab.

Session Length Model:. To understand properties of PPLive sessions, we use Matlab to model the PDF of session lengths. Due to the nature of our crawler, node's session lengths were measured only multiples of 10 minute periods, thus we an appropriate model would be a *discrete* mathematical series, rather than a continuous distribution. To fit our session length data, we tried out a variety of Matlab models, and finally settled on the geometric series as the best fit. Figure 8 shows fitting curve obtained from Matlab for three different channels A, B, and C. While the fitted curve is an exponential function of time (since Matlab offers only continuous fits of data), we express the session length distribution as the (equivalent) geometric series due to the nature of our collected data.

Concretely, the geometric series can be expressed as follows. Let y be the probability that a node's session length is measured as $x \cdot 10$ minutes (where $x > 0$). Our Matlab models reveal the relationship between y and x as:

$$y = a \cdot e^{10 \cdot b \cdot x} \tag{3}$$

Here, a and b are constants. a is the base of the geometric series, and the multiplicand in the geometric series is $r = e^{10 \cdot b}$. Factor 10 in the above equation arises from our discretized session lengths that are multiples of 10 minutes.

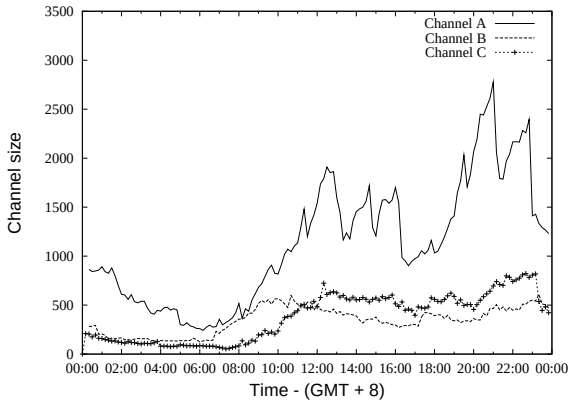

Figure 9: Channel population size distribution (or Channel size). Channel population size peaks at noon and night, and is smallest in the morning.

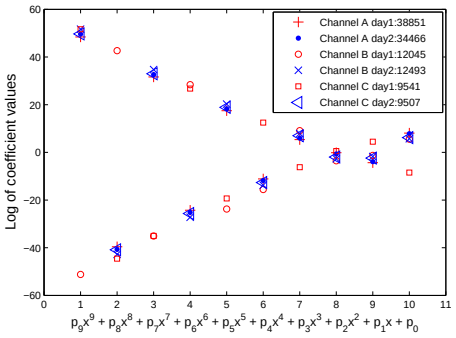

Figure 10: Logarithmic of values of coefficients of 9th degree polynomial fitting A,B, and C. $y = sign \cdot Log(abs(coefficient))$**,** $sign = -1$ **if coefficient is negative.**

We obtained the values of a and b by fitting the data to a continuous exponential curve in Matlab. We verified that this indeed leads to a geometric series by verifying, for each channel, that the value of $\Sigma_{i=1}^{\infty} a \cdot r^i$ turned out to sum to 1.0. For instance, channel A's exponential fit gives us a = 0.6378 and b = -0.05944, and the above sum turns out to be approximately 1.

In conclusion, the application characteristics and user behaviors cause *very short session lengths and consequently a higher degree of churn in PPLive than in p2p file-sharing overlays*. The geometrically distributed session lengths of nodes (per channel) can be used to accurately model node arrival/departure behavior in simulations of media streaming p2p systems. This can be used to improve the "believability" of simulation set-ups for media streaming p2p systems, as well as to incorporate session-length-based optimizations at run-time in real deployments. The geometrically distributed session length times also indicate a high degree of *homogeneity* across nodes in the session lengths, and this indicates that homogenous protocol designs have substantial promise and are a good match for media streaming p2p overlays - this does not of course preclude benefits from heterogeneous protocol designs. Future designs for both streaming p2p overlays and "generic p2p routing substrates" will have to keep these in mind.

7. CHANNEL POPULATION SIZE VARIES WIDELY OVER A DAY

Studies on file-sharing p2p systems [6] showed that diurnal patterns and churn exist, but the size of a p2p overlay stays stable in spite of these features. The findings in this section show that PPLive-style networks have highly variable channel population size (as well as high churn and diurnal patterns).

This section studies time variation pattern of channel population size in PPLive channels. The channel population size variation pattern is also modeled by Matlab in later part of this section. Figure 9 shows the variation of channel size for the three PPLive channels over the course of a day. We observe that all channels have peak populations at noon and evening/night, and are smallest in the morning. This is clearly true because streaming content in PPLive is for entertainment and users usually use PPLive in spare time (at noon and evening/night.)

Next, notice that the distribution of channel population sizes for B and C look much flatter than A – the former two vary between 100 and 600, while A shows a variance of over 2000. This pattern arises from the content of these channels. B and C are series of equal length programs while A consists of longer movies. As a consequence, compared to A, people join channel B to watch short skits and leave more quickly, while viewers that join A stay on for longer, perhaps long enough for the movie to end. Finally, notice that the channel population size peaks are both unevenly spaced and are different in height - this is because A's programs have variable lengths (and popularity.)

Channel Population Size Variation Model:. To learn the variation pattern, we make use of Matlab to model channel population size variation over the course of a 24 hour period (1 day) for each given channel. Due to the complexity of variation pattern, the obtained results from Matlab show that only polynomials whose degree is higher than $8th$ can support more than 0.9 of "Adjusted R-Square Statistic". Moreover, when we increase degree over 9, the "Adjusted R-Square Statistic" plateaus out. We, therefore, opt for a $9th$ degree polynomial to fit data of channels A,B, and C. Specifically, our $9th$ degree polynomial is:

$$y = \sum_{i=0}^{9} p_i x^i$$

Here, p_is are coefficients. We fitted the channel population size variations over the course of a day for 3 channels, each for 2 separate days. These six $9th$ degree polynomials obtained from 3 channels in two days show a very important (and surprising) property: *for any i, the coefficient* p_i *is similar for most channels on any day, except for the sign*. This is shown in Figure 10, which plots on the y-axis the logarithmic of the value of coefficients ($y = sign \cdot Log(abs(coefficient))$), with $sign = -1$ if the coefficient is negative. Notice that in the majority of cases, coefficients of a given channel appear to not change from one day to the next. Yet, there are exceptions such as channel B on day 1 and day 2 have different signs for the first 5 terms. A hypothesis we had was that perhaps coefficients are related to channel population. This hypothesis was not borne out by the Figure 10-the largest channel (A) on day 1 has many high degree coefficients matching with the smallest channel (C) on day 2.

The reasons behind such partly-predictable channel size population variation comes from various causes. These include user behavior, diurnal patterns, the nature of the PPLive protocol and system, and channel content. Although it is not possible to pinpoint any one of these causes as the primary one for the predictability in polynomial coefficients, this feature is a boon for media streaming

p2p systems such as PPLive. For instance, even though the same channel may have different coefficients on different days, it may be possible to design "Channel Population Size predictors" for such systems which predict coefficients by looking at (or sampling) the first few hours of population variation for a particular day. This would address the churn problem more directly than possible in today's p2p file-sharing systems.

In conclusion, unlike p2p file-sharing systems which only show diurnal patterns, p2p media streaming environments show a higher variation in channel population size, besides predictable absolute values of coefficients in a polynomial fitting curve.

8. CONCLUSIONS AND LESSONS LEARNED

Results obtained from our extensive experiments indicate that PPLive overlay characteristics differ from those of p2p file-sharing. Our major findings are: (1) Small PPLive overlays (as many as 500 nodes) are similar to random graphs in structure, (2) Average degree of a peer in the overlay is independent of channel population size, (3) The availability correlation between PPLive peer pairs is bimodal, i.e., some pairs have highly correlated availability, while others have no correlation, (4) Unlike p2p file-sharing users, PPLive peers are impatient, and (5) Session lengths are typically geometrically distributed, (6) Channel Population Size variations are larger than in p2p file-sharing networks, yet they can be fitted with polynomial mathematical models. Followings are our lessons learned and implications:

1. Since the availability correlations among node pairs is bimodal, this can be used to *fingerprint*, at run-time, which pairs of nodes are correlated and which not. The bimodality of the behavior means that a few (random) sample points will suffice in categorizing each node pair as either "correlated" or "not correlated". This availability information can then be used to create overlays (or sub-overlays) that are either present all at once, or to route media streams (for a given channel) to a recipient node via other correlated nodes that are likely to also be up at the same time. This means simulations of media streaming p2p systems need to account for this bimodal availability correlation in the injected churn models.

2. Geometrically distributed session lengths of nodes (per channel) can be used to accurately model node arrival/departure in simulations of media streaming p2p systems. Further, since the geometric distribution is indicative of *memoryless* session lengths (per node), this means that nodes are homogeneous w.r.t. their availability. Thus, *homogeneous* protocol designs for p2p overlays in this application space are reasonable. In other words, protocols that treat participating nodes equally are simpler and work effectively. This doesn't of course preclude benefits of heterogeneous protocol designs based on metrics such as bandwidth, CPU speed, etc.

3. As indicated by our conclusion (1), small PPLive overlays work well by creating random overlay graph structures - thus, simple and homogeneous solutions work well at medium-scale and not too large channel size. Further, even when overlays are large, our conclusion (2) above indicates that homogeneous designs work well too. Notice that this does not preclude the use of heterogeneous protocol design.

In conclusion, the differences between PPLive overlays and p2p file-sharing overlays drawn from our study show that p2p systems designers may need to account for application nature. This study is also indicative of the challenge in designing "generic" p2p substrates catering to a wide variety of applications. Since custom-built substrates are too wasteful, it may thus be important for systems designers to address classes of p2p applications with common characteristics. Finally, a deeper study of user behavior (e.g., via HCI research) may yield novel p2p overlay design principles.

9. REFERENCES

[1] Multimedia research group inc.

[2] PPLive homepage, http://www.pplive.com/.

[3] A. Ali, A. Mathur, and H. Zhang. Measurement of commercial peer-to-peer live video streaming. In *Workshop in Recent Advances in Peer-to-Peer Streaming*, 2006.

[4] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *ACM SOSP*, pages 131–145, 2001.

[5] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *ACM SIGCOMM*, 2002.

[6] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. In *IPTPS*, 2003.

[7] M. Chandy and L. Lamport. Distributed snapshots: Determining global states of distributed systems. *ACM Transactions on Computer Systems*, pages 131–145, 1985.

[8] M. Hefeeda, A. Habib, D. Xu, B. Bhargava, and B. Botev. Collectcast: A peer-to-peer receiver-driven overlays. In *ACM Multimedia*, 2003.

[9] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. A measurement study of a large-scale P2P IPTV system. Technical report, Polytechnic U., 2006.

[10] Y. hua Chu, S. G. Rao, and H. Zhang. A case for end system multicast. *ACM SIG-METRICS*, 2000.

[11] J. Liang and K. Nahrstedt. Dagstream: Locaity aware and failure resilient peer-to-peer streaming. In *SPIE/ACM MMCN*, 2006.

[12] X. Liao, H. Jin, Y. Liu, L. M. Ni, and D. Deng. Anysee: Peer-to-peer live streaming. In *IEEE Infocom*, 2006.

[13] R. Rejaie and S. Stafford. A framework for architecting peer-to-peer receiver-driven overlays. In *ACM NOSSDAV*, pages 42 – 47, 2004.

[14] M. Ripeanu, I. Foster, and A. Iamnitchi. "Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design". *IEEE Internet Computing Journal*, 6(1), 2002.

[15] Saroiu, P. K. Gummadi, and S. D. Gribble. Measuring and analyzing the characteristics of Napster and Gnutella hosts. *Multimedia systems*, 2003.

[16] D. A. Tran, K. A. Hua, and T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. In *Infocom*, 2003.

[17] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt. Mapping the pplive network: Studying the impacts of media streaming on p2p overlays. Technical report, UIUC, 2006.

[18] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 1998.

[19] X.Hei, C.Liang, J. Liang, Y. Liu, and K. W. Ross. Insight into PPLive: Measurement study of a large scale P2P IPTV system. In *WWW 06 Wshop. IPTV Svcs. over WWW*, 2006.

[20] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming. In *IEEE Infocom*, 2005.

[21] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. Donet: A data-driven overlay network for efficient live media streaming. *IEEE Infocom*, 2005.